XQuery 中 FLWOR 式的查询重写研究

席凤磊[1],毛宇光[1,2],廉成洋[1]
(1. 南京航空航天大学 信息科学与技术学院,江苏 南京 210016;
2. 南京大学 计算机软件新技术国家重点实验室,江苏 南京 210093)

摘 要:FLWOR 表达式由 for、let、where、order by 和 return 子句组成,它的重写对 XQuery 查询优化起着重要的作用。对它的研究先前主要在 FLWR 表达式上,然而缺少了对 order by 子句的探讨,此子句是对返回节点集内的数据进行排序,并且执行是比较复杂和耗费时间的。因此,文中按 for 子句循环嵌套内和嵌套外把 order by 子句分成 3 种情况进行讨论,并提出了它与其他子句结合后如何对 FLWOR 表达式进行重写的方法,目的是减少节点排序的多个不必要的重复执行。实验结果表明重写后的语句执行时间总是优于之前的。

关键词:FLWOR;查询优化;重写;XQuery
中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2009)06-0025-04

Query Rewriting of FLWOR Expressions in XQuery

XI Feng-lei[1], MAO Yu-guang[1,2], LIAN Cheng-yang[1]
(1. College of Information Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China;
2. State Key Laboratory for Novel Software Technology at Nanjing University, Nanjing 210093, China)

Abstract: The expression of FLWOR, whose rewriting has an important impact upon optimizing XQuery, is made up of clauses of For, Let, Where, Order By and Return. The previous research has focused on the expression of FLWR while lacking enough discussion on the clause of Order By. The clause is designed to sort the data in returned node-set. Besides, it is complicated and time-consuming to carry out this clause. Hence, divides the clause of Order By into three situations according to the inner and outer of For loop, then discusses them respectively. In addition, puts forward the method of rewriting the expression of FLWOR since the clause of Order By is combined with other clauses, which is intended to reduce the unnecessary repeated execution. The result of the experiment shows that the executing time after rewriting the clause always excels the previous one.

Key words: flwor; query optimization; rewriting; XQuery

0 引 言

随着 Internet 的发展,XML(eXtensible Markup Language)已经成为网络上信息描述和信息交换的标准。许多用户开始使用大量的 XML 文档,为了查询这些 XML 文档,出现了许多查询语言,其中 XQuery 适合于各种类型的 XML 数据源,是一种功能极强的查询语言,因此,很快就成为了具有代表性的 XML 数据查询语言。它是在多种语言的基础上开发形成的,具有过程化程序语言、SQL 数据库查询语言等的优点,具有灵活方便、符合 XML 数据特点、功能全面等优点,且对 XML 的作用类似 SQL 对数据库的作用[1]。XQuery 主要由模块声明、函数声明、变量声明和表达式组成[2],其中表达式又由 FLWOR 式、基本式、路径式、算术式、序列式和一些构造体组成。在表达式中主要是 FLWOR 表达式,它由 for 子句,let 子句,where 子句,order by 子句,return 子句组成。例如下面的 FLWOR 表达式:

```
for $x in doc("books.xml")/bookstore/book
where $x/price > 30
order by $x/title
return $x/title
```

这个 FLWOR 表达式的执行是从 XML 文件 books.xml 中检索书的价格大于 30 的书名,得到的结果再按书名排序。文中的研究也主要对上述表达式中

收稿日期:2008-09-17;**修回日期**:2009-02-20
基金项目:国家计算机软件重点研究基金(A200711)
作者简介:席凤磊(1979-),男,江苏徐州人,硕士研究生,研究方向为数据库系统、XML 数据库;毛宇光,副教授,博士后,硕士生导师,研究方向为数据库系统及理论、数据挖掘与数据仓库。

的 for,order by,return 子句,还有 let 子句的查询重写,核心内容主要为以下两点:

