

一种基于 MapReduce 的 XML 查询的优化方法

魏永山, 张 峰, 陈 欣, 孙忠林

(山东科技大学 信息科学与工程学院, 山东 青岛 266590)

摘 要: 基于 MapReduce 的 XML 查询逐渐成为研究热点, 如何提高 XML 文档查询的效率是关键问题之一. 在保障查询结果正确和 XML 数据分块一致的前提下, 采用 XPath 投影的方法对 XML 数据分块预处理, 使得 XML 分块中仅包含与查询有关的数据. 实验表明, 该方法有效缩小了数据分块的规模、缩短了 XML 数据服务的查询处理时间.

关键词: MapReduce; XPath 投影; XML 查询; 查询优化

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2013)03-0102-04

A Optimization Method for XML Query Based MapReduce

WEI Yong-shan, ZHANG Feng, CHEN Xin, SUN Zhong-lin

(College of Information and Engineering, Shandong University of Technology and Science, Qingdao 266590, China)

Abstract: Research on xml query based MapReduce gets more and more attentions. It is an important issue to promote performance of xml query. In order to reduce split size of xml documents, xml splits are pre-processed by xpath projection in the precondition of correct query result and consistency of xml splits. Experiments show that the optimization method effectively reduces query processing time of xml data.

Key words: MapReduce; XPath projection; XML query; query optimization

1 引言

企业在运营过程中积累了大量数据, 许多数据会以 XML 文档形式存在. 通过分析这些大量数据, 企业可以提高服务质量、提供更优的选择、发现欺诈活动, 使得大规模数据挖掘变得可行. 面对 XML 大数据量的处理需求, 在云计算环境中提供 XML 数据查询逐渐成为研究热点.

MapReduce 编程模型^[1]是一个可以在云计算环境中进行并行数据分析的框架, 利用由普通计算机组成的、通过高速网络连接的无共享集群, 执行大规模数据分析任务. 不同于并行数据库中, 在处理数据之前需要程序员先建模、加载数据, MapReduce 模型更适合对一次性写入数据的单次即时查询. 与传统的 DBMS 相比, MapReduce 实现了更好的容错性和异构环境的操作性.

当前基于 MapReduce 的 XML 查询优化注重研究通过变换或重写的优化方法. 从数据规模角度考虑, 减少集群节点间的数据传输也可以有效提高查询性能, 在基于 MapReduce 的 XML 查询过程中, 在分块阶段可以减少数据源 XML 文档分块的数据规模, 中间数据传输阶段也可以减少中间数据的数据规模.

本文从减少数据规模角度出发, 在保障查询结果正确和 XML 数据分块一致的前提下, 采用基于 XPath 投影的方法对 XML 数据分块预处理, 从而达到减少分块数据规模的目的. 本文的主要贡献有:

(1) 通过分析查询语句中的变量引用, 获得 XPath 集合, 将此 XPath 用于指导 XML 数据的动态分块;

(2) 以 XPath 集合为基础对 XML 数据分块预处理, 减少分块中的数据规模和中间数据的规模.

收稿日期: 2012-06-08; **修回日期:** 2012-07-12

基金项目: 山东省自然科学基金(ZR2011FQ028); 山东省优秀中青年科学家科研奖励基金(BS2010DX009)

2 基于 MapReduce 的 XML 查询

在云计算环境中提供 XML 数据查询能力,存在两种实现方式:(1)在云计算环境中引入已有的 XML 查询处理引擎,比如 XQuery 引擎 Saxon,(2)在 MapReduce 框架之上重新构造 XML 查询处理引擎。

在云计算环境中引入已有的 XML 查询处理引擎可以减少开发工作量,在 Map 函数中调用已有的 XQuery 引擎执行 XQuery 查询语句处理 XML 数据分片(Split),在 Reduce 函数中合并 Map 函数处理的结果并做汇聚运算。其优点是不必重新实现 XQuery 引擎,但也存在不容易实施针对 MapReduce 框架的优化等缺点。在 MapReduce 之上重新构造 XML 查询处理引擎需要重新实现引擎的核心操作,明显的优点是可以针对 MapReduce 等框架实施查询优化。

