针对区块链应用的查询优化模型

王泓机[1,2],戴炳荣[2],李 超[2],张绍华[2]
1.复旦大学 计算机科学技术学院,上海 201203
2.上海计算机软件技术开发中心,上海 201112

摘 要:区块链应用有去中心化、可追溯、不可篡改等优点,但现有的区块链系统难以满足在大数据量下数据的查询、访问需求。针对以上问题,结合ETL流程与区块链自身特点,提出一种区块链应用查询优化流程模型ETLVQ(Extract-Transform-Load-Validation-Query)。模型分为三个阶段:在第一阶段将区块链应用中异构数据源的数据进行抽取、转换并加载到查询优化数据仓库中;在验证阶段对加载后数据进行一致性校验;在查询阶段对外提供查询访问服务。该模型在保证区块链数据安全性的前提下,显著提升了区块链应用的查询访问效率与并发能力。

关键词:区块链;智能合约;ETL流程;查询优化
文献标志码:A **中图分类号:**TP311 doi:10.3778/j.issn.1002-8331.1905-0361

王泓机,戴炳荣,李超,等.针对区块链应用的查询优化模型.计算机工程与应用,2019,55(22):34-39.
WANG Hongji, DAI Bingrong, LI Chao, et al. Query optimization model for blockchain applications. Computer Engineering and Applications, 2019, 55(22):34-39.

Query Optimization Model for Blockchain Applications

WANG Hongji[1,2], DAI Bingrong[2], LI Chao[2], ZHANG Shaohua[2]
1.School of Computer Science, Fudan University, Shanghai 201203, China
2.Shanghai Development Center of Computer Software Technology, Shanghai 201112, China

Abstract: Blockchain applications have the advantages of decentralization, traceability, and non-tampering. However, the existing blockchain system is difficult to meet the query and access requirements of data under large data volume. Aiming at the above problems, combined with the characteristics of ETL process and blockchain, a blockchain application query optimization process model ETLVQ(Extract-Transform-Load-Validation-Query) is proposed. The model is divided into three phases. In the first phase, the data of the heterogeneous data source in the blockchain application are extracted, converted and loaded into the query optimization data warehouse. During the verification phase, the loaded data are checked for consistency, and the query access service is provided externally during the query phase. Under the premise of ensuring the security of blockchain data, this model significantly improves the query access efficiency and concurrency of blockchain applications.

Key words: blockchain; smart contract; ETL process; query optimization

1 引言

区块链本质上是一个去中心化的分布式账本,由该区块链网络中的各个节点共同维护[1]。每一个节点上都具有一份同样的账本数据,该账本由一个个的区块组成。每一个区块中通常包含多笔交易,在这个网络中各个挖矿节点不断通过计算而获取记账权和记账奖励[2]。区块链由其所具有的去中心化、不可篡改等特性受到广泛关注,并被应用到金融、物联网、人工智能等多个领域[3-5]。

基金项目:上海区块链产业研究与评估验证平台研发应用项目(No.18DZ1112101);上海市软件技术创新服务平台项目(No.17DZ2292100)。

作者简介:王泓机(1995—),男,硕士研究生,研究领域为区块链,E-mail:wang.hongji@foxmail.com;戴炳荣(1984—),男,博士,高级工程师,研究领域为大数据、区块链、云计算;李超(1989—),男,工程师,研究领域为大数据、区块链;张绍华(1974—),男,博士后,副研究员,研究领域为区块链、数据治理。

收稿日期:2019-05-24 **修回日期:**2019-09-30 **文章编号:**1002-8331(2019)22-0034-06
CNKI**网络出版:**2019-10-10, http://kns.cnki.net/kcms/detail/11.2127.TP.20191010.1355.014.html

基于现有的区块链平台，人们构筑了大量的区块链实际应用，如支持可信溯源查询的供应链系统、数字版权保护平台、智能电网系统等。部分应用并不仅仅将数据存储到单一区块链网络中，如ethDrive[6]、Blockchain-IoT[7]等将摘要数据存储到区块链中，文件数据存储到分布式文件系统中以最优化存储性能，而ECISB（Electricity Consumption Information Storage Blockchain）[8]采用双链存储电网数据，在保护隐私的同时采集了用电信息。

