# XQuery 实现技术研究综述

李小青 廖湖声 张晓博
(北京工业大学计算机学院 北京 100124)

**摘 要** XML 数据的广泛应用,使得高性能的 XQuery 实现成为 XML 数据处理领域的重要课题,但 XQuery 的灵活性和复杂性为其实现技术研究提出了巨大挑战。XQuery 语言的高性能实现需要利用 XML 查询代数提供的查询优化方法,也需要采取高效的树模式整体匹配算法。给出了 XQuery 语言实现的基础架构,探讨了原生 XML 数据库系统中 XQuery 实现的关键技术——查询代数和树模式查询的国内外研究现状,展望了未来的研究方向及面临的挑战。

**关键词** XML,XQuery 语言,查询代数,树模式
**中图法分类号** TP311 **文献标识码** A

## Survey of XQuery Implementation

LI Xiao-qing LIAO Hu-sheng ZHANG Xiao-bo
(College of Computer Science, Beijing University of Technology, Beijing 100124, China)

**Abstract** As increasing amounts of information are stored, exchanged, and presented using XML, the ability to query XML data sources becomes increasingly important in XML database field. The flexibility and complexity of XQuery language are the great challenge to XQuery implementation. High-performance implementation of XQuery needs to use query optimization methods provided by XML query algebra, also needs to use efficient holistic twig matching algorithm. This paper presented a basic architecture of XQuery implementation, and then gave an overview of the current status of two key technology for native XQuery implementation——XML query algebra and tree pattern matching. Finally, this paper prospected future research directions and presented some viewpoints of XQuery implementation.

**Keywords** XML, XQuery language, Query algebra, Tree pattern

## 1 引言

XML(Extensible Markup Language) 可扩展标记语言凭借其内容与形式分离、易于扩展和平台无关等诸多特点,被广泛地应用于分布式计算、电子商务、电子政务等很多领域,成为互联网环境中数据交换和共享的事实标准。如何有效地实现对 XML 数据的查询,则成为当前 XML 数据处理研究领域的重要课题[1,2]。目前主流 XML 数据的查询语言是 W3C 发布的 XPath 和 XQuery 语言(XML Query Language)。相对于 XPath 语言,XQuery 语言具有更强大的查询描述能力,在 XML 数据库领域得到越来越普遍的重视。人们正在研究数据集成、XML 流数据、移动环境和 XML 数据库等环境下 XQuery 语言的优化实现技术。

现有的 XQuery 查询实现研究大致可以分为两类:基于关系的 XQuery 查询实现和原生的 XQuery 查询实现[3]。基于关系的 XQuery 实现技术一般应用于基于关系的 XML 数据库中,其基本思想是将 XQuery 查询转换为 SQL 查询,在底层的关系数据库中执行查询计划,最后将返回的关系元组重新组织成为 XML 片段;原生 XQuery 实现技术针对原生 XML 数据库,其基本思想是为 XQuery 查询设计一套查询代数,将 XQuery 查询转换成代数表达式,进行各种优化后生成与原 XQuery 程序等价的查询计划,在查询引擎上执行并返回查询结果。

本文主要探讨原生 XQuery 实现技术。XML 数据的半结构化特征以及 XQuery 语言的函数式特征,使得 XQuery 语言的实现技术面临很多挑战。首先,研究者提出了多种查询代数,用于 XQuery 查询请求的处理与优化;其次,针对 XQuery 查询请求的树模式特征,探讨这种核心查询操作的实现算法。将高效的树模式查询算法应用于 XML 查询代数计算,是实现高性能 XQuery 的重要方法之一。本文第 2 节给出了原生 XQuery 语言实现的基础架构;第 3 节介绍并分析了当前 XML 查询代数和树模式查询算法的发展情况;最后对 XQuery 实现相关技术的发展进行了展望。

## 2 XQuery 语言实现架构

文献[4]根据关系数据库的查询处理思想,将 XML 查询处理分为 4 个大阶段。由于 XML 数据结构的复杂性以及 XQuery 语言的函数式特征,本文将 XQuery 语言的实现划分

到稿日期:2011-04-19 返修日期:2011-07-17 本文受北京市自然科学基金(4082003)资助。
李小青(1983—),女,博士生,主要研究方向为 XML 数据库技术,E-mail:li_xiaoqing@emails.bjut.edu.cn;廖湖声(1954—),男,教授,博士生导师,主要研究方向为程序理论及编译技术、XML 数据库技术等;张晓博(1986—),男,硕士生,主要研究方向为 XML 数据库技术。

