基于 MapReduce 的 GML 并行查询

许 斌 关佶红
(同济大学计算机科学与技术系 上海 201804)

摘 要 针对应用地理标记语言(Geography Markup Language,GML)表示的海量空间数据查询问题,提出一种基于MapReduce的GML并行查询方法。通过提取GML空间特征集合,实现GML文档查询到GML空间特征集合查询的查询转化,并利用MapReduce实现空间特征并行查询。

关键词 地理标记语言,GML查询,MapReduce,并行查询
中图法分类号 TP311.12 **文献标识码** A

GML Parallel Query Based on MapReduce

XU Bin GUAN Ji-hong
(Department of Computer Science and Technology, Tongji University, Shanghai 201804, China)

Abstract A novel GML distributed query method based on MapReduce model was proposed to deal with queries with large amount of GML data. Some pretreatments are taken with the original GML files to generate GML feature set before the queries, and the queries are converted from GML files to GML Feature Set. The queries can execute in a parallel way and be more efficient.

Keywords GML, GML query, MapReduce, Parallel query

1 引言

GML(Geography Markup Language)即地理标记语言[1],是由OGC(开放式地理信息系统协会)制定的一种基于XML编码格式,用于地理信息的编码、传输、存储和发布的标记语言。目前GML已成为Internet上地理信息表示的国际标准,广泛应用于Web和分布式地理信息系统中交换和集成空间数据。随着越来越多的空间数据以GML作为表示、存储与交换手段,GML数据变得越来越复杂,GML文档也随之变得越来越大,传统的基于单节点的GML查询方法已无法胜任处理如此庞大数据量的任务。因此,一种高效处理大量和海量GML空间数据的方法变得极为重要。

GML采用特征(Feature)或特征集来描述现实世界中的地理对象或现象。GML文档中的空间包含空间和非空间属性等内容。空间特征构成了GML文档的核心,同时GML查询也往往以空间特征的某个属性或某些属性作为研究对象。因此,GML文档查询也可以认为是对GML文档内所包含的GML空间特征集合的查询。空间特征集合内的任一特定空间特征查询,与其他特征的查询是相对独立的,因此可以采用并行处理方式对整个GML空间特征集合执行查询处理。

鉴于GML文档存储特点,本文提出一种基于GML空间特征的并行查询策略,旨在利用MapReduce[2]计算模型对GML文档中的GML空间特征实现并行解析查询,以支持对GML大文档的查询工作。

2 相关工作

目前针对GML的查询主要有两种方法:第一种方法是利用模式映射将GML文档数据存储至数据库,从而能够利用相对较成熟的数据库技术来实现GML数据查询,如Oracle的Oracle Spatial、IBM的DB2 Spatial Extender和PostgreSQL的PostGIS等都支持基本的空间数据操作;第二种方法即直接对GML文档进行解析查询,一般通过扩展XML[3]查询的XPath、XQuery[4]方法,如加入空间操作算子等,其主要思想是对GML文档进行DOM或SAX解析,直接从文档内获取目标信息。

第一种方法需要定义GML文档到空间数据库的映射规则,如文献[5,6]中都是采用该思想实现GML文档的存储和查询。该方法的映射时间往往较长,随着GML文档的增大,这个过程将消耗更多的时间。同时相比于文档形式存储,其灵活性也有所下降;而第二种方法主要用于处理小文件,对于GML大文档的查询时间往往比较长,且限于机器性能对于过大文档将无法实现查询,如文献[7,8]等提出了基于XQuery扩展的GML文档查询方法,并实现了相关查询引擎,但是对于所能处理的GML文档大小依然有限。

近年来针对XML海量数据分布式处理的相关研究越来越多,罗静等[9]提出一种基于文档拆分的XML文档分布式存储与查询机制;Leonidas Fegaras等[10]也提出了一种基于MapReduce的MRQL语言,并将其在XML文档查询上加以

