doi:10.6041/j.issn.1000-1298.2024.03.036

基于区块链的农产品供应链溯源数据多条件查询优化方法研究

高官岳[1,2]　孙传恒[2,3]　罗　娜[2,3]　徐大明[2,3]　邢　斌[2,3]

(1.上海海洋大学信息学院,上海 201306;2.国家农业信息化工程技术研究中心,北京 100097;
3.农产品质量安全追溯技术及应用国家工程研究中心,北京 100097)

摘要:随着基于区块链的农产品溯源系统迅速发展,区块链查询能力面临着巨大挑战。对于供应链参与方来说,区块链中保存的数据多为编码或序列化的数据,使得供应链参与方的审计和监督等存在多条件查询的工作变得十分困难。通常情况下,原生区块链并未提供满足多条件查询的查询方式。因此,为了实现多条件查询并提高查询效率,本研究提出一种农产品溯源数据多条件查询优化方法。首先,该方法采用一种优化的 Merkle 树结构(n-Tree)对交易信息进行重构,从而提供更高效的条件验证能力。其次,通过自适应多条件区块布隆过滤器判断交易信息中查询条件的存在性,进而快速过滤区块。最后,提出一种应用 TWTN-Heap(Time weight and transaction number based heap)结构的索引构建方法,以区块权重为序构建主条件相关的区块号索引列表。产品数据的查询过程包括遍历区块号索引列表、过滤非相关区块以及验证特定查询条件,从而获得条件查询结果。实验结果表明,本研究提出的产品数据条件查询优化方法能够有效地解决农产品供应链面临的条件查询问题,同时保证查询时间消耗维持在 15 ms 左右,查询效率较默克尔语义字典树(Merkle semantic trie,MST)方法提高 60.9%,较原始遍历(Orignal traverse,OT)方法提高 87.7%。

关键词:农产品供应链;区块链溯源;条件查询;n-Tree;布隆过滤器

中图分类号:TP311.11　　**文献标识码**:A　　**文章编号**:1000-1298(2024)03-0362-13　　**OSID**:

Blockchain-based Multi-condition Query Optimization Method for Traceability Data of Agricultural Product Supply Chain

GAO Guanyue[1,2]　SUN Chuanheng[2,3]　LUO Na[2,3]　XU Daming[2,3]　XING Bin[2,3]

(1. *College of Information Technology*, *Shanghai Ocean University*, *Shanghai* 201306, *China*
2. *National Engineering Research Center for Information Technology in Agriculture*, *Beijing* 100097, *China*
3. *National Engineering Laboratory for Agri-product Quality Traceability*, *Beijing* 100097, *China*)

Abstract: With the rapid development of blockchain-based agricultural product traceability systems, blockchain query capabilities face great challenges. For supply chain participants, most of the data stored in the blockchain are coded or serialized data, which makes the process of multi-condition query such as audit and supervision of supply chain participants very difficult. In general, native blockchains do not provide a query method to satisfy multi-condition queries. Therefore, in order to realize multi-condition query and improve query efficiency, an optimization method for agricultural product traceability data was proposed. Firstly, the method used an optimized Merkle tree structure (n-Tree) to reconstruct the transaction information, so as to provide more efficient conditional verification ability. Secondly, the adaptive multi-condition block Bloom filter was used to judge the existence of query conditions in the transaction information, and then the blocks were quickly filtered. Finally, an index construction method using time weight and transaction number based heap structure was proposed, and the block number index list related to the main condition was constructed in the order of block weight. The process of querying product data included traversing the block index list, filtering irrelevant blocks, and validating

收稿日期:2023-11-24　修回日期:2023-12-25

基金项目:国家重点研发计划项目(2022YFD2001304)和江苏省科技计划(重点研发计划现代农业)项目(BE2023315)

作者简介:高官岳(1998—),男,硕士生,主要从事区块链技术研究,E-mail:gaogy670@gmail.com

通信作者:孙传恒(1978—),男,研究员,博士,主要从事区块链、物联网、大数据追溯技术研究,E-mail:sunch@nercita.org.cn

specific query conditions to obtain conditional query results. The experimental results showed that the query method proposed can effectively solve the problem of conditional query in the supply chain of agricultural products. At the same time, the query time consumption was maintained at about 15 ms, and the query efficiency was improved by 60.9% compared with Merkle semantic trie method and 87.7% compared with original traverse method.

Key words: agricultural product supply chain; blockchain traceability; conditional query; n-Tree; Bloom filter

0　引言

农产品供应链管理是一种对农产品的生产、运输、存储等相关活动进行协调、监督和优化的过程，其有助于资源优化、降低成本、提高效率，为企业的长期稳定发展打下坚实基础[1]。其参与主体众多，包括原材料供应商、制造商、批发商、零售商和最终用户，这些主体共同构成了产品生产和流通过程中的关键角色。区块链是一种分布式数据存储技术，旨在提供一种去中心化、不可篡改、可追溯和可审计的能力。区块链技术的快速发展推动了其在各个领域的应用，尤其在农产品供应链管理方面起到了至关重要的作用[2-3]。应用区块链技术，可以为农产品供应链提供更为强大的数据管理能力、数据溯源能力并促进跨组织的协作，提升企业竞争力。

然而，随着农产品供应链数据量的急剧增加，促使基于区块链的供应链管理系统需要为参与方提供更强的审计及监督能力[4-6]。这种以数据为驱动的需求导致参与方对农产品溯源数据查询的依赖不断增加，并提出了更为细粒度的查询要求。原生区块链如比特币和以太坊，通常使用键值(Key - Value，KV)数据库存储区块链数据。在查询交易数据时，需要通过交易 ID 进行检索。对交易内部多个信息进行查询时，只能通过遍历区块及交易内容来实现。再如 Hyperledger Fabric 等联盟链项目，虽然提供了 KV 存储方式来满足结构化数据的查询，但在面临未知结构化数据的多条件查询时，单一的 KV 查询方式无法充分满足查询要求。由于交易内部数据的数据类型复杂字段未知，且多为序列化数据，通常不具有可读性，这在一定程度上增加了查询的难度。在农产品供应链管理过程中，如何提升查询效率、满足区块链多条件查询并精准定位查询内容成为了迫切需要解决的问题[7-8]。一些方法包括同步交易数据到链下数据库、构建多级索引等，能够较好地提升查询效率，但多数情况下不能满足多条件查询需求。一些方法替换了底层数据库引擎可以实现交易的富查询，但对于原生区块链来说成本巨大，一方面数据迁移过程较长，另一方面需要额外针对某些特殊字段组合构建索引结构，不利于参与方的类轻节点结构进行审计及监督工作。

国内外的研究主要集中于两方面，一方面为引入外部数据库来扩展区块链的查询能力，另一方面为根据特定需求构建索引结构，实现更加高效的数据查询[9]。将区块链与外部数据库结合的研究中，一些学者提出链上链下协同的方法[10-11]。杨信廷等[12]提出为农产品供应链追溯系统构建“数据库 + 区块链”的链上链下双存储结构，依赖外部数据库查询引擎和优化功能实现快速查询。XU 等[13]提出将旧区块上云实现动态扩容，设计查询效率和存储代价之间关系的目标函数，通过优化目标函数将问题转移为区块选择过程，从而提高查询效率。贾大宇等[14]提出为区块链模型增加查询层，通过外部缓存技术提升数据的二次查询效率。一些学者通过应用数据查询中间件，来实现高效的数据查询能力。ZHOU 等[15]提出设计一种数据查询中间件用于检索区块和交易，同时提供对区块链账本状态模式的分析和聚类，使用户能够对账本数据执行富查询。

多数研究聚焦于构建更加高效的数据索引结构。王俊陆等[16]提出了一种用于多链查询的多级索引结构，从而提升主从多链的查询能力。WANG 等[17]提出一种用于验证查询完整性的框架 vChain，设计两种索引来聚合块内和块间数据，以实现高效的数据查询验证。随后，XU 等[18]对 vChain 进行实现和优化，使该系统能够允许轻量级用户验证来自潜在不可信服务提供者的查询结果。WANG 等[19]针对 vChain 存在的最坏性能问题和公钥管理问题，提出一种改进的 vChain + 系统，能够支持高效的可验证布尔范围查询。XING 等[20]针对基于账户交易轨迹链的查询方法进行优化，提出一种基于子链的账户交易链索引结构，将原始的逐块查找模式转变为子链查找模式，缩短查询路径。PEI 等[21]通过引入默克尔语义字典树索引技术，利用提取的链下数据语义信息构建链上索引结构，实现链上链下数据的实时查询。GUO 等[22]针对食品供应链溯源场景，设计一种优化溯源查询的双层索引结构，该结构通过外部索引定位区块，内部索引定位交易的方式，实现数据的快速查询。LIU 等[23]提出一种区块链应用的存储引擎 FlyDB，通过封装多种存储和查询

算法为键值索引查询提供更高的性能。JAGDISH 等[24]提出为区块链构建语义模型,设置连接索引来提升数据查询效率,解决大数据环境下多宿区块链之间关联查询处理的有效性问题。SUN 等[25]为保证物流信息的快速查询,提出了一种可搜索加密的物流信息区块链查询算法,从数据文件中提取关键字并构建索引,上传到区块链用于后续的查询。

