doi:10.6041/j.issn.1000-1298.2021.08.019

森林生态站大数据快速存储与索引方法

王新阳[1,2]　贾相宇[1,2]　陈志泊[1,2]　崔晓晖[1,2]　许　福[1,2]

(1.北京林业大学信息学院,北京 100083;2.国家林业和草原局林业智能信息处理工程技术研究中心,北京 100083)

摘要:针对森林生态站中大量图像、视频、GIS 数据等非结构化数据以及生态指标等结构化数据存储效率低、检索性能差的问题,提出了基于 Hadoop 和 HBase 的森林生态站大数据存储框架。基于所提出的框架,给出了森林生态数据存储业务流程,并对森林生态大数据平台涉及的核心技术进行了优化:①设计预分区算法保证数据在集群中均匀分布。②根据生态数据特点科学设计了 RowKey,实现生态数据的快速检索。③针对原生 HBase 不支持多条件查询问题,设计基于索引数据和服务器性能评估的 ElasticSearch 索引分片放置策略,以此基于 ElasticSearch 的二级非主键索引技术优化多条件检索 HBase 生态数据库。④针对生态站海量小图像存储困难问题,提出基于数据站点及时间关联性的打包合并策略。⑤解析 GIS 数据使之进行高效存储。通过实验对以上理论进行验证。结果表明,ElasticSearch 索引分片放置策略比默认分片策略的查询时间平均减少 20 ms,比基于改变 ElasticSearch 评分策略的查询时间平均减少 20 ms。结构化数据规模为 1×10^8 条时,系统的检索时间为 1.045 s,比原生 HBase 检索速度提升 3.99 倍,在非结构化数据为 1×10^7 条时,采用数据站点及时间关联性的打包小图像策略是基于 SequenceFile 合并效率的 1.15 倍,是原生 HBase 的 1.79 倍;在 1×10^4 次并发用户的情况下,优化后的每秒查询数是原来的 1.88 倍,每秒吞吐量是优化前的 1.74 倍,系统响应时间比优化前降低 69.5%。结果表明,本文所提出的方案在集群负载均衡、海量结构化和非结构化数据检索效率以及系统吞吐量等方面都有了明显的性能提升,为森林生态数据的存储和管理提供了必要的理论基础和技术实现。

关键词:森林生态;大数据;快速存储;数据索引;分布式平台

中图分类号:TP392　　**文献标识码**:A　　**文章编号**:1000-1298(2021)08-0195-10　　**OSID**:

Fast Storage and Indexing Method of Big Data in Forest Ecological Station

WANG Xinyang[1,2]　JIA Xiangyu[1,2]　CHEN Zhibo[1,2]　CUI Xiaohui[1,2]　XU Fu[1,2]

(1. *College of Information Science and Technology, Beijing Forestry University, Beijing* 100083, *China*
2. *Engineering Research Center for Forestry-oriented Intelligent Information Processing, National Forestry and Grassland Administration, Beijing* 100083, *China*)

Abstract: Aiming at the problems of low storage efficiency and poor retrieval performance of a large number of unstructured data such as images, videos, GIS data and ecological indicators in the forest ecological station, a forest ecological station big data storage framework was proposed based on Hadoop and HBase. Based on the proposed framework, the business process of forest ecological data storage was given and the core technologies involved in the forest ecological big data platform was optimized. A pre-partitioning algorithm was designed to ensure that the data was evenly distributed in the cluster. According to the characteristics of ecological data, the RowKey was scientifically designed to achieve rapid retrieval of ecological data. Aiming at the problem that native HBase did not support multi-condition query, an ElasticSearch index shard placement strategy was designed based on index data and server performance evaluation, and the multi-condition search HBase ecological database was optimized based on ElasticSearch's secondary non-primary key index technology. In view of the difficulty of storing large amounts of small pictures in the ecological station, a package and merge strategy was proposed based on data sites and time relevance. GIS data was analyzed for efficient storage. The above theory was verified through experiments. The results showed that the ElasticSearch index shard placement strategy reduced

收稿日期:2021-02-08　修回日期:2021-05-04

基金项目:中央高校基本科研业务费专项资金项目(BLX201923)和国家自然科学基金项目(32071775)

作者简介:王新阳(1985—),男,讲师,博士,主要从事大数据集成、并行计算故障诊断与容错等研究,E-mail:wxyyuppie@bjfu.edu.cn

通信作者:陈志泊(1967—),男,教授,博士生导师,主要从事大数据技术与人工智能、计算机软件与理论研究,E-mail:zhibo@bjfu.edu.cn

the query time by an average of 20 ms compared with the default shard strategy. The average query time was reduced by 20 ms compared with that based on changing the ElasticSearch scoring policy. When the structured data size was 1×10^8, the retrieval time of the system was 1.045 s, which was 3.99 times faster than the native HBase retrieval, and when the unstructured data was 1×10^7 pieces, the based on data site and time correlation package small picture strategy was 1.15 times that of SequenceFile-based merging efficiency and 1.79 times that of native HBase. In the case of 1×10^4 concurrent users, after optimization, the number of queries per second was 1.88 times as much as before, the throughput per second was 1.74 times as much as before, and the system response time was 69.5% lower than that before optimization. From the above results, it can be seen that the solution proposed had significant performance improvements in cluster load balancing, massive structured and unstructured data retrieval efficiency, and system throughput, which provided the necessary theoretical foundation and technical realization for the storage and management of forest ecological data.

Key words: forest ecological; big data; fast storage; data index; distributed platform

