Journal of Computer Applications
计算机应用,2015,35(7):1849 - 1853
ISSN 1001-9081
CODEN JYIIDU
2015-07-10
http://www.joca.cn

文章编号:1001-9081(2015)07-1849-05
doi:10.11772/j.issn.1001-9081.2015.07.1849

基于 HBase 的海量地形数据存储

李振举[1*],李学军[1],谢剑薇[1],李雁南[2]
(1.装备学院 信息装备系,北京 101416; 2.96275 部队,河南 洛阳 471003)
(*通信作者电子邮箱 belovings@163.com)

摘 要:随着遥感技术的发展,遥感数据的类型和量级发生了巨大变化,对于传统的存储方法产生了挑战。针对 HBase 中海量地形数据管理效率不高的问题,提出一种四叉树-Hilbert 相结合的索引设计方法。首先,对传统地形数据管理方式和基于 HBase 的数据存储国内外研究现状进行了综述;然后,在基于四叉树对全球数据进行组织的基础上,提出了四叉树和 Hilbert 编码相结合的设计思想;其次,设计了根据经纬度求地形数据的行列号和根据行列号计算 Hilbert 编码的算法;最后,对设计的索引的物理存储结构进行了设计。实验结果表明,利用设计的索引进行海量地形数据入库,数据入库速度与单机情况相比,提高了 63.79% ~78.45%;在地形数据的范围查询中,设计的索引与传统的行序索引相比,查询时间降低了 16.13% ~39.68%。查询速度最低为 14.71 MB/s,可以满足地形数据显示的要求。

关键词:HBase;地形数据;云存储;四叉树-Hilbert 索引;三维地形显示
中图分类号:TP391.1 **文献标志码**:A

Massive terrain data storage based on HBase

LI Zhenju[1*], LI Xuejun[1], XIE Jianwei[1], LI Yannan[2]
(1. *Department of Information Equipment, Equipment Academy, Beijing* 101416, *China*;
2. 96275 *Troops, Luoyang Henan* 471003, *China*)

Abstract: With the development of remote sensing technology, the data type and data volume of remote sensing data has increased dramatically in the past decades which is a challenge for traditional storage mode. A combination of quadtree and Hilbert spatial index was proposed in this paper to solve the the low storage efficiency in HBase data storage. Firstly, the research status of traditional terrain data storage and data storage based on HBase was reviewed. Secondly the design idea on the combination of quadtree and Hilbert spatial index based on managing global data was proposed. Thirdly the algorithm for calculating the row and column number based on the longitude and latitude of terrain data, and the algorithm for calculating the final Hilbert code was designed. Finally, the physical storage infrastructure for the index was designed. The experimental results illustrate that the data loading speed in Hadoop cluster improved 63.79% -78.45% compared to the single computer, the query time decreases by 16.13% - 39.68% compared to the traditional row key index, the query speed is at least 14.71 MB/s which can meet the requirements of terrain data visualization.

Key words: HBase; terrain data; cloud storage; quadtree-Hilbert index; 3-dimensional terrain data visualization

0 引言

地形数据是遥感数据的一个重要类型,作用是表征地形起伏特征,是进行遥感数据三维可视化的基础,可以作为遥感应用的地形和地貌显示、虚拟城市建设的数据源。在地理信息系统(Geographic Information System, GIS)中,地形数据主要包括数字高程模型(Digital Elevation Model, DEM)和数字正射影像(Digital Orthophoto Map, DOM),通过 DEM 和影像数据的共同使用,可以进行三维场景的构造。目前地形数据的特点是几何复杂度、纹理数据庞大,在进行存储时,一般采用金字塔模型进行分级存储,将分辨率高、数据量大的图像放在底层。近年来,随着遥感技术的发展,对地观测技术的进步,可以获得多分辨率、多光谱、多时相和多模式的遥感影像和高程数据。遥感数据的分辨率越来越高,数据量级越来越大,遥感地形数据满足大数据的典型特征[1],这对于传统的数据存储方式提出了新的挑战。

