DOI:10.13203/j.whugis20180034 文章编号:1671-8860(2018)12-2295-08

基于 Spark 的分布式空间数据存储结构设计与实现

乐 鹏[1] 吴昭炎[1] 上官博屹[1]

1 武汉大学遥感信息工程学院,湖北 武汉,430079

摘 要:Apache Spark 分布式计算框架可用于空间大数据的管理与计算,为实现云 GIS 提供基础平台。针对 Apache Spark 的数据组织与计算模型,结合 Apache HBase 分布式数据库,从分布式 GIS 内核的理念出发,设计并实现了分布式空间数据存储结构与对象接口,并基于某国产 GIS 平台软件内核进行了实现。针对点、线、面数据的存储与查询,与传统空间数据库系统 PostGIS 进行了一系列对比实验,验证了提出的分布式空间数据存储架构的可行性与高效性。

关键词:Spark;云 GIS;分布式空间数据组织;分布式 GIS 内核;空间大数据

中图分类号:P208 **文献标志码**:A

近年来,随着传感器技术及对地观测技术的迅猛发展,地理空间大数据已经成为大数据的重要组成部分。空间信息系统管理和处理的数据量已经从 TB 级增加到 PB 乃至 EB 级。空间大数据在类型、采集速度、价值密度、准确性及变化性等方面的特点增加了地理空间数据管理与处理的复杂性,传统的数据管理系统和计算能力难以满足这些需求[1-2]。传统 GIS 空间数据存储管理或直接依赖于已有的数据库(如 PostGIS、Oracle Spatial),或在其上构建空间数据引擎中间件(如 ArcSDE 等)。但这些方案在分布式地理空间大数据管理与计算上存在不足。近年来,面向云环境的分布式空间数据组织管理已成为空间大数据管理的趋势[3-5]。

随着 GIS 数据走向云环境下的分布式存储,分布式计算框架 Apache Hadoop 及其改进版本 Spark、分布式文件系统 HDFS、分布式数据库 HBase 等云计算技术和软件设施的发展为分布式空间数据组织管理带来了前景[6-7]。其中 Spark 框架采用基于内存的分布式处理模式,展现了比 Hadoop 更好的性能和容错性,在此基础上开展高效的空间数据管理与计算研究显得十分必要[8]。

如何将传统 GIS 软件以较小代价迁移到以 Spark 为代表的分布式云环境下,对于云 GIS 平台软件的研制具有重大意义。目前已有部分研究开始着手 Spark 环境下的空间数据组织[9-10]。然而,现有的工作尚缺乏对已有 GIS 软件功能的复用,从头开始构建 GIS 功能的解决方案往往成本高昂、耗时耗力且易出错;其次,目前基于 Spark 的研究多采用分布式文件系统存储空间数据,而利用分布式数据库接入 Spark 处理的流程仍有待研究。采用分布式数据库 HBase,相较于分布式文件系统 HDFS 的优势在于:①在检索时可根据行键来缩小检索范围,不需要将数据文件全部读入;② HBase 数据库的创建、读取、更新、删除(CRUD)操作简便,支持增量更新,在更新时比 HDFS 分布式文件系统更便捷;③HBase 提供了版本控制功能,对于矢量数据的更新、回滚十分重要;④HBase 支持对属性数据的检索。

本文从分布式空间数据存储结构的设计与实现出发,提出外存 HBase 支持持久化存储、内存 Spark 支持计算的空间数据管理方法,探讨其中的分布式存储和分布式内存对象设计等关键问

收稿日期:2018-06-07

项目资助:国家重点研发计划(2017YFB0504103);国家自然科学基金(41722109);武汉黄鹤英才科技创新专项(2016);湖北省杰出青年自然科学基金(2018CFA053)。

第一作者:乐鹏,博士,教授,主要从事地理信息系统软件、网络地理信息与位置智能服务、空间数据库、高性能地理计算的研究。pyue_whu@163.com

题,设计了面向列存储的空间数据表结构和基于弹性分布式数据集(resilient distributed dataset,RDD)的分布式内存空间对象SpatialRDD,或称空间弹性数据集,实现了HBase数据表映射转换SpatialRDD对象进行操作的方法,并通过复用某国产GIS平台软件内核实现对SpatialRDD的操作,为存储和处理空间大数据提供了一种涵盖内外存设计的较为全面的解决方案。

1 分布式空间数据存储与计算架构

本文设计的分布式空间数据存储与计算架构如图1所示,从云计算资源层、空间数据外存层、空间数据内存计算层等多个层次进行划分,在兼顾内外存数据结构的同时,支持对传统GIS软件功能的复用。

云计算资源层作为整个框架的平台底层,为分布式空间数据存储架构提供硬件资源及资源管理,包括平台虚拟化、节点管理、资源分配等功能。该层可以采用公有云或私有云平台(如OpenStack等)。