0 引言

为了更好地对森林生态系统的服务功能进行评估,我国在各地建立了大量的森林生态定位观测站,针对典型生态系统类型进行了长期连续定位观测[1-2]。迄今为止,中国已有 180 多个国家级森林生态定位观测站(以下简称生态站),分布在不同气候区,覆盖不同类型的生态系统[3-4]。生态观测站点能够进行长期、连续地观测并自动感知和获取有关观测区的水、土、气、生等方面的生态因子数据,所累积的数据具有海量性、多样性等特点[5-10]。然而,采用单一站点对生态数据进行存储和管理的模式已经无法满足海量异构生态数据的存储和管理需求,且各生态站点之间相互独立,逐渐形成信息孤岛,更不能满足面向生态服务功能评价所需要的多站联合分析、数据挖掘、实时检索以及高并发的访问需求,并且海量生态数据具有多样性,庞杂且无法重构,生态数据处理过程中容易产生运算负担大、检索时间慢等问题[11-13]。因此,研究面向森林生态大数据的存储和索引模型,并基于此建立海量生态数据管理平台显得非常有必要。目前大多数生态站存储架构以 MySQL 为代表的关系型数据库为主,如文献[14]提出通过 MS-SQL Server 复制技术实现数字化森林生态站的应用,文献[15]提出通过调用云平台提供的 REST Servce API 来实现西天山森林生态站数据管理系统建设,可以保证事务的一致性,但在拓展性、容错性和可用性方面都不能满足海量异构数据的存储和管理。针对传统关系型数据库的劣势,采用 Hadoop[16-17] 分布式平台和 HBase[18] 分布式 NoSQL 数据库,在扩展性、容错性和可用性方面具有巨大优势,如文献[19]为解决海量 GIS 数据设计的基于 Hadoop 的 GIS 平台,文献[20]通过 Hadoop 构建了森林资源信息平台,为各级部门提供更加有效科学、准确的数据参考。Hadoop 和 HBase 技术的诞生为解决生态大数据高效存储和快速索引奠定了重要技术的基础,但原生 Hadoop 不能很好地处理小文件问题,原生 HBase 默认只支持一级索引[21]。

以上这些平台主要存在 2 个问题:①原生 Hadoop 在存储方面没有解决海量小图像的存储问题和没有对海量数据进行分区处理。②原生 HBase 在面对海量数据多维查询时没有提供很好的快速检索方案。

生态站产生的大量小文件会导致 NameNode 内存瓶颈和检索性能低下,文献[22]提出 Harballing 技术,通过一种打包技术将小文件打包成大文件,但预处理花费时间比较长。文献[23]提出将相同类型的小文件合并为大文件,并建立小文件到合并大文件的索引关系,索引关系存储于 HashMap 中,这种方式如果没有命中缓存,读取性能并不高。文献[24]提出了利用 SequenceFile 技术实现海量物联网图像打包合并策略,这种方式可以解决 NameNode 内存过大问题,但并未考虑图像之间的关系,不利于关联查询;对于 GIS 数据,文献[25]利用传统的关系型数据库存储,但该种架构面临生态站海量 GIS 数据时,会有拓展性差、存取效率低等问题。另外,对于海量生态数据应用场景,如果没有做到合理的预分区,将会造成数据倾斜,极端情况下,分布式存储变成了单节点存储。

直接利用 HBase 提供的列簇和主键(RowKey)机制组织和存储生态数据会存在很多问题,如 HBase 并未提供对多维生态数据的原生支持,只支持对数据记录的主键上建立一维索引,在对主键之外的其他属性信息进行查询时,效率极低。根据实现方式的不同,目前比较主流的方式可以分为双层索引、全局分布式索引、位图索引和基于 Hadoop 框架的索引[26]。双层索引的代表是 RT-CAN[27] 和 A-Tree[28],但这种索引维护起来代价高,实现很复

杂，面对生态领域多维查询时，性能急剧下降；全局分布式索引实现起来相对简单，但对于生态领域大数据量的场景下，无法满足数据频繁插入；位图索引典型的是 RBI[29]，可以适合高并发的场景，但对于生态数据划分规则复杂的场景时，会造成 false positives 的场景；而基于 Hadoop 框架实现的索引，大多是借助 Hadoop 生态组件来构建索引，以此来提高性能，其中包括基于 Coprocessor 方案、ApachePhoenix 方案[30]和华为 HIndex 方案[31]等，该方案需要为每一个索引列构建一个索引表，以索引的值作为新的 RowKey，实现索引列与原有 RowKey 的映射，但数据存储和索引进行耦合，具有一定的入侵性，会对 RegionServer 性能产生一定影响。文献[32]提出了一种基于 Solr 的 HBase 海量数据的二级索引方案，将数据存储和索引进行分离，但 Solr 实时建立索引时，会产生 I/O 阻塞，查询性能较差，并且随着数据量增加，检索效率会变得更低，同时面对生态数据时并没有解决对多条件查询优化的问题；文献[33]提出互补聚簇式索引，将数据详细信息也放入索引表，这样把随机读变成了顺序读，但当索引列比较多时，空间开销会更大，索引更新代价会比较高，影响了吞吐量；MD－HBase 利用 KD－tree 对多维空间数据进行划分，根据最长公共前缀的方式计算得到每个子空间名称，但该方法会造成数据一致性问题[34]。以上二级索引方案都不适合生态站的应用场景，ElasticSearch 对系统的耦合度低且简单易行，在实时搜索方面性能很高，同时可以定制化专门生态数据的评分机制，所以采用 ElasticSearch 作为 HBase 的二级索引解决方案。

针对生态大数据存储和快速索引方面的迫切需求，以及现有森林生态数据方案存在的弊端问题，本文在存储方面科学地设计 RowKey，其次设计预分区算法保证数据分布一致；针对海量小文件处理问题，提出基于数据站点及时间关联性的关联合并存储方法（Redis based station time cooperate，RBSTC）；针对 GIS 数据存储问题，提出 HBase 和 GeoTools 存储方法[35]；在索引方面，根据生态站点的特点设计 RowKey，并设计基于索引数据和服务器性能评估的 ElasticSearch 分片算法，优化多条件检索二级索引失效问题[36]。以期能够满足海量异构生态数据的存储和高效检索，为生态监测数据高效存储、管理和分析提供技术支撑。

1　森林生态大数据平台总体架构

从森林生态数据特征出发，提出基于 Hadoop 的森林生态大数据平台，可用于分布在全国各地的生态站数据管理。平台深度集成大数据、物联网、人工智能等技术，为用户提供森林生态数据的快速检索、处理和可视化分析。其总体架构如图 1 所示，主要包括应用层、服务层、存储层。

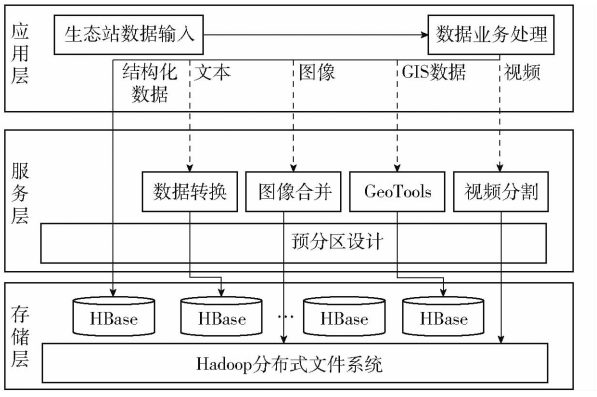

图 1　总体架构图

Fig. 1　General architecture diagram

存储层是森林生态大数据模型中最重要的部分，其主要作用是用来持久化存储。该层包括基于分布式的 HDFS 和面向列的 HBase 数据库。HDFS 存储森林生态数据中的视频、图像等非结构化数据；HBase 数据库用来存储森林生态中产生的结构化数据。