到稿日期:2013-01-31 返修日期:2013-05-09 本文受基金项目面向特定领域的xml数据组织和检索技术(2009AA01Z135)资助。
许 斌(1988—),男,硕士生,主要研究方向为海量GML空间数据管理,E-mail:1020080244@tongji.edu.cn;关佶红(1969—),女,博士,教授,主要研究方向为空间数据库与数据挖掘、分布计算、生物信息学、地理信息系统。

实现。上述方法均是利用集群处理能力来提高 XML 查询效率。GML 是由 XML 发展而来，针对 GML 查询的优化研究同样可以适当借鉴 XML 文档查询的现有成果。相较于 XML 文档，GML 文档内所存储的 GML 空间特征包含了更加丰富的空间信息，如几何属性、拓扑属性和时间属性等，如何对这部分属性进行处理，也需要在查询实现过程中根据需要采用特殊处理方式。

3 基于空间特征的 GML 查询

根据前文介绍，GML 文档的查询也等同于对 GML 文档中所存储的 GML 空间特征进行查询。可以对 GML 空间特征进行匹配筛选，并从符合查询条件的 GML 空间特征内提取相关 GML 信息。

3.1 GML 空间特征简介

GML 是用于描述地理实体或地理现象的一种特殊的 XML 文档，主要描述的是地理空间特征。GML3.0 标准模式中对地理数据的描述主要针对空间特征或特征集进行，通过各种属性描述各特征或特征集，空间特征所包含的主要属性有空间属性(包括地理位置属性、边界属性、拓扑属性、时间属性等)和非空间属性。对于带有模式的 GML 文档而言，各空间特征及其各属性描述都已确定。图 1 显示了 GML 文档内部节点由大到小的嵌套关系。

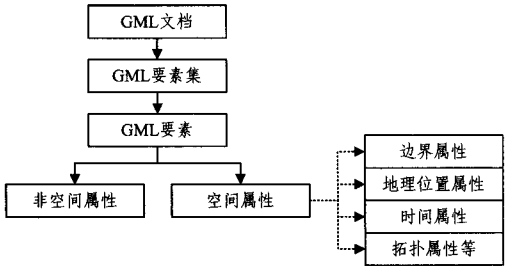

图 1 GML 文档节点嵌套结构

```
<?xml version="1.0" encoding="UTF-8"?>
<CityModel>
  <gml:name>Cambridge</gml:name>
  <gml:boundedBy>
  <gml:Box>
  <gml:coordinates>0.0,0.0 100.0,100.0
  </gml:coordinates>
  </gml:Box></gml:boundedBy>
<cityMember>
<River>
   <gml:description>
    The river that runs through Cambrid
  </gml:description>
   <gml:name>Cam</gml:name>
   <gml:centerLineOf>
     <gml:LineString>
     <gml:coordinates>0,50 70,60 100,50
     </gml:coordinates>
     </gml:LineString></gml:centerLineOf>
</River>
</cityMember>
  <cityMember>
   <Road>
   <gml:name>M11</gml:name>
   <linearGeometry> <gml:LineString>
   <gml:coordinates>0,5.0 20.6,10.7 80.5,60.9
   </gml:coordinates>
   </gml:LineString></linearGeometry>
   <classification>motorway</classification>
   <number>11</number>
  </Road>
</cityMember>
.....
</CityModel>
```

GML空间特征
River:
属性包括:
gml:description,
gml:name, 等
gml:centerLineOf等

图 2 GML 文档片段及空间特征

一个 GML 文档片段("cambridge. xml")和其中所包含 GML 空间特征的简单示例如图 2 所示，其中黑色粗体字部分即为一空间特征的示例，在该文档中为一个 River(表示一个河流对象)。

根据图 2 所示，GML 文档整体由若干 GML 空间特征组成，一个 GML 文档根据需要往往可以包含多类空间特征，如图 2 GML 片段中包含了 River 和 Road 等。而实际使用的 GML 文档绝大多数要比图 2 所示文档片段更加复杂。

3.2 GML 查询

对 GML 文档进行查询，主要是为了从 GML 中提取符合查询要求的 GML 空间特征的属性信息。因此，针对 GML 文档的查询，首先需要定位与查询相关的 GML 空间特征，随后对其提取查询返回信息。