HBase 是开源云平台 Hadoop 的一个重要组成部分,设计灵感来自谷歌文件系统 Bigtable[2],不同于传统的基于行存储的关系数据库管理系统(Relational Database Management System, RDBMS),在磁盘上使用基于列的存储格式。HBase 中的数据表理论上可以拥有数百万列元素,在普通微机组成的集群上即可处理上亿行数据[3],可以解决 Hadoop 分布式文件系统(Hadoop Distributed File System, HDFS)存在的不支持随机访问和实时性不高的问题。

1 相关研究

1.1 传统的地形数据管理方式

传统的地形数据存储方式主要包括两类:基于文件系统的方式和基于关系型数据库的方式[4]。文件管理需要对影像和 DEM 数据进行物理和逻辑分块,而后建立索引进行查

收稿日期:2015-02-05;修回日期:2015-04-05。

作者简介:李振举(1987 -),男,河南安阳人,助理工程师,博士研究生,CCF 会员,主要研究方向:云计算、遥感数据管理; 李学军(1967 -),男,湖北监利人,教授,博士,CCF 会员,主要研究方向:通信与信息系统、计算机图形学; 谢剑薇(1970 -),女,安徽黄山人,副教授,硕士,主要研究方向:遥感数据管理; 李雁南(1988 -),男,河南叶县人,助理工程师,硕士,主要研究方向:遥感数据管理。

询。基于关系型数据库的存储方式主要使用大型数据库进行管理,将 DEM 数据转换成高程比例一致的灰度图后和影像数据统一进行管理[5]。现有的地形数据管理系统中,地理信息系统输入、处理和输出工具(A Geographic Input, Processing and Output Tools for Geographic Information System, ArcGIS)可以采用两种方式进行存储,超图(SuperMap)采用了文件系统的管理方式,Oracle Spatial、地理信息系统输入、处理和输出工具的空间数据引擎(A Geographic Input, Processing and Output Tools for Spatial Database Engine, ArcSDE)和天地图系统采用关系型数据库的方式进行管理。其中,基于文件系统的管理方式依托计算机的本地磁盘进行存储,按照特定的数据组织方式进行管理,优点是管理方便,访问方式和操作系统访问普通文件的方式相同,不足之处是维护成本较高,同时存在一定的安全性问题;基于关系型数据库的地形数据存储依托成熟的数据库技术进行,以独立记录的方式对每个地形数据进行管理,检索按照瓦片划分的层级和金字塔结构分辨存储的方式。优点是安全性较高,不足之处是索引构建效率不高,对网络速度要求较高,一定程度上会导致瓦片的查询和处理速度下降。传统的地形数据管理方式在地形数据量过大时会出现磁盘存储碎片化现象,进而导致 I/O 性能降低,而且数据可迁移性差,数据备份和恢复比较复杂[6],使用 HBase 进行数据存储可以较好地解决这类问题。

1.2 基于 HBase 的数据存储研究现状

HBase 是谷歌非结构化数据库 BigTable 的开源实现,已经在海量网页信息的管理中得到应用。考虑到三维显示的需要,在进行存储时,地形数据一般都比较小,使用 Hadoop 分布式文件系统(HDFS)进行存储会产生小文件问题,降低整个云集群的效率。采用 HBase 进行瓦片数据存储,既可以避免小文件问题的产生,又可以解决 HDFS 不支持并发读写的问题。

HBase 系统架构如图 1 所示[7],分为用户层、服务层和支撑层。其中:支撑层是系统运行的基础,分布式文件系统作为底层的数据存储系统保证数据的可靠性,分布式锁系统 Zookeeper 用来保证数据的同步服务和配置维护;主服务器是系统的核心部分,负责处理来自客户端的连接请求,维护与子表服务器的连接和子表的分配。元数据表类似于数据字典,用来维护子表的记录范围;子表服务器负责处理所有操作的读写操作,随着数据量的急剧增加,可以通过增加子表服务器的数量动态扩容,客户端对数据的操作通过直接和子表服务器交互实现。