然而随着应用范围的扩大，使用场景的增多，区块链应用在访问、查询方面暴露出了短板[9]。在设计之初为了提升区块的同步速度，许多区块链平台选用levelDB作为区块数据的存储数据库[10]。该数据库是一个基于LSM（Log Structured Merge）树的key-value数据库，由于写入时只需要在memTable中追加数据，该数据库的写入性能远远高于其读取性能。在区块链保持最新的同步状态时，由于节点数量众多，达成共识速度慢，其写入性能大量过剩。同时区块链平台在levelDB上并没有封装丰富的查询接口[11]，仅仅支持根据区块或交易的哈希值查询对应内容，无法支持复杂的如关系查询、聚合查询等，更进一步限制了区块链应用对数据的访问和分析操作。

针对查询层面存在的问题，国内外学者从不同角度提出了可能的解决思路。

EthernityDB[12]通过智能合约的编写模拟了数据库的部分查询功能，但其支持的查询由于受到智能合约编写逻辑的制约，仍然极为有限，且数据的读取仍在区块链上完成。文献[13]提出一种高性能、易开发的区块链及可分叉应用的存储引擎ForkBase，该引擎采用全新的索引结构SIRI，并提供了数据版本控制、篡改记录、分叉语义等功能。

王千阁等人[10]受Myisam启发，提出了在levelDB中添加额外索引的方式来优化查询效率，该方法会面临写入功能退化，查询性能受区块链系统瓶颈限制等问题。

Li等人[9]提出了针对以太坊的附加查询层etherQL，该方法将区块链数据拷贝到链外数据库并进行查询，Pratama等[14]在此基础上增强和扩展了etherQL的查询功能。林妤儒提出了hyperQL[15]，为HyperLedger Fabric添加了外接查询层。但这些外接查询层仅考虑了某一特定区块链的数据查询，且选择性忽略了较难处理的智能合约中的用户数据。

针对当前区块链应用存在的一些问题，受传统数据仓库中的ETL（Extract-Transform-Load）流程的启发，提出了一种查询优化流程模型（Extract-Transform-Load-Validation-Query，ETLVQ）。

本文的主要贡献如下：

（1）提出了兼容单一/多数据源的区块链应用查询优化模型，设计了区块链数据提取、加载、转换元模型。该优化模型能够在提升查询效率的基础上，保证写入功能不受影响，且能够较好地处理智能合约中的用户数据，对查询密集型应用具有很大的价值。

（2）对装载到数据仓库中的区块链数据设计了一致性验证模型，给出了区块、交易的验证算法。该验证模型弥补了附加查询层可能带来的数据不一致问题，在数据出现不一致情况（如被外部篡改等）时能够及时发现和响应，提升数据安全性。

（3）在一种具有代表性的区块链应用上进行性能测试，实验表明，本文设计的查询优化模型能较好地解决应用在大量访问、查询的性能瓶颈，且能够保证数据的安全有效性。

2 ETL技术

ETL是指数据的提取、转换、加载的一个处理流程[16]，通常用于将多个系统中的多个异构源数据库中的数据提取后经过处理模块清洗、整理后加载到单一的目标数据库中，即数据仓库[17]。进一步对数据的聚合、分析、查询将直接对数据仓库进行。

ETL的提取过程主要解决异构数据源中不同的数据组织方式的问题，以尽量将提取出的数据统一化[17]。

ETL的转换过程主要解决数据质量问题，结合一定的转换规则对数据进行清洗、过滤、去重、删除、合并等。

ETL从装载过程上分类可分为全量ETL和增量ETL[18]。增量ETL在性能和效率上更优，但设计和实现上较为复杂，通常有如下几种思路：

（1）基于Snapshot；

（2）基于数据库中的日志；

（3）基于数据库中的时间戳。

整个ETL过程通常由元数据驱动[19]，因此在进行该流程前需要对源数据库结构、目标数据库结构、源表结构、源字段描述、目标表结构、目标字段描述、转换与映射规则等信息在元模型中进行定义。

本查询优化模型在结合ETL技术并做适当改进的基础上，能够有效解决当前查询密集型区块链应用中的以下问题：

（1）查询效率低：采用本文提出的ETLVQ之后，数据的存储结构不再采用LSM树，采用对查询更加友好的数据组织方式，从而提升查询效率。

（2）查询代价高：为了保证数据的安全性、一致性，区块链系统在每次查询时需要前向遍历区块。采用ETLVQ之后，数据通过额外的验证模块保证安全，与查询部分充分解耦，不必再进行前向遍历。