图 3 是一个针对图 2 中所表示的 GML 文档的简单查询实例，查询 cambridge. xml 中长度不小于 50 的道路。

```
for $x in document("cambridge. xml")//Road
where Length( $x//gml:LineString)>=50
return//返回查询结果信息
〈RoadInfo〉
  〈Name〉{ $x/gml:name/text()}〈/Name〉
  〈Length〉{Length( $x//gml:LineString)}〈/Length〉
  〈Boundary〉{Boundary( $x//gml:LineString)}〈/Boundary〉
〈/RoadInfo〉
```

图 3 GML 查询实例

基于传统 GML 文档查询方法，对整个 GML 文档进行解析，随后当访问到 Road 空间特征时，利用空间算子计算该 GML 空间特征的长度属性并与查询条件中的 50 进行比较，若条件符合，则提取返回信息；反之直接跳过，寻找下一个 Road 特征，直到整个 GML 文档解析完成。显然，这样一种处理方式随着 GML 文档的增大，将耗费更多时间用于完成对所有空间特征的解析和条件匹配。若我们能够将所有相关 GML 空间特征利用预处理提取成集合，并采用并行方式解析匹配该集合内的所有元素，则有助于提高查询效率。

4 MapReduce 并行处理模型

MapReduce 计算模型是由谷歌公司 Jeffery Dean 等首先公开发表的一种分布式并行处理模型，主要用于大规模数据集的并行运算。在 MapReduce 模型中，用户需要自行定义 Map 函数和 Reduce 函数的具体工作内容，通过 Map 函数处理输入键值对(key/value)，产生一系列的中间键值对，然后利用 Reduce 函数来对具有相同键的中间键值对进行规约。

执行 Map 函数前，集群首先对输入数据进行分割(split)，不同的分割被分配到不同的 Map 处理器上并行处理，从而保证任务处理效率，在 Map 函数之后还会有一个洗牌(Shuffle)过程，其对于提高 Reduce 的处理效率以及减小集群内部数据的传输压力有很大帮助。图 4 所示为 MapReduce 处理模型的工作流程。

用户可以利用特定的 InputFormat 对原始数据进行格式化，生成相应的键值对列表作为集群处理的源数据；其后 MapReduce 调用预先定义好的 Map 函数对所有键值对进行 Map 处理，得到中间结果 list(k2,v2)；最后将具有相同键值的中间键值对传输到同一个 Reduce 处理器上，调用 Reduce 处理中间结果而获得最终结果。

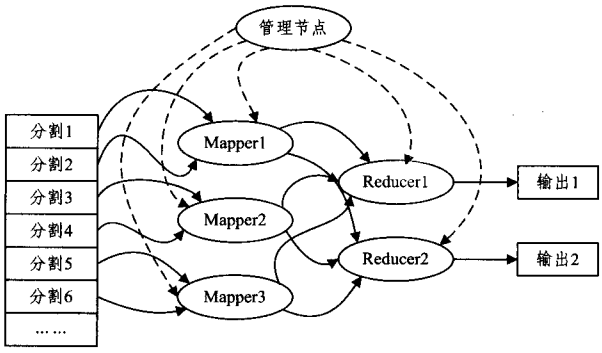

图 4　MapReduce 框架

结合 MapReduce 处理模型和 GML 文档存储结构，GML 文档内任一空间特征均能作为 MapReduce 处理器的最小处理单元，处理节点利用查询条件对 GML 空间特征进行匹配筛选，得到最终符合所有查询条件的空间特征或特征集合，并利用其构造查询结果。

5　基于空间特征 GML 并行查询

通过对 GML 文档提取空间特征集合，实现 GML 查询与 MapReduce 并行处理模型的完美结合。本节介绍基于 MapReduce 处理模型的 GML 空间特征并行查询策略，将从空间特征集合提取、查询分类、并行查询工作原理以及 GML 结果构造等方面具体展开。