服务层主要包括数据转换、图像合并、GeoTools、视频分割、预分区设计和 HBase 二级索引。数据转换主要是因为接入数据复杂且海量，同时各传感器型号类型不同，数据传输格式不统一，利用数据转换模块将数据继续进行统一，有利于系统后期拓展性能增强；图像合并模块主要解决的是 Hadoop 处理海量小文件时性能不足的问题，通过基于数据站点及时间关联性算法对小文件进行合并，达到阈值后存储 HDFS 上；GeoTools 主要针对 GIS 数据解析后存入 HBase 数据库；视频分割是将大文件按照 HDFS 数据块大小进行分割后，直接储存在 HDFS 上；HBase 二级索引模块主要解决 HBase 在查询主键之外的数据索引失效的问题，可以实现数据的高效检索。

应用层主要是面向各类用户对森林生态大数据平台中的各种业务进行统一数据处理。用户可以对森林生态数据进行查询、分析、管理和数据下载等。

2　森林生态大数据平台关键技术

2.1　业务流程设计

目前森林生态站点采集的数据主要分为图像、视频、GIS 数据、文本数据的非结构化数据及结构化数据 5 大类，文本数据主要包括 Excel 文件和 TXT

文件等,业务处理模块需要统一数据访问接口对需要存储的数据类型进行判断,并针对不同的数据类型,采取不用的存储策略,数据写入业务算法的处理伪代码:

```
输入:生态站实时传入数据
输出:HDFS 和 HBase 数据库
令 RealtimeData = 实时数据流集合
BlockCapacity 为每个文件块的容量
while RealtimeData ≠ ∅
    if RealtimeData 为图像类型数据
        将图像加入合并队列基于 Redis 合并;
    if BlockCapacity 到达阈值
        构建索引信息并写入 HDFS;
    if RealtimeData 为视频类型数据
    if BlockCapacity > 128MB
        视频分块;
        构建索引信息并写入 HDFS;
    if RealtimeData 为 GIS 类型数据
        GeoTools 工具解析;
        构建索引信息并写入 HDFS;
    if RealtimeData 为文本类型数据
    if 为 Excel 类型
        Excel 解析器解析;
    if 为 TXT 类型
        TXT 解析器解析;
        ⋮
        构建索引写入 HBase
    if RealtimeData 为结构化数据
        构建索引写入 HBase
end while
```

当存储图像时,因为生态监测中的图像都是以海量的小文件为主(每幅图像通常在几兆字节以内),而 HDFS 的数据块(Block)默认容量为128 MB,当把一幅图像存储到一个数据块中时,虽然不会占满整个数据块,海量小文件不会对硬盘存储产生压力,但会增加 HDFS 中 NameNode 的内存消耗,而且,读取小尺寸图像会浪费大量时间。

一般情况下 NameNode 的元数据容量为 250 B,默认 3 个副本的情况下,2 个新副本的元数据容量为 368 B。当 n 个小图像在 HDFS 上进行 3 副本存储时,NameNode 的内存消耗 M 随着 n 的增大而增大。

$$M = a + 250n + (368 + b)\sum \frac{F}{B} \tag{1}$$

式中 a——HDFS 没有数据时 NameNode 所占内存容量

b——每个数据块在 NameNode 的内存消耗

B——HDFS 数据块容量

F——HDFS 中存储的 n 个图像所占的内存容量

因此,当数据为图像时,采用对图像进行基于数据站点及时间关联性合并存储的策略,即:首先将图像写入一个图像队列,判断图像队列的值是否大于一个 Block 容量,如果不够一个 Block 的容量则继续写入;如果大于一个 Block 的容量,则直接存入 HDFS 中,并建立索引,将图像元信息存入 HBase 中。

当写入数据为视频时,首先判断是否超过一个 Block 容量,如果没有超过,直接按照一个 Block 的容量来处理,如果大于一个 Block 的容量,则直接按照 HDFS 分块策略进行分块,建立索引,并将视频元数据信息存入 HBase 中。

当写入数据为 GIS 数据时,通过 GeoTools 工具进行解析,存入 HBase 数据库。

当写入数据为文本数据时,通过对应服务层解析,将文本数据转成结构化数据后,存入 HBase 数据库。

当写入数据为结构化数据时,直接将数据存入 HBase 数据库。

2.2 预分区设计算法

HBase 在默认情况下创建数据表时,会创建一个没有起始和终止的 Region,数据会按照键值对的字典升序写入该 Region,如果 HBase 的 Region 到达阈值,就会频繁触发分裂(Split)操作,会造成热点倾斜,取值范围为 0 ~ 60。假设拟划分成 k(k 为整数)个分区,将 0 ~ 60 范围内数据 splitKey 从 1 开始,按照 HBase 预分区算法初步进行预分区,算法伪代码为:

输入:拟划分分区 k,数据量为 num(n),平均写入请求数为 avg;

输出:splitKey

$\text{splitKey}(n) = \left\lfloor n\frac{60}{k} \right\rfloor$;($n = 1, 2, \cdots, k-1$;$k$ 为整数)

$$M = \left| \frac{\text{num}(n+1) - \text{num}(n)}{\text{avg}} \right|;$$

```
if M > 1   //进行微调
  if num(n) < num(n+1)
    x = (num(n+1) - avg)/num(n+1) * 60/k;
    splitKey(n+1) = splitKey(n+1) + x;
    return;
```

else

$$x=\frac{\text{num}(n+1)-\text{avg}}{\text{num}(n)}\frac{60}{k};$$

splitKey(n+1) = splitKey(n+1) − x;

return;