MRQL^[2]采用第二种方式实现云计算环境中 XML 数据查询。MRQL 提供一种类似 SQL 的描述式 XML 查询语言,采用两种方式执行 MRQL 查询:单机模式和集群模式。MRQL 通过扩展数据源接口可以支持多种数据资源,比如 XML, CSV 和 JSON 等数据类型。由于其易扩展性,可以很方便的支持不同的 XML 分块方法,因此本文选择 MRQL 为基础实现基于 XPath 投影的数据分块算法。

3 基于 XPath 投影的数据分块算法

动态的 XML 分块方法的基础是 XPath 投影。而 XPath 需要通过对 XML 查询语句获得。本小节首先给出针对给定的 XML 查询的 XPath 分析算法,再讨论基于 XPath 的动态 XML 分块算法。

3.1 XPath 分析算法

XML 查询的投影算法是由文献[3]提出。主要思想是通过对 XML 查询语句的分析得出查询语句所涉及到的 XML 文档结构,然后在执行 XML 查询过程中,为查询引擎提供影响查询结果的数据,而不必包含对查询结果无贡献的数据。

MRQL 中采用 λ 语言作为查询的中间变换语言,因此本文以 XML 查询的 λ 表达式为基础分析 XPath。基本思想是先对 λ 语言表示的 XML 查询进行 β -规约,将 XML 变量重写为对包含全部路径的表示方式,再搜集所有的对 XML 变量的引用,再通过合并 XPath 变量得到 XPath 集合。本质上是统计对 XML 变量直接引用和间接引用的路径中出现的

路径标记,再统计在各类变量和表达式中出现的路径标记,最终形成具有层次关系的路径和标记结构,得到的 XPath 集合实际是 XML 文档模式的一个子模式。以下面的类 SQL 查询说明算法的思想。

```
select( d. name, sum(select e. salary
from e in Employees
where e. dno=d. dno) )
from d in Departments
```

Select 语句的功能是按照 Departments 统计薪水的和,转换为 λ 语言表示是:

```
cmap(  $\lambda$ d. d. dno,
 $\lambda$ e. e. dno,
 $\lambda$ (ds, es). cmap( $\lambda$ d. {( d. name,
sum(cmap( $\lambda$ e. {e. salary}) es) )}
ds )
(Departments, Employees)
```

采用 β -规约方法将 λ 表达式中的 Departments 和 Employees 分别代入,得到

```
cmap(  $\lambda$ Departments. d. dno,
 $\lambda$ Employees. e. dno,
 $\lambda$ (ds, es). cmap( $\lambda$ Departments. {( Departments. d.
name,
sum(cmap( $\lambda$ Employees. {Employees. e. salary}) es)
)})
ds )
```

此表达式中形如 Departments. d. name 的即是所求的 XPath 路径,然后收集规约后的 λ 表达式中出现的 XPath 路径,可以得到:

$$SET_{XPath} = \{\{ Departments. d. dno, Departments. d. name \}, \{ Employees. e. dno, Employees. e. salary \} \}$$

SET_{XPath} 的逻辑意义可以解释为,在计算此 λ 表达式的过程中,仅与 SET_{XPath} 中包含的 XPath 对应的数据有关,其他数据可以不参加计算,而不影响最终的计算结果。因此在对 XML 数据分块时,可以以 SET_{XPath} 为基础对 XML 数据分块做预处理投影,在确保 XML 数据分块完整性的前提下,只保留 XPath 集合中的数据路径,以减少 XML 数据分块的规模。XPath 方法的正确性在文献[3]中有所论述。

在 λ 表达式之上进行 XPath 路径分析的算法见算法 1,采用递归方式遍历 Tree,最后获得 XPath 列表。

算法 1 XPath 路径分析算法 xPathAnalysis.