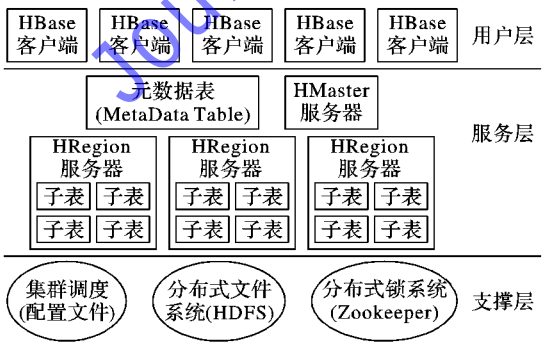

图 1 HBase 系统体系架构

通过查阅相关文献,现阶段使用 HBase 进行数据存储的研究如表 1 所示。其中:RDF(Resource Description Framework)为海量资源描述框架。

表 1 基于 HBase 的存储研究现状

存储数据类型	解决的主要问题	文献
地图瓦片数据	提出了一种地图瓦片数据缓存方案,解决了单机瓦片缓存效率不高的问题	文献[8]
海量图片数据	提出了一种海量图片数据存储方案,解决了 HBase 无校验码和键值对字节数组未对齐问题,在城市监控系统中得到应用	文献[9]
无线传感信息	提出了两层存储架构,可以满足大规模传感器的存储需要	文献[10-11]
海量影像数据	提出了适用于海量影像数据分布式存储的索引结构 P2H,可以进行影像数据的存储和快速查询	文献[12]
矢量空间数据	设计了基于 HBase 的矢量空间数据存储模型和空间索引方法	文献[13]
RDF 数据	提出一种基于基于 HBase 的 RDF 数据存储模型,实现了基于该模型的 8 种查询算法并进行了可行性验证	文献[14]
海量空间数据	提出一种 HGrid 数据模型,设计了基于四叉树和网格索引的双重索引结构	文献[15]

通过表 1 可以看出,HBase 适于存储海量数据,尤其对于海量空间数据的存储有较好的支持能力。

2 基于 HBase 的地形数据索引及存储结构设计

2.1 基于四叉树的全球地形数据组织

进行地形数据存储首先需要对地形数据的逻辑存储结构进行设计。本文采用四叉树结构对全球数据进行存储。四叉树结构的核心思想是对一个区域反复进行 4 等分,直至满足用户的需求。本文中为了更好地进行全球数据的三维显示和数据处理,四叉树结构的第一级分为东半球和西半球两个部分,从第 3 层开始,每一级的存储按照正常的四叉树结构进行存储。全球地形数据的四叉树存储结构如图 1 所示。

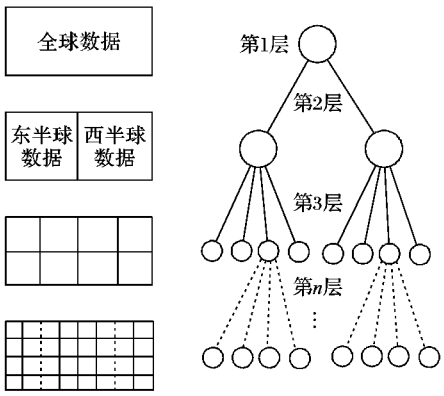

图 2 基于四叉树的全球地形数据划分

采用这种逻辑结构可以对多种类型的地形数据进行表示,同时数据的读写操作也比较方便,可以根据数据的分辨率和坐标系直接定位四叉树节点,减少数据查询的时间,提高数据访问速度。

2.2 基于四叉树-Hilbert 方法的行键索引设计

空间索引是提高地形数据存储和数据检索效率的重要方式。常见的索引结构包括 R 树、四叉树等。本文在全球数据四叉树划分结构的基础上提出了一种四叉数和 Hibernate 编码方式相结合的索引结构。

在进行索引结构设计之前,首先对 HBase 的物理存储结

构进行研究。HBase 中基本存储单元是 CELL,可以视为行键 RowKey、列族 ColumnFamily 和时间版本识别符 Version 的集合,即:

$$CELL = \{RowKey, ColumnFamily, Version\} \quad (1)$$

其中:行键 RowKey 是确定行的标识符,列族 ColumnFamily 需要预先定义,时间版本 Version 用于表征单元 CELL 的时间版本特性,是 HBase 进行驻留时间(Time-To-Live, TTL)操作的基础。