综合分析以上研究,尽管外部数据库在提高查询效率方面起到了一定作用,但每条交易数据在链上和链下都存在特殊的映射关系,需要分别保存在链上和链下以用于数据查询和验证过程。随着时间推移,将导致存储大量的消耗,增加参与方数据维护压力进而导致数据丢失概率变大。建立数据索引可以避免供应链参与方维护外部数据库并提供更高的查询效率。然而,大多数基于区块链的农产品溯源系统无法实现交易中任意多个条件的联合批量查询,导致农产供应链溯源管理系统的查询能力不足。因此,本文提出一种基于区块链的农产品供应链溯源数据多条件查询优化方法。首先,采用 n - Tree 作为一种优化的 Merkle 树结构用于交易结构的重构,使其具有高效的细粒度数据验证能力。其次,根据交易中的属性字段值,构建自适应多条件区块布隆过滤器。该布隆过滤器可以在确定主条件的情况下对多个副条件进行判断,从而快速筛选区块中是否包含满足条件的交易,同时该布隆过滤器以动态方式分配长度,避免存储空间浪费。最后,采用一种快速定位与主条件相关区块的索引结构 TWTN - Heap,该索引结构基于时间和交易数量权重进行排序,以提高目标区块的命中效果,进而保证数据的响应能力。

1 农产品供应链产品数据分析

农产品供应链管理系统中,区块链交易中包含的内容大多为产品数据信息[26-27],且与上传的参与方密切相关,其格式由智能合约中产品数据的定义决定。由于各参与方上传的数据格式不同,其查询方式也有差异。通常情况下,农产品供应链产品信息应该至少包含表 1 中提到的属性字段[28-29]。

在解决区块链多条件查询问题的过程中,首先需要明确的属性为产品所有者身份。由于本研究为面向参与方的查询优化,所以查询的主条件即为参与方的身份。例如,参与方可能进行自身审计工作,同时分析产品在供应链中的流通情况。因此,需要以参与方自身所有的产品为基础进行细粒度的查询,即多条件查询。确定产品所有者的身份具有多重优势。一方面,可以限定数据的查询范围,避免了非必要数据查询带来的性能开销。另一方面,这有助于为参与方构建外部索引,实现区块的快速定位。其次,需要从产品名称、产品 ID、批次号和时间戳这 4 个固有属性分析。产品名称是指一个产品所使用的特定命名,用以标识、区分和传达产品的特征、功能、品牌、定位或其他重要信息,是查询过程的必选条件。产品 ID 是赋予产品的唯一编号,用于在产品和产品之间进行区分,对于持有该产品的参与方来说具有重要意义,也是溯源过程的核心信息。批次号有助于区分产品在大批量生产过程中的不同批次,通常视作必要条件。时间戳记录了产品在供应链中活跃的时间点,充分记录产品在各个阶段的时间戳,可以有效地提高溯源信息可信度,提升消费者对产品的信任程度。综合分析产品供应链普遍的数据存储方式,一般将产品 ID 作为键,剩余的产品数据作为值,以键值对的方式存储到 KV 数据库中。因此,在查询的过程中,以产品 ID 为条件的普通查询只需要通过 KV 数据库即可。对于其他信息,包括产品所有者身份、产品名称、产品批次号等,需要提供额外的条件查询方法。其中,一些非必要字段依据不同参与方的定义各有不同,可以简要地总结为操作人员、产品数量、产品价格等,这些信息具有极大的重复性,通常情况下很少作为独立查询条件进行查询。产品溯源信息包括生产信息、加工信息、销售信息、物流信息等,其本身可能具有额外的字段结构。通常情况下,这些信息多为合约执行过程中通过自动填充方式进行追加,一般不作为多条件查询的条件,而是作为其查询结果出现。

表 1 农产品供应链产品交易属性

Tab. 1 Product transaction attributes of agricultural product supply chain

交易组成	交易属性字段			
交易固有字段	交易 ID	交易类型	通道信息	状态码
	合约名称	函数名称	函数参数	
	创建者身份	背书信息	交易时间戳	
交易数据字段	产品名称	产品 ID	产品批次号	产品原始时间戳
	产品所有者身份	产品品质信息	产地信息	设备信息
	操作人员信息	产品数量/质量	(产品价格)	产品实物信息链接
	产品合格证书	备注信息		
	生产信息	加工信息	销售信息	物流信息

2 产品溯源数据多条件查询优化方法

2.1 问题分析及定义

农产品供应链多采用原生区块链,如以太坊或

联盟链，作为底层架构进行系统开发。由于原生区块链不具备多条件查询这一能力，极大地限制了查询功能的可扩展性。通常，交易在区块链网络中借助 KV 数据库作为存储介质，查询则依赖于交易 ID，即交易的哈希值。以该方式获取的查询结果通常为未经反序列化和解码的二进制数据，需要经过解码和反序列化才能获得原始的交易数据。由于交易数据包含多个数据字段，且数据类型不明确，不采用特殊的优化方式则无法实现交易的多条件查询。综上，可以总结为 3 个问题定义：

定义 1：原生区块链。原生区块链是指独立构建的全新区块链网络，具有自主设计的区块链技术、共识算法、加密机制和网络规则，不依赖于现有区块链平台的改造或二次开发。比特币和以太坊是常见的原生区块链，分别为数字货币和智能合约提供基础设施。

定义 2：多条件查询。多条件查询由一个主条件和多个副条件联合构成，其含义为在满足主条件的情况下，查询满足的多个副条件的数据内容。在农产品供应链管理溯源场景中，主条件一般为产品所有权的归属方，即参与方的身份。多条件查询可以表示为

$$Q = M_c \cap \left(\bigcap_{i \in I} C_i \right) \quad (I = 1, 2, \cdots, n)$$

其中　　$n < N - 1$

式中　Q——条件查询

M_c——条件查询主条件

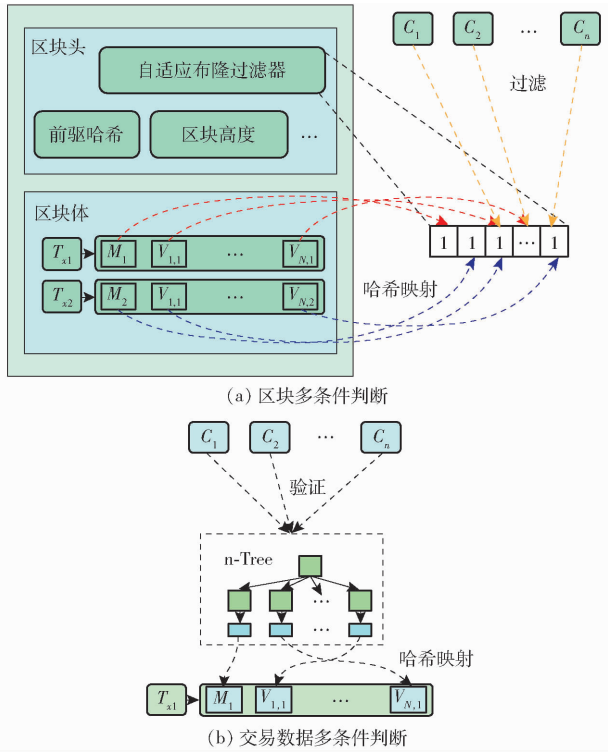
C_i——查询副条件

N——构成交易验证的 n-Tree 叶子节点数

如查询“生产商 1”在“某一时间”生产的“葡萄品种 1”，品质为“高”的所有相关产品。其多条件形式可以表示为一个多元组，即（生产商 1，某一时间，葡萄品种 1，高），其中生产商 1 为主条件。

定义 3：多条件判断。多条件判断是在给定多个条件的情况下，判断某一区块或交易是否具有满足这些条件的内容。本研究中分为两种情况，第 1 种情况为区块的多条件判断，如图 1a 所示，方式为逐一对单条件的存在情况进行过滤。第 2 种情况为交易数据的多条件判断，如图 1b 所示，逐一对单条件在某交易中存在性进行验证。图中 T_{xi} 表示第 i 个交易，$V_{k,j}$ 表示第 k 个属性字段的第 j 个具体属性值，不同交易同一属性位置的属性值可以相同，如 T_{x1} 和 T_{x2} 的第一个非主属性均为 $V_{1,1}$。

针对问题定义 1，为扩展农产品供应链溯源底层原生区块链的查询能力，提出构建一种基于索引的多条件查询优化方法。针对问题定义 2 和 3，提出多条件查询优化方法的具体实现从以下 3 方面出发，分别是交易结构、区块多条件判断和快速定位满足主条件区块。在查询过程中，则按照查询优化方法的反向过程进行查询，依次为定位满足主条件的区块列表，根据区块中的多条件判断方法快速判断区块列表中满足条件的区块，最后通过重构的交易结构验证并查询对应交易，完成查询过程。