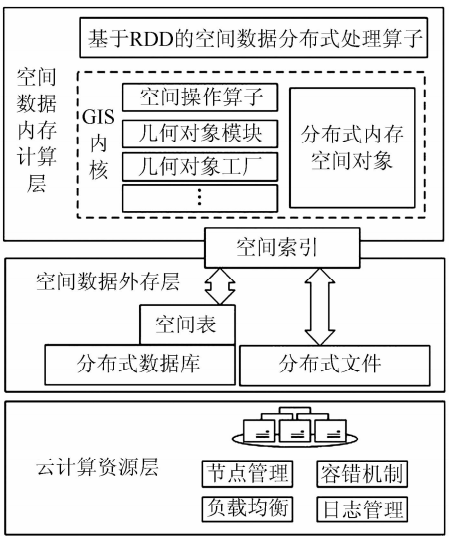

图1 分布式空间数据存储与计算架构

Fig. 1 Distributed Geospatial Data Storage and Computation Architecture

在外存层的设计上,可采用分布式数据库或文件系统,如HBase或HDFS。对于分布式数据库而言,可进一步设计空间表,增加空间数据与索引的字段。本文以国际开放地理空间信息联盟(Open Geospatial Consortium,OGC)的简单要素标准(Simple Features)为基础[11],将空间实体分为点(Point)、线(LineString)、面(Polygon)、点集合(MultiPoint)、线集合(MultiLineString)、面集合(MultiPolygon)6种类型,以WKT/WKB(well-known text/well-known binary)的形式存储在表字段中。对于海量的空间数据存储,通常需要增加空间索引字段以提升空间数据的检索效率,常用的空间索引类型有R树索引、四叉树索引、网格索引等。对于分布式处理架构中的空间数据存储,一般可采用以下几种方式建立和存储空间索引。①一次性构建,并以文件方式持久化存储在分布式文件系统中;②每次从分布式存储系统中读取空间数据后,动态构建空间索引并缓存于内存中,以供多次查询使用;③将空间索引编码后与空间数据一同存储在分布式数据库中。本文选用了第三种方法,在HBase中设计实现了一种顾及空间范围的变长GeoHash编码作为空间索引,后续也可以进一步探讨便于邻近分块存储的索引。在建立索引后,以空间索引GeoHash编码为键,空间数据及其属性为值,将空间信息及其索引以<键,值>的形式存储于空间数据表中,并在分布式数据库中持久化存储。

在内存计算层上,空间数据的组织结构应服从计算的需求,同时兼顾已有GIS软件功能的复用。该层包含了分布式GIS内核以及构建在内核之上的分布式空间处理算子。分布式GIS内核由传统的GIS内核以及分布式内存空间对象组成,而传统的GIS内核包括了几何对象工厂、集合对象模块和空间操作算子等。内核中也可加入符号化与空间转换等模块,提供空间数据的序列化、对象构建以及基本几何操作计算能力。传统GIS内核相对稳定,封装为相对独立的软件实体向外部提供接口,作为可重用的构件构造其他软件。在此基础上,扩展基于RDD的SpatialRDD模块,将几何对象模块映射到分布式内存中进行半持久化存储,使其支持分布式空间计算。在分布式GIS内核之上,设计分布式空间数据处理算子,采用基于构件技术的软件复用方法,调用GIS内核中的空间操作算子实现分布式空间数据的计算。

2 基于HBase的分布式空间数据存储设计

HBase可以管理超大规模的稀疏表,并提供了基于MapReduce的输入输出(I/O)接口[12]。本文基于这些优点设计了面向空间数据的HBase表结构,并实现了分布式的读取、写入接口来支持空间数据的读写。

2.1　空间数据存储结构设计

与传统的关系型数据库不同,HBase 是一种面向列存储的数据库。在 HBase 中,数据表由行和列构成,表的每一行具有唯一的行键(Rowkey)用于标识;而行中的列以列簇(ColumnFamily)的形式组织,同一列簇中的列都具有相同的前缀,如 Point:Attribute1 和 Point:Attribute2 均是列簇 Point 的成员。行与列的交叉是单元格(cell),它具有版本信息,默认为数据插入单元格时的时间戳(timestamp, Ts),因此 HBase 表中的数据由行键、列簇名、列名与时间戳唯一确定。

在 HBase 的表结构基础上,本文结合 OGC 的简单要素空间数据结构,设计了空间数据的存储表结构(表 1)。与传统的空间数据表类似,每种要素类对应一张表,对于不同的要素几何类型,定义不同的列簇。列簇中的列由要素的属性所构成,每个地理实体对应一个行键,由变长 GeoHash 索引和要素标识符 ID 组合构成。

表 1　基于 HBase 的空间数据存储表结构

Tab. 1　Table Structure for Geospatial Data Storage in HBase

行键	列簇			
	几何字段	属性 1	…	属性 n
GeoHash_1	Ts:几何体 1	Ts:属性值 1	…	…
GeoHash_2	Ts:几何体 2	Ts:属性值 2	…	…
⋮	⋮	⋮	⋮	⋮