3 区块链应用查询优化流程模型

在区块链语境下，首先需要对模型中的以下概念进行定义：

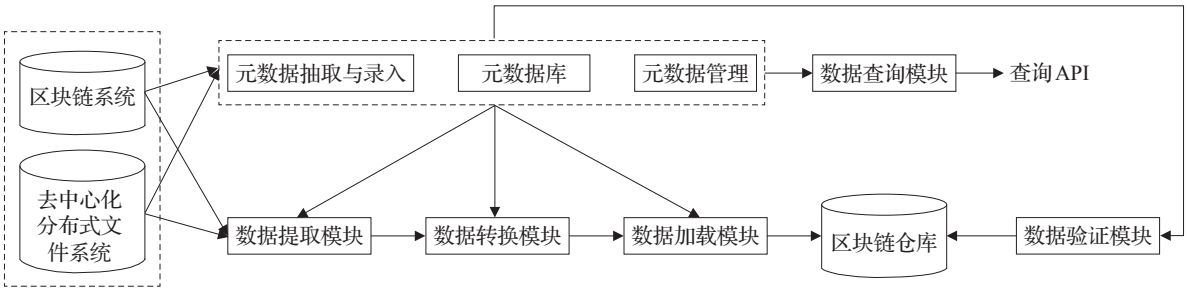

图1　查询优化模型流程设计

(1)数据源(Data Source):传统的ETL流程中,数据源通常指异构的关系型数据库,如MySQL、SQL Server等,但有时也指如XML、文本文件等数据来源[19]。在ETLVQ模型中,数据源指与区块链应用相关的数据来源,如区块链系统和去中心化分布式文件系统。

(2)区块链仓库(Blockchain Warehouse):指将区块链应用中的数据经过提取、转换和加载后的目标存储区域,为了提升查询性能,通常选用对查询支持良好的关系型数据库或NoSQL数据库。

由于区块链应用本身底层存储的限制,其查询性能不甚理想,将其数据加载到区块链仓库中的方式能较为有效地解决这个问题。这个查询优化流程的模型分为如下几个模块:

(1)数据提取模块:该模块负责从异构的区块链数据源中提取关键数据,包括区块链系统中的区块信息、交易信息、智能合约信息以及去中心化分布式文件系统中的文件信息等。

(2)数据转换模块:该模块负责对前一模块中提取出的原始数据进行清洗和转换,包括格式处理、冗余处理、合约解析、数据解码等过程。

(3)数据加载模块:根据目标区块链仓库的不同参数调整连接方式、数据组织方式等行为,并将数据加载到仓库中。

(4)数据验证模块:该模块对数据仓库中的区块链数据进行一致性验证,以保证同步无误且数据安全。

(5)数据查询模块:该模型主要是为了优化数据查询而设计的,因此设计有丰富的基础查询接口以满足对一般区块链应用的访问、查询需求,同时也能够根据应用的不同特点增加查询功能。

以上的整个流程模型都是由元数据驱动的,即在每一个流程模块中都由元数据进行指导。元数据描述了区块链数据、区块链仓库的结构、特征、构造,定义了数据清洗、转换的映射规则和一致性校验规则,贯穿整个ETLVQ过程。

对元数据的建模本文采用了通用仓库模型(Common Warehouse Model,CWM)[20],该模型是由OMG(Object Management Group)提出的一个元模型框架,描述了元数据应该如何组织、管理和交换。

如图1所示,通过元数据库对该查询优化流程模型中所涉及到的元数据进行全生命周期管理,在将数据从区块链数据源经提取、转换和加载后,通过数据验证模块保证安全性,使用数据查询模块对外部应用提供查询API。

3.1　数据提取模块

数据的提取过程关注如何从数据源中提取含有目标数据的原始数据的过程。数据提取又称为数据抽取,通常分为全量抽取和增量抽取。在区块链系统中通常会含有某一区块或交易的生成时间戳,新生成的合法区块的时间戳一定晚于已有的最新合法区块。因此在数据提取模块中可以直接利用区块数据的这一特性,使用基于时间戳的增量抽取方式。同时由于区块链已有的数据链条已完全确定,具有无法篡改性,因此在数据抽取中也不必考虑对已抽取数据的更新、删除等情况,很大程度上简化了模块设计。

传统ETL流程中的数据通常来源于关系型数据库,因此在对抽取过程建立元模型时通常使用CWM中资源层的关系型包。关系型包描述了通过原生的关系型接口如RDBMS、ODBS或JDBC所能访问到的所有数据。

由于区块链系统中的数据通常不采用关系型数据库存储,而更多地体现为一系列集合的形式,因此采用了CWM中资源层的记录包(Record Package)来建立元模型。该包下的UML图见图2。