为3大阶段，如图1所示。

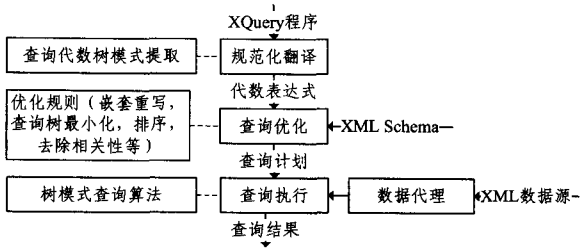


图1 XQuery语言实现架构

规范化翻译阶段，将XQuery查询请求翻译为某种XML查询代数表示的内部表达方式，同时进行树模式提取，以达到分离表达式求值与数据抽取的目的，便于下一步的逻辑优化以及最终的查询执行。

查询优化阶段，重点是基于查询代数的优化，即制定一系列优化规则，将翻译后的查询代数表示的XQuery查询请求根据规则进行程序变换，实现查询重写，从而得到优化的查询计划。典型的优化规则包括XQuery查询请求的嵌套重写及最小化、排序上浮以及去除相关性等，从而达到易于查询的高性能实现以及便于任务划分、代价计算和分析等目的。同时，可以根据数据源的模式信息，制定相应的优化规则，简化计算。

查询执行阶段，将优化后的查询计划传入执行引擎进行求值计算，得到查询结果。该阶段需要高性能查询算法的支撑，以便快速地从复杂的XML数据中提取出数据信息。由于采用整体模式匹配的树模式查询算法展现出较大优势，因此在查询代数中导入树模式是十分必要的。

## 3 XQuery实现技术研究进展

作为W3C推荐标准，XQuery语言的实现技术研究得到了广泛的重视。研究者们从树模式查询算法、查询代数、树自动机[5,6]、索引技术[3,7]、查询优化[4,8-12]等各个方面，研究提高XML查询性能的方法。还有学者将XML数据看作单词符号序列，采用正规式类型[13,14]来描述XML数据的树结构特征，为静态类型化XML处理提供了理论依据。如XDuce[15]、CDuce[16]等语言的正规式模式匹配机制为XML数据处理逻辑的描述提供了强大的手段，并已用于XQuery语言的实现当中。同时，针对XQuery语言的并行实现技术[17]、字节码编译技术[18]等也得到了很大的发展。本文主要对树模式查询和查询代数两大关键技术的研究现状进行分析和概括。

### 3.1 XML查询代数

XML查询代数是对遵循一定数据模型的XML文档集合的操作集。XML查询代数提供根据请求在文档集合中选择一个或多个文档或者文档片段的能力。XML查询代数应支持对查询结果的重构[4]。目前有很多种XML代数，其主要思想来源于关系代数、面向对象代数、半结构化代数和函数式程序设计语言等。文献[4]总结并比较了几种XML查询代数，由于篇幅有限，不再一一介绍。早期的XML查询代数研究，或侧重于规范XML查询语义，不利于查询优化及查询执行，或过多地依赖于系统本身独特的数据存储和索引技术，很难应用于其他系统。本节介绍当前主流的查询代数相关研究，并指出其存在的问题。

Timber数据库[19]中应用的代数，是XML查询代数领域最具有代表性和影响力的查询代数之一。TAX代数[20]是其第一代查询代数，其操作对象是树集合，而不是采用关系代数中的元组集合。它提供了一组基于树模式的代数算子，通过模式树和实例树的概念来表示树查询。但TAX中的模式树存在很大的局限性，难以表示XQuery查询中的复杂语义。文献[21]提出的GTP通用树模式，在TAX代数的模式树基础上进行了扩展。GTP能表示XQuery程序所描述的弱绑定、筛选谓词、结果分组和嵌套层次等相关信息，从而增强了树模式的描述能力。文献[22]在GTP的基础提出了树逻辑类代数TLC，其操作对象APT(Annotated Pattern Tree)在模式树边上关联一个匹配规格mSpec(matching specification)，使得它的匹配是可变的、异构的，从而更大程度上满足了XQuery查询的需要。Timber等数据库中应用的一系列以树模式为数据模型的查询代数，其树模式的表示形式为之后树模式查询算法的发展奠定了基础，但其查询代数存在一定缺陷。一方面，以树集合作为操作对象，难以实现去除相关性等基于元组的逻辑优化；另一方面，其查询执行算法采用结构连接的方式，涉及复杂路径分解及连接等相关问题。