2.2　空间数据索引及检索方法设计

GeoHash[13]编码(http://geohash.org/site/tips.html)是一种地理编码,它将二维地理空间按照规则格网进行划分,并利用一维空间填充曲线进行编码填充,常作为空间索引应用于空间点数据的存储和检索中。然而,GeoHash 编码对于空间线/面状数据的索引效果不佳,这是由于 GeoHash 编码是对点状数据进行编码,而重心这类特征点仅能表达线/面状数据的大致位置,不能表达其空间范围,从而导致检索结果缺失。为了适应空间线/面状数据的空间范围特征,本文设计了一种顾及空间范围的变长 GeoHash 编码与检索方案。

如图 2 所示,$GeoHash_i$($i=1,2,3,4$)分别为地理要素的最小外接矩形 4 个顶点的 GeoHash 编码。不同于重心这类特征点,最小外接矩形可以很好地表达线/面状地理要素的空间位置及其空间范围:

$$GeoHash = GeoHash_1 \cap GeoHash_2 \cap GeoHash_3 \cap GeoHash_4 \quad (1)$$

如式(1)所示,变长 GeoHash 编码为最小外接矩形 4 个顶点编码的最长前缀交集(如 w4gh12、w4gk12、w4gj12 和 w4gm12 的前缀交集为 w4g),代表了完全包含该地理要素的最小格网范围。

图 3 展示了本文提出的变长 GeoHash 空间范围检索策略。在进行检索时,首先根据检索矩形框的最大长宽确定粗检索使用的 GeoHash 编码长度,保证该编码长度所对应的格网大小能够完全覆盖检索矩形框;然后计算矩形框重心所在的格网及其相邻八方向格网的 GeoHash 编码,从而判断并获得检索矩形框所在的格网范围及编码

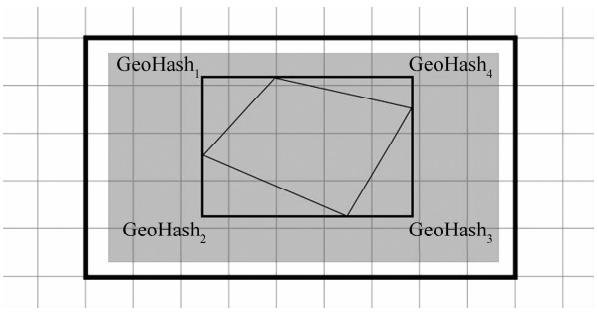

图 2　顾及空间范围的变长 GeoHash 编码方案

Fig. 2　Variable Length GeoHash Coding Scheme Considering Space Range

列表;根据对应编码在 HBase 空间表中匹配以该编码或其子集加"_"为前缀的行键,如图 3 的中心格网编码 w4gh 将匹配以 w4gh、w4g_、w4_、w_与_(跨越 0 经度和 0 纬度区域的矩形框会出现 4 个顶点无共同前缀的情况)为前缀的行键,从而可以从 HBase 中获取对应的空间数据;最后,将粗检索出的空间数据集与检索矩形框进行空间相交判断便可完成空间范围检索。经实验验证,这种变长 GeoHash 索引在保证查全率的同时,也提高了检索效率。

2.3　空间数据导入方法设计

尽管 HBase 提供了基于 MapReduce 的输入输出接口,但在 HBase 写入数据时由于预写日志系统(write-ahead logging, WAL)机制会导致 I/O瓶颈问题,因此本文设计了一种基于 MapReduce 的批量空间数据导入方法,跳过 WAL 机制,将空间数据文件转换为 HBase 对应的存储格式 HFile,实现分布式空间数据的快速预处理及入库。具体步骤如下:①将存储空间数据的文件(如

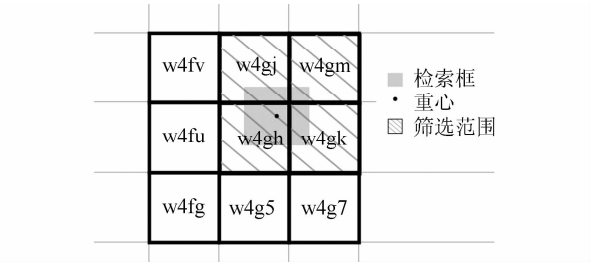

图3 基于GeoHash编码的空间范围检索策略

Fig. 3 Spatial Range Query Method Based on GeoHash Coding

shapefile、csv等文件)存储于HDFS中;②创建表1结构的空间数据存储表;③使用MapReduce框架对空间数据文件进行解析,计算各空间要素的GeoHash索引后将数据写入HFile;④将HFile导入到对应的HBase空间数据存储表中。实验表明,对于亿级数据记录的导入,时间可以从小时级降低到分钟级。