(a) 区块多条件判断

(b) 交易数据多条件判断

图 1　多条件判断

Fig. 1　Multi-condition judgment

首先，使用 n-Tree 结构重构交易结构提供查询验证能力，使交易结构能够在满足通用场景的情况下满足多条件查询的要求。一方面，为保证查询效率，需保证该特殊交易结构的构建时间和验证时间足够短；另一方面，考虑到多数情况下参与方执行查询功能的节点多为类轻节点，不能提供足够多的存储资源来同步完整的区块链，需要控制存储单元的消耗。

其次，需要考虑解决区块的多条件判断问题。针对多条件判断问题，提出使用一种存储自适应多条件区块布隆过滤器。该布隆过滤器保存在区块头中，便于快速筛选区块。

最后，需要考虑如何快速定位可能存在条件的相关区块。满足多条件的交易可能存在于多个块中，因此该过程的目的是完整搜集满足主条件的区块，并对满足主条件的区块进行排序。这一排序过程需要依赖于每个块的出块时间，以及区块中包含的满足主条件的交易数量，通过这两个指标计算权重来决定区块号的排列顺序。

2.2 基于 n-Tree 的交易结构设计

区块链交易可以分为 2 部分，一部分为固有字段，另一部分为数据字段。以比特币为例，交易的固有字段包括交易 ID、版本号、交易字节数和时间戳等，数据字段包括输入脚本和输出脚本。在联盟链中以 Hyperledger Fabric 为例，交易的固有字段包括版本号、时间戳、通道 ID、交易 ID、背书身份等，数据字段则主要包含区块链数据的读写集。在实际应用中，本研究将农产品供应链的交易分为交易头和交易体，交易头中存储交易的固有字段，交易体中存储交易的数据字段。交易头能够为存储能力较差的节点提供查询和验证能力，在同步区块时只同步区块头和交易头即可。交易结构的优化主要在数据字段，提出采用 n-Tree 对数据字段进行优化，将数据字段中各属性值组织为 n-Tree 的叶节点，将构建n-Tree 得到的 n-Tree 根存入交易头，将 n-Tree 的节点存储在交易体中，用于后续的条件验证。

n-Tree[30] 是一种优化的 Merkle 树结构，在 n-Tree 中每个非叶节点都有 n 个子节点，在多叶节点情况下选择合适的 n 值能够在性能和存储方面都优于传统的 Merkle 树。由于产品数据属性字段个数对不同参与方来说是变量，因此可以将属性个数表示为 N，每个非叶节点的子节点个数表示为 n，树的高度为 H（从 0 开始计数），H_i^j 表示第 j 层第 i 个节点的哈希值，Hash(·) 表示进行哈希计算，每个非叶节点的值都为其所有子节点哈希值的拼接再计算其哈希值得到，叶节点则存储交易数据字段中每个属性的哈希值，A_i 表示交易数据字段的第 i 个属性，具体的组织结构如图 2 所示。

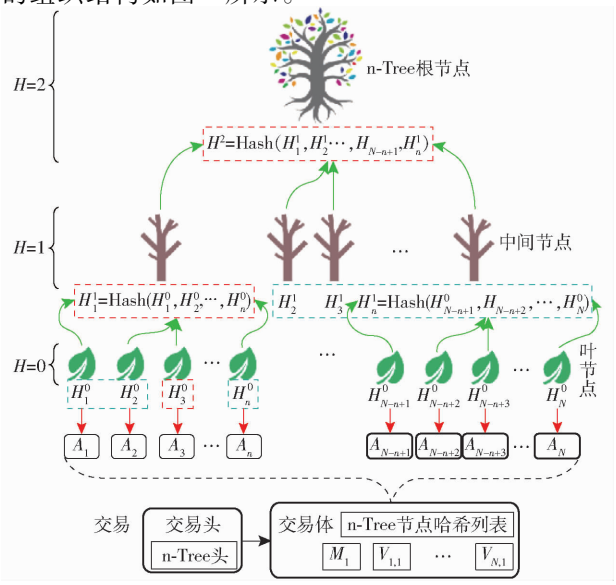


图 2 使用 n-Tree 构建的交易结构

Fig. 2 A transaction structure built using n-Tree

在 Merkle 验证过程当中，验证路径需要的验证节点个数为 $H\times(n-1)$，每一层需要 $n-1$ 个兄弟节点哈希值。以图 2 为例，属性 A_3 的 Merkle 验证路径为蓝色线框部分，因此该验证路径可以表示为矩阵

$$\boldsymbol{P}=\begin{bmatrix} H_2^1 & H_3^1 & H_4^1 & \cdots & H_n^1 \\ H_1^0 & H_2^0 & H_4^0 & \cdots & H_n^0 \end{bmatrix}_{2\times n} \tag{1}$$

构造最优的 n-Tree 交易结构，需要选择合适的 H 和 n。H 由 N 和 n 共同决定，因此有

$$\begin{aligned} &N=n^H \\ \Rightarrow &H=\lceil \log_n N\rceil = c(\log_n N) \\ \Rightarrow &N=n^{c(\log_n N)} \end{aligned} \tag{2}$$

式中 $\lceil\cdot\rceil$——向上取整运算符

然而多数情况下 N 不可开方，为保证 n-Tree 的最大叶子节点数 n^H 不小于 N，则令

$$g(n)=c(\log_n N) \tag{3}$$

其中函数 $c(\cdot)$ 为取上限函数，在 N 确定的情况下 $g(n)$ 是非递增的，因此存在 n_0 使得

$$g(n)=c(\log_n N)\leqslant 2 \quad (\forall n\geqslant n_0) \tag{4}$$

当且仅当 $n=N$ 时 $g(n)=1$，即树高为 1 时 Merkle 树会退化为一个哈希值序列，其验证过程变为线性，因此需要保证 $n_0<N$。综上所述，当 $n\in\{n_0,n_1,\cdots,N-1\}$ 时，均能保证选择 $g(n)=2$ 作为 H 最优值。$H=2$ 时可以最小化中间节点的数量，维持较低的存储开销。综上可得

$$\begin{aligned} &1<H\leqslant 2 \\ \Rightarrow &1<c(\log_n N)\leqslant 2 \\ \Rightarrow &1<\frac{\ln N}{\ln n_0}\leqslant 2 \\ \Rightarrow &n_0<N\leqslant n_0^2 \end{aligned} \tag{5}$$

当 $n>N$ 时需要补充额外节点满足满多叉树条件，增加开销。一般情况下 $n\in\{2,3,\cdots,N-1\}$，理想情况下 $n\in\{2,3,\cdots,c(N^{0.5})\}$。由于 $H=2$ 时存储的中间节点最少，同时能够满足 $n^H=N$ 这一条件，因此在实际过程中将 n 取为 $c(N^{0.5})$。

2.3 自适应多条件区块布隆过滤器

在优化查询的过程中，快速确定某条件是否在区块的某个交易中是优化查询的核心工作。本研究提出的自适应多条件区块布隆过滤器在满足判断元素存在性的同时，使用自适应优化的动态存储空间降低各个属性域的误判率，从而保证提供更加准确的判断结果。为方便描述，下文将其简称为自适应布隆过滤器。

自适应布隆过滤器的长度由交易数据字段的属性值个数 n 和区块最大交易数 N_{max} 决定，经后续分析可以将其取值为区块交易数量平均值的两倍，避

免更高的误判率。自适应布隆过滤器长度可以表示为 $M = n \times N_{\max}$，该长度可以满足各个交易中任意属性均为不同值的情况。为充分利用其空间并为不同值较多的属性提供更大的哈希空间，本文提出通过提升各个属性域在自适应布隆过滤器中的占比达到降低误判率的目的。以动态的方式重新计算实际场景中的属性域占比，天然适用于区块链农产品溯源数据。区块链农产品溯源数据的部分属性域存在大量重复数据字段，例如农产品名称、价格、数量、产地等，缩减上述属性域的长度能够动态地扩充其他属性域长度，提高其他域在自适应布隆过滤器中的占比进而降低各个域的原始误判率。提出采用 BitMap 统计对应属性字段中非重复值的个数 m_{Ai}（A_i表示第 i 个属性字段），BitMap 的大小由区块最大交易数量决定，可以通过并行执行方式来快速统计多个属性字段的非重复值个数。分别计算各属性字段非重复值个数占所有域中非重复属性值总数的比值，即为每个属性字段在布隆过滤器中的长度占比，从而确定每个属性在布隆过滤器中占有空间的长度 L_i，其中 L_i的计算结果多为非整数，因此当 L_i小数部分大于等于 0.5 时对 L_i向上取整，小于 0.5 时对 L_i向下取整，综上 L_i满足

$$L_i = \begin{cases} \left\lceil \dfrac{m_{A_i}}{\sum_{j=1}^{N} m_{A_j}} M \right\rceil & \left(\dfrac{m_{A_i}}{\sum_{j=1}^{N} m_{A_j}} M - \left[\dfrac{m_{A_i}}{\sum_{j=1}^{N} m_{A_j}} M \right] \geqslant 0.5 \right) \\ \left\lfloor \dfrac{m_{A_i}}{\sum_{j=1}^{N} m_{A_j}} M \right\rfloor & \left(\dfrac{m_{A_i}}{\sum_{j=1}^{N} m_{A_j}} M - \left[\dfrac{m_{A_i}}{\sum_{j=1}^{N} m_{A_j}} M \right] < 0.5 \right) \end{cases} \tag{6}$$