最后根据预分区算法获取 splitKey,并创建预分区表,以避免热点倾斜。

2.3 RowKey 设计方案

HBase 数据库主要由行键、列族、列族限定符和时间戳组成。在满足长度原则、哈希原则和唯一性原则的前提下,行键可以提高内存利用率。由于生态站点查询使用较为频繁,所以在设计时添加该字段,同时为了记录数据产生的时间以及数据的版本控制,将时间也放在主键 RowKey 内,所以本系统设计的 RowKey 格式为:站点 + 时间(YYYYMMDDHHMM)。森林生态结构化数据根据特点将其分为 4 个列族,分为土壤列族、气象列族、生物多样性列族、水文列族,共包括 742 个要素,如温度、相对湿度、风速、降水量等。一个站点在特定时间的标识 ID 存储一个 RowKey,每一个 RowKey 会有多个列族(Column qualifer),表示不同时间的要素值。HBase 结构及数据样例如表 1 所示。

表 1　RowKey 设计

Tab. 1　RowKey design

行键	时间戳	气象列族		土壤列族		…
		空气温度/℃	空气相对湿度/%	土壤含水率/%	土壤类型	…
站点 + 时间	2020 − 01 − 12　01:11	14.5	60.77	22.4	壤土	…

2.4 HBase 二级索引选取及 ElasticSearch 索引分片算法

HBase 原生不支持二级索引,对非主键查询时会出现索引失效问题。但由于生态数据检索时常常需要对数据某一个维度进行统计分析,如根据温度、湿度等信息进行统计查询,如果没有二级索引,则会进行全表扫描,效率极其低下,与传统的关系型数据系统索引类似。HBase 二级索引存储新坐标和现有坐标之间的映射关系,因此,从索引的角度出发,选择合适的二级索引尤为重要。

在现在主流的 HBase 二级索引方案中,Phoenix 方案并没有将数据与索引分开存储、耦合性很强;在实时建立索引时,Solr 会产生 I/O 阻塞,查询性能较差,随着数据量的增加,Solr 的搜索效率会降低,而 ElasticSearch 没有明显的变化,因此选择 ElasticSearch 作为二级索引。

其利用 ElasticSearch 提供主要索引设计的索引结构为

```
PUT create_index{
"settings":{
"number_of_shards"x, //根据算法修正
    "number_of_replicas":3
},
"mappings":{
"data_type":{
"properties":{
"dataId":{"type":"long","index":"yes","
store":true},
"datayear":{type:"Int","index":"yes","
store":true},
"datamonth":{type:"Int","index":"yes","
store":true},
"dataday":{type:"Int","index":"yes","
store":true},
"datahour":{type:"Int","index":"yes","
store":true},
"datamin":{type:"Int","index":"yes","
store":true},
"dataTemp":{type:"Int","index":"yes","
store":true}
"rowkey":{type:"String","index":"yes","
store":true}
 ⋮
}}}
}
```

由于生态站观测具有长期性,所以需要在架构设计之初,就考虑未来存储容量的问题,但原生 ElasticSearch 不易确定索引分片问题,索引主分片数量设置后不能修改,分片过多资源浪费;分片过少,会导致索引性能和集群性能降低,且不能保证生态站未来扩容,故针对原生 ElasticSearch 分片问题,提出基于索引数据和服务器性能评估的索引分片放置策略,具体 ElasticSearch 索引分片算法伪代码为

输入:数据量 D,磁盘使用率 $\text{rate}_{\text{disk}}$,虚拟机堆内存容量 J

输出:索引分片配置

令 nodeList = 可用节点集合,$\text{node}_{\text{current}}$表示现有节点,$\text{node}_{\text{current}}$(shardNum)表示现有节点的

shardNum;

if $rate_{disk}$ <85% && $node_{current}$(shardNum) < $J*20$

//ES 官方建议节点磁盘使用率小于 85%

//1GB 堆内存对应 20shardNum

nodeList( ). add($node_{current}$);

$$shardNum = \left[\left(k_1 * |nodeList| + k_2 * \frac{D}{20} + k_3 * M \right) * k_4 \right];$$

$//k_1, k_2, k_3, k_4$为权重系数

if shardNum > $|nodeList|^2$

shardNum = $|nodeList|^2$;

return shardNum;

同时 ElasticSearch 的每一个单元称之为 Document(doc),doc 和 HBase 中的 RowKey 对应。结合生态站的常用业务,将年(year)、月(month)、日(day)、时(hour)、分(min)、气温(Temp)、纬度(Latitude)、湿度(Hum)作为字段建立索引,通过 ElasticSearch 的多条件检索快速过滤到符合条件的 RowKey,通过指定的 RowKey 在 HBase 中检索符合条件的结果,以实现海量生态数据的高效检索,基于 ElasticSearch 实现非 RowKey 查询如图 2 所示。

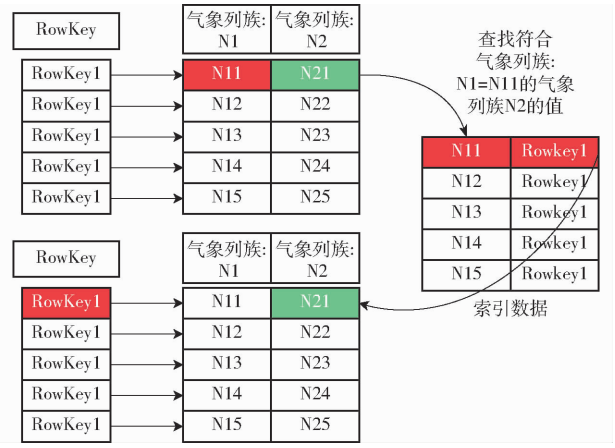

图 2 ElasticSearch HBase 查询过程图

Fig. 2 ElasticSearch HBase query process diagram

当要对气象列族 N1 这列建立索引时,只需要建立气象列族:N1 各列值到其对应行键的映射关系,如 N11→RowKey1 等,这样就完成了对气象列族 N1 列值的二级索引的构建,当要查询符合气象列族 N1 = N11 对应的气象列族 N2 的列值时(即根据 N1 = N11 来查询 N2 的值,图 2 绿色部分)其查询步骤如下:①根据 N1 = N11 到索引数据中查找其对应的 RowKey,查询得到其对应的 RowKey = RowKey。②得到 RowKey1 后即能根据 RowKey1 来查询 N2 的值。

2.5 图像合并索引算法

由于生态站内存储的图像大多都是小图像,会出现 NameNode 瓶颈问题和检索性能低等问题。现有的算法一般都是为了解决内存占用较高等问题,将小文件合并成大文件,再存储成大文件的方式,但没有考虑到图像之间的问题,比如取某一生态特征数据图像,可能分散在不同的大文件里面,存储效率低下。