Natix系统[23]使用了针对XPath的完备的查询代数，能提高XPath的查询效率，但不支持XPath以外的XQuery查询语义。XAT[24]基于树集合概念构造了逻辑代数操作集，其操作分为3种：抽取操作从XML文档中获得必要的数据，如选择、投影等；元操作控制表达式求值过程，并非针对XML数据的抽取或者构造操作，而是为其他操作符准备输入或者控制其他操作的操作，如映射、迭代等；构造操作用于构造查询结果。XAL为查询优化提供了一组启发式转换规则。

OrientX数据库系统中应用的OrientXA代数[25]，同样用无序的树集合作为数据模型，采用基于树模式的查询算子，在TAX数据模型上扩展了强弱绑定、节点绑定与序列绑定等功能，扩展了树模式的描述能力，并且引入了用于结果构造操作的算子，但同样难以实现基于元组的逻辑优化。文献[26]提出了一种基于语法的代数XAlgeba，并在此基础上实现了OrientX系统中XQuery的导航式处理。

Michiels[27]等提出的方法能将树模式结合到查询代数中，在基于元组的查询代数中扩充了树模式算子，该方法能够支持较多的逻辑优化方法，从而进行关系代数方面的优化。但该方法对树模式有一定的限制，要求树模式只能有单一的输出节点。

在XQuery查询中，由于表达式可以任意嵌套，XML查询代数需要将嵌套的查询转换为逻辑树形式，因此不可避免地会面临嵌套结构的非嵌套化表示问题。同时，XML数据的有序性要求查询代数支持排序优化，而上述查询代数不能很好地处理这些复杂的XQuery查询语义。

文献[28]中定义的查询代数，解决了之前TAX查询代数不支持有序的问题。文献[29]提出了一种NAL查询代数和基于NAL的针对嵌套查询的优化方法。文献[30]中定义的查询代数，支持具有嵌套结构的XQuery查询优化，生成最小化查询树。Arion[31]等定义的查询代数同样能够表示嵌套结构的XQuery程序，并给出了提取树模式的算法，其提取的

树模式描述能力强于GTP，能够保持更多的XQuery查询的语义信息，同时支持查询最小化优化。该查询代数支持由于IDs引用产生的P-C或者A-D关系的推导，从而扩展基于视图的查询重写优化技术。此外，Wang[32]等人在XAT代数的基础上，提出了包含排序子句的嵌套XQuery查询重写技术。

Re等人[33]从标准XQuery语法角度出发，定义了一套完备的XML查询代数，并在此基础上提出了一系列优化技术，以支撑XQuery编译实现系统的构建。其中包括针对嵌套XQuery的查询重写以及带有复杂谓词XQuery的处理。

综上所述，XML查询代数研究已取得了很大的成果，但也存在一些问题：(1)目前出现的查询代数仅支持XQuery语言的子集，不支持条件语句、自定义函数等函数式语言功能；(2)当前主流的查询代数普遍采用面向集合的算子，对于XQuery查询请求中多级for子句间有依赖关系的情况，元组方式表示的数据会有大量重复项，从而占用过多的内存空间；(3)XML查询代数一个重要的作用就是能为XQuery查询实现提供更多的优化机会，但目前出现的XML查询代数基本不支持跨函数的逻辑优化；(4)XML查询代数既要有足够支持XQuery语义的描述能力，又要利于查询优化和高性能的查询执行。鉴于近来高效的树模式查询算法的飞速发展，在丰富查询代数处理能力的同时，如何实现XML查询代数与树模式查询的有机结合，也面临一些挑战。

### 3.2 树模式查询

XQuery语言的实现必须支持高性能的XML数据处理。其中，Twig查询又称作树模式查询(tree pattern query, TPQ)，其充分反映了这种半结构数据处理的特点，被认为是XML数据查询的核心操作，也是近年来XML查询领域研究的热点之一。图2给出了一个简单的树模式查询在一个XML文档树实例中的匹配情况，图2(a)中双线边表示祖先-后代(A-D)关系，单线边表示父亲-孩子(P-C)关系；图2(b)中两种不同虚线的椭圆表示获得的2个查询结果。