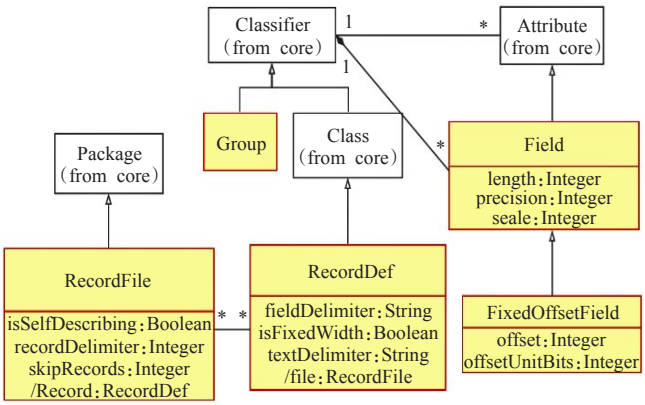

图2　CWM资源层记录包结构

图中RecordDef表示一组有序的Field的集合,这组集合表达了某一条(或一组)Record的结构。Field是在

RecordDef中的基本信息载体,是对数据域的抽象。Group是在一条记录内对多个Field一种结构化的描述。图3展示了某一区块链系统中的数据记录的元模型实例。

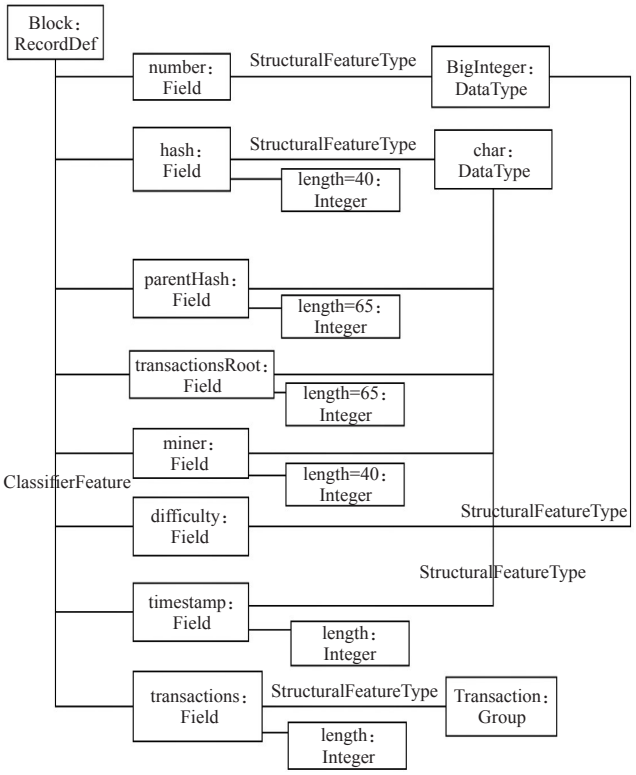

图3 区块链系统数据记录元模型实例

在该实例中一个区块的存储结构表达为一条RecordDef,其中包含多个Field,如区块号number,区块哈希值hash,区块的父哈希值parentHash,区块中包含交易形成的交易Merkle树的根哈希值等。其中transactions域中包含多个Transaction,每一个Transaction的具体结构在此省略。

用户在使用查询优化流程模型前首先需要预定义好如图3所示的提取元模型实例,在提取过程中则能识别和处理来自于区块链系统的异构数据,为下一步的数据转换做准备。

3.2 数据转换模块

数据转换模块关注如何从原始数据中清洗、提取和加工出数据仓库所关心的高质量数据。这个阶段的元数据模型在CWM的转换包的核心类基础上构建。该包中的部分核心类和它们之间的关系如图4。其中TransformationActivity表示某个转换行为,该行为包含一组转换步骤(TransformationStep),每个转换步骤中包含一个或多个转换任务(TransformationTask)。一个转换任务是转换的执行逻辑单元,包含多条映射规则(TransactionMap)。ClassifierMap与ClassifierFeatureMap(CFMap)分别表示RecordDef到Recordef、RecordDef与Field之间的映射规则。