因此本文提出一种基于数据站点及时间关联性的合并存储方法 RBSTC。首先,创建临时队列,存入初始图像。然后,判断待存储的图像是否来自相同站点并且同一天,若是,则将存储的图像合并到临时队列中,否则新建一个新的队列,待存储图像作为新建队列中的初始图像,重复以上操作,直到所有待上传图像都传至 HDFS 上。图像合并索引算法具体操作伪代码为

输入:生态站实时传入图像数据

输出:HDFS

令 RealtimeData = 实时数据流集合

BlockCapacity 为每个文件块的容量

QueueCapacity 为每个队列块的容量

while RealtimeData

判断 RealtimeData 类型

if BlockCapacity 到达阈值

构建索引信息并写入 HDFS;

else

if ReatimeData 类型存在

合并文件到临时队列块

else 创建新的临时队列

if QueueCapacity 到达阈值

构建索引信息并写入 HDFS;

清除队列块信息

end while

2.6 GIS 数据存储设计

GeoTools 是用 Java 语言开发的 GIS 工具包,基于标准的 GIS 接口,支持多种 GIS 数据源的访问,GIS 数据一般由坐标数据、属性数据、拓扑关系数据组成,根据矢量数据的特点,设计适合 HBase 的矢量数据存储模型,GIS 数据 RowKey 设计中 RowKey 为站点 + 时间戳,分为 3 个列族,空间信息列族、属性信息列族、拓扑信息列族。具体生态站 GIS 数据存储算法伪代码为

输入:生态站实时传入 GIS 数据

输出:HBase 数据库

令 RealtimeData = 实时数据流集合

while RealtimeData ≠ ∅

GeoTools 开始解析 GIS 数据

创建 HBase 链接

```
if 创建表成功
   创建列族
   for(1 to 列族 . length){
      添加列族属性
      }
  else 表创建失败
end while
```

3　森林生态站系统性能测试

3.1　集群部署

为了评估本文技术方案的性能，基于不同数据量级的数据对存储模型、预分区、RowKey 设计、二级检索方案、图像合并策略等进行性能测试，本文配置了服务器相关环境搭建了 Hadoop 集群、HBase 集群、Zookeeper 集群；为了进行二级索引对比实验，还搭建了 ElasticSearch 集群、Solr 集群、Phoenix 集群，服务器配置如表 2 所示。

表 2　服务器配置

Tab. 2　Server configuration

设备	参数
CPU	Inter(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz
内存容量/GB	32(Server configuration)
硬盘容量/GB	500
网络类型	千兆网络
操作系统	Centos Linux Server 7.4
Hadoop	3.1.2
HBase	2.1.0
ElasticSearch	6.7.0
Zookeeper	3.4.10
Solr	6.0.0
Phoenix	5.0.0

3.2　实验结果与分析

3.2.1　ElasticSearch 索引分片放置策略验证

通过校验集群节点性能因素以及索引数据量，来构建索引分片数量计算模型得到合理的索引分片数量，其中索引预估生态数据量是分片结果的关键因素。为了验证索引放置策略模型优化的提升度，与文献[37]改变原有 ES 评分机制提升 ES 性能的方法进行对比。首先对比不同生态数据量下，分片优化模型和 ElasticSearch 原生默认的分片数据数量结果，如图 3 所示。

从图 3 可知，随着数据量的增大，分片模型得到的分片数量也随之线性增长，但不仅仅是单纯的线性增长，还考虑了索引的拓展性，当数据量小于 60 GB，可以减小资源浪费，大于 60 GB 时，可以避免单片数据量过大，造成性能下降。

索引查询是衡量索引性能的重要指标，因此本文也做了原生默认和分片策略模型结果生成索引的查询，以此验证分片模型对性能的影响，索引查询性能对比如图 4 所示。

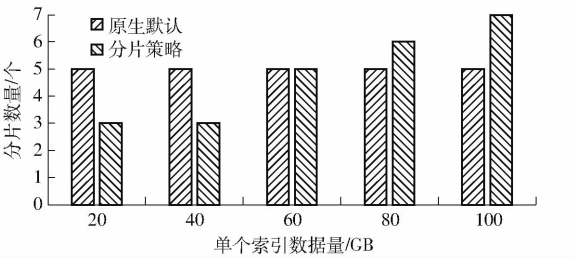

图 3　原生与索引分片数据模型对比

Fig. 3　Comparison diagram of native and indexed sharded data models

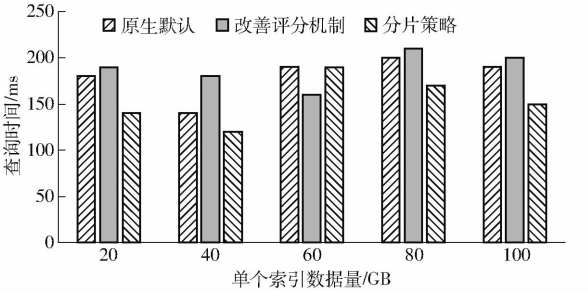

图 4　原生与索引分片数据模型查询时间对比

Fig. 4　Comparison diagram of native and indexed sharded data model queries

从图 4 中可看出，相同的索引数据量，优化后模型生成的索引查询普遍比原生默认查询低 20 ms，并且随着索引数据量的增大，两种索引的查询延迟应该更大。但通过改变 ES 评分机制来提高性能的方法并没有起到太好的效果，因为生态数据本身就比较繁杂，检索评分不易判断并且本身检索计算基于内存，所以提升效果并不明显。因此，优化后的分片数量模型，对生态站的索引查询性能有提升。

3.2.2　数据插入及检索方案性能验证

3.2.2.1　插入性能对比

实验通过 4 台客户端同时向 HBase 表插入数据，并在 4 台客户端上统计每 10^7 条数据的 Put 时间，重复了 10 次实验后，取平均值，分别测试 HBase 原生、基于 ElasticSearch 的索引、Solr 的索引和 Phoenix 的索引在相同条件下的 Put 时间，其结果如图 5 所示。

由图 5 可以看出，Put 执行效率最高的是 HBase 原生方式，这是因为在 Put 操作时，无需再分配资源进行索引的构建，而其它方式都需要额外构建二级索引。可以看出在同样增加 10^7 条数据时，插入时间越来越长，这是因为随着数据量的增加，索引数据也越来越多，在进行索引插入时困难。同时，因为 Phoenix 底层需要在协处理器构建合适的存储索引结构，额外消耗计算资源。而基于 ElasticSearch 和