3 基于Spark RDD的分布式空间数据内存组织与映射

在Spark中,分布式数据通过RDD进行管理,它是分布式内存的一个抽象概念,是只读的记录分区的集合[13-14]。对于RDD的基本操作可以分为转换操作(transformation)与行动操作(action)。Spark对于RDD的操作是惰性的,在执行转换操作时并不会直接对RDD进行计算,而是记录计算子RDD与父RDD之间的溯源关系(lineage),直到行动操作时才依据之前的转换操作对RDD进行计算。这样的计算框架在保证RDD鲁棒性的同时,也减少了数据的冗余,带来了性能方面的提升。基于Spark RDD的特性,本文设计实现了基于RDD的分布式空间数据内存组织与映射,并为GIS内核提供针对分布式存储的空间大数据的数据接口。

3.1 分布式GIS内核

吉奥之星(GeoStar)内核是武汉大学与吉奥公司为国产地理信息平台软件GeoStar编写的跨平台地理信息处理内核,可以在Windows系统和Linux系统下实现地理信息处理功能,在生产环境中的应用证明了该内核的健壮性[15-16]。为了在分布式环境下尽可能地复用原有功能,保证软件健壮性的同时实现分布式、高性能的优点,本文设计实现了面向GeoStar GIS内核的分布式空间数据访问接口,重用内核中的算子来对存储于分布式空间数据表中的空间数据进行操作。图4为总体框架实现统一建模语言(unified model language,UML)图。

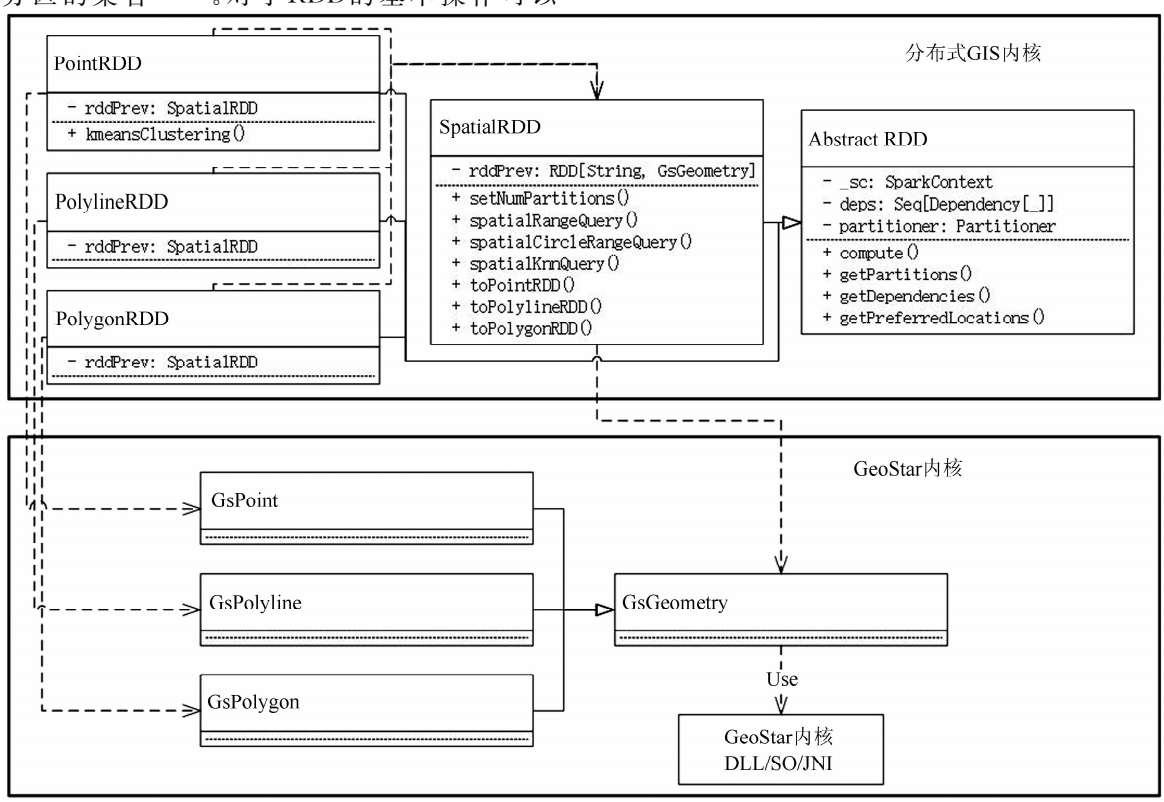

图4 基于GeoStar的分布式GIS内核实现示意图

Fig. 4 A UML Diagram of the Framework Using GeoStar Kernel

由于 GeoStar 内核底层基于 C/C++进行开发,不适合直接在分布式系统中调用。本框架通过利用 Java 本地调用接口调用 GeoStar 内核提供的动态链接库来访问其底层接口,从而为顶层的 Spark 分布式空间数据组织和处理(或称分布式 GIS 内核)提供支撑。