5.1　空间特征集提取

并行处理要求处理对象必须是简单可拆分的，其可以被分成多个子对象由不同处理器同时处理。MapReduce 模型的工作原理也同样是将一个大文件拆分成若干个相互独立的分割(Spilits)，这些分割被分配到不同的 Map 处理器上，每个分割经过处理都会产生一个相应的中间结果，并在 shuffle 阶段传给指定的 Reduce 处理器。为了让 GML 文档能够在 MapReduce 模型下得到有效处理，需首先对原始 GML 数据进行一些预处理，提取待处理的 GML 空间特征集合以替代原 GML 文档作为 MapReduce 的处理输入数据。

本文采取的 GML 文档预处理策略是基于 Hadoop[12,13] 的 XmlInputFormat，根据查询处理要求，选取 GML 文档内部特定的 GML 空间特征片段构成 GML 特征集合。利用算法 source(fileName, nodeName) 进行空间特征提取，其中 fileName 为 GML 原始文档名(该文档需预先存储于 HDFS, Hadoop Distribute File System[11])，nodeName 为提取节点名，亦即 GML 空间特征名。GML 查询对象的转换如图 5 所示。

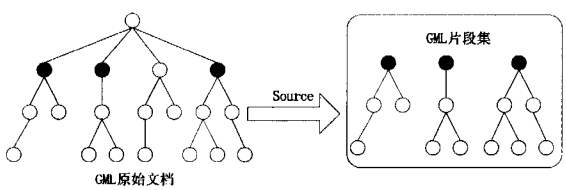

图 5　空间特征集合提取

存储于 HDFS 中的 GML 文档通过对构成该 GML 文档的块进行并行的遍历，获取所有满足条件的空间特征构成空间特征集合，Source 算法如图 6 所示。

```
Algorithm Source:Retrieve GML Feature Set
Input:GM file location:gml & Feature List:list
Output:GML Feature Set gset
1. Procedure locate(gml)/ * 定位 gml 对应的所有文档 block * /
2.    For each block of blocklist/ * 对存储于本节点上的文档 block 处理 * /
3.      For each feature in block
4.        If(isFeture in list) / * 验证当前特征是否属于 l 指定 GML 空间特征 * /
5.          gset. insert/ * 将当前特征插入返回集合中 * /
6.      End for each feature
7. End for each block
```

图 6　GML 特征提取算法 Source

利用上述算法，可以从图 2 所示的 cambridge. xml 文档中提取所有 Road 空间特征，如 ROAD=source("cambridge. xml",{"Road"})，生成用于并行查询的 GML 空间特征集合，并将该数据集合命名为 ROAD，如图 7 所示。

```
1. 〈Road〉
2.   〈gml:name〉M11〈/gml:name〉
3.   〈linearGeometry〉
4.   〈gml:LineString〉
5.     〈gml:coordinates〉0,5.0 20.6,10.7 80.5,60.9
6.     〈/gml:coordinates〉
7.   〈/gml:LineString〉
8. 〈/linearGeometry〉
9.   〈classification〉motorway〈/classification〉
10.  〈number〉11〈/number〉
11. 〈/Road〉
12. 〈Road〉
13.  〈gml:name〉M23〈/gml:name〉
14.  〈linearGeometry〉
15.  〈gml:LineString〉
16. 〈gml:coordinates〉3.5,2.0 20.6,－10.7 80.5,－60.9
17. 〈/gml:coordinates〉
18.  〈/gml:LineString〉〈/linearGeometry〉
19.  〈classification〉motorway〈/classification〉
20.  〈number〉11〈/number〉
21. 〈/Road〉
22. ……
```

图 7　GML 特征集合 ROAD

5.2　GML 查询分类

根据所涉及的空间特征属性，GML 查询可以分为空间查询和非空间查询，其中对于特征的非空间属性查询与 XML 查询完全相同，因此该类查询可以直接采用 XML 的 XPath 和 XQuery 等查询方法进行处理。