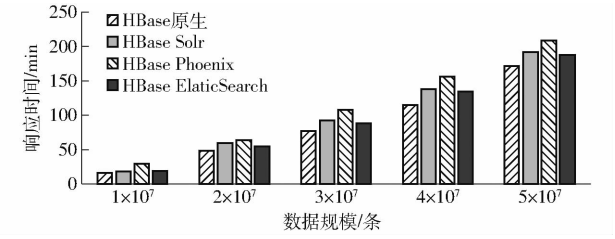

图 5 插入数据响应时间对比

Fig. 5 Comparison of response time of inserted data

Solr 只需要在自己的集群中构建索引，不需要程序额外的计算资源，因此 Phoenix 二级索引对插入性能损耗最大。

3.2.2.2 不同二级索引的单条件查询性能对比分析

检索数据仍采用上述数据，其检索性能对比如图 6 所示。

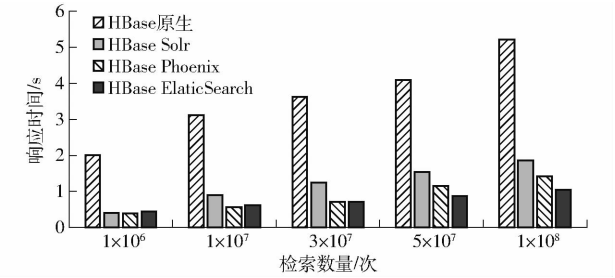

图 6 检索性能对比

Fig. 6 Comparison of retrieval performance

由图 6 可看出，原生 HBase 响应速度降低较为明显，当数据量达到 1×10^8 时，响应时间大于 5 s，从理论层面分析，HBase 是一个面向列的数据库，它的底层是建立一个基于 RowKey 的 B + 树索引，可以很高效地对数据进行检索，但对应非行键的索引，系统将进行全表扫描，整体数据检索效率低下。针对二级索引的对比，Solr 和 ElasticSearch 底层都是基于 lucence，但 ElasticSearch 的框架设计进行了进一步优化，数据检索效率也具有更好的性能，所以当数据量达到 1×10^8 条，ElasticSearch 的效率是 Solr 的 1.72 倍。Phoenix 的检索效率跟 ElasticSearch 速度接近，但 Phoenix 的耦合性比较强，因此最终选择 ElasticSearch 作为二级索引。

3.2.2.3 多条件查询、性能分析及其优化

检索数据仍采用上述数据，检索条件数分别为 2、5、10，取 10 次实验后的平均值，其平均检索性能对比如图 7 所示。

把数据横向对比，发现检索时间与检索条件成负相关，检索条件越多检索时间越短，比如在数据为 5×10^7 条时，检索条件数取 2、5、10，HBase 的检索时间为 4.721、4.328、4.295 s，HBase 索引优化的时间为 0.925、0.884、0.888 s，这是由于在 HBase 中检索条件增多，符合条件的数据检索量减少，就相当于减

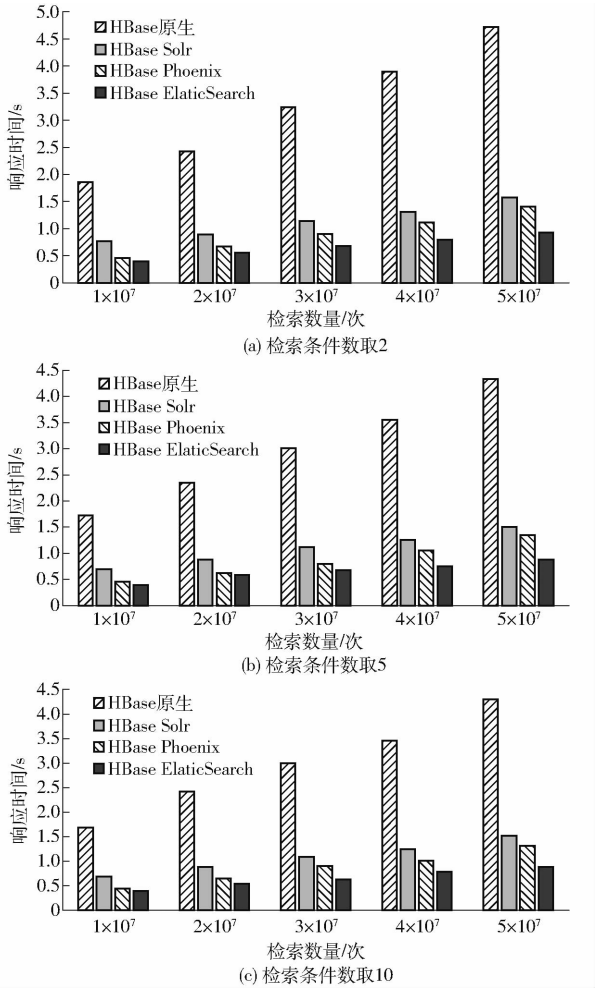

图 7 多条件检索性能对比

Fig. 7 Comparison of multi-condition retrieval performance

少用户并发量，因此，检索时间也一直降低。

3.2.3 非结构化数据存储方案性能验证

数据存储验证部分，针对图像的合并策略性能验证，即验证 HDFS 默认的 SequenceFile 合并和本实验使用 RBSTC 算法对比测试。

采用生态站系统中图像共 1×10^7 幅，占用存储空间 100 ~ 500 KB，将图像采用 HDFS 默认的 SequenceFile 合并和 RBSTC 合并索引分别测试，做 10 次读写实验，取 10 次读写的平均值作为最终时间消耗，其平均读写验证对比如图 8 所示。

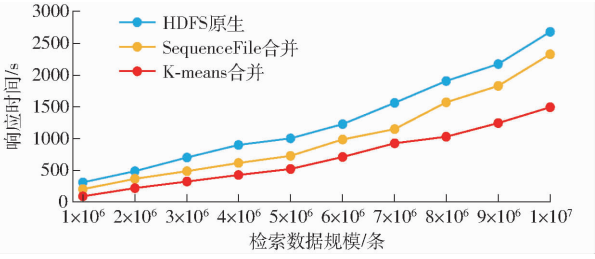

图 8 读写性能对比

Fig. 8 Reading and writing performance comparison

由图 8 可看出，随着图像数据规模不断增大，基于数据站点及时间关联性的合并方式优势越来越明

显，当图像数量为 1×10^7 幅时，基于数据站点及时间关联性的小文件合并是原生读写的 1.79 倍，是 SequenceFile 的 1.15 倍，因此基于数据站点及时间关联性的合并策略更适合生态站海量图像的存储场景。