式中运算符“⌊⌋”表示向下取整，运算符“[ ]”表示取整。

根据 L_i设置各个属性域的哈希函数，计算各交易数据字段中各属性值对应的哈希值 H_{Ai}，使其映射到自适应布隆过滤器对应的属性域中。为了实现各个属性到属性域的映射，需要将 L_i作为各个域在自适应布隆过滤器中的偏移量，记录在一个 N 维数组 Range Array 中，并将其写入区块头。

构建自适应布隆过滤器的过程中，首先将内存池中待打包交易以遍历的方式获取每条交易中 n-Tree 结构的所有叶节点哈希值，每个叶节点代表对应位置属性域中的值，通过 BitMap 统计各个域中非重复属性的个数，并计算每个域在自适应布隆过滤器中的长度，将各个域的偏移记录在 Range Array 结构当中。随后，将每个叶节点哈希值对小于该属性域长度 L_i且距 L_i最近的素数进行取模运算，得到各个属性在自适应布隆过滤器属性域中的位置，将这些属性哈希值映射到自适应多条件布隆过滤器中。然后，将该布隆过滤器域 Range Array 结构写入区块头中，并准备发布区块。最后，通过共识算法进行区块广播，使区块在区块链网络中达成共识。

农产品供应链中，交易内容多为参与方传输的产品数据，在给出多个查询条件之前需要明确参与方身份这一主条件内容。在条件判断的过程中，首先通过主条件进行筛查，判断区块中是否含有对应主条件的交易，若没有则直接跳过该区块，若存在则通过逐一判断副条件的方式来检查是否存在相关属性，其结构如图 3 所示（为简化图例，哈希函数个数 k 为 1）。

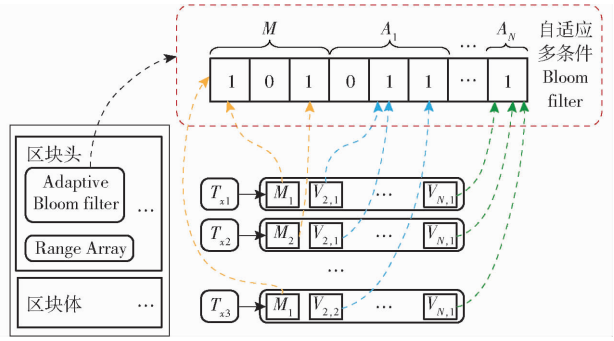


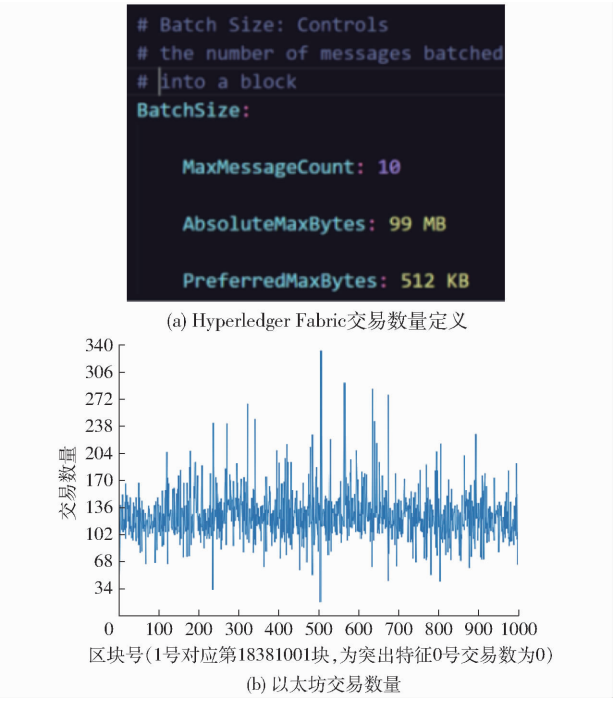
图 3　自适应多条件布隆过滤器

Fig. 3　Adaptive multi-condition Bloom filter

2.4　基于 TWTN-Heap 的区块索引结构

为快速查询目标区块，提出采用基于时间权重与交易数量的小顶堆结构。区块产生时间距第一个区块产生的时间越短权重越小，时间越长则权重越大，即时间段与权重成正比。区块中相关交易数量越多权重越大，交易数量越少权重越小，即交易数量与权重成正比。时间在权重计算的过程中具有重要意义，由于出块时间不定可能导致早期产生的区块权重相对较小，实际场景中仍旧存在多数情况参与方需要查询更久更远的数据，因此需要降低时间对权重计算的影响。采用对数形式处理时间间隔，可以有效降低该影响。通过分析，交易数量在每个区块中的数量稳定在某个特定数值附近。以 Hyperledger Fabric 为例，区块中交易数量由配置文件决定，如图 4a 所示。默认情况下最大交易数量为 10，串行执行的情况下交易数量稳定在 1，并行情况下产生的交易数可以维持在 7 ~ 10 之间，主要由智能合约执行和共识过程决定。为使权重计算公式具有通用性，同时分析了以太坊交易数量的特性。图 4b（2023 年10 月 19 日）展示了遍历以太坊中从第 18381001 号区块开始的 1 000 个区块，记录各个区块的交易数量，该结果表明以太坊区块中的交易

量在某个特定值附近浮动。对该 1 000 个区块的交易数量取平均值可得,每个区块的平均交易量为 126。在处理交易数量的过程中,为降低交易数量对权重的影响将其以比值的方式定义(相关交易数量 N_r除以区块最大交易数量 N_m,如 Hyperledger Fabric 可以取 $N_r/10$,以太坊交易数量的最大值可以定义为 2 倍的交易平均值,因此可以取$\frac{N_r}{126\times2}$)。其中,相关交易数量的计算需要在区块共识之后,通过主条件筛选新区块中的交易进行统计,这一过程依赖于交易结构。由于多条件查询是由参与方决定查询条件的内容,因此不能通过一般条件进行相关交易的判断,相关性判断的条件依赖于主条件,即前文提到的交易所有者身份。



(a) Hyperledger Fabric交易数量定义

(b) 以太坊交易数量

图 4 交易数量分析

Fig. 4 Transaction quantity analysis

因此,可以得到权重计算公式

$$w_i=\frac{N_r}{N_m}\ln(T_i-T_1) \tag{7}$$

式中 T_i——第 i 个区块的出块时间

为了消除权重数值波动跨度较大的问题,使用 z-score 标准化算法处理权重,即

$$w_i'=\frac{w_i-\overline{w}}{\sigma} \tag{8}$$

其中

$$\overline{w}=\frac{1}{i-1}\sum_{j=1}^{i-1}w_j$$

$$\sigma=\sqrt{\frac{1}{i-1}\sum_{j=1}^{i-1}(w_i-\overline{w})^2}$$

分析供应链管理中的查询任务,为满足参与方对自身持有产品进行数据查询的需求,需要构建与其身份相关的交易所在区块的区块号索引。由于索引结构对查询和插入具有更高的依赖性,因此需要选择时间复杂度稳定的索引排序算法。堆排序在最好、最坏以及平均情况下时间复杂度均为 $O(n\mathrm{lb}n)$,可以保证索引构建时间的稳定。计算区块权重并以权重为序加入索引堆中,通过小顶堆排序来实现索引值的排序,结合上述要求该索引结构如图 5 所示。该索引列表中,每个索引元素的结构包括区块号、权重和权重计算函数,排序关键字为每个区块的权重 w_i',其余部分为排序过程中的卫星数据。

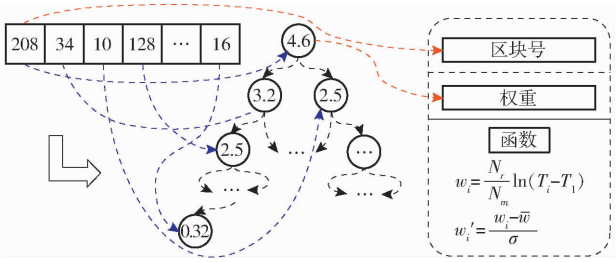


图 5 TWTN-Heap 索引结构

Fig. 5 TWTN-Heap index structure

全节点在加入网络的初始阶段或需要构建以特定主条件为索引的情况下,需要通过初始化算法实现索引结构的初始化。该初始化过程涉及遍历已有区块链结构、区块权重计算、TWTN-Heap 结构索引排序,具体算法为:

算法 A:TWTN-Heap 索引结构初始化

输入:区块链结构 bc,主条件 c_M

输出:TWTN-Heap 结构 s

1. if len(bc) = =0 then
2. return nil
3. end if
4. $n\leftarrow0$, $i\leftarrow0$
5. s←new(TWTN-Heap)
6. while n ! = len(bc) do
7. if c_M不满足 bc[n] then //使用自适应布隆过滤器判断条件
8. continue
9. else
10. item ← TWTN-Heap. element //产生一个新的索引结构元素
11. N_r← 计算 bc[n]的相关交易数量
12. item. weight ← 使用出块时间 T_i与 N_r计算 bc[n]权重
13. item. BlockNumber ← bc[n]. BlockNumber
14. s. append(item)