在空间数据查询时,按照行键、列族、列限定符和时间戳这样的顺序进行定位,按照它们在键中的排列顺序,从左至右查询数据的性能在降低,因此本文设计的行键是查询空间信息的关键,由行键快速定位一行。图 3 展示了 HBase 键值对中不同字段的排列。

键				值
行键	列族	列限定符	时间戳	单元值

图 3　键值对的不同字段

HBase 把它的文件透明地存储到分布式文件系统(HDFS)中,HDFS 以块(Block)为单位对数据进行管理,块的大小可以根据输入数据的特征进行设置,默认值为 64 MB,可以由用户定义。在建立四叉树模型时,分辨率自底层向上逐渐缩小,这样使得剖分后不同层的空间数据块大小不同,有些层的空间数据块大于 Block 大小,有些层的空间数据块小于 Block 大小,因此对空间数据金字塔中每个空间数据块进行存储时,需要将小的空间数据块合并存储在 Block,大的空间数据块分 Block 进行存储。

本文设计的空间索引机制,在四叉树金字塔模型构建的基础上,对每层空间数据再分块,每一层用 Hilbert 曲线进行填充,首先由经纬度计算空间数据块的行列号,再由行列号计算其 Hilbert 编码,将东西半球的空间数据按照行列号进行标识,东西半球的空间数据块可以得到各自唯一的 Hilbert 排列码,就得到了该层空间数据块的四叉树- Hilbert 索引,编码规则为:

$$RowKey = f(k, hilbert, row, column) \quad (2)$$

其中:k 为瓦片数据所在的层号,*hilbert* 为该瓦片数据的希尔伯特码,*row* 为所在的行,*column* 为所在的列,f 为编码规则。下面以实例进行说明。如图 4 所示,以四叉树金字塔逻辑结构的第 2 到第 4 层为例给出索引设计的实例。

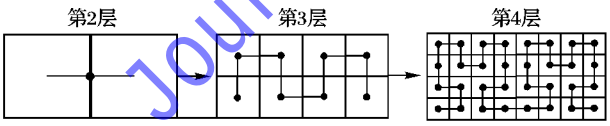

图 4　四叉树与 Hilbert 模型编码示例

首先是由经纬度计算空间数据块的行列号方法。算法思想如下。

假设地形瓦片数据的经纬度为(x,y),层号为k,第 1 层空间数据像元格网的经纬度尺寸为δ,则第k层空间数据格网的经纬度尺寸为$\delta/2^k$。

对于第k层金字塔,已知某一点的经纬度坐标(x,y),瓦片数据的行列号(R,C)可以由如下公式计算得到。式中对R和C的计算结果向下取整,使计算结果不超出行列号范围。

$$\begin{cases} R = \lfloor (x + 90)/\delta \rfloor \\ C = \lfloor (y + 180)/\delta \rfloor \end{cases} \quad (3)$$

其次是根据行列号计算 Hilbert 编码。Hilbert 编码其实就是计算该空间数据块之前 Hilbert 曲线上的空间数据块数。把全球空间数据每层分为东西两半球,对每个半球的空间数据分块计算其 Hilbert 编码,东半球计算的结果就是其包括的空间数据块的 Hilbert 编码,西半球的 Hilbert 编码是其计算结果加上该层东半球空间数据块的总数。算法思想如下。

根据瓦片所在层号k,经纬度坐标(x,y),计算得到瓦片的行列编码。东西半球的 Hilbert 编码只有列的区别,因此只需计算一个半球的列对应的编码即可。每层空间数据划分的像元阵列为 8 × 8,第k层瓦片位置为$2^{k-4} \times 2^{k-3}$,因此根据计算出的列号和2^{k-4}比较即可确定该瓦片所在的东西半球。如果在东半球,则直接列用行列号计算 Hilbert 编码即可;如果在西半球则将$C = c - 2^{k-4}$作为列号,以R为行号计算该瓦片数据的 Hilbert 编码。最后得到计算结果。算法的最终输出为该坐标的 Hilbert 值。