对于 GML 的空间属性查询，按照查询中所涉及的 GML 空间特征个数，又可以分为以下两类：(1)GML 空间属性查询，如：Length(geometry)、Area(geometry)，用以查询某个特征的几何体的长度和面积；(2)GML 空间关系查询，涉及到两个空间对象之间属性间关系的比较，如：Overlap(geometry1, geomety2)、Crosses(geometry1, geomety2)，用以判断两几何体是否有覆盖和重叠等。

根据上述 GML 查询分类，Query1 和 Query2 是基于特征集合 ROAD、RIVER 的两类简单的 GML 查询示例。GML 空间属性查询和 GML 空间关系查询如图 8 和图 9 所示。

```
Query1：查询 ROAD 集合内，长度大于 50 公里的公路信息
1. for $x in ROAD //对 ROAD 特征集合内每个元素进行遍历
2. where Length($x//gml:LineString)>=50
3. return//返回查询结果信息
4. <RoadInfo>
5.   <Name>{$x/gml:name/text()}</Name>
6.   <Length>{Length($x//gml:LineString)}</Length>
7.   <Boundary>
8.     {Boundary($x//gml:LineString)}
   </Boundary>
9. </RoadInfo>
```

图 8　GML 空间属性查询

```
Query2：查询 11 号公路经过的所有河流
1. for $x in ROAD
2. let $y in RIVER
3. where $x/number=11 and
4.   Crosses($x//gml:LineString, $y//gml:LineString)=1
5. return
6. <RiverInfo>
7.     <Name>{$y/gml:name/text()}</Name>
8.     <Length>{Length($y//gml:LineString)}
       </Length>
9. </RiverInfo>
```

图 9　GML 空间关系查询

5.3　GML 并行查询处理

GML 并行查询实际查询对象为 GML 空间特征集合 DataSet(G)。根据 5.1 节介绍，GML 特征集合是通过利用标签对从 GML 文档中提取所有符合条件的 GML 片段而获得的。GML 并行查询的剩余工作就是根据查询要求建立相应的 MapReduce 任务对空间特征集合进行筛选处理和获取查询结果。在完成预处理后，所有的空间特征将以流(stream-like)的形式传递给 Mapper 处理器或者 Reducer 处理器进行查询处理工作，经过 Map 处理后所得到的中间结果则会以集群内部定义的二进制格式进行暂存和传输，以提高集群内部效率，Mapper 处理器端的输出将在 Shuffle 阶段传递给 Reducer 处理器进行最后的规约(reduce)，构建最终查询结果，或将结果传输给后续 MapReduce 任务。

整个 GML 并行查询过程可以分成两个部分：(1)空间特征筛选；(2)查询结果构造。查询处理器将 GML 查询转换为一系列 MapReduce 任务，其中每一个 MapReduce 任务处理一个或多个查询条件，用以对空间特征进行筛选。最后查询处理器将满足所有查询条件的空间特征传输给查询结果构造器，构造器从符合所有查询条件的空间特征内获取结果信息。

以 5.2 节内 Query1 为例，该查询可以翻译为一个简单的 MapReduce 任务完成：在 Map 过程中，每一个 Mapper 处理器对传输过来的 Road 进行筛选，若当前处理的 Road 长度属性小于 50 则直接忽略，否则将该空间特征添加至中间结果集合并传输给 Reducer 处理器，由 Reducer 处理器将 GML 片段构造成查询结果并返回。具体工作流程如图 10 所示。