15.　　$i \leftarrow i+1$

16.　　end if

17. end while

18. return s ← HeapSort_Descend(s) //使用降序的堆排序,即小顶堆

在后续过程中,每共识一个区块就使用权重计算方法计算该区块权重,并生成一个新的索引元素结构,将该元素结构的各个字段赋值后,添加至已有的 TWTN-Heap 中,使用降序堆排序算法实现索引列表的重排。

2.5　多条件交易查询过程

(1)全节点在参与共识的过程中,依据主条件即各自所属参与方的身份信息,使用每个新产生的区块在本地构建 TWTN-Heap 索引结构。

(2)执行查询的用户将条件信息如(M_3、M_6、…)发送给类轻节点,随后类轻节点将条件信息与对应条件的位置索引(3、6、…)发送给全节点,全节点通过条件信息在索引结构中筛选出与条件相关的区块。遍历该区块列表,使用主条件对交易进行 n-Tree 验证进而筛选交易,筛选出的交易根据类轻节点提供的位置索引,获取交易中对应位置索引的 n-Tree 验证路径以及交易的 ID,将其组合成一个三元组,即(交易 ID,索引位置,验证路径),将每个筛选出的交易产生的三元组构成三元组列表,随后向类轻节点返回该结果。

(3)类轻节点收到该三元组列表后,依次遍历该列表,使用副条件内容取哈希值,结合每个三元组提供的 n-Tree 验证路径验证条件的存在性,当某一交易满足所有条件时,则该交易被选定为目标交易。

(4)类轻节点将该交易 ID 发送给全节点,全节点通过其底层的 KV 数据库,获取到该交易 ID 对应的交易内容,将其返回给轻节点。

(5)类轻节点获得序列化的字节数据或编码数据后,将该数据进行解码得到原始结构化数据的字节数据,将字节数据依据其保留的结构化交易数据类型模板进行反序列化,得到原始的结构化数据内容。

(6)当所有筛选出的区块完成以上过程后,则完成了条件查询过程,类轻节点将查询结果批量返回给用户。

该多条件查询过程的时序图如图 6 所示。

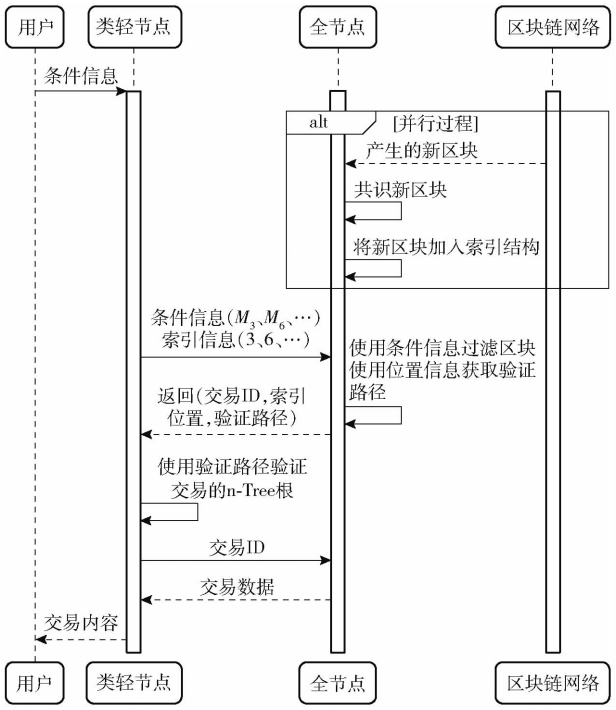


图 6　多条件查询过程时序图

Fig. 6　Temporal diagram of multi-condition query process

3　实验测试与分析

3.1　实验环境

本研究的实验完成于 Intel CORE i5 - 8265U CPU@1.60 GHz 和 8 GB 内存的 PC 机,区块链的模拟以及算法的实现均采用 Golang 语言实现,由于索引构建不完全依赖于区块链的共识过程,其消耗的时间对查询结果影响较小,因此部分实验具有独立性。

3.2　n-Tree 构建性能及存储分析

3.2.1　n-Tree 构建性能

本文提到交易分为固有字段和数据字段,因此在实现的过程中将交易分为交易头和交易体。交易头中包含固有字段的内容以及数据字段的 n-Tree 根,交易体包含 n-Tree 的节点和数据字段。在 n-Tree 构建实验中,对比了传统的 Merkle 树即满二叉树,以及随属性个数变化的 n-Tree 的构建时间,实验结果以微秒为单位,测试了属性字段个数从 10 到 100 范围内的构建性能如图 7a 所示。在大多数情况下,两种树形结构在构建性能上并无明显差别。测试过程中,由于编译器的优化能力较强导致多数情况下构建时间为 0,因此该实验结果中,每个交易字段个数对应的构建时间均为多次实验中出现非零结果的平均值。

在存储性能测试中,以交易数据段中包含 10 个属性为例,与传统的 Merkle 树相比叶节点数量不变,均需要额外复制节点满足的构建条件。对于非叶节点,n-Tree 需要存储根节点哈希值以及其 4 个中间节点哈希值,则共需存储 5 个节点哈希值。传统的 Merkle 树在该情况下为一个 4 层树(根为 0 层),需要存储根节点在内的非叶子节点共 15 个。

相比于传统 Merkle 树,n-Tree 在构建过程中能够节省 73.3% 的空间。图 7b 展示了属性字段个数从 0 至 49 过程中,传统 Merkle 树和 n-Tree 非叶节点数的变化曲线,可以观察到 Merkle 的非叶节点存储消耗程度总体上大于 n-Tree,因此 n-Tree 可以有效优化传统 Merkle 树的存储消耗问题。

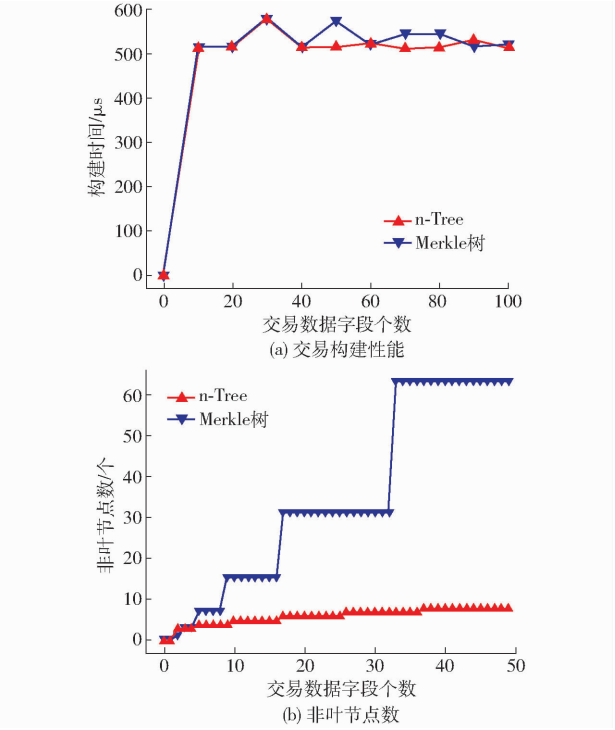


图 7 n-Tree 构建性能分析

Fig. 7 The n-Tree build performance analysis

3.2.2 Merkle 验证性能

虽然 n-Tree 在结构上与传统的 Merkle 树略有区别,但其执行 Merkle 验证的过程是类似的。例如,溯源交易数据中具有 10 个属性字段,则 n-Tree 的子节点数至少为 4。为保证验证路径长度,n-Tree 的高度要始终为 2。当验证交易中某个位置属性值时,首先需要获取与其在同一个子树的另外 3 个属性的哈希值,拼接后通过哈希计算得到该子树父节点的哈希值,再以相同的方式获取该父节点的另外 3 个兄弟节点的哈希值,将该父节点哈希值与这 3 个节点的哈希值拼接后进行哈希计算,得到最终的根哈希值,将其与交易头中的根哈希值进行比较,从而完成 n-Tree 的 Merkle 证明过程。

实验中,n-Tree 的验证路径层数始终为 2,该验证路径的长度与 n-Tree 的高度一致,每一层路径中待拼接的节点数和树的阶数相关,即每一层验证都需要拼接验证路径中的 $n-1$ 个节点。

传统的 Merkle 树的验证路径层数根据叶节点个数进行变化,树高度不确定,因此验证路径较长。其优点是每层拼接的数据量较少,由于其阶数为 2,所以验证路径包含的节点数为 1。但哈希计算过程则相对较多,由于哈希计算数量与树高度密切相关,因此在叶节点数量较多的情况下,其性能略有下降。由于 Merkle 树与 n-Tree 高度为倍数关系,因此在实际场景中,验证过程时间复杂度均为 $O(H)$。