分布式 GIS 内核通过封装 GeoStar 内核,提供了 GsGeometry、GsPoint、GsPolyline、GsPolygon 等类,为 Spark 框架提供多种类型空间对象表示方法以及一系列空间数据处理接口,实现分布式空间数据内存组织方式和分布式处理。

3.2　SpatialRDD 设计

本文设计的基于 RDD 的 SpatialRDD 为空间数据提供分布式处理方法,并以其为基础类,针对不同的空间数据类型(点、线、面)扩展了相应的点弹性数据集(PointRDD)、线弹性数据集(PolylineRDD)以及面弹性数据集(PolygonRDD)。同时,基于 GeoStar 内核提供的空间计算操作,为 SpatialRDD 扩展空间计算能力,从而实现单机式空间操作到分布式空间操作的转换。如在 PointRDD 提供了针对空间点的分布式 k 均值聚类计算方法,在 SpatialRDD 中使用空间相交判断、空间距离计算等操作实现分布式空间范围查询、分布式空间 K 近邻查询等通用的分布式空间查询方法。

图 5 展示了 HBase 与 SpatialRDD 之间的映射关系,为 SpatialRDD 提供了一个与 HBase 中的空间数据表交互的数据接口(CreateSpatialRDDFromHBase),通过这一接口,系统能从 HBase 中的空间数据表直接生成 SpatialRDD 对象。在该接口中,首先将 HBase 表中的每行数据转换为 Hadoop 分布式弹性数据集(HadoopRDD),然后通过转换操作解析空间数据对象和索引,生成分区映射弹性数据集(MapPartitionRDD)。最后根据 MapPartitionRDD 对象转换生成包含空间对象数据集和其索引的 SpatialRDD。SpatialRDD 可以通过相应的转换操作进一步细化为 PointRDD、PolylineRDD 和 PolygonRDD,它们之间具有父子溯源关系。

3.3　分布式空间数据处理

图 6 展示了基于 SpatialRDD 的空间数据处理工作流。首先,根据 HBase 中的空间数据存储结构,利用 Spark 的 HBase 访问接口创建 SpatialRDD 对象(图 6 中的 CreateSpatialRDDFromHBase 操作);其次,SpatialRDD 对象调用数据重分区接口,利用空间索引对空间数据集在分布式

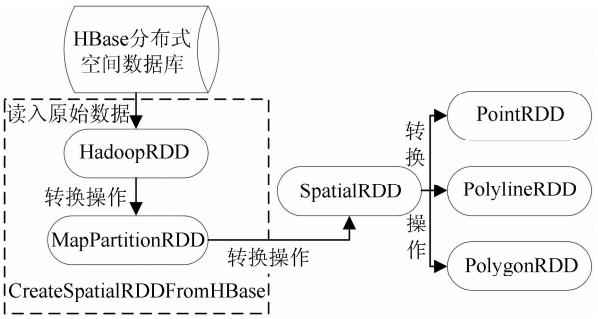

图 5　HBase 表与 SpatialRDD 映射图

Fig. 5　Mapping from HBase Table to SpatialRDD

集群上进行重分区(图 6 中的 SetNumPartitions 操作),减少数据倾斜对分布式处理性能的影响;之后,根据点、线、面数据类型通过相应的转换接口将 SpatialRDD 对象转化成相应空间类型的弹性数据集对象;最后,根据用户对空间大数据的处理需求,利用 GeoStar 内核(Gs Kernel 1)实现分布式空间数据处理接口对其进行处理(图 6 中的 Processing Interface)。

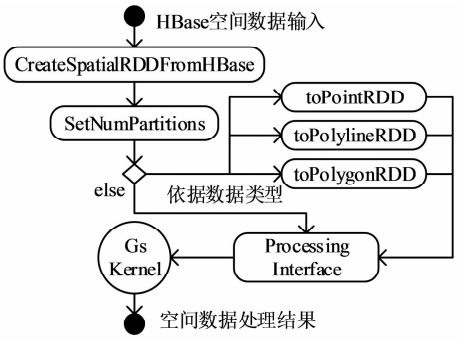

图 6　空间数据处理工作流

Fig. 6　Workflow of Geospatial Data Processing

基于 SpatialRDD 的分布式空间数据处理流程如图 7 所示。存储在 HBase 表中的空间数据通过空间索引预筛选后生成 SpatialRDD,数据在重分区后分布缓存在集群各个节点的内存中,基于 GeoStar 内核实现的空间数据处理算子被分发到各个数据分区中对数据进行处理,最后汇聚各个分区的数据处理结果得到最终的结果进行输出。

本文以空间圆形范围查询为例介绍其实现流程。①获取用户输入的查询中心点与查询半径,通过圆形范围的最小外接矩形获取其所在格网范围的 GeoHash 编码列表;②基于 GeoHash 编码列表从 HBase 空间数据表中初步筛选出行键,为这些编码及其子集加上字符"_"为前缀的空间数据对象,生成 SpatialRDD 对象并将数据均匀分区;③将查询中心点、查询半径、空间对象距离计算函数分发到各个数据分区所在的计算单元;