3.2.4　系统压力测试

为了验证系统稳定性，采用 Postman 工具对大数据平台进行压力测试，大数据平台如图 9 所示。选取每秒查询率（QPS）、吞吐量（TPS）和响应时间（RT）作为主要参数，并发量为 1×10^4 次，测试时间为 3 min，取 10 次实验的平均值，并发测试如表 3 所示。

图 9　大数据平台页面

Fig. 9　Home page of big data platform

表 3　并发压力测试结果

Tab. 3　Pressure test result

参数	HBase	HBase 二级索引
每秒查询数	3 475	6 520
每秒吞吐量/MB	42	73
响应时间/ms	852	260

通过测试结果可以发现，在 1×10^4 次并发用户的情况下，优化后的每秒查询数是原来的 1.88 倍，每秒吞吐量是优化前的 1.74 倍，系统响应时间比优化前降低 69.5%，说明该系统在高并发的条件下同样可以稳定运行。

4　结论

（1）面对海量生态数据的存储和快速检索的需求，传统模型架构无法保证生态数据平台的数据处理性能，采用分布式大数据技术，搭建了生态大数据平台，通过对 RowKey 的科学设计，采用 Hadoop 和 HBase 作为数据存储层，实现了海量数据的存储；提出并设计预分区算法保证数据分布一致，避免了热点数据倾斜问题。

（2）提出基于索引数据和服务器性能评估的索引分片放置策略，并通过 ElasticSearch 优化了多条件检索二级索引失效的问题，优化后的分片策略比默认分片策略和基于改变 ElasticSearch 评分的策略平均查询时间减少 20 ms；结构化数据规模为 1×10^8 条时，系统的检索时间为 1.045 s，比原生 HBase 检索速度提升 3.99 倍；在 1×10^4 次并发用户的情况下，优化后的每秒查询数是原来的 1.88 倍，每秒吞吐量是优化前的 1.74 倍，系统响应时间比优化前降低 69.5%。

（3）针对海量图像数据的存储需求，提出一种基于数据站点及时间关联性的关联合并存储方法，在非结构化数据为 1×10^7 条时，采用数据站点及时间关联性的打包小图像策略是基于 SequenceFile 合并效率的 1.15 倍，是原生 HBase 的 1.79 倍。从实验结果可以看到，本文所提出的方法可以有效地满足生态数据存储和检索的实时需求，访问效率取得良好效果，能够根据需要高效地从各个维度对海量生态数据进行查询和统计分析，为应用深度学习等人工智能技术对生态数据进行多维度特征抽取提供足够多的数据资源，进而为后期开展生态数据智能决策分析提供了必要的数据支撑和处理能力。

参　考　文　献

[1]　王兵，丁访军. 森林生态系统长期定位观测标准体系构建[J]. 北京林业大学学报，2010，32(6)：141－145.
WANG Bing, DING Fangjun. The construction of long-term positioning observation standard system of forest ecosystem [J]. Journal of Beijing Forestry University, 2010, 32(6): 141－145. (in Chinese)

[2]　傅伯杰，于丹丹，吕楠. 中国生物多样性与生态系统服务评估指标体系[J]. 生态学报，2017，37(2)：341－348.
FU Bojie, YU Dandan, LÜ Nan. Ecological indicators for assessment of biodiversity and ecosystem services in China [J]. Acta Ecologica Sinica, 2017, 37(2): 341－348. (in Chinese)

[3]　邵明，董宇翔，林辰松. 基于 GWR 模型的成渝城市群生态系统服务时空演变及驱动因素研究[J]. 北京林业大学学报，2020，42(11)：118－129.
SHAO Ming, DONG Yuxiang, LIN Chensong. Spatial-temporal evolution and driving factors of ecosystem services in Chengdu－Chongqing urban agglomeration based on GWR model [J]. Journal of Beijing Forestry University, 2020, 42(11): 118－129. (in Chinese)

[4]　李惠萍，刘小林，张宋智，等. 小陇山生态站森林生态系统服务功能及其价值评估[J]. 西北林学院学报，2012，27(5)：15－20.
LI Huiping, LIU Xiaolin, ZHANG Songzhi, et al. Assessment on value of forest ecosystem services in Xiaolongshan ecological research station[J]. Journal of Northwest Forestry University, 2012, 27(5): 15－20. (in Chinese)

[5]　宋庆丰，牛香，王兵. 基于大数据的森林生态系统服务功能评估进展[J]. 生态学杂志，2015，34(10)：2914－2921
SONG Qingfeng, NIU Xiang, WANG Bing. Forest ecosystem service function assessment based on big data: a review [J]. Chinese Journal of Ecology, 2015, 34(10): 2914－2921. (in Chinese)

[6]　牛腾，岳德鹏，张启斌，等. 潜在生态网络空间结构与特性研究[J/OL]. 农业机械学报，2019，50(8)：166－175.
NIU Teng, YUE Depeng, ZHANG Qibin, et al. Spatial structure and characteristics of potential ecological networks [J/OL].

Transactions of the Chinese Society for Agricultural Machinery, 2019, 50(8): 166 - 175. http://www.j-csam.org/jcsam/ch/reader/view_abstract.aspx? flag=1&file_no=20190819&journal_id=jcsam. DOI: 10.6041/j.issn.1000-1298.2019.08.019. (in Chinese)

[7] 宫晨,李新武,吴文瑾. 基于大数据分析框架的生态脆弱型人地系统模式研究[J]. 遥感技术与应用,2020,35(5):1187 - 1196.
GONG Chen, LI Xinwu, WU Wenjin. Study on the model of ecological vulnerable human - land system based on big data analysis framework[J]. Remote Sensing Technology and Application, 2020, 35(5): 1187 - 1196. (in Chinese)

[8] LIU Yanxu, FU Bojie, WANG Shuai, et al. Research progress of human - earth system dynamics based on spatial resilience theory [J]. Acta Geographica Sinica, 2020, 75(5): 891 - 903.