3.3 自适应多条件区块布隆过滤器误判率及存储

3.3.1 误判率及哈希函数数量

假设布隆过滤器长度为 n_0,则布隆过滤器中任意 bit 被置位的概率为 $p_{set}=1/n_0$,反之,该 bit 未被置位的概率为 $p=1-1/n_0$。假设该布隆过滤器有 k 个哈希函数,当插入一个元素时,k 个哈希函数都未将该 bit 置位的概率为 $p^k=\left(1-\frac{1}{n_0}\right)^k$,向布隆过滤器中添加 m 个元素后,仍未将该 bit 置位的概率为 $p^{km}=\left(1-\frac{1}{n_0}\right)^{km}$,反之将该 bit 置位的概率为 $p_{set}^{km}=1-\left(1-\frac{1}{n_0}\right)^{km}$。因此,误判率即 k 个哈希函数分别计算哈希值后映射到布隆过滤器中的值均将对应位置置位的概率,即

$$p_{false}=(p_{set}^{km})^k=\left[1-\left(1-\frac{1}{n_0}\right)^{km}\right]^k \tag{9}$$

因此,通过极限原理,当 $n_0\to\infty$ 时,则

$$p_{false}\approx(1-e^{\frac{-mk}{n_0}})^k \tag{10}$$

由此可以推断,增大布隆过滤器长度或者减少待插入元素个数,均可降低误判率。令

$$f(k)=(1-e^{\frac{-mk}{n_0}})^k \tag{11}$$

对该函数求极限可以得

$$k=\frac{n_0}{m}\ln 2\approx 0.7\frac{n_0}{m} \tag{12}$$

误判率极限为

$$p_{false}=2^{-\frac{n_0}{m}\ln 2}\approx 0.615\,8^{\frac{n_0}{m}} \tag{13}$$

本研究提出的自适应布隆过滤器,需要根据区块中交易数量来确定布隆过滤器长度。以前文以太坊交易数量为例,将 m 取为以太坊中交易数量平均值 126,交易数量最大值为 332,为了降低误判率将 n_0 取为 336 来保证容纳一般情况下的所有交易数据,同时满足字节格式。因此可以得到 $n_0/m\approx 2.7$,$k\approx 2$,$p_{false}\approx 0.27$,在未进行自适应调整下的布隆过滤器误判率在 27% 左右。通过自适应方式以各属性不同值的数量比例来划分布隆过滤器,由于在供应链实际场景下一些属性会存在大量重复,因此可以通过缩减大量重复属性的布隆过滤器空间,来弥补少量重复属性值所面临的较高误判率缺陷,这种方式提高 n_0,从而降低误判率。在极限情况下,即各属性字段的值均为非重

复值，可以得到如图 8 中所示黑色线条表示的误判率回归曲线和函数个数回归曲线，误判率回归值在 25% 左右变化，哈希函数数量回归值在 2.08 左右变化。

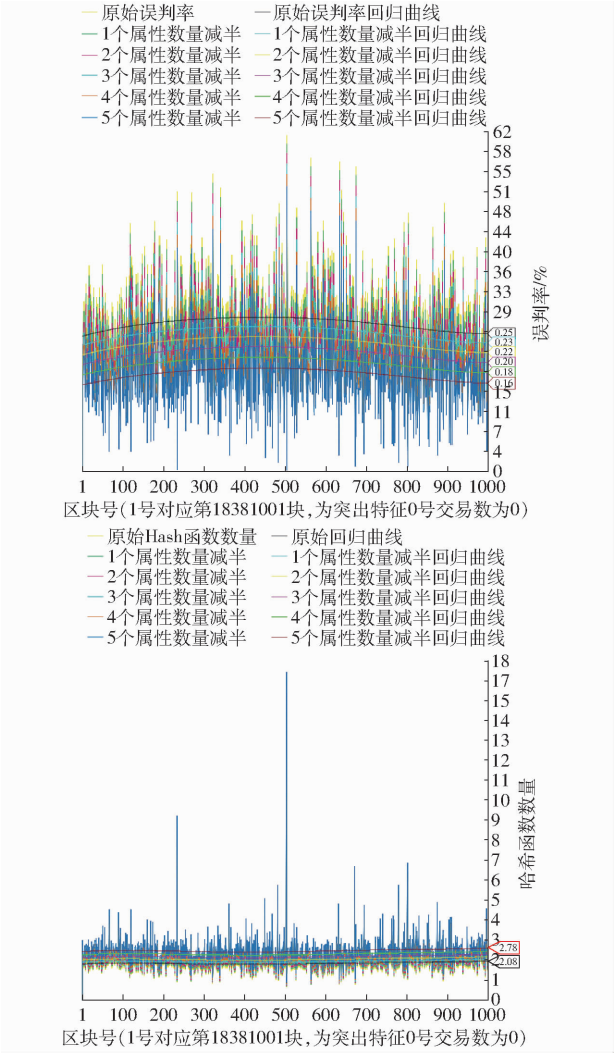


图 8　误判率及哈希函数数量分析

Fig. 8　Analysis of false error rate and number of Hash functions

为了测试自适应布隆过滤器在降低误判率上的提升，实验模拟了供应链中某些字段出现重复值的情况，假设某供应链交易字段包含 10 个属性，分别将其中 1 到 5 个属性字段各自包含的不同属性值个数降低至原来的 1/2，来测试自适应过程带来的误判率的优化和哈希函数数量的变化，为了更加清晰地展示其变化能力，使用回归曲线来表示误判率的稳定状态。由图 8 可知，当属性字段中重复值越多其误判率越低，哈希函数数量越多。在 5 个属性同时降低其不同值数量到原来数量的 1/2 的时候，误判率的回归值可以达到 16% 左右，哈希函数数量的回归值可以达到 2.78 左右。

由于自适应布隆过滤器的生成在区块打包阶段，交易的数据字段是非可读的，因此需要将交易构建阶段保存的属性字段的哈希值作为数据，用于自适应布隆过滤器的映射。

3.3.2　存储分析

前文提到的误判率为 n_0 取特定值的情况，其具有一定的主观性。选取合适的 n_0 值需要根据误判率来确定，对此也可以从误判率角度出发进行反向选择，误判率 p 与布隆过滤器长度满足公式

$$n_0 = -\frac{m\ln p}{(\ln 2)^2} \tag{14}$$

为测试满足低误判率的 n_0 的取值，选取待插入元素个数 $m = 126$ 进行计算。如图 9a 所示，在误判率为 1% 的情况下，m 的取值在 2 005 左右即 251 B，在属性字段个数为 10 的情况下需要消耗 2.51 kB 左右的存储空间。为了能够确定该布隆过滤器在区块中的存储占用比例，实验测试了连续 1 000 个区块的存储消耗比例，如图 9b 所示，通常情况下其区块存储消耗比例在 6% 以下，在实验数据中最大存储消耗比例在 22% 左右，因此在通常情况下存储消耗是可以接受的。

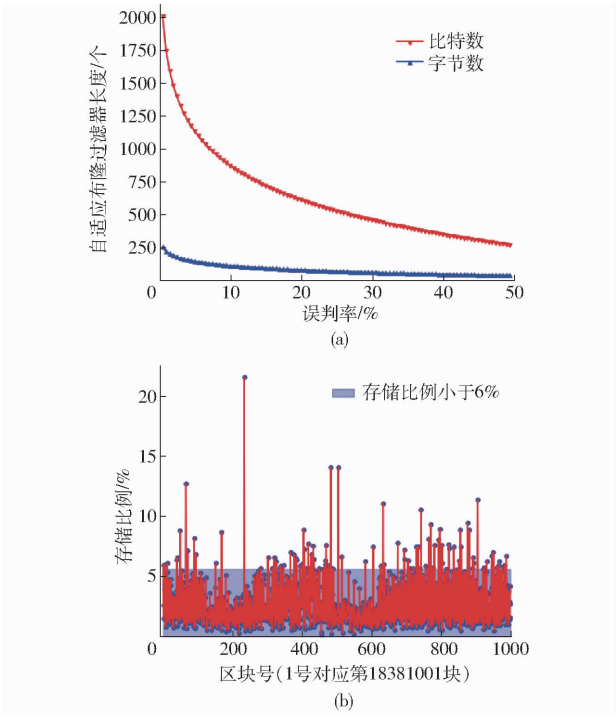


图 9　自适应布隆过滤器存储分析

Fig. 9　Storage analysis

为了确定自适应布隆过滤器的实用性，实验将自适应布隆过滤器与布谷鸟过滤器进行对比。如图 10a、10b 所示，在元素数量从 1 000 到 5 000 的增长过程中，自适应布隆过滤器在存储和元素添加方面具有明显优势。如图 10c 所示，自适应布隆过滤器在元素验证过程相对于布谷鸟过滤器时间消耗较高，但总体差距并不大。在功能方面，布谷鸟过滤器支持删除操作，但本文场景中并不涉及元素删除，同