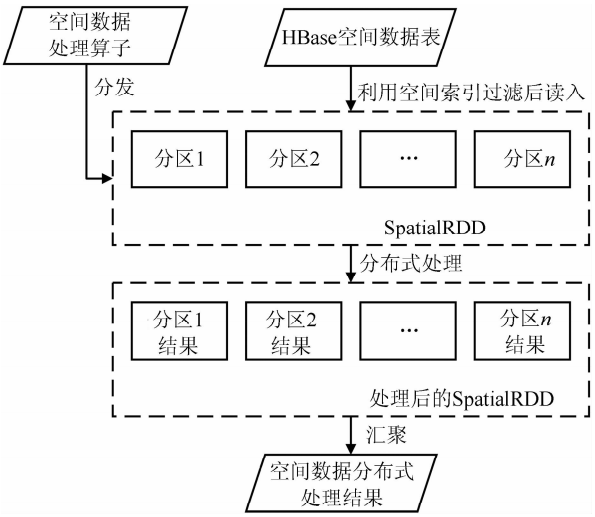

图7 分布式空间数据处理流程

Fig. 7 Distributed Geospatial Data Processing Flow

④在各个计算单元上计算分区中各个空间对象到查询中心点的距离，精确筛选出位于查询范围内的空间对象；⑤将各个分区的计算结果汇聚到主节点，输出结果。

4 实验与分析

4.1 实验环境

本文实验所用的Spark与HBase集群部署于OpenStack私有云平台之上，由1个主节点与4个从节点组成，Hadoop版本为Hadoop2.7.3，HBase版本为HBase1.3.1，Spark版本为Spark2.1.0，Centos系统版本为Centos7 X86_64，OpenStack版本为OpenStack Newton，主节点配置为4核、8 GB内存，从节点配置为8核、16 GB内存。

4.2 实验结果分析

本文的实验数据集是从公开地图OSM(OpenStreetMap)(http://www.openstreetmap.org)中抽取出的空间数据。点数据集包含1.8亿个兴趣点，线数据集包含0.76亿条道路，面数据集包含1.14亿个建筑物，可视化效果如图8所示。本文将各类型数据集分别导入到HBase表中进行持久化存储后，进行了一系列性能测试。

首先，本文对比了分布式空间数据存储计算框架下有索引和无索引进行分布式空间查询的效率。随机选取空间点50次，查询半径10 000 m内的空间数据，平均查询时间如图9(a)所示。由于使用了变长GeoHash编码对空间数据进行预筛选后，仅有小部分数据导入SpatialRDD中进行分布式空间检索。虽然查询时间会与查询框的选取位置有关(比如面数据较密集区域的查询时间为30 s左右，而面数据稀疏区域只需要几秒)，但相较于不使用GeoHash索引整表检索的方式，已经大大减少了检索数据量，查询效率提高两个量级。

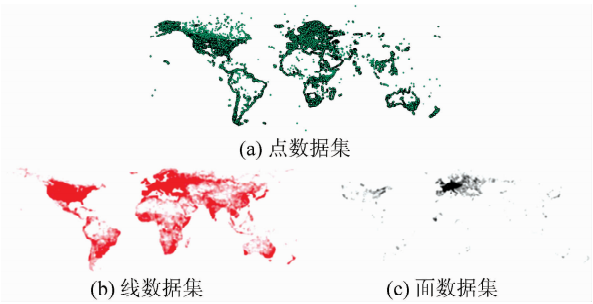

图8 实验数据集可视化

Fig. 8 Visualization of Experimental Datasets

其次，就本文提出的分布式空间数据存储计算框架和传统的基于PostGIS的空间数据存储框架进行性能对比。随机选取空间点50次，按不同查询半径分别对点、线和面状空间数据集进行空间范围查询，HBase+Spark与PostGIS查询返回结果相同，平均查询时间如图9(c)、9(d)、9(e)所示。由于PostGIS空间检索的策略为全表扫描，其查询速度与查询半径大小无关，稳定在某个时间区间；而本文框架进行空间范围查询时，由于使用了变长GeoHash编码作为空间索引，并利用了对应的检索策略对空间数据进行了预筛选，结合分布式计算实现精筛，体现出了较好的查询性能。且本文提出的检索策略克服了传统GeoHash编码突变性造成的性能瓶颈，相较于传统空间数据库检索方式有一个量级的性能提升。最后，对比了本文提出的分布式空间数据存储与计算框架与传统单机空间数据存储与计算组合的效率，根据空间矩形范围查询获取了约100万个空间点对象，进行不同参数的k均值聚类计算，结果如图9(b)所示，由于采用了分布式查询与计算架构，本框架的查询与计算效率明显高于传统的单机框架，而且计算量越大，性能优势越明显。