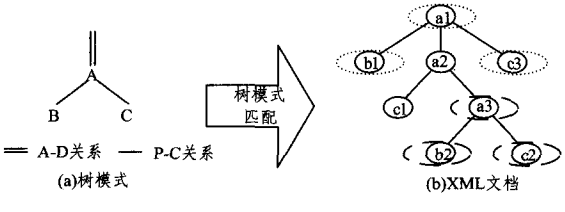


图2 树模式匹配实例

目前，树模式查询算法的研究可以分为两大类，即基本的二元结构连接处理方法和整体Twig查询算法。其中，整体Twig查询算法是当前处理树模式查询的主流方法。

(1)基本的二元结构连接方法。它将Twig模式分解为一系列的二元结构关系，最后将两两结构连接的结果进行合并过滤后得到最终的查询结果。这种方法会产生大量无用的中间结果，时空开销较大。其代表是基于区域编码的多谓词结构连接MPMGJN算法[34]以及基于栈结构的Stack-Tree-Desc/Anc[35]算法。

(2)整体匹配的方法。将整个Twig模式进行整体的处理，从而减少了只满足局部结构的中间结果。根据是否需要对复杂路径进行分解，整体Twig查询又分为二阶段算法和一阶段算法两类。表1对整体Twig算法进行了简单的概括。

表1 Tiwg查询算法比较

| 算法 | 索引 | 优点 | 存在的问题 |
|---|---|---|---|
| TwigStack | 区域编码 | A-D关系最优 | 2阶段算法<br>P-C关系无用中间结果 |
| TSGeneric+ | 区域编码 | 减少无用节点访问 | 2阶段算法 |
| TwigStackList | 区域编码 | 部分P-C最优 | 2阶段算法 |
| TJFast | 扩展Dewey编码 | 部分P-C最优<br>I/O代价低 | 部分P-C无用中间结果 |
| iTwigJoin | 标记+层次前缀路径 | 提高P-C边效率 | 2阶段算法 |
| Twig²Stack | 区域编码 | 1阶段算法<br>支持GTP查询 | 数据结构复杂空间开销大 |
| TwigList | 区域编码 | 1阶段算法<br>数据结构简单 | 不支持GTP查询结果不是文档序 |
| TwigVersion | 版本编码 | I/O代价低 | 版本树构造过程复杂 |
| GTwigMerge | 区域编码 | 支持“OR” | 2阶段算法 |
| TreeMatch | 扩展Dewey编码 | 支持“*、NOT、<” | |

Bruno最早提出了基于区间编码的整体Twig查询算法TwigStack[36]，但该算法对只含A-D关系的查询是最优的。TSGeneric+[37]算法对TwigStack算法进行了改进，有效地利用索引跳过了不参与连接的数据节点，从而降低了I/O开销。TwigStackList[38]算法在TwigStack的基础上采用了一种前瞻的方法，使得一部分包含P-C关系的查询达到最优。文献[39]提出了基于扩展Dewey编码的TJFast算法，它能有效减少扫描数据元素个数，降低I/O代价，在处理部分包含P-C关系的Twig查询时达到最优，但在处理包含分支节点与其子节点呈P-C关系的Twig查询时会产生大量无用中间结果。iTwigJoin算法[40]按照标记+层次(Tag+Level)和前缀路径(prefix path)对XML文档进行细分，从而提高了包含P-C关系的树模式的查询效率。这些Twig查询算法都是分支合并的两阶段算法，在对第一阶段产生的中间结果进行合并连接时会消耗大量的时间和空间，所以研究者们又提出了各种整体处理算法。

文献[41]提出的整体Twig查询算法Twig²Stack，采用了一种层次栈结构组织中间结果，避免了大量不必要的路径归并，但其数据结构复杂，空间开销较大。TwigList[42]算法改进了中间结果数据结构和算法，其计算复杂度与树模式中查询节点个数和匹配的XML数据节点个数呈线性关系，但TwigList不支持GTP查询，且不能保证结果的文档序。文献[43]提出一种版本树，用于区别不同文档树结构，通过版本树来筛选符合Twig模式的数据，其效果优于Twig²Stack。

早期对Twig查询算法的研究主要集中在算法本身和数据结构的优化方面，很少考虑如何将整体Twig查询应用于XQuery。Twig查询可以很好地描述XPath查询请求，但无法表示XQuery程序中由FLWOR语句描述的丰富的查询语义。对于XQuery查询请求，往往将其分解成多个Twig查询，单独执行每个独立的Twig查询，最后进行连接，这造成了大量冗余的中间结果，从而降低了查询效率。因此，当前针对Twig查询的研究倾向于对Twig查询中的树模式表示进行扩展，并发展相对应的树模式匹配算法，以便更好地描述XQuery语言表示的查询需求。