·对 order by 子句重写的分类讨论。

·整体考虑 FLWOR 表达式的重写。

1 相关工作

XML 已经成为网上信息描述和信息交换的标准,与此同时,它的查询优化也是数据库领域研究的一个重要课题,文献[3]综合论述了 XML 查询优化技术的现状,指出了它的特点和研究的关键性问题,并描述了物理和逻辑优化各个方面的重要研究成果。然而文献[4]主要针对 XQuery 查询优化技术进行研究,在已有的 XQuery 查询语句的基础上,对它进行逻辑上的优化,然后通过重写机制进行重写,生成 XQuery 语句,最后将重写后的语句进行最小化处理,以达到优化的目的。

XML Schema 是用来描述 XML 结构、约束等因素的语言,许多 XML 文档都有一个 Schema,文献[5]研究了如何按照 Schema 减少不可能的路径表达式或者删除多余的查询判断条件优化 XQuery。他们的研究主要偏向于除 XQuery 本身之外的信息,使用这些信息来优化查询。

Song Wang 和 Elke A. Rundensteiner[6]等对嵌套的 XQuery 表达式的优化进行研究,主要从查询代数上来考虑的,先对 XQuery 标准化,然后再转化成 XAT 代数,对其中的表达式进行解相关,最小化,以至达到优化的目的。

Jong-Hyun Park[7]主要从查询重写规则上优化 XQuery,介绍了 3 个查询重写规则,分别进行讨论,他们的方法不依赖于任何查询引擎和中间件,不会对语义进行修改,最终的目的是减少由 for 子句产生的多次重复执行。然而,文中只研究了 XQuery 中 FLWR 表达式的查询重写,没有考虑 order by 子句的重写,以及含有 order by 子句后,对其他子句的重写方法的影响。

2 FLWOR 式的查询重写规则

对 XQuery 的查询优化主要讨论了如何对 XQuery 中 FLWOR 表达式的查询重写,分别从 for,let,where,order by,return 子句进行考虑,以及它们互相的影响。下面几节将分别对它们进行讨论。

2.1 FOR 子句中变量的分类

在 FLWOR 表达式中,for 子句中的变量是否在其他子句中出现是非常重要的,它关系到应对哪些子句进行重写。下面详细对 for 子句进行分类:

1) for 子句中的变量在 where,order by 和 return 中出现;

2) for 子句中的变量在 where,order by 中出现;

3) for 子句中的变量在 where 和 return 中出现;

4) for 子句中的变量只在 where 中出现;

5) for 子句中的变量在 order by 和 return 中出现;

6) for 子句中的变量只在 order by 中出现;

7) for 子句中的变量只在 return 中出现;

8) for 子句中的变量在 where,order by 和 return 中都未出现。

其中 3)、4)、7)和 8)的情况与文献[7]中的情况类似,并且文献[7]中还把 for 子句循环的位置分为最外层、中间层和最内层,而文中,不再对 for 子句循环的位置进行分类,因此,就以上给出的 8 类情况,在 2.2 节中将使用实例来详细阐述它们的重写方法。其中,C1($ x)表示 where 中第一个判断条件内含有 x 变量,Xquery1 表示 for 子句中的变量在 where, order by 和 return 中出现,也就是第一种情况。

2.2 FLWOR 式的详细分类

FLWOR 式的详细分类见图 1。

1	● xquery1 for $ x in Expr1 for $ y in Expr2 where C1($ x) and C2($ y) order by $ x return $ x	5	● xquery5 for $ x in Expr1 for $ y in Expr2 where C($ y) order by $ x return $ x
2	● xquery2 for $ x in Expr1 for $ y in Expr2 where C1($ x) and C2($ y) order by $ x return $ y	6	● xquery6 for $ x in Expr1 for $ y in Expr2 where C($ y) order by $ x return $ y
3	● xquery3 for $ x in Expr1 for $ y in Expr2 where C1($ x) and C2($ x, $ y) order by $ y return $ x	7	● xquery7 for $ x in Expr1 for $ y in Expr2 where C($ y) order by $ y return $ x
4	● xquery4 for $ x in Expr1 for $ y in Expr2 where C1($ x) and C2($ x, $ y) order by $ y return $ y	8	● xquery8 for $ x in Expr1 for $ y in Expr2 where C($ y) order by $ y return $ y