5 结 语

为高效存储并应用海量时空数据，本文提出了一种基于Spark的分布式空间数据存储架构，利用OpenStack、Spark、HBase、Hadoop等云计算方案，搭建了分布式空间数据存储设施，并复用传统地理信息系统软件GeoStar内核进行空间数据的操作，与传统的基于PostGIS搭建的空间数据

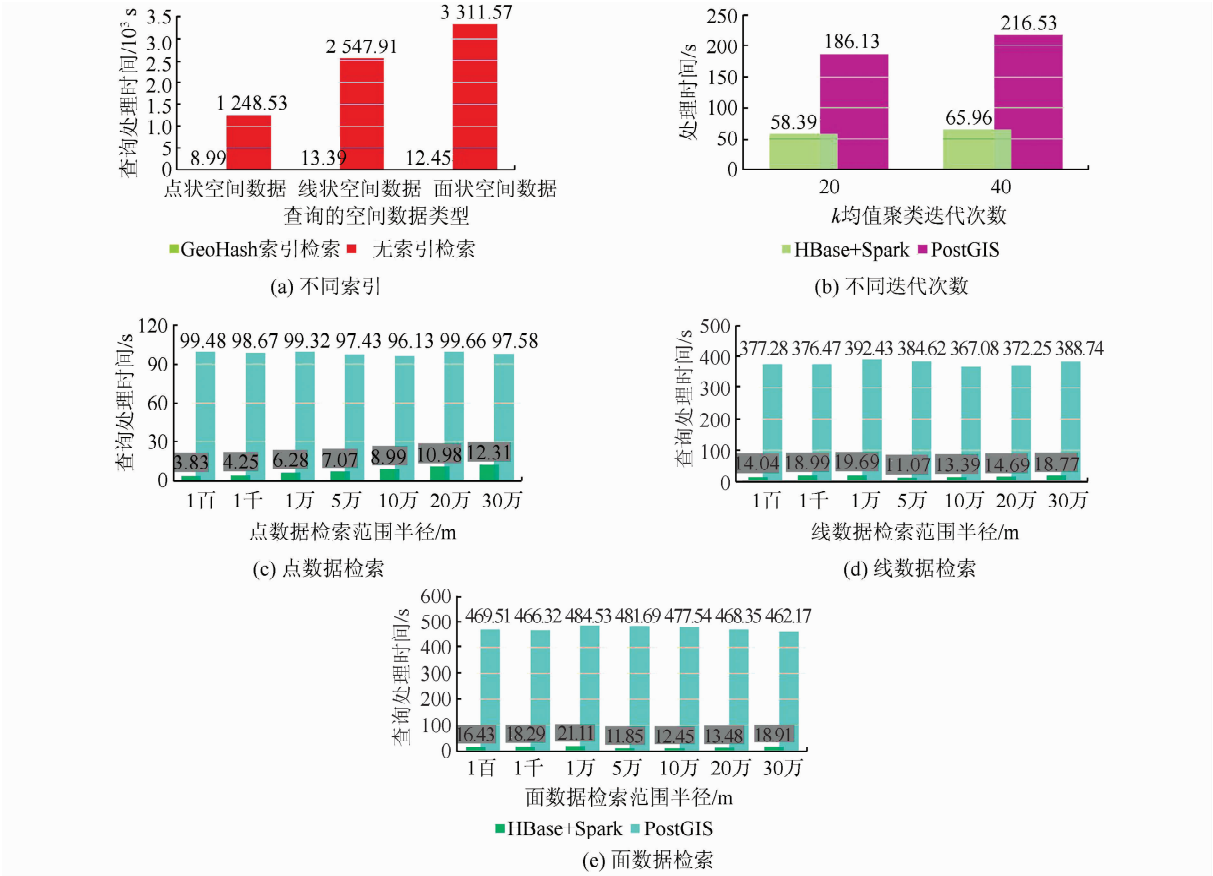

图 9　HBase+Spark 与传统 GIS 架构性能对比图

Fig. 9　Performance Comparison Between HBase+Spark and Traditional GIS Architecture

库系统进行对比实验，验证了本文提出的分布式空间数据存储架构的高效性。在之后的研究中，将考虑使用 R 树等更高效的空间索引来对 HBase 空间数据表的索引方案进行优化，以得到更好的查询性能。

本文提出的框架设计一方面兼容了传统的地理信息软件，提高了软件的复用性，另一方面运用了先进的云计算软件为空间数据的存储及计算带来高性能与高可用的特色，有助于从传统 GIS 内核走向分布式 GIS 内核，为促进地理信息系统适应大数据时代提供了思路。

参　考　文　献

[1] Li Deren. Towards Geo-spatial Information Science in Big Data Era[J]. *Acta Geodaetica et Cartographica Sinica*, 2016, 45(4): 379-384 (李德仁. 展望大数据时代的地球空间信息学[J]. 测绘学报, 2016, 45(4): 379-384)

[2] Yue P, Ramachandran R, Baumann P, et al. Recent Activities in Earth Data Science[J]. *IEEE Geoscience & Remote Sensing Magazine*, 2016, 4(4): 84-89