GTP[21]通过扩展边和节点的类型来扩展Twig查询。实线边是表示强绑定，与Twig查询相同；虚线边表示弱绑定，

即并不要求弱绑定边连接的子树一定被匹配。节点类型分为普通节点、返回节点、分组返回节点。由于 GTP 对 Twig 查询进行了扩展，Twig 查询模式不仅可以描述相当多的 XPath 查询语义，也可以在一定程度上描述复杂的 XQuery 查询语义。GTwigMerge[44]算法中树模式扩充了“或”节点的表示，从而能够支持带有“OR”谓词的树模式查询。文献[45]提出一种扩展的树模式，增加了树模式中对 XQuery 语言的否定函数“NOT”、通配符“*”和顺序“<”的表示，并给出了 TreeMatch 处理该扩展树模式的查询算法。

## 4 进一步研究方向

### 4.1 查询代数与树模式结合

鉴于 XQuery 语言的灵活性和复杂性，仅仅依靠对树模式表示进行扩展，很难完全支持 XQuery 的全部查询语义。XQuery 语言的高性能实现需要利用 XML 查询代数提供的查询优化方法，也需要采取高效的树模式查询。将 XML 查询代数的表达式求值与基于 Twig 查询的数据抽取相结合，是 XQuery 实现研究的重要方向。在 XQuery 语言处理系统框架中，实现查询代数和树模式的有机结合将面临以下问题：

(1)在 XML 查询处理中，树模式查询请求采用树形的请求方式，查询结果通过树模式匹配产生。由于 XQuery 是一种函数式语言，查询计划描述难以采用函数调用的方式来描述树查询请求和结果引用。

(2)存在函数调用、谓词和弱绑定等复杂信息的 XQuery 查询请求，可能具有相同或类似的树模式表示。谓词可以出现在任意地方，同一个谓词在不同的地方会产生不同的查询结果，相同的模式树表示难以区分出语义不同的 XQuery 查询请求。如图 3(a)中的 Q1、Q2 和 Q3 都对应图 3(b)所示的查询模式，但却分别返回 A、B、C 节点。Q4 中存在弱绑定问题，其对应的树模式与图 3(b)类似，但需要进行相应的扩展。

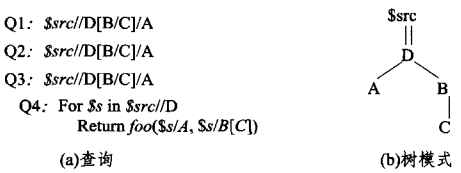


图 3 查询与树模式

(3)鉴于树模式查询的结果允许采用不同的组织方式，并且同时出现在表达式中的不同位置，计算结果组织方式的表示以及不同的求值策略都应该得到查询计划支持。

### 4.2 查询代数

XML 数据模型本身具有的半结构化特点是定义完善的代数运算的最大障碍。而 XML 查询语言中的不确定性和一些算法描述功能的引入是另一个难以克服的困难。研究者根据不同的应用场景定义了多种 XML 查询代数，但基本都是 XQuery 查询的子集，且不能支持跨函数的逻辑优化。本文针对 XML 查询代数存在的问题，提出以下 XML 查询代数发展方向：

(1)基于集合的查询代数和函数式的查询计划描述的统一表示。目前主流的 XML 查询代数都以集合为数据模型，查询语言经翻译转换为由查询代数的算子构成的操作树或者操作序列。由于 XQuery 是一种函数式语言，基于集合的 XML 查询代数难以采用函数调用的方式来描述树查询请求和结果引用，因此有必要发展基于集合的查询代数与函数式查询计划的统一表示。

(2)编译优化技术和基于查询代数的逻辑优化技术的交叉引用。以集合作为数据模型的 XML 查询代数具有很好的优化基础，结合 XQuery 语言自身的函数式程序设计语言特点，借鉴编译优化技术，发展基于查询代数的逻辑优化与编译优化交叉引用的新型优化技术，为 XQuery 实现提供更多的优化方案。

(3)对查询计划描述数据模型进行扩展，使其支持树模式实例树及其代数算子。复杂 XQuery 查询请求需要对其提取的树模式查询结果进行操作，因此可以考虑扩展查询计划的数据模型及代数算子，支持以树模式查询抽取出的 XML 数据实例树为操作对象，在上一次树模式查询结果的基础上进行计算或进一步的树模式查询。