HBase 表中行键设计为采用四叉树- Hilbert 索引编码的形式,因此 Block 的索引信息只用记录起始行键值和结束行键值。使用这种索引管理空间数据,一次读取就可以得到 Block 中存储的所有空间数据的信息,并且保存该 Block 下的索引结构,缓存到内存。这样在下次调用该 Block 数据时,就可以跳过名称节点,直接到该 Block 下的索引结构对应的数据节点访问数据。这样就减少了与名称节点的交互,避免造成性能瓶颈。

2.3　物理存储结构设计

HBase 采用键值对进行存储,键值对存储是一个简单的存储模型,键与值之间是一种映射的关系。根据分布式领域一致性-可用性-分区容错性(Consistency-Availability-Partition tolerance, CAP)[16] 理论,在三个指标中,最多只能同时满足两个,键值模型选择了可用性和分区容错性,一致性方面性能相对较弱。

HBase 表中的数据以列为单位聚合数据,存储时按列顺序地存入磁盘,一行中列族的数据需要物理地存储到一起,四叉树结构中位于同一层级的同一节点的 DEM 和 DOM 数据,由于它们空间位置相同,因此可以用一个列族里面存放它们,进而保证空间位置上相邻的空间数据物理位置也是相邻的。这样在访问浏览空间数据时,可以做到相邻的空间数据块快速被调用,充分体现列存储节省 I/O 带宽的优势。

空间数据存储结构选择键值对存储的形式,四叉树金字塔模型中的每层空间数据存储到不同的表,由于四叉树金字塔模型中的一个四叉树节点存储的空间数据包括地理坐标范围匹配的 DEM 数据和 DOM 数据,因此它们的行键值相同,地理空间范围元数据信息一致,将它们存储到一行里。根据相关研究成果,在 HBase 中建表应该满足以下原则[15]:在 HBase 中建立的 RowKey 名称和列族名称尽可能要短,每一行数据中存储的数据尽可能要少,列族的数量最好只有一个,列的数量也应该有所限制,过多的列也会降低存储效率,采用压缩技术也可以提高数据传输的性能。基于此,设计的海量地形数据物理存储结构如图 5 所示。

在图 5 中,行键是四叉树金字塔模型中节点空间数据块的四叉树- Hilbert 索引编码(对应图 3 中的行键),D 代表空间数据列族,包括 DEM、DOM、GEO、MET 等列,只设计一个列族可以保证海量数据入库和查询时的效率(对应图 3 中的列

族),其中:DEM 和 DOM 列分别负责存储 DEM 数据和 DOM 数据(对应图 3 中的单元值);GEO 列用来存储地形数据的几何位置信息,包括坐标系统、行列数、格网间隔和经纬度等[17];MET 列用来存储空间数据分辨率、数据层级和显示精度等元数据信息;t_1、t_2 是记录单元的时间戳,表示空间数据块在同一区域的不同时间成像标签,由系统自动生成,是数据的时间版本标识(对应图 3 中的时间戳)。下面给出 HBase 中地形数据存储的一个实例,如表 2 所示。用户需要读取数据,只需通过行键即可调用相应区域的不同时刻,不同元数据信息的 DEM 和 DOM 数据。

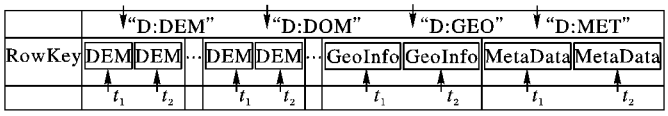

图 5 海量地形数据物理存储结构

表 2 基于 HBase 的海量地形数据存储实例

行键	D			
	DEM	DOM	GEO	MET
Row_1	DEM_v1	DOM_v1	GEO_v1	MET_v1
Row_2	DEM_v2	DOM_v2	GEO_v2	MET_v2
Row_3	DEM_v3	DOM_v3	GEO_v3	MET_v3