[3] Stonebraker M. SQL Databases v. NoSQL Databases[J]. *Communications of the ACM*, 2010, 53(4): 10-11

[4] Yue P, Baumann P, Bugbee K, et al. Towards Intelligent GIServices[J]. *Earth Science Informatics*, 2015, 8(3): 463-481

[5] Huang B. Comprehensive Geographic Information Systems[M]. US: Elsevier, 2018: 50-79

[6] Aji A, Wang F, Vo H, et al. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce[J]. *Proceedings of the VLDB Endowment*, 2013, 6(11): 1 009-1 020

[7] Eldawy A, Mokbel M F. Spatial Hadoop: A MapReduce Framework for Spatial Data[C]. 31st IEEE International Conference on Data Engineering, Seoal, Korea, 2015

[8] You S, Zhang J, Le G. Large-Scale Spatial Join Query Processing in Cloud[C]. 31st IEEE International Conference on Data Engineering, Seoal, Korea, 2015

[9] Yu J, Wu J, Sarwat M. GeoSpark: A Cluter Computing Framework for Processing Large-Scale Spatial Data[C]. 23rd ACM Sigspatial International Conference on Advances in Geographic Information Systems, Seattle, Washington, 2015

[10] Tang M, Yu Y, Malluhi Q M, et al. Location-

Spark: A Distributed In-memory Data Management System for Big Spatial Data[J]. *Proceedings of the VLDB Endowment*, 2016, 9(13): 1 565-1 568

[11] Hall G B, Leahy M G. Open Source Approaches in Spatial Data Handling[M]. Berlin, Heidelberg: Springer, 2008: 87-104

[12] Vora M N. Hadoop-HBase for Large-Scale Data [C]. 1st IEEE International Conference on Computer Science and Network Technology, Harbin, China, 2011

[13] Zaharia M, Chowdhury M, Franklin M J, et al. Spark: Cluster Computing with Working Sets[C]. 2nd ACM Usenix Conference on Hot Topics in Cloud Computing, Boston, USA, 2010

[14] Zaharia M, Chowdhury M, Das T, et al. Resilient Distributed Datasets: A Fault-tolerant Abstraction for In-memory Cluster Computing[C]. 9th ACM Usenix Conference on Networked Systems Design and Implementation, San Jose, USA, 2012

[15] Li D, Gong J, Zhu Q, et al. GeoStar—A China Made GIS Software for Digital Earth[C]. International Symposium on Digital Earth, Beijing, China, 1999

[16] Zhu Xinyan, Gong Jianya, Huang Juntao, et al. Spatial Data Organization and Management in GeoStar[J]. *Geomatics and Information Science* of Wuhan University, 2000, 25(2): 122-126 (朱欣焰, 龚健雅, 黄俊韬, 等. GeoStar空间数据组织与管理[J]. 武汉大学学报·信息科学版, 2000, 25(2): 122-126)

Design and Implementation of a Distributed Geospatial Data Storage Structure Based on Spark

YUE Peng[1] *WU Zhaoyan*[1] *SHANGGUAN Boyi*[1]

1 School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China

Abstract: In recent years, with the rapid development of sensor web and earth observation technologies, geospatial data has become an important part of the big data, traditional geospatial data storage and processing systems are increasingly unable to meet the requirements of big geospatial data. The Apache Spark, which is a unified analytics engine for large-scale data processing, can provide both the management and processing capabilities of big geospatial data. And based on the Apache Spark, a fundamental platform for developing cloud-based GIS can be developed to move conventional GIS kernel to distributed GIS kernel in the era of cloud computing. On the basis of the data organization and computation models of the Apache Spark system, this paper couples it with the Apache HBase distributed database, and presents the approaches of the design and implementation of a distributed geospatial data storage and processing architecture by leveraging data management and computing paradigm between Apache Spark and Apache HBase. In the architecture, a variable-length GeoHash index method is proposed to improve the query performance of geospatial point, polyline and polygon data, and the SpatialRDD is presented to manage and process the geospatial data queried from the Apache HBase in a distributed manner. The GIS kernel of the architecture is realized based on a Chinese-brand GIS software, in view of the storage and processing of different kinds of geospatial data, such as point, polyline and polygon, a series of contrast experiments with the traditional geospatial database, PostGIS, are performed, and the results demonstrate the applicability and efficiency of the approaches.

Key words: Spark; cloud GIS; distributed spatial data organization; distributed GIS kernel; big geospatial data

First author: YUE Peng, PhD, professor, specializes in the geographical information system software, web geographic information and location intelligence services, spatial database, high performance geographical computing. E-mail: pyue_whu@163.com

Foundation support: The National Key Research and Development Program of China, No. 2017YFB0504103; the National Natural Science Foundation of China, No. 41722109; the Huanghe Talents Science and Technology Innovation Project of Wuhan (2016); the Science Foundation for Distinguished Young Scholars of Hubei Province, No. 2018CFA053.