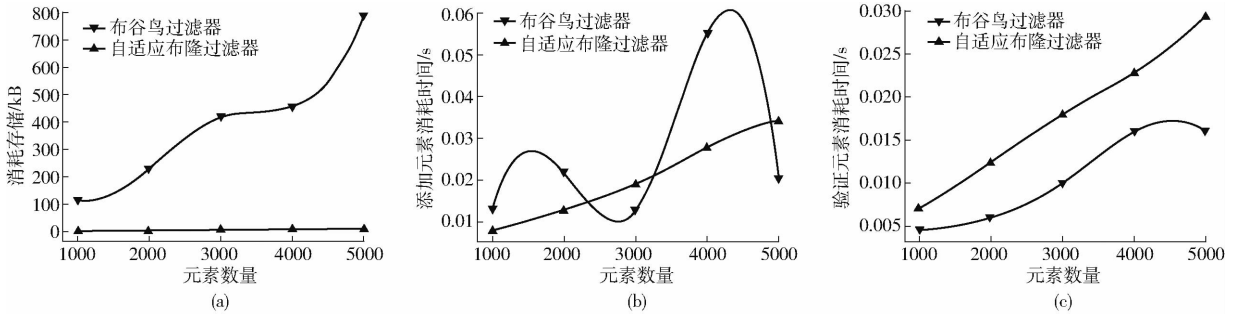


图 10 自适应布隆过滤器与布谷鸟过滤器性能对比

Fig. 10 Comparative analysis of adaptive Bloom filter and Cuckoo filter

时区块链本身是不可篡改的,因此自适应布隆过滤器更加满足实际的场景需求。

3.4 TWTN-Heap 区块索引构建性能分析

索引构建性能分析实验中,以某农产品溯源项目的区块链数据集为例,测试了区块个数从 10^5 到 10^6 个区块的索引构建时间。该索引结构以时间和相关交易数量结合计算的权重为序,对相关交易区块进行排序。如图 11a 所示,区块数为 10^5 的情况下,索引结构构建时间为 0.04 s,当区块数到达 10^6 时,索引结构构建时间在 4.17 s。由于实际场景中多为动态的索引排序过程,即每增加一个区块就更新索引列表,因此实验分别测试了在特定区块数量的情况下,新增 1 个区块后索引排序的时间消耗。如图 11b 所示,当达到 10^6 个区块时,新增一个区块需要消耗 2.18 s 来重新排列索引结构。

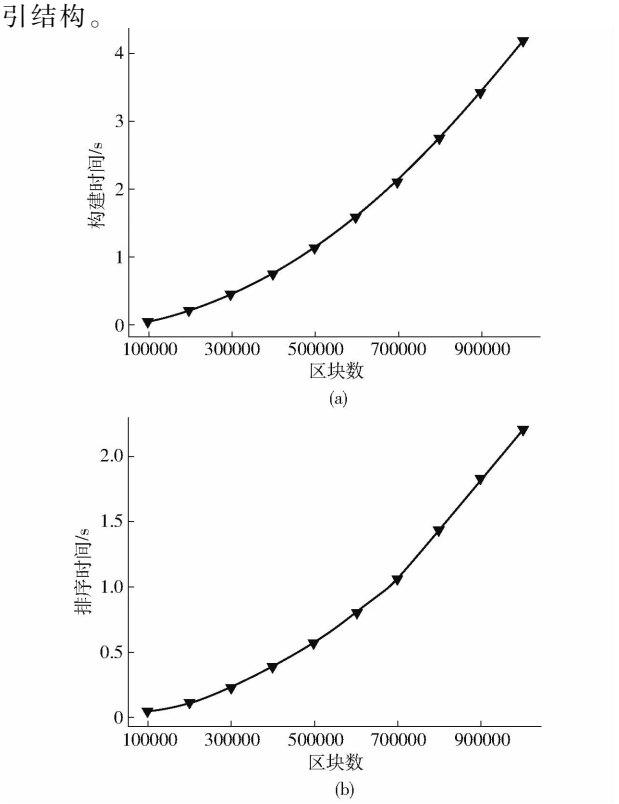


图 11 TWTN-Heap 区块索引性能分析

Fig. 11 TWTN-Heap block index performance analysis

3.5 条件查询性能分析及测试

查询过程涉及 3 个主要步骤:①全节点遍历索引结构,该时间复杂度为 $O(r)$,r 为索引结构中区块个数。②使用副条件筛选出相关区块,该时间复杂度为 $O(fk)$,f 为副条件个数,k 为布隆过滤器中的哈希函数个数,由前文实验可得 k 值大于 3,同时该过程可以得到相关区块中筛选出的区块个数为 r'。③类轻节点使用 n-Tree 路径的验证过程,该时间复杂度为 $O(fHr't)$,H 通常情况下取值为 2,t 为区块中的交易个数。由于网络传输时延非稳定值,因此在计算时间复杂度的过程中将其忽略,从而可以得到交易查询时间复杂度为 $O(rfk+fHr')$。在最坏情况下,即相关区块个数及筛选出的区块个数均为区块链中区块总数 n 时,时间复杂度为 $O(afn)$,其中 $a=2t_{\max}+3$,为方便对比去掉常数项,可以将其近似为 $2t_{\max}$。在不考虑网络时延的情况下,传统的实现多条件查询的查询方式需要遍历整个区块链,再遍历所有交易及其反序列化后的各个数据字段,令返序列化数据及数据判断比较需要消耗的时间为 m,交易中属性字段个数为 p,因此传统的查询方式其时间复杂度为 $O(nt_{\max}fpm)$,为避免不确定的因素 m 的影响,可以将 $t_{\max}pm$ 近似为 $pt_{\max}$。当 $p\leqslant 2$ 时,交易的数据字段属性个数不多于 2 个,对于农产品供应链来说这种交易不具有任何意义,其不满足前文提到的交易数据中的必要字段个数。同时,在实际情况中反序列化问题更加复杂,在非获知数据类型的情况下,几乎不可能做到获取原数据。当 $p>2$ 时,本文提出的优化方式在最坏情况下使查询效率可以达到传统查询方法的常数倍,同时赋予原生区块链数据条件查询能力。

如图 12 所示,实验将本文提出的多条件查询方法(Multi-condition query method, MCQM)与文献[11]的外置数据库(External database, EDB)方法、文献[21]的默克尔语义字典树(MST)方法和原始遍历查询(OT)方法进行对比,使用交易属性个数

为 17 的某农产品供应链数据集，测试了区块数目从 1 000 至5 000 范围内查询时间消耗对比图。随着区块数增多，本文提出的方法与 EDB 方法和 MST 方法均保持较为稳定的效果。由于 EDB 方法使用外置数据库，因此本文采用 MySQL 数据库对该方法进行模拟测试，本文提出的 MCQM 方法与 EDB 方法相比，在查询时间消耗上相差不大，均在 15 ms 左右。MST 方法的查询时间消耗维持在 30 ms 左右，本文提出的方法与之相比在查询时间上消耗更少，整体平均优化率为 60.9%。与 OT 方法相比，当区块数为 5 000 时 OT 方法的查询时间为 218.2 ms，MCQM 方法的查询时间为21.5 ms，查询优化率达到 90.1%，整体的平均优化率为 87.7%，可以有效地提升多条件查询的效率。

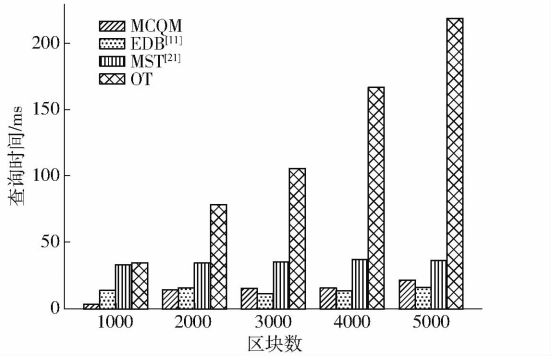


图 12　不同区块数下查询时间对比

Fig. 12　Comparison of query time under different numbers of blocks

4　结论

(1)针对农产品供应链管理中交易数据的多条件查询需求，为参与方提出了一种基于索引的解决方案。该方案从交易本身的构建、区块的快速筛选以及相关区块索引列表外部索引的快速构建 3 方面出发，构建满足多条件查询的索引结构。首先，使用 n-Tree 构建更短且存储消耗更少的 Merkle 验证路径，同时将交易划分为交易头和交易体，通过同步交易头方式进而保证类轻节点的查询能力。其次，使用自适应多条件区块布隆过滤器构建区块筛选器，从而跳过遍历区块内部数据的过程。最后，以参与方身份为主条件构建相关区块的 TWTN-Heap 索引结构，从而进一步缩小条件查询范围，以时间间隔和相关交易数量为参数计算区块权重，保证更高的区块命中率，实现交易查询结果的快速响应。

(2)实验结果表明，该索引结构的各个部分从构建时间到存储消耗上都具有良好的表现。同时，在查询效率的理论分析中，在最坏情况下，即索引结构包含每个区块的权重计算结果时，该索引结构仍然能够保证其查询效率是传统遍历查询方式的常数倍。查询时间测试中，本文提出的方法查询时间消耗维持在 15 ms 左右，与默克尔语义字典树相比查询效率平均提高 60.9%，与原始遍历方法相比查询效率平均提高 87.7%。

参　考　文　献

[1] KRITHIKA L. Survey on the applications of blockchain in agriculture[J]. Agriculture, 2022, 12(9): 1333.