输入: Tree; // 树形结构的 λ 表达式

Map<String, Object> V; // lambda 变量集合,初始为空

```

输出:List xPaths; //XPath 列表
Begin
For EACH node IN Trees.Children Begin
IF (node is variable) Then Begin
V.put(node.name, node);
End Else if (node is reference) Then Begin
For Each variable IN node Begin
v = replace(variable, V.get(variable.name));
IF Not Exists Variable IN v Then
Begin XPaths.add(v)
End
Eed Else (node is tree) Begin
xPathAnalysis (node, V, xPaths)
End;
End

```

3.2 XML 文档分块算法

通过对 λ 表达式的 XPath 路径分析可以获得与查询相关的数据节点以及属性,需要在对 XML 分块时以算法 1 得到的数据节点以及属性控制分块并减少数据规模。

算法 2 用于将 XML 文档分割为多个 Split,使得每个 Split 保持标记的封闭性和完整性,且块中不包含对查询结果没有影响的 XML 标记和数据。其中 schema 是由算法 1 获得的 XPath 组成的模式。

算法 2 XML 文档分块算法。

输入:XML 文档 D,XML 文档子模式 schema

输出:XML 数据块

```

Begin
While (node != null) Begin
find node tag in D of schema;
buffer.append(D[nod])
node = schema.getNextNode();
End;
Return buffer
End

```

4 实验与评价

实验分为单机实验和集群实验两组。实验数据是 DBLP 的 1G 的数据集 dblp.xml^[4]。实验测试对象有:mrql 以及采用改进分块算法的 mrql-f。

单机和集群中的实验都采用相同的测试数据和查询语句。通过三类查询的实验,分别在 mrql 和 mrql-f 中执行相同的查询,且采用相同数据格式的 dblp.xml 文档。这三个查询具有一定的代表性:查询(a)从 dblp.xml 选择一条 article 记录,查询(b)的返回结果

有 14023 条数据记录并要求排序,查询(c)主要测试查询的分组统计功能。如图 1 所示。

```

select x.author
from x in source(xml, "/dblp.xml", {"article"})
where x.@key = "journals/cacm/Titus67b";
(a)查询单条记录

select x.author, x.@mdate, x.@key
from x in source(xml, "/dblp.xml", {"article"})
where x.@mdate = "2002-01-03"
order by x.@key;
(b)查询 2002-01-03 日期的 article 并按照 key 排序

select (x, count(a))
from a in source(xml, "/dblp.xml", {"article"})
group by x: a.@mdate;
(c)按照日期统计 article 的数量

```

图 1 测试查询语句

4.1 单机实验比较

单机的测试环境是: Intel E7200 2.53GHz, 内存 3G, JVM 基本配置是 Xms512M, Xmx1600M。

单机实验结果如图 2 所示。从图中可以看出,采用改进的分块方法的 mrql-f 比 mrql 的查询时间要少,查询(a)的时间减少 21.9%,查询(b)的时间减少 20.7%。对于查询 c,由于在 mrql 中执行时内存溢出因此没有相应的数据,而在采用改进的分块方法的 mrql-f 中执行时没有溢出问题,说明在单机环境中改进的分块算法可以有效减少数据规模。

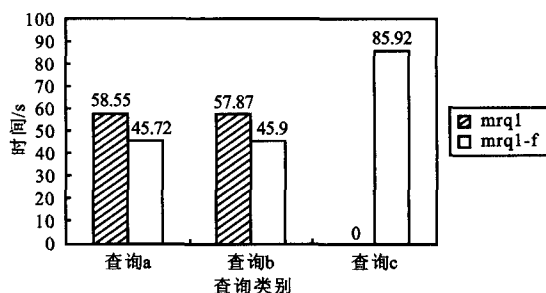


图 2 单机环境下 mrql 与 mrql-f 的实验比较

4.2 集群试验比较

实验在四台普通 PC (Intel P4 2.8GHz, 内存 2G, Ubuntu 10.04, 千兆网卡连接) 组成的 Hadoop 集群 (一台 namenode, 三台 datanode 和 jobtracker) 上完成。JVM 基本配置是 Xms512M, Xmx1024M。

集群环境中的实验测试结果如图 3 所示。查询(a)和查询(b)两者的查询时间相差不大, mrql-f 的查询时间稍小一些,但在查询(c)中 mrql-f 比 mrql 要节省 21.7% 的时间。