根据上述数据转换元模型给出转换实例,详见表1。

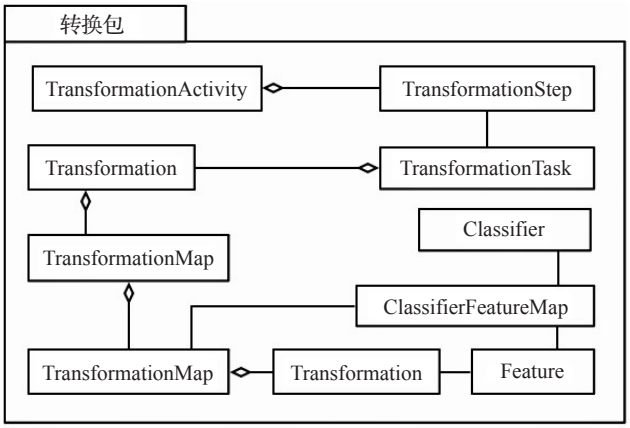

图4 CWM转换包结构

表1 数据转换映射规则集

类别	映射规则
ClassifierMap	Block:RecordDef → Block:Table
	transactions:Group → Transaction:Table
FeatureMap	timestamp:Field(String) → timestamp:Field(Date)
	miner:Field(String) → miner:Field(Address)
	gasPrice:Field(String) → gasPrice:Field(BigInteger)
CFMap	smartContractData:Field(String) → ApplicationData:Table
	gasPrice:Field Union gasUsed:Field Aggregation → TransactionFee:Field

表1中定义的规则包括从记录到表之间的转换,从某个数据域到某一列的转换,以及单一列的数据拆分为某张表的数据等转换,如将智能合约中的输入域数据(inputData)转换为区块链仓库中智能合约部分的用户应用数据表。在映射规则建立后,通过如下算法进行转换。

算法1 数据转换算法

输入:待转换数据集合 *SourceSet*,数据转换规则集 *MapSet*。

输出:转换后数据集合 *TargetSet*。

```
1. for every map in MapSet do
2. source=GetSourceFrom(SourceSet,map)
3.   if map instanceOf ClassifierFeatureMap
4.     if source instanceOf Classifier /*多数据域聚合*/
5. target=Aggregate(map,source)
6. TargetSet.put(target)
7.     end if
8.   if source instanceOf Feature /*单一数据域拆分*/
9. TargetSet.put(map.function(source))
10. end if
11.  if map instanceOf FeatureMap
12. target=map.function(source) /*调用 map 中定义的处理函数对源数据进行处理*/
13. TargetSet.put(target)
14.   end if
15. end for
```

3.3 数据装载模块

数据装载模块关注如何将转换后的数据正确高效地加载到区块链仓库中。一个区块中通常带有四部分数据:区块自身描述数据、交易数据、用户状态数据和智能合约数据。区块自身描述数据又叫区块元数据,包含区块本身的一些状态信息,如时间戳、区块号、区块哈希值等。交易数据即挖出该区块的矿工所打包记录的链上交易,交易中包含了交易的发起方、接收方等内容。用户状态数据指由用户状态树(State Trie)的根哈希值获得的链上用户的具体状态数据,如余额、地址等内容。智能合约数据指用户在与链上智能合约互动时产生的调用信息。这四部分的内容在装载过程中以对象的方式存在,通过使用不同的持久层连接驱动(如JDBC、MongoDriver等)将对象转化为表内容或文档的形式存入区块链仓库中。

3.4 数据验证模块

在将数据装载到区块链仓库后,需要对数据进行两方面的校验:

(1)区块链仓库与区块链系统的同步进度一致性验证。

(2)区块数据是否有效,即区块数据是否满足其原本区块链系统中的所有约束。

对于上述第一点的数据进度验证,在模型中设置装载进度日志,该日志中保存当前已装载区块号及时间戳。在仓库遭遇故障停机恢复后,能够及时从上次断点位置继续装载数据。

下面详细叙述如何进行区块数据有效性验证。首先对区块数据建立一系列检验规则,主要分为两类,单独区块头校验规则(Block Header Rule)和依赖区块头校验规则(Dependent Block Header Rule)。定义的部分校验规则和说明见表2和表3。

表2 单独区块头校验规则

规则名	规则描述
TransactionFeeLimitRule	交易费用限制规则,需大于最小值
TransactionFeeValueRule	交易费用需小于限制值
ConsensusRule	共识算法规则,如PoW规则等
BlockHeaderHashRule	区块头哈希规则
FullHashRule	区块内哈希规则
ExtraDataSizeRule	区块携带的额外数据大小规则
ExtraDataPresenceRule	校验区块额外数据是否与提供的一致

表3 依赖区块头校验规则

规则名	规则描述
BestNumberRule	最新区块号规则
ParentNumberRule	父区块号规则
DifficultyRule	难度规则
ParentHashRule	父哈希规则