图 1 FLWOR 式的详细分类

在图 1 中,对各种情况进行了详细分类,其中的 4 种情况已经在文献[7]中研究,文中不再讨论,这里主要讨论 order by 子句内出现 for 子句中变量的情况。

2.3 ORDER BY 子句的重写方法

文献[6]介绍了一些重写方法,主要从查询代数、标准化上来考虑的,在 XATTable 连接时,对底层的 order by 操作进行向上层抽取,而文中的查询重写 order by 子句是不依赖于任何查询引擎的。因此,对 order by 子句的考虑分为 for 循环嵌套内层和嵌套外层,

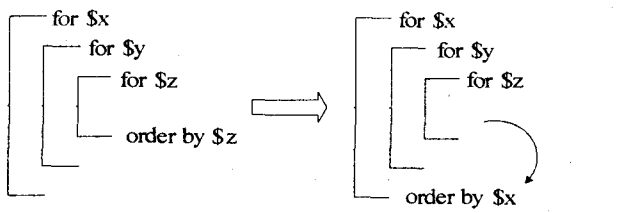
目的是减少多次重复执行的排序操作。

下面分三种情况重写 order by 子句:

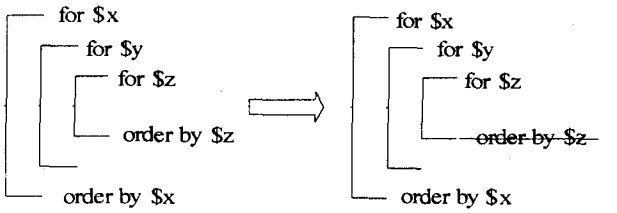
1) 如果 for 循环嵌套外层没有 order by 子句,内层有,则需要抽取到嵌套外层,因为外层的集合元素总是小于或等于内层集合元素的,见图 2(a)。

2) 如果嵌套内外层都有 order by 子句,则不需要抽取,而直接删除内层的 order by 子句,见图 2(b)。

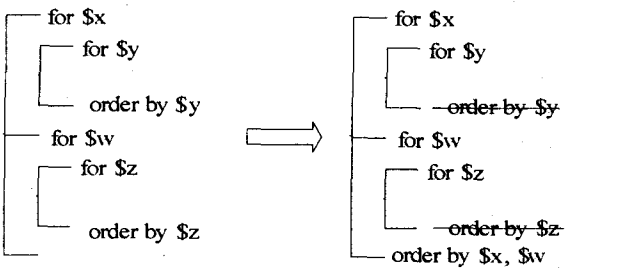
3) 如果两个嵌套内 for 子句都有 order by 子句,则向嵌套循环外层抽取 order by,见图 2(c)。



(a) 重写order by第一种情况



(b) 重写order by第二种情况



(c) 重写order by 第三种情况

图 2　重写 order by 的三种情况

2.4　FLWOR 式的 4 类重写规则

对 2.2 节中的 8 种情况进行组合可以得到 4 类重写规则,如下:

1) for 子句的重写规则;

2) let 子句的重写规则;

3) order by 的重写规则;

4) return 子句的重写规则。

在这一节中,用例子详细阐述这 4 类重写规则,对于仅对 let 和 return 子句的重写,文献[7]已经详细描述,这里不再讨论,下面举一个例子,其中包含 for 和 order by 子句的重写规则,在这里,结合 2.3 节的重写方法,分两种情况进行讨论,第一种情况的重写步骤:

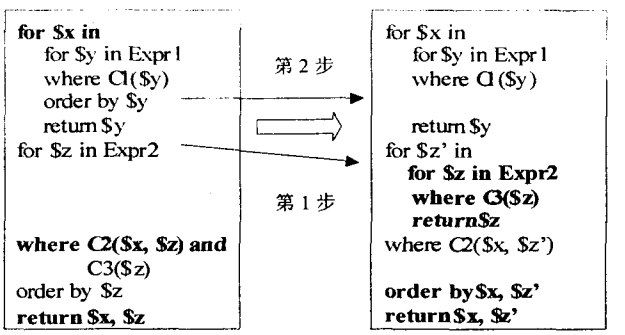
第 1 步:对变量 $z 所在的 for 子句进行重写。

第 2 步:使用 2.3 节的 3 种方法对 order by 子句重写。

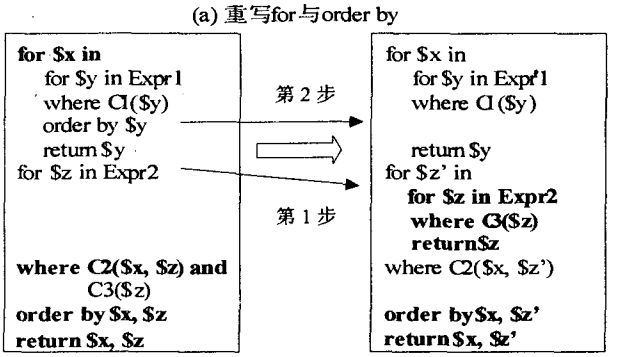
具体重写见图 3(a)。

以下是第二种情况,最外层的 order by 子句内含有 for 子句的变量 $x,因此,不管 return 子句中有没有变量 $x,都直接删除嵌套内的 order by $y 子句,见图 3(b)。

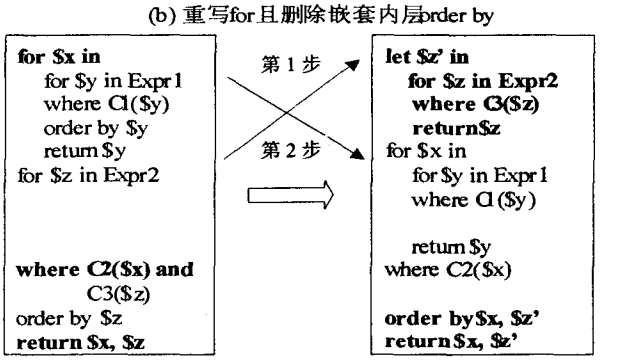
图 3(c)的情况,是 where 中没有变量 $z 的处理方法,其中含有 let 子句的重写。



(a) 重写for与order by



(b) 重写for且删除嵌套内层order by



(c) 重写let与order by

图 3　order by 子句的重写

3　实　验

在这一节里,通过实验对上述的重写进行验证,使用的硬件为 CPU 迅驰单核 1.7GHz,主存 1G,操作系统为 Windows XP。因查询重写不依赖于任何查询引擎,所以选择了 IBM 公司的 DB2 9.5 版本的数据库管理系统作为软件支持。数据是两个 XML 文档:一个是存在表 BOOKS2 中的 BOOKST 字段里,含有 500 个 BOOK 要素节点;另外一个存在表 AUTHOR1 中的 AUTHXML 字段里,含有 60 个 AUTHOR 要素节点。

以上是实验所需的数据，下面就输入语句执行来比较它们的执行时间。首先在命令编辑器内输入命令设置一个事件监视器 dbevent 用来检测语句的执行时间；然后使用 2.4 节的查询重写规则对源语句进行重写并且执行，经检验，得出的输出结果是相同的。图 4 中的语句是程序员输入的原始语句。

```
for $x in
    for $y in db2 - fn: xmlcolumn ( 'BOOKS2. BOOKST')/
bookstore/book
    where $y/price > 30
    order by $y/title
    return $y
for $z in db2 - fn: xmlcolumn( 'AUTHOR1. AUTHXML')/au-
thordept/author
where $x/author = $z/name and $z/age < 31
order by $z/birthdate
return <rslt> | $x/title, $z/birthdate | </rslt>
```