### 4.3 树模式查询

针对 XQuery 语言的树模式表示的扩展，是当前树模式查询算法研究的热点。研究工作主要集中在支持逻辑谓词、顺序、强弱绑定以及通配符等方面，它是 XQuery 语言一个很小的子集。XQuery 是函数式程序设计语言，查询表达式支持自定义函数调用和条件选择结构，可以描述复杂的查询请求，并允许对查询结果进行后期操作，如查询结果的结构化、排序、分组、求和以及算数和逻辑运算等。以下几个方面可能会成为今后 XQuery 实现中树模式查询研究的方向：

(1)面向 XQuery 语言的树模式的表示问题的研究。在当前树模式表示研究成果的基础上，使树模式能够包含更多的 XQuery 查询语义，同时发展相应的高性能树模式查询算法。

(2)支持多数据源的整体树模式表示及其查询算法的研究。XQuery 查询经常需要从不同的数据源中提取数据并组织成最终结果。多数据源 XML 查询一般通过谓词对要抽取的数据进行筛选，最后根据条件进行连接。单独在每个数据源中进行树模式查询后再对其进行连接，必定会产生大量的无用中间结果。用一个树模式表示查询需求，对多数据源进行整体 Twig 查询，将能够有效减少冗余的中间结果，从而降低查询的时空复杂度。

(3)支持多级查询的树模式表示及其查询算法的研究。鉴于 XQuery 程序中包含根据条件判断选择程序片段进行查询等编程风格，实际的查询请求并非整个 XQuery 中提取的树模式查询，而是其子集，因此有必要分级进行数据提取，下一级的树模式查询输入作用在上一级树模式查询的中间结果上，查询执行阶段动态选择需要进行匹配的树模式。

(4)支持递归的树模式表示及其查询算法的研究。现有的树模式表示及其查询算法不能很好地处理 XQuery 语言表示的 XML 查询中包含递归的情况，递归树模式的表示及提取以及相应的匹配算法，也是 XQuery 实现需要解决的问题之一。

这些研究方向并不是相互独立的，一个复杂的 XQuery 程序可能包含其中一部分或者全部，其中还有很多问题有待进一步探讨。

**结束语** XML 语言已经成为事实上的数据交换和数据共享标准。互联网技术的飞速发展，使得 XML 数据与日俱增，作为 W3C 推荐标准的 XML 数据查询语言 XQuery 也得

到 XML 数据库研究领域的普遍重视。XQuery 实现不同于关系数据库的 SQL 语言，其既要适应复杂的数据结构和更灵活的变化，又要适应更丰富的查询语义。原生 XML 数据库系统的研究是目前在 XML 研究领域的一个热点，已经出现了一批相对独立的系统，这些系统采用的查询和处理方法也将对 XQuery 实现技术产生越来越重要的影响。而目前对 XQuery 实现技术的研究工作还远未成熟，在查询代数和树模式查询方面仍有众多尚待解决的问题或需要完善的技术。因此，未来的 XQuery 实现技术研究将以更广泛、更深入的方式展开。

## 参 考 文 献


[1] Gang G, Chirkova R. Efficiently querying large XML data repositories: a survey[J]. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(10): 1381-1403

[2] 孔令波，唐世渭，杨冬青，等. XML 数据的查询技术[J]. 软件学报，2007，18(6)：1400-1418

[3] 孟小峰. XML 数据管理[M]. 北京：清华大学出版社，2009

[4] 孟小峰，王宇，王小锋. XML 查询优化研究[J]. 软件学报，2006(10)：2069-2086

[5] Schwentick T. Automata for XML—A survey [J]. Journal of Computer and System Sciences, 2007, 73(3): 289-315

[6] Maneth S, Nguyen K. XPath whole query optimization [C]// Proc. VLDB Endow, 2010, 3: 882-893

[7] Catania B, Maddalena A, Vakali A. XML document indexes: a classification[J]. Internet Computing, 2005, 9(5): 64-71

[8] 高军，杨冬青，唐世渭，等. 一种基于 DTD 的 XPath 逻辑优化方法[J]. 软件学报，2004，15(12)：1860-1868

[9] Levin M Y, Pierce B C. Type-based optimization for regular patterns[C]// 10th International Symposium Database Programming Languages (DBPL 2005). Berlin, Germany: Springer-Verlag, Aug. 2005

[10] Kader R A, van Keulen M. Overview of query optimization in XML database systems[Z]. Enschede: Centre for Telematics and Information Technology, University of Twente, 2007