表2和表3分别给出了部分单独区块头校验规则和依赖区块头校验规则。单独区块头校验规则是指,该规则的校验仅需要提供某一区块头的数据即可完成,不依赖其他数据。依赖区块头校验规则是指,该规则的校验需要依赖于其他区块(比如父区块)的额外信息。

系统在准备数据验证模块时将会首先加载上述区块头校验规则,并根据算法2做出有效性校验,以保证区块链数据的安全性和一致性。

算法2 区块数据有效性验证算法

输入:单独区块头校验规则集合 $R_{\text{block}}=\{r_{b1}, r_{b2}, \cdots, r_{bm}\}$,依赖区块头检验规则集合 $R_{\text{dep}}=\{r_{d1}, r_{d2}, \cdots, r_{dn}\}$,区块 $block$。

输出:校验结果 VR。

1. VR=false
2. for every r in R_{block} do
3. VR_{r_b}=Validate($block$, r)
4. if VR_{r_b}=false
5. return VR
6. end if
7. end for
8. $depBlock$=GetParentBlock($block$)
9. for every r_d in R_{dep} do
10. $VR_{r_d b_d}$=Validate($block$, r_d, $depBlock$)
11. if $VR_{r_d b_d}$=false
12. return VR
13. end if
14. end for
15. return true

算法2首先读取系统中已加载的单独区块头验证规则,并将该规则应用到目标区块上进行校验。在验证通过后获取到目标区块的依赖区块,并依次使用系统中已加载的依赖区块头校验规则,对目标区块和依赖区块进行校验。在该过程中如有任何一条规则校验失败,则整个校验过程返回失败。

数据验证模块即是整合了上述规则和算法,并在自动化任务调度的支持下定时对模型中的数据进行一致性和安全性保障。

3.5 数据查询模块

查询建立在区块链仓库之上,对外提供查询API,同时将热点数据缓存以提高查询效率。

查询API主要分为三类:

获取查询(Retrieval Query):该查询根据一个或多个字段获取对应的区块、交易、用户和智能合约数据,字段匹配可以是具体值精确匹配,也能支持范围查询,如获取某一时间段内的区块数据。

聚合查询(Aggregate Query):通过聚合函数对区块链仓库中数据进行分析性处理,支持多种聚合函数如Count、Sum、Max、Min、Avg等。同样可以分别对区块、交易、用户以及智能合约数据进行处理。

排序查询(Ranking Query):该查询根据指定的查询条件和排序条件返回已排序的数据。

在进行数据查询时,首先检查查询层缓存是否命中,当缓存命中时则直接返回。若缓存中不存在,则从区块链仓库中查询,并将结果同步到内存缓存中。

4 测试与分析

测试采用的硬件环境为Intel Core i7-6500 CPU(3.2 GHz),内存为16 GB,系统为Windows 10。所有算法由Java语言实现,使用多线程技术模拟多个查询请求。区块链应用选用某区块链信息服务平台,该应用数据源采用以太坊(Ethereum)和IPFS。区块链仓库选用MongoDB作为数据库。以太坊选用Rinkeby测试网络,客户端选用Geth v1.8.23。

实验主要分为三组:(1)数据同步测试,测试区块链应用中的数据经过提取、转换和加载到区块链仓库中在不同区块数量下需要花费的时间。(2)一致性验证测试,对同步到区块链仓库中的数据进行校验,测试篡改区块的检出率。(3)查询效果测试,测试应用中具有代表性的查询接口在不同并发线程下的每秒查询数(Query Per Second,QPS),并与不使用查询优化模型之前进行比较。

第一组测试对网络最末尾区块(当前为第4 206 914号区块)前1 000、10 000、100 000个区块进行数据同步并记录完成提取、转换和加载过程需要的总时间,结果如图5。

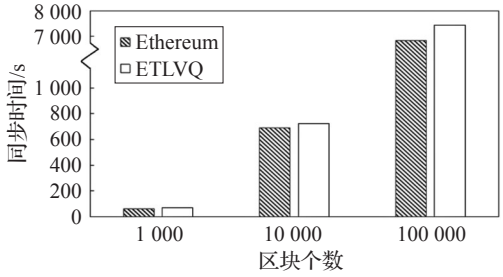

图5 数据同步测试结果

从图5中可以看出,无论是否采用本文的查询优化模型,同步时间均与区块数量这一因子正相关,即区块数量越多,同步时间越长。采用ETLVQ模型后的同步时间较不采用该模型会有所增加,但总体增加比率不高。这一增加的部分主要是由于额外的ETL过程所带来的。表明ETLVQ模型在同步上主要受区块链系统客户端自身同步瓶颈限制,ETL过程耗时相比而言不占据主要部分。