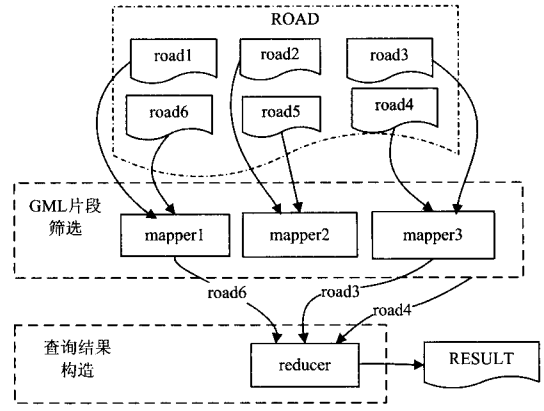

图 10　Query1 查询过程

在 Query1 查询执行过程中，mapper1－mapper 3 所处理的 6 个 Road 特征中只有 road3、road4 与 road6 符合查询条件，因此这 3 个 GML 片段被传输至 reduce 端进行最后结果的构造与输出。实际查询工作往往要比上述查询复杂得多，查询处理器将把整个 GML 查询翻译成多个 MapReduce 任务的组合，最终完成源数据的筛选，并对满足条件的 GML 片段进行结果构造。

6　实验及性能

6.1　试验环境及数据集

实验集群由 6 台普通 PC 机(AMD Athlon64X2 双核处理器，2G 内存)组成，采用 Hadoop 开源框架实现集群管理：设置其中 1 台 PC 作为 namenode 实现对整个集群的管理，其他 5 台 PC 作为数据处理节点 datanodei 完成数据存储和处理工作。节点之间利用百兆交换机实现互连，用于各节点之间的数据传输和通信。

本文利用 ArcGIS 空间数据格式转换工具，将 ArcGIS 中部分美国国家地图 Shapefile 文件数据(包括：铁路网信息、城市信息、街道信息以及水域信息等)转换为一组实验用 GML 文档，文档内所有 GML 数据的保存形式均符合 GML2.0 规范。表 1 给出了实验所用到的 GML 数据集的信息。

表 1　实验数据集信息

数据名称	文件大小	包含特征数
USA_Railways. xml	235M	262,960
USA_Cities. xml	522M	69,646
USA_Streets. xml	1.9G	1,478,256
USA_Waters1. xml	4.2G	3,249,180
USA_Waters2. xml	8.6G	6,134,918

6.2　实验与分析

针对实验数据集中的文档，我们分别对其进行：1)非空间查询，如河流、城市名称等；2)空间属性查询，如河流长度和城市面积等，该类查询一般不需要进行对象间的连接查询，直接对 GML 片段集内元素进行逐个处理便能得到最终结果；3)空间关系查询，如某一河流经过的所有城市等，该类查询通常需要通过一次甚至多次 GML 片段集之间的连接查询来完成。

实验主要验证文档大小、集群规模和集群配置等因素对查询的影响。

6.2.1 文档大小

根据图11结果显示，查询时间长短与文件大小正相关。空间关系查询一般涉及到两个甚至多个空间对象的连接操作，因此该类查询相比于非空间查询和空间属性查询所消耗的时间要多得多。同时根据结果显示，查询时间与GML文档大小并不成正比，这是由于GML片段集提取过程并没有对整个文档所有内容进行提取，只提取了与查询相关的那部分GML文档片段。基于MapReduce和GML片段集的分布式查询策略有效分摊了空间查询中的查询负荷，相比于单机空间查询，其查询效率明显提高。

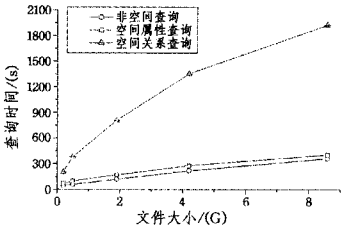

图11 查询效率与文件大小关系

6.2.2 集群规模

研究USA_Water.xml文档的非空间及空间属性查询，进行集群规模对查询时间的影响分析。根据图12所示：随着集群规模的增大，查询消耗时间逐渐缩短，但是时间减少幅度逐渐降低。因此在决定集群规模过程中需要综合考虑效率提升收益以及硬件投入的成本，最终选取较优方案。本组实验中，我们设置集群文件系统的Block大小为64M，并将每个处理节点的最大Mapper和Reducer数置为固定常数。由于本集群中所有处理节点均配置双核处理器，因此将该值设为2，将充分利用处理器的能力。