3 实验和结果

3.1 实验环境和数据

HBase 采用 0.90.5 版本,Hadoop 采用 0.20.205 版本,ZooKeeper 采用 3.3.2 版本,JDK 版本为 OpenJDK 1.6,操作系统为 Kylin Linux 3.2.1,PC 机内存 2.0 GB,主频 3.0 GHz,共有 6 台相同配置的机器组成 HBase 集群。实验数据来自课题组数字地球项目的海量地形数据。

3.2 地形数据入库性能测试

本实验目的是对空间数据的入库性能进行测试。采用实验室已有的 DOM 数据作为实验数据,包括多种分辨率的 DOM 数据,大小分别为 23.2 MB、58.0 MB、106.4 MB、420.6 MB、970.4 MB、2.1 GB、2.8 GB、3.3 GB,在 HBase 单机和 HBase 集群条件下分别对不同大小的 DOM 数据进行入库,集群条件下采用并行入库的方式,首先将空间数据分块上传到 HDFS 中,然后利用 Hadoop 编程模型将空间数据并行入库。单机环境和集群环境下空间数据的导入速度的对比如图 6 所示。根据实验结果可以看出,随着数据量的增加,单机环境下输入导入速度比较稳定,集群环境下的数据导入随着数据量的增加速度不断变快。

3.3 空间范围查询结果分析

本实验的目的是对设计的存储系统进行查询性能的测试。选取最底层的空间数据作为查询对象,在空间数据经纬度范围(-180°,180°)和(-90°,90°)上,利用随机函数每次产生比例相同的 6 对空间范围(比例依次取值为 1%、5%、15%、25%、35%、45%),记录每个空间范围查询的执行时间。查询时间 t_{query} 由两部分组成:空间数据块索引查询的时间 t_{index} 和空间数据块从 HBase 中显示到三维浏览客户端的时间 t_{show},每次查询的总时间为 $t_{query} = t_{index} + t_{show}$。按照比例计算查询总时间的均值,得到查询结果如图 7 所示。

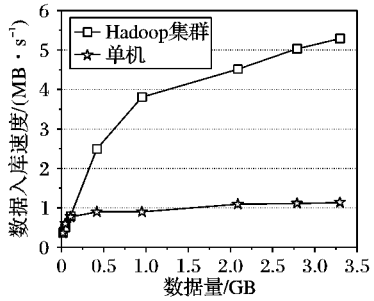

图 6 不同环境下空间数据的导入速度对比

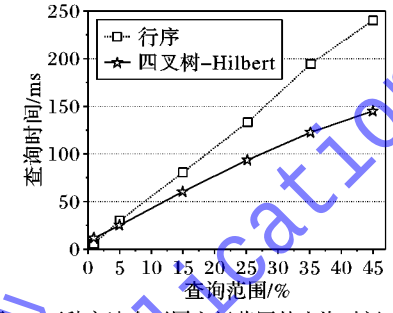

图 7 两种方法在不同空间范围的查询时间对比

从图 7 中可以看出,当查询的空间范围为 1% 时,按照行序组织索引查询性能更佳,这是由于空间数据量较少,t_{index} 是查询时间的主要影响因素,四叉树-Hilbert 索引结构的编码计算量比行序编码大,因此当空间数据量较小时本文设计的四叉树-Hilbert 索引效率较低。随着查询的空间范围增大和空间数据量增加,t_{show} 成为查询时间的主要影响因素,本文设计的四叉树-Hilbert 索引结构优势越来越明显。该实验证明本文设计的索引和物理存储模型能够满足海量空间数据快速查询的要求。

3.4 范围查询结果测试

本实验的目的是测试设计的存储结构对三维浏览速度的实施浏览能力的支持。本文采用 256×256 进行空间数据分块,每个节点由 DEM 和 DOM 数据组成。分别选取四叉树结构中 512,1024,2048 个节点进行查询性能的测试,测试 10 次取平均值,实验结果如表 3 所示。

表 3 三维浏览的查询测试

查询节点数	空间数据量/MB	查询时间/ms	查询速度/(MB·s⁻¹)
512	224	12960	17.28
1024	448	28976	15.46
2048	896	60889	14.71