[11] Paparizos S, Patel J M, Jagadish H V. SIGOPT: Using schema to optimize XML query processing[C]// 23rd International Conference on Data Engineering (ICDE 2007). Istanbul, Turkey: Computer Society, April 2007

[12] Geng K, Dobbie G, Meng Y. Survey of XML semantic query optimization[C]// 4th International Conference on Internet Computing for Science and Engineering (ICICSE 2009). Harbin, China: IEEE Computer Society, 2010

[13] Murata M, Lee D, Mani M, et al. Taxonomy of XML schema languages using formal language theory[J]. ACM Transactions on Internet Technology, 2005, 5(4): 660-674

[14] Hosoya H, Vouillon J, Pierce B C. Regular expression types for XML[J]. ACM Transactions on Programming Languages and Systems, 2005, 27(1): 46-90

[15] Hosoya H, Pierce B C. XDuce: a typed XML processing language [C]// Third International Workshop on World Wide Web and Databases (WebDB 2000). Berlin, Germany: Springer-Verlag, May 2001

[16] Benzaken V, Castagna G, Frisch A. CDuce: An XML-centric General-purpose Language[C]// Eighth ACM SIGPLAN International Conference on Functional Programming. Uppsala, Sweden: Association for Computing Machinery, 2003

[17] Li X. Efficient and parallel evaluation of XQuery[D]. The Ohio State University, 2006

[18] Bothner P. Compiling XQuery to Java bytecodes[C]// The First International Workshop on XQuery Implementation, Experience and Perspectives. Paris, France, 2004: 31-36

[19] Paparizos S, Jagadish H V, Patel J M, et al. TIMBER: A Native System for Querying XML[C]// 2003 ACM SIGMOD International Conference on Management of Data. San Diego, CA, United states Association for Computing Machinery, 2003: 672

[20] Jagadish H V, Lakshmanan L V S, Srivastava D, et al. TAX: a Tree Algebra for XML[C]// 8th International Workshop on Database Programming Languages (DBPL 2001). Berlin, Germany: Berlin, Germany: Springer-Verlag, 2002

[21] Chen Z, Jagadish H V, Lakshmanan L V S, et al. From tree patterns to generalized tree patterns: on efficient evaluation of XQuery[C]// Proceedings of the 29th International Conference on Very Large Data Bases. Volume 29. VLDB Endowment, 2003

[22] Paparizos S, Lakshmanan L V S, Wu Y, et al. Tree logical classes for efficient evaluation of XQuery[C]// Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2004). Paris, France: Association for Computing Machinery, 2004

[23] Brantner M, Helmer S, Kanne C C, et al. Full-fledged algebraic XPath processing in Natix[C]// Proceedings 21st International Conference on Data Engineering. Los Alamitos, CA, USA: IEEE Comput. Soc, April 2005

[24] Frasincar F, Houben G, Pau C, et al. XAL: an algebra for XML query optimization[C]// (ADC '02) Darlinghurst, Australia, AustraliaLos Alamitos, CA, USA: Australian Computer Society, IEEE Computer Society Press, 2002: 49-56

[25] Meng X, Luo D, Hang Y, et al. OrientXA: an effective XQuery algebra[J]. Journal of Software, 2004, 15(11): 1648-1660

[26] 陆世潮，孟小峰，林灿，等. OrientX 中 XQuery 的导航式实现[J]. 计算机研究与发展，2004，41(10)：1815-1822

[27] Michiels P, Mihaila G A, Simeon J. Put a tree pattern in your Algebra[C]// 23rd International Conference on Data Engineering (ICDE 2007). Istanbul, Turkey: Inst. of Elec. and Elec. Eng. Computer Society, April 2007

[28] Paparizos S, Jagadish H V. Pattern tree algebras: Sets or sequences? [C]// 31st International Conference on Very Large Data Bases (VLDB 2005). Trondheim, Norway: Association for Computing Machinery, August 2005

[29] May N H S, Moerkotte G. Nested queries and quantifiers in an ordered context [C]// Proceedings. 20th International Conference on Data Engineering. Los Alamitos, CA, USA: IEEE Comput. Soc, March 2004

[30] Yannis A D, Papakonstantinou Y, Xu Y. The NEXT Framework for Logical XQuery Optimization[C]// 2004

[31] Arion A, Benzaken V, Manolescu L, et al. Algebra-based identification of tree patterns in XQuery[C]// 7th International Conference on Flexible Query Answering Systems (FQAS 2006). Milan, Italy: Springer Verlag, June 2006


(下转第 18 页)