第二组测试分别对前述装载好的区块描述数据、交易数据、用户状态数据、智能合约数据进行改动,均被验证模块成功检查出为篡改数据。表明ETLVQ模型能在一定程度上保证数据的一致性和安全性。

第三组测试首先测试获取查询的吞吐量,在保证查询TP99(Top Percentile 99%,即99%的查询响应时间)小于100 ms,成功率为100%的情况下逐步增大并发查询的线程数。每组进行10轮测试,取10轮QPS的平均值,得到结果如图6所示。

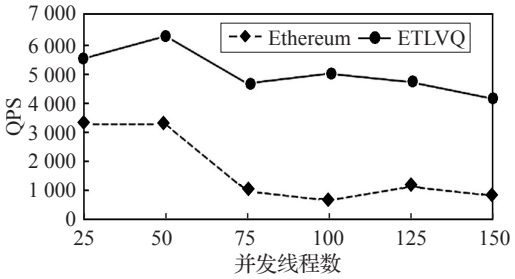

图6 吞吐量测试结果

从图6中可见,随着并发线程数的增加,ETLVQ模型的QPS先小幅增加到6 500,而后下降到5 000左右。而原生以太坊在低并发线程数时QPS维持在3 000,而后在1 000左右内波动。

ETLVQ的查询QPS基本在原生以太坊的两倍以上,在高并发情况下查询效率优势体现得更为明显。表明本文提出的查询优化方案能切实提升查询效果。

5 总结

目前的区块链应用面临着查询效率不足、查询语义少等瓶颈问题,本文结合传统的ETL流程和区块链的特点进行创新,提出了一种ETLVQ查询优化流程模型。该模型可以有效地指导数据从源区块链系统到目标区块链仓库并进行数据查询的全生命周期。结合以太坊和分布式文件系统完成了模型效果测试,测试结果表明,区块同步时间略有上升,但是查询效率大幅提高,对于查询密集型区块链应用有较大实际意义。在未来工作中将继续深入研究区块链的查询优化技术,并着眼于设计开箱即用的区块链应用查询优化工具。

参考文献:

[1] Crosby M,Pattanayak P,Verma S,et al.Blockchain technology:beyond bitcoin[J].Applied Innovation,2016,2:71.

[2] Iansiti M,Lakhani K R.The truth about blockchain[J].Harvard Business Review,2017,95(1):118-127.

[3] Guo Y,Liang C.Blockchain application and outlook in the banking industry[J].Financial Innovation,2016,2(1):24.

[4] Huh S,Cho S,Kim S.Managing IoT devices using blockchain platform[C]//2017 19th International Conference on Advanced Communication Technology,2017:464-467.

[5] Swan M.Blockchain thinking:the brain as a decentralized autonomous corporation [commentary][J].IEEE Technology and Society Magazine,2015,34(4):41-52.

[6] Yu X L,Xu X,Liu B.EthDrive:a peer-to-peer data storage with provenance[C]//29th International Conference on Advanced Information Systems Engineering,2017:25-32.

(下转第171页)