通过测试,表 3 中选取不同查询节点的查询速度最低 14.71 MB/s,系统查询性能满足三维地形实时浏览需求,能够支持全球三维地形实时浏览。

4 结语

云计算技术的出现为海量地形数据的存储提供了一种新的解决方案。本文利用 Hbase 解决海量地形数据的存储问题,通过设计四叉树-Hilbert 相结合的索引结构提高数据存储的效率,结合三维地形数据的显示需求和金字塔原理,设计了地形数据的物理存储结构,最后通过实验证明了该文的设计满足地形数据的存储要求。下一步研究方向是在基于 HBase 的地形数据存储的基础上,构建云计算环境下的数字

地球三维显示和数据处理系统。

参考文献:

[1] CHEN C L P, ZHANG C-Y. Data-intensive applications, challenges, techniques and technologies: a survey on big data [J]. Information Sciences, 2014, 275: 314 - 347.

[2] CHANG F D J, DEAN J, GHEMAWAT S, *et al.* Bigtable: a distributed storage system for structured data [J]. ACM Transactions on Computer Systems, 2008, 26(2): Article No. 4.

[3] O'DRISCOLL A, DAUGELAITE J, SLEATOR R D. 'Big data', Hadoop and cloud computing in genomics [J]. Journal of Biomedical Informatics, 2013, 46(5): 774 - 781.

[4] SHANG X. Design and implementation of WebGIS massive tile data management engine [D]. Jinhua: Zhejiang Normal University, 2012. (商秀玉. WebGIS 海量瓦片数据管理引擎的设计与实现[D]. 金华: 浙江师范大学, 2012.)

[5] YU W. Digital earth oriented key technologies for 3D landscape development [D]. Beijing: University of Chinese Academy of Sciences, 2006. (于文洋. 面向数字地球的三维景观构造关键技术研究[D]. 北京: 中国科学院研究生院, 2006.)

[6] LUO Z, LI X. High performance tile map service based on database storage scheme [J]. Geography and Geo-Information Science, 2013, 29(3): 48 - 51. (罗智勇, 黎小东. 基于数据库存储方案的高性能瓦片地图服务研究[J]. 地理与地理信息科学, 2013, 29(3): 48 - 51.)

[7] ZHAO S. Typical Hadoop cloud computing [M]. Beijing: Publishing House of Electronics Industry, 2013: 362. (赵书兰. 典型 Hadoop 云计算[M]. 北京: 电子工业出版社, 2013: 362.)

[8] CHEN H, LI Y, ZHU M. Research on tile buffer policy supporting large number of concurrent [J]. Computer Engineering & Science, 2012, 34(12): 144 - 149. (陈华, 李艳明, 朱美正. 一种支持大量并发用户的瓦片缓存方案研究[J]. 计算机工程与科学, 2012, 34(12): 144 - 149.)

[9] ZHU X, ZHAO Z. Massive image storage based on HBase [J]. China CIO News, 2013(8): 22 - 24. (朱晓丽, 赵志刚. 一种基于 HBase 的海量图片存储技术[J]. 信息系统工程, 2013(8): 22 - 24.)

[10] ZHOU L, CHEN Q. HBase-based storage system for wireless sensor information of agriculture [J]. Computer Systems & Applications, 2012, 21(8): 6 - 9. (周利珍, 陈庆奎. 基于 HBase 的农业无线传感信息系统[J]. 计算机系统应用, 2012, 21(8): 6 - 9.)

[11] CHEN Q, ZHOU L. HBase-based storage system for large-scale data in wireless sensor network [J]. Journal of Computer Applications, 2012, 32(7): 1920 - 1923. (陈庆奎, 周利珍. 基于 HBase 的大规模无线传感网络数据存储系统[J]. 计算机应用, 2012, 32(7): 1920 - 1923.)

[12] HUO S. Research on the key techniques of massive image data management based on Hadoop [D]. Changsha: National University of Defense Technology, 2010. (霍树民. 基于 Hadoop 的海量遥感数据管理关键技术研究[D]. 长沙: 国防科学技术大学, 2010.)