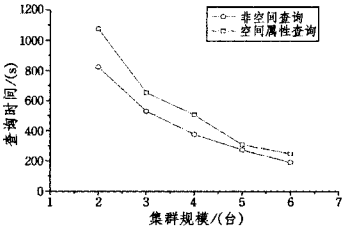

图12 集群规模对查询的影响(文档大小4.2G，集群块配置64M)

6.2.3 集群配置(HDFS块大小配置)

图13中实验结果显示，随着HDFS中块(Block)大小的增大，查询时间逐渐减小。这是由于在本实验中，块大小在一般情况下与MapReduce任务实际工作中所处理的数据分割(Split)保持一致；当HDFS的块设置较大时，Hadoop集群能够利用较少的Map和Reduce任务来处理GML空间特征集合，从而减少了任务配置时间以及Suffer阶段中间结果的传输配置时间，因此查询总时间有所减少。但是若块大小设置过大，当GML文档过小时将有可能导致集群内各节点负载不均衡，从而影响整个集群查询效率。因此选择合适的集群文件系统分块Block大小，可以有效提高查询效率，如上述查询设置Block大小为64M或者128M较为合适，有利于充分利用处理器的处理能力。

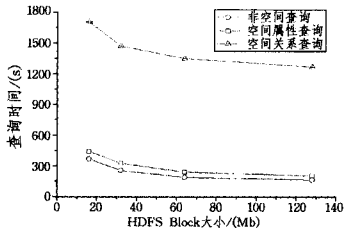

图13 块大小对并行查询的影响

根据以上实验结果，集群的规模与配置均将对GML并行查询产生影响。在本集群实验环境中，6台普通PC机组成实验集群，并将HDFS块大小设置为128M，相比于其他集群配置，能够更好地发挥集群并行处理能力。

结束语 本文将对GML文档的查询转换为对空间特征集合的查询，利用MapReduce并行计算模型处理每一个空间特征，支持对GML大文档的查询。在查询过程中将GML查询语句转换为一系列筛选GML空间特征的MapReduce任务，最终利用结果构造器从符合查询条件的GML空间特征中获取查询结果。

参考文献

[1] Geography Markup Language(GML)[OL]. http://opengis.net gml/01-029 GML2.html,2000

[2] Dean J,Ghemawat S. MapReduce: simplified data processing on large clusters[C]//Six Symposium on Operating System Design and Implementation. San Francisco,CA. Berkeley: USENIX Association,2004:10-24

[3] World Wide Web Consortium. Extensible Markup Language (XML) 1.0(Fourth Edition)[S/OL]. http://www.w3.org/TR/2004/REC-xml-20060816/. W3C Recommendation 16 August 2006,edited in place 29 September 2006,2009-04-28

[4] W3C(2005)XQuery 1.0:an XML query language[OL]. http://www.w3.org/TR/xquery/

[5] 李俊，关佶红，李玉珍. GML空间数据存储映射模型研究[J]. 武汉大学学报：信息科学版，2004(12)

[6] 朱付宝，关佶红，周水庚. 基于模型映射的GML文档存储和查询方法[J]. 计算机研究与发展，2006,43(z3):510-516

[7] 陈慧，朱付宝，关佶红，等. GML查询处理器的设计与实现[C]//第二十五届中国数据库学术会议(NDBC2008). 桂林，中国，2008

[8] 兰小机，闻国年，刘德儿，等. 基于XQuery的GML查询语言研究[J]. 测绘科学，2005,30(6):99-102

[9] 罗静，陈宁. 一种分布式环境中海量XML数据的有效查询机制[J]. 重庆交通大学学报：自然科学版，2009,28(4):807-812

[10] Fegara L,Li C,et al. XML Query Optimization in Map-Reduce [C]//WebDB'11,14th International Workshop on the Web and Database. Athens,Greece,2011

[11] Borthakur D. HDFS Architecture[OL]. http://hadoop.apache.org/common/docs/.20.0/hdfs design.html,April 2009

[12] Apache. Apache Hadoop[OL]. http://hadoop.apache.org/

[13] White T. Hadoop,the definitive guide [M]. O'Reilly Media,Inc,2009