2007,14(10):707-710.
[9] Chen X,Ge D,Wang Z,et al.Complexity of unconstrained L_2-L_p minimization[J].Mathematical Programming,2014,143(1/2):371-383.
[10] Ge D,Jiang X,Ye Y.A note on the complexity of L_p minimization[J].Mathematical Programming,2011,129(2):285-299.
[11] Xu Z B.Data modeling:visual psychology approach and $L_{1/2}$ regularization theory[C]//International Congress of Mathematicians,2010:3151-3184.
[12] Xu Z B,Chang X Y,Xu F M,et al. $L_{1/2}$ regularization:a thresholding representation theory and a fast solver[J].IEEE Transactions on Neural Networks and Learning Systems,2012,23(7):1013-1027.
[13] Jiang Y,Shen P,Zhao P,et al.An improved algorithm of search for compressive sensing image recovery based on l_p norm[C]//2015 Chinese Automation Congress,Wuhan,2015:1962-1968.
[14] Sun Q.Recovery of sparsest signals via l_q minimization[J].Applied and Computational Harmonic Analysis,2012,32(3):329-341.
[15] Chen X,Xu F,Ye Y.Lower bound theory of nonzero entries in solutions of ℓ_1-ℓ_p minimization[J].SIAM Journal on Scientific Computing,2010,32(5):2832-2852.
[16] Chen X,Ng M K,Zhang C.Non-lipschitz ℓ_p-regularization and box constrained model for image restoration[J].IEEE Transactions on Image Processing,2012,21(12):4709-4721.
[17] Lai M,Xu Y,Yin W.Improved iteratively reweighted least squares for unconstrained smoothed ℓ_q minimization[J].SIAM Journal on Numerical Analysis,2013,51(2):927-957.
[18] Zeng J,Lin S,Xu Z.Sparse regularization:convergence of iterative jumping thresholding algorithm[J].IEEE Transactions on Signal Processing,2016,64(19):5106-5118.
[19] 王宜举,修乃华.非线性最优化理论与方法[M].北京:科学出版社,2012.
[20] 袁亚湘,孙文瑜.最优化理论与方法[M].北京:科学出版社,2016.
[21] Tropp J,Gilbert A.Signal recovery from partial information via orthogonal matching pursuit[J].IEEE Transactions on Information Theory,2005,15(2):419-439.
[22] Do T T,Gan Lu,Nguyen N,et al.Sparsity adaptive matching pursuit algorithm for practical compressed sensing[C]//42nd Asilomar Conference on Signals,Systems and Computers,Pacific Grove,2008:581-587.
[23] Needell D,Tropp J A.CoSaMP:iterative signal recovery from incomplete and inaccurate samples[J].Applied and Computational Harmonic Analysis,2009,26(3):301-321.
[24] Bian W,Chen X,Ye Y.Complexity analysis of interior point algorithms for non-Lipschitz and nonconvex ℓ_1 minimization[J].Mathematical Programming,2015,149:301-327.
[25] Yang J,Zhang Y.Alternating direction algorithms for l_1-problems in compressive sensing[J].SIAM Journal on Scientific Computing,2011,33(1):250-278.
[26] Aharon M,Elad M,Bruckstein A. K-SVD:an algorithm for designing overcomplete dictionaries for sparse representation[J].IEEE Transactions on Signal Processing,2006,54(11):4311-4322.

(上接第39页)

[7] Dorri A,Kanhere S S,Jurdak R,et al.Blockchain for IoT security and privacy:the case study of a smart home[C]//2017 IEEE International Conference on Pervasive Computing and Communications Workshops,2017:618-623.
[8] 张利华,蓝凡.基于双区块链的用电数据收集方案设计[J].计算机工程与应用,2019,55(21):110-114.
[9] Li Y,Zheng K,Yan Y,et al.EtherQL:a query layer for blockchain system[C]//International Conference on Database Systems for Advanced Applications.Cham:Springer,2017:556-567.
[10] 王千阁,何蒲,聂铁铮,等.区块链系统的数据存储与查询技术综述[J].计算机科学,2018,45(12):12-18.
[11] Singh S,Singh N.Blockchain:future of financial and cyber security[C]//2016 2nd International Conference on Contemporary Computing and Informatics,2016:463-467.
[12] Helmer S,Roggia M,El Ioini N,et al.EthernityDB integrating database functionality into a blockchain[C]//European Conference on Advances in Databases and Information Systems.Cham:Springer,2018:37-44.
[13] Wang S,Dinh T T A,Lin Q,et al.Forkbase:an efficient storage engine for blockchain and forkable applications[J].Proceedings of the VLDB Endowment,2018,11(10):1137-1150.
[14] Pratama F A,Mutijarsa K.Query support for data processing and analysis on Ethereum blockchain[C]//2018 International Symposium on Electronics and Smart Devices,2018:1-5.
[15] Lin Y R.Efficient blockchain query[D].National Central University,2018.
[16] Vassiliadis P,Simitsis A.Extraction,transformation,and loading[M]//Encyclopedia of database systems.Berlin,Heidelberg:Springer,2009:1095-1101.
[17] Kimball R,Ross M.The data warehouse toolkit:the complete guide to dimensional modeling[M].Hoboken:John Wiley & Sons,2011.
[18] 徐俊刚,裴莹.数据ETL研究综述[J].计算机科学,2011,38(4):15-20.
[19] Li L.A framework study of ETL processes optimization based on metadata repository[C]//2010 2nd International Conference on Computer Engineering and Technology,2010,6:125-129.
[20] Poole J,Chang D,Tolbert D,et al.Common warehouse metamodel[M].Hoboken:John Wiley & Sons,2002.