[9] 陈元鹏,任佳,王力. 基于多源遥感数据的生态保护修复项目区监测方法评述[J]. 生态学报,2019,39(23):8789 - 8797.
CHEN Yuanpeng, REN Jia, WANG Li. A review of monitoring methods for ecological conservation and restoration project area based on multi-source remote sensing data [J]. Acta Ecologica Sinica, 2019, 39(23): 8789 - 8797. (in Chinese)

[10] 申文明,孙中平,张雪,等. 生态环境移动数据采集系统研究与实现[J]. 生态学报,2013,33(24):7846 - 7852.
SHEN Wenming, SUN Zhongping, ZHANG Xue, et al. Research and implementation of mobile data acquisition system for ecological environment [J]. Acta Ecologica Sinica, 2013, 33(24): 7846 - 7852. (in Chinese)

[11] 唐秀美,郝星耀,潘瑜春,等. 基于生态需求评价的北京市生态区位划分研究[J/OL]. 农业机械学报,2016,47(1):170 - 176.
TANG Xiumei, HAO Xingyao, PAN Yuchun, et al. Ecological regionalization based on ecological demanding evaluation in Beijing City[J/OL]. Transactions of the Chinese Society for Agricultural Machinery, 2016, 47(1): 170 - 176. http://www.j-csam.org/jcsam/ch/reader/view_abstract.aspx? flag=1&file_no=20160122&journal_id=jcsam. DOI: 10.6041/j.issn.1000-1298.2016.01.022. (in Chinese)

[12] 熊丽君,袁明珠,吴建强. 大数据技术在生态环境领域的应用综述[J]. 生态环境学报,2019,28(12):2454 - 2463.
XIONG Lijun, YUAN Mingzhu, WU Jianqiang. A review of the application of big data technology in the field of ecological environment [J]. Journal of Ecology and Environment, 2019, 28(12): 2454 - 2463. (in Chinese)

[13] CRAVERO A, SALDANA O, ESPINOSA R, et al. Big data architecture for water resources management: a systematic mapping study[J]. IEEE Latin America Transactions, 2018, 16(3): 902 - 908.

[14] 张慧萍,陈志泊. 分布式数据库技术在数字化森林生态站的应用[J]. 北京林业大学学报,2007,29(3):131 - 135.
ZHANG Huiping, CHEN Zhibo. Application of distributed database technology in digital forest ecological station [J]. Journal of Beijing Forestry University, 2007, 29(3): 131 - 135. (in Chinese)

[15] 杨铭伦,张文革,于新文,等. 基于物联网大数据云平台的西天山森林生态站数据管理系统[J]. 林业科技通讯,2020(8):14 - 16.
YANG Minglun, ZHANG Wenge, YU Xinwen, et al. Data management system of forest ecological station in West Tianshan Mountains based on big data cloud platform of Internet of things [J]. Bulletin of Forestry Science and Technology, 2020(8): 14 - 16. (in Chinese)

[16] Theapache software foundation. Apache hadoop documentation[EB/OL]. 2020. http://hadoop.apache.org.

[17] ZHU Nan, LIU Xue, LIU Jie, et al. Towards a cost-efficient mapReduce: mitigating power peaks for Hadoop clusters [J]. Tsinghua Science and Technology, 2014, 19(1): 24 - 32.

[18] VORA M N. Hadoop-HBase for large-scale data [C]// Proc. of the 2011 Int'l Conf. on Computer Science and Network Technology. Piscataway: IEEE, 2011: 24 - 26.

[19] 李涛,冯仲科,孙素芬,等. 基于 Hadoop 的气象大数据分析 GIS 平台设计与试验[J/OL]. 农业机械学报,2019,50(1):180 - 188.
LI Tao, FENG Zhongke, SUN Sufen, et al. Design and experiment of GIS platform for meteorological big data analysis based on Hadoop [J/OL]. Transactions of the Chinese Society for Agricultural Machinery, 2019, 50(1): 180 - 188. http://www.j-csam.org/jcsam/ch/reader/view_abstract.aspx? flag=1&file_no=20190119&journal_id=jcsam. DOI: 10.6041/j.issn.1000-1298.2019.01.019. (in Chinese)

[20] 吴庭天,田蜜,陈宗铸,等. 基于 Hadoop 的森林资源信息平台研究[J]. 热带林业,2019,47(1):43 - 47.
WU Tingtian, TIAN Mi, CHEN Zongzhu, et al. Research on forest resource information platform based on Hadoop [J]. Tropical Forestry, 2019, 47(1): 43 - 47. (in Chinese)

[21] DONG B, QIU J, ZHENG Q, et al. A novel approach to improving the efficiency of storing and accessing small files on Hadoop: a case study by PowerPoint files [C]// SCC 2010: Proceedings of the 7th IEEE International Conference on Services Computing. Piscataway: IEEE Press, 2010: 65 - 72.

[22] 李三淼,李龙澍. Hadoop 中处理小文件的四种方法的性能分析[J]. 计算机工程与应用,2016,52(9):44 - 49.
LI Sanmiao, LI Longshu. Performance analysis of four methods for processing small files in Hadoop [J]. Computer Engineering and Applications, 2016, 52(9): 44 - 49. (in Chinese)

[23] 秦加伟,刘辉,方木云. 大数据平台下基于类型的小文件合并方法[J]. 软件工程,2020,23(10):12 - 14,11.
QIN Jiawei, LIU Hui, FANG Muyun. Small file merging method based on type in big data platform[J]. Software Engineering, 2020, 23(10): 12 - 14, 11. (in Chinese)

[24] 许鑫,时雷,何龙,等. 基于 NoSQL 数据库的农田物联网云存储系统设计与实现[J]. 农业工程学报,2019,35(1):172 - 179.
XU Xin, SHI Lei, HE Long, et al. Design and implementation of cloud storage system for farmland Internet of things based on NoSQL database [J]. Transactions of the CSAE, 2019, 35(1): 172 - 179. (in Chinese)

[25] 戴建旺,白晓飞. 第二次全国土地调查国家级数据库管理系统关键技术研究[J]. 中国土地科学,2010,24(6):74 - 80.
DAI Jianwang, BAI Xiaofei. Research on key technologies of national database management system for the second national land survey[J]. China Land Science, 2010, 24(6): 74 - 80. (in Chinese)

[26] PAPADOPOULOS A, KATSAROS D. A-Tree: distributed indexing of multi-dimensional data for cloud computing environments [C]// Proc. of the 3rd IEEE Int'l Conf. on Cloud Computing Technology and Science. Washington: IEEE Computer Society, 2011: 407 - 414.

(下转第 212 页)