单机环境中的查询时间相比集群环境中的查询

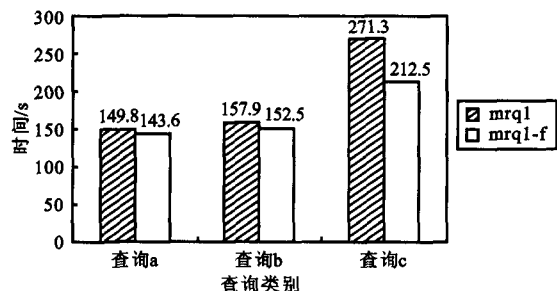


图3 集群环境中 mrql 和 mrql-f 的部分查询时间对比
时间要短,主要原因可能有以下几点:(1)集群规模较小,1G的dblp数据集默认被分为16个map任务处理,由于实验采用的Hadoop集群规模较小,因此没能有效发挥并行处理效能;(2)数据规模较小,数据规模并未超出单机的处理能力,当数据规模远大于单机的处理能力时,集群中查询时间应该比单机的时间要小,仍然需要在更大规模的集群并采用更大的数据集的实验中进一步的验证。(3)集群配置与单机配置差异,当然这并不是主要因素。

本文仅测试了简单查询的对比实验,针对复杂查询(比如多表连接、PageRank等查询)将在进一步的研究工作中完成。

基于投影的数据分块算法依靠对XML数据的投影减少被处理的数据规模减少查询时间,对不返回全部数据属性的查询有明显效果,如图1(a)所示,select子句中仅包含x.author属性,而不关心其他属性,如果仅包含x,则基于投影的数据分块算法也无法有效减少分块的数据规模。

5 相关工作

基于MapReduce的XML数据查询相关的研究工作主要有Xadoop^[5],ChuQL^[6]和MRQ^[2]。

Xadoop是在Hadoop之上的支持XML数据查询的一个框架,结合已有的XQuery引擎表示MapReduce程序,当前仅支持以行为单位数据分块方法,且查询执行效率较低。

ChuQL采用混合的方式表示对XML的查询,在XQuery查询语句中显示指定在Map或Reduce中处理。而MRQL是类似于SQL的一种表达查询方式,支持嵌套数据结构的访问。

MRQL提供一种类似SQL的语言操纵XML数据,提供较为完整的XML数据操作功能,基于MapReduce构造编译器和优化器。

本文工作在MRQL的基础上改进了其XML

分块算法,借鉴XPath投影的方法减少数据分块的规模,使得查询时间减少。其他常见的MapReduce框架上的数据处理语言,如Hive^[7],Pig^[8],Sawzall^[9]等,但都不直接支持XML数据处理。

6 结束语

本文引入基于投影的分块算法减少XML数据分块的规模和中间数据的传输规模,有效提高了XML数据服务的查询处理效率。实验表明针对简单查询,单机实验环境中,采用改进分块算法的mrql-f与mrql相比,同等条件下大约可以缩短20%的查询时间。在进一步的工作中,如何将基于投影的分块算法应用于更多的查询类型、更大规模的集群、更丰富的数据集是下一步工作的重要内容。

参考文献:

- [1] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1):107-113.
- [2] Fegaras L, Li C, Gupta U, et al. XML query optimization in map-reduce[C]// [s. l.]: IEEE. WebDB'11, 2008.
- [3] Marian A, Siméon J. Projecting XML documents[C]// VLDB Endowment, [s. l.]: IEEE, 2003: 213-224.
- [4] DBLP. [EB/OL]. [2012-04-20]. <http://dblp.uni-trier.de/xml/>.
- [5] Xadoop. [2012-04-20]. <http://www.xadoop.org>.
- [6] Khatchadourian S, Consens M, Siméon J. ChuQL: processing XML with uery using hadoop[C]// IBM Corp. [s. l.]: IEEE. 2011: 74-83.
- [7] Thusoo A, Sarma J S, Jain N. Hive: a warehousing solution over a map-reduce framework[J]. Proceedings of the VLDB Endowment, 2009, 2(2): 1626-1629.
- [8] Gates A F, Natkovich O, Chopra S. Building a high-level dataflow system on top of map-reduce: the pig experience[J]. Proceedings of the VLDB Endowment, 2009, 2(2): 1414-1425.
- [9] Pike R, Dorward S, Griesemer R, et al. Interpreting the data: parallel analysis with sawzall[J]. Scientific Programming, 2005, 13(4): 277-298.

作者简介:

魏永山 男, (1977-), 博士, 讲师. CCF 会员 (No. E200014902M). 研究方向为大规模数据处理、服务计算。