[13] FAN J, LONG M, XIONG W. Research of vector spatial data distributed storage based on HBase [J]. Geography and Geo-Information Science, 2012, 28(5): 39 - 41. (范建永, 龙明, 熊伟. 基于 HBase 的矢量空间数据分布式存储研究[J]. 地理与地理信息科学, 2012, 28(5): 39 - 41.)

[14] PAPAILIOU N, TSOUMAKOS D, KONSTANTINOU I, *et al.* H2RDF +: an efficient data management system for big RDF graphs [C]// Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. New York: ACM, 2014: 909 - 912.

[15] HAN D, STROULIA E. HGrid: a data model for large geospatial data sets in HBase [C]// Proceedings of the 2013 IEEE Sixth International Conference on Cloud Computing. Piscataway: IEEE, 2013: 910 - 917.

[16] DeCANDIA G, HASTORUN D, JAMPANI M, *et al.* Dynamo: Amazon's highly available key-value store [C]// ACM SIGOPS Operating Systems Review, 2007, 41(6): 205 - 220.

[17] TANG G, ZHANG Q, WANG G. Research of storage and schedule of huge terrain based on 3D GIS [J]. Science of Surveying and Mapping, 2008, 33(3): 110 - 120. (唐桂文, 张庆娟, 王功明, 等. 基于三维 GIS 的海量地形数据存储和调度的研究[J]. 测绘科学, 2008, 33(3): 110 - 120.)

(上接第 1842 页)

[6] LI J, JING Y, ZHANG C, *et al.* A trust model based on similarity-weighted recommendation for P2P environments [J]. Journal of Software, 2007, 18(1): 157 - 167. (李景涛, 荊一楠, 张根度, 等. 基于相似度加权推荐的 P2P 环境下的信任模型[J]. 软件学报, 2007, 18(1): 157 - 167.)

[7] ZHANG Q, ZHANG X, WEN X, *et al.* Construction of peer-to-peer multiple-grain trust model [J]. Journal of Software, 2006, 17(1): 96 - 107. (张骞, 张霞, 文学志, 等. Peer-to-Peer 环境下多粒度 Trust 模型构造[J]. 软件学报, 2006, 17(1): 96 - 107.

[8] JIANG S, LI J. A reputation-based trust mechanism for P2P e-commerce systems [J]. Journal of Software, 2007, 18(10): 2551 - 2563. (姜守旭, 李建中. 一种 P2P 电子商务系统中基于信誉的信任机制[J]. 软件学报, 2007, 18(10): 2551 - 2563.)

[9] CHAISIRI S, LEE B, NIYATO D. Optimization of resource provisioning cost in cloud computing [J]. IEEE Transactions on Services Computing, 2011, 99(7): 1 - 32.

[10] SHI X, XU K. Utility maximization model of virtual machine scheduling in cloud environment [J]. Chinese Journal of Computers, 2013, 36(2): 252 - 261. (师雪霖, 徐恪. 云虚拟机资源分配的效用最大化模型[J]. 计算机学报, 2013, 36(2): 252 - 261.)

[11] LIN J, NI H, SUN P, *et al.* Adaptive resource allocation based on neural network PID control [J]. Journal of Xi'an Jiaotong University, 2013, 47(4): 112 - 117. (林军, 倪宏, 孙鹏, 等. 一种采用神经网络 PID 控制的自适应资源分配方法[J]. 西安交通大学学报, 2013, 47(4): 112 - 117.)

[12] YANG W, LI J, WANG K. Domain-adaptive service evaluation model [J]. Chinese Journal of Computers, 2005, 28(4): 514 - 523. (杨文军, 李涓子, 王克宏. 领域自适应的 Web 服务评价模型[J]. 计算机学报, 2005, 28(4): 514 - 523.)

[13] TAN Y, ZENG G, WANG W. Policy of energy optimal management for cloud computing platform with stochastic tasks [J]. Journal of Software, 2012, 23(2): 266 - 278. (谭一鸣, 曾国荪, 王伟. 随机任务在云计算平台中能耗的优化管理方法[J]. 软件学报, 2012, 23(2): 266 - 278.)