图 4 XQuery 查询语句

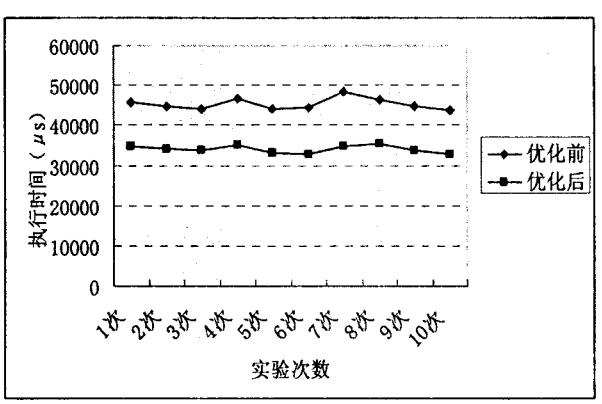


图 5 实验结果

图 5 显示出了实验结果，横坐标表示查询语句执行次数，纵坐标表示查询执行的时间（μs），通过对 order by 子句重写后，正方形代表了重写后的 XQuery 语句执行时间，从图可以看出，经过对 for 子句和 order by 子句的重写后，执行时间总是低于之前的 25%～30% 左右，由此证明我们的重写方法是可行的。

4 结束语

提出了 XQuery 查询优化时对 FLWOR 表达式的重写规则，包括 for，let，return，order by 子句的分别重写规则，其着重介绍了含有 order by 子句的重写方法，这也是扩展了先前学者的研究，从而在 FLWOR 表达式的整体上考虑了重写规则。以上提出的这种对 XQuery 查询优化的方法，是不依赖于任何数据库查询引擎的，它可以适用于任何支持 XQuery 查询标准的 XML 数据库中。最后，通过使用 IBM 公司的 DB2 进行实验，也证明了我们的查询优化方法是正确和可行的。然而，这里只考虑了在 XQuery 查询里的 FLWOR 表达式的重写规则，在未来的研究中，还需要更深入研究 XQuery 查询里的其他表达式的查询重写方法，例如验证式和序列表达式等。

参考文献：

[1] W3C. Introduction to XQuery[EB/OL]. 2001. http://www.w3schools.com/xquery/xquery_intro.asp.

[2] W3C. XQuery 1.0: An XML Query Language[EB/OL]. 2007. http://www.w3.org/TR/xquery.

[3] 孟小峰，王 宇，王小锋. XML 查询优化研究[J]. 软件学报，2006，15(11)：2069－2086.

[4] 周 正. 基于重写机制的 XQuery 查询优化技术研究[D]. 南昌：江西财经大学，2006.

[5] Kong A，Gertz M. Schema－based optimization of XPath expressions[R]. USA：department of Computer Science，University of California，2001.

[6] Song W，Rundensteiner E A，Mani M. Optimization of Nested XQuery Expressions with Orderby Clauses[J]. Data & Knowledge Engineering，2007，60(2)：303－325.

[7] Jong－Hyun P，Ji－Hoon K. Optimization of XQuery Queries including FOR Clauses[C]//In Proceedings of Second International Conference on Internet and Web Applications and Services. Washington，DC，USA：IEEE Computer Society，2007.

（上接第 24 页）

参考文献：

[1] Heaton J. 网络机器人 JAVA 编程指南[M]. 童兆丰，李纯，刘润杰译. 北京：电子工业出版社，2002.

[2] 叶勤勇. 基于 URL 规则的聚集爬虫及其应用[D]. 杭州：浙江大学计算机软件学院，2007.

[3] 邱 哲，符滔滔. 开发自己的搜索引擎[M]. 北京：人民邮电出版社，2007.

[4] 汪 涛，樊孝忠. 链接分析对主题爬虫的改进[J]. 计算机应用，2004，24：174－176.

[5] 斯图尔特 G W. 矩阵计算引论[M]. 王守根等译. 上海：科学技术出版社，1980.

[6] 徐树方. 矩阵计算的理论与方法[M]. 北京：北京大学出版社，1995.

[7] 俞 辉，赵玉国. 基于 LSA 和 PLSA 的网页聚类算法研究[J]. 计算机应用，2008(4)：65－68.

[8] 朱志国，孔立平. 面向电子商务的 Web 使用挖掘技术应用研究[J]. 计算机技术与发展，2008，18(6)：228－232.

[9] Han Jiawei，Kamber M. 数据挖掘概念与技术[M]. 范 明，孟晓峰等译. 北京：机械工业出版社，2001.

[10] 吴 瑛，王秋生. 模糊 C 均值聚类算法在 Web 使用挖掘上的应用研究[J]. 计算机技术与发展，2008，18(6)：32－35.