[2] 孙传恒，于华竟，罗娜，等. 基于智能合约的果蔬区块链溯源数据存储方法研究[J]. 农业机械学报, 2022, 53(8): 361 – 370.
SUN Chuanheng, YU Huajing, LUO Na, et al. Blockchain traceability data storage method of fruit and vegetable foods supply chain based on smart contract[J]. Transactions of the Chinese Society for Agricultural Machinery, 2022, 53(8): 361 – 370. (in Chinese)

[3] KAMATH R. Food traceability on blockchain: Walmart's pork and mango pilots with IBM[J]. The Journal of the British Blockchain Association, 2018, 1(1): 47 – 53.

[4] TAN Yiheng, HUANG Xiying, LI Wei. Does blockchain-based traceability system guarantee information authenticity? An evolutionary game approach[J]. International Journal of Production Economics, 2023, 264: 108974.

[5] 于华竟，徐大明，罗娜，等. 杂粮供应链区块链多链追溯监管模型设计[J]. 农业工程学报, 2021, 37(20): 323 – 332.
YU Huajing, XU Daming, LUO Na, et al. Design of the blockchain multi-chain traceability supervision model for coarse cereal supply chain[J]. Transactions of the CSAE, 2021, 37(20): 323 – 332. (in Chinese)

[6] 李修华，罗潜，杨信廷，等. 面向小麦区块链追溯系统的分级监管模型设计与实现[J]. 农业机械学报, 2023, 54(3): 363 – 371.
LI Xiuhua, LUO Qian, YANG Xinting, et al. Design and implementation of blockchain hierarchical supervision model for wheat supply chain[J]. Transactions of the Chinese Society for Agricultural Machinery, 2023, 54(3): 363 – 371. (in Chinese)

[7] ZHOU Enyuan, HONG Zicong, XIAO Yang, et al. MSTDB: a hybrid storage-empowered scalable semantic blockchain database[J]. IEEE Transactions on Knowledge and Data Engineering, 2022, 35(8): 8228 – 8244.

[8] 董思含，信俊昌，郝琨，等. 多区块链环境下的连接查询优化算法[J]. 浙江大学学报(工学版), 2022, 56(2): 313 – 321.
DONG Sihan, XIN Junchang, HAO Kun, et al. A join query optimization algorithm in multi-blockchain environment[J]. Journal of Zhejiang University(Engineering Science), 2022, 56 (2): 313 – 321 . (in Chinese)

[9] 潘恒，钱海洋，姚中原，等. 典型区块链存储与查询技术综述[J]. 郑州大学学报(理学版), 2022, 54(6): 34 – 50.
PAN Heng, QIAN Haiyang, YAO Zhongyuan, et al. A survey of typical blockchain storage and query technologies[J]. Journal of Zhengzhou University(Natural Science Edition), 2022, 54(6): 34 – 50. (in Chinese)

[10] 焦通,申德荣,聂铁铮,等. 区块数据库[J]. 软件学报, 2019, 30(9): 2671 - 2685.
JIAO Tong, SHEN Derong, NIE Tiezheng, et al. BlockchainDB: querable and immutable database[J]. Journal of Software, 2019, 30(9): 2671 - 2685. (in Chinese)
[11] YANG Xinting, LI Mengqi, YU Huajing, et al. A trusted blockchain-based traceability system for fruit and vegetable agricultural products[J]. IEEE Access, 2021, 9: 36282 - 36293.
[12] 杨信廷,王明亭,徐大明,等. 基于区块链的农产品追溯系统信息存储模型与查询方法[J]. 农业工程学报, 2019, 35(22): 323 - 330.
YANG Xinting, WANG Mingting, XU Daming, et al. Data storage and query method of agricultural products traceability information based on blockchain[J]. Transactions of the CSAE, 2019, 35(22): 323 - 330. (in Chinese)
[13] XU Mengtian, FENG Guori, REN Yanli, et al. On cloud storage optimization of blockchain with a clustering-based genetic algorithm[J]. IEEE Internet of Things Journal, 2020, 7(9):8547 - 8558.
[14] 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.
JIA Dayu, XIN Junchang, WANG Zhiqiong, et al. Efficient query model for storage capacity scalable blockchain system[J]. Journal of Software, 2019, 30(9): 2655 - 2670. (in Chinese)
[15] ZHOU Ence, SUN Haoli, PI Bingfeng, et al. Ledgerdata refiner: a powerful ledger data query platform for hyperledger fabric [C]// 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS). IEEE, 2019: 433 - 440.
[16] 王俊陆,张桂月,杜立宽,等. 面向主从区块链的多级索引构建方法[J/OL]. 计算机研究与发展, 1 - 10[2023 - 11 - 09] http://kns.cnki.net/kcms/detail/11.1777.TP.20230512.1617.010.html.
WANG Junlu, ZHANG Guiyue, DU Likuan, et al. A multi-level index construction method for master-slave blockchain[J/OL]. Journal of Computer Research and Development, 1 - 10[2023 - 11 - 09] http://kns.cnki.net/kcms/detail/11.1777.TP.20230512.1617.010.html. (in Chinese)
[17] WANG Haixin, XU Cheng, ZHANG Ce, et al. vChain: a blockchain system ensuring query integrity[C]// Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, 2020: 2693 - 2696.
[18] XU Cheng, ZHANG Ce, XU Jianliang. vChain: enabling verifiable boolean range queries over blockchain databases[C]// Proceedings of the 2019 International Conference on Management of Data, 2019: 141 - 158.
[19] WANG Haixin, XU Cheng, ZHANG Ce, et al. vChain +: optimizing verifiable blockchain boolean range queries[C]//2022 IEEE 38th International Conference on Data Engineering (ICDE). IEEE, 2022: 1927 - 1940.
[20] XING Xiaogang, CHEN Yuling, LI Tao, et al. A blockchain index structure based on subchain query[J]. Journal of Cloud Computing, 2021, 10: 1 - 11.
[21] PEI Qingqi, ZHOU Enyuan, XIAO Yang, et al. An efficient query scheme for hybrid storage blockchains based on Merkle semantic trie[C]//2020 International Symposium on Reliable Distributed Systems (SRDS). IEEE, 2020: 51 - 60.
[22] GUO Chaopeng, LIU Yiming, NA Meiyu, et al. Dual-layer index for efficient traceability query of food supply chain based on blockchain[J]. Foods, 2023, 12(11): 2267.
[23] LIU Yang, XIE Zehui, XIAO Nong, et al. FlyDB: query optimization of blockchain system based on hybrid storage architecture[C]//2023 4th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT). IEEE, 2023: 233 - 237.
[24] JAGDISH M, ANAND N, GAURAV K, et al. Multihoming big data network using blockchain-based query optimization scheme [J/OL]. Wireless Communications and Mobile Computing, 1 - 10[2022 - 08 - 21] https://www.hindawi.com/journals/wcmc/2022/7768169/.
[25] SUN Yibo, LI Xiaofang, LV Furu, et al. Research on logistics information blockchain data query algorithm based on searchable encryption[J]. IEEE Access, 2021, 9: 20968 - 20976.
[26] 殷文杰,都牧,贾凯文,等. 基于区块链的野生农产品溯源系统[J]. 电子科技大学学报(社科版), 2022, 24(5): 88 - 98.
YIN Wenjie, DU Mu, JIA Kaiwen, et al. A blockchain-based traceability system for wild agricultural products[J]. Journal of University of Electronic Science and Technology of China(Social Sciences Edition), 2022, 24(5): 88 - 98. (in Chinese)
[27] HADER M, TCHOFFA D, El MHAMEDI A, et al. Applying integrated blockchain and big data technologies to improve supply chain traceability and information sharing in the textile sector[J]. Journal of Industrial Information Integration, 2022, 28: 100345.
[28] 孙传恒,万宇平,罗娜,等. 面向追溯主体的果蔬全供应链区块链多链模型研究[J]. 农业机械学报, 2023, 54(4): 416 - 427.
SUN Chuanheng, WAN Yuping, LUO Na, et al. Blockchain multi-chain model of fruit and vegetable supply chain for traceability subjects[J]. Transactions of the Chinese Society for Agricultural Machinery, 2023, 54(4): 416 - 427. (in Chinese)
[29] 陈明,孙浩,邹一波,等. 基于区块链的河豚供应链可信溯源优化研究[J]. 农业机械学报, 2022, 53(9): 295 - 304.
CHEN Ming, SUN Hao, ZOU Yibo, et al. Trusted traceability optimization of pufferfish supply chain based on blockchain[J]. Transactions of the Chinese Society for Agricultural Machinery, 2022, 53(9): 295 - 304. (in Chinese)
[30] RIPOSO J. On improving the Merkle trees: the n-Trees[M]// Some Fundamentals of Mathematics of Blockchain. Cham: Springer Nature Switzerland, 2023: 89 - 104.