[34] Kleinberg J, Tardos E. Algorithm Design[M]. Boston: Addison-Wesley, 2005

[35] Bodlaender H L. A linear time algorithm for finding tree-decompositions of small treewidth[J]. SIAM J. Computer, 1996, 25: 1305-1317

[36] Rohrig H. Tree decomposition: A feasibility study [J]. Saarbrucken. Germany: Max-Planck-Institut fur Informatik, 1998

[37] Tarjan R E, Yannakakis M. Simple linear time algorithms to test chordiality of graphs, test acyclicity of graphs, and selectively reduce acyclic hypergraphs[J]. SIAM J. Comput., 1984, 13: 566-579

[38] Berry A, Blair J, Heggernes P, et al. Maximum cardinality search for computing minimal triangulations of graphs[J]. Algorithmica, 2004, 39: 287-298

[39] Rose D J, Tarjan R E, Lueker G S. Algorithmic aspects of vertex elimination on graphs[J]. SIAM J. Comput., 1976, 5: 266-283

[40] Villanger Y. Lex M versus MCS-M[J]. Disc. Math., 2006, 306: 393-400

[41] Bertele U, Brioschi F. Nonserial dynamic programming [M]. New York: Academic Press, 1972

[42] Hell P, Kirkpatrick D G. Algorithms for degree constrained graph factors of minimum deficiency[J]. Journal of Algorithms, 1993, 14(1): 115-138

[43] Becker A, Geiger D. A sufficiently fast algorithm for finding close to optimal clique trees[J]. Artificial Intelligence, 2001, 125 (1): 3-17

[44] Lauritzen S L, Jensen F V. Local computation with valuations from a commutative semi group[J]. Annals of Mathematics and Artificial Intelligence, 1997, 21: 51-69

[45] Meila M, Jordan M I. Triangulation by continuous embedding [M]. Advances in Neural Information Processing System. Cambridge: MIT Press, 1997

[46] Meila M, Jordan M I. An objective function for belief net triangulation[C]// Proceedings of the 1997 Conference on Artificial Intelligence and Statistics. Ft. Lauderdale, FL, 1997

[47] Reed B. *Finding approximate separators and computing tree-width quickly*[C]// Proceedings of the 24th Annual Symposium on Theory of Computing. New York: ACM Press, 1992: 221-228

[48] Koster A. Frequency Assignment-Models and Algorithms[D]. Maastricht, The Netherlands: Univ. Maastricht, 1999

[49] Amir E. Efficient approximation for triangulation of minimum treewidth[C]// Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence. 2001: 7-15

[50] Bodlaender H L, Gilbert J R, Hafsteinsson H, et al. Approximating treewidth, pathwidth, frontsize, and minimum elimination tree height[J]. J. Algorithms, 1995, 18: 238-255

[51] Diestel R, Jensen T R, Gorbunov K Y, et al. Highly connected sets and the excluded grid theorem[J]. J. Comb. Theory, Series B, 1999, 75: 61-73

[52] Kloks T. Treewidth, Computations and Approximations [C]// Lecture Notes in Computer Science, Vol. 842. Berlin: Springer-Verlag, 1994

[53] Bouchitte V, Kratsch D, Muller H, et al. On treewidth approximations[J]. Disc. Appl. Math., 2004, 136: 183-196

[54] Amir E. Approximation algorithms for treewidth[J]. Algorithmica, 2010, 56(4): 448-479

(上接第 13 页)

[32] Wang S, Rundensteiner E A, Mani M. Optimization of nested XQuery expressions with orderby clauses[J]. Data & Knowledge Engineering, 2007, 60(2): 303-325

[33] Re C, Simeon J, Fernandez M. A complete and efficient algebraic compiler for XQuery[C]// 22nd International Conference on Data Engineering(ICDE '06). Atlanta, GA, United states: Institute of Electrical and Electronics Engineers Computer Society, April 2006

[34] Chun Z, Naughton J, Dewitt D, et al. On supporting containment queries in relational database management systems[C]// 2001 ACM SIGMOD International Conference on Management of Data. ACM, May 2001

[35] Al-Khalifa S, Jagadish H V, Koudas N, et al. Structural joins: a primitive for efficient XML query pattern matching[C]// Proceedings 18th International Conference on Data Engineering. Los Alamitos, CA, USA: IEEE Comput. Soc, Feb. 2002

[36] Bruno N, Koudas N, Srivastava D. Holistic twig joins: Optimal XML pattern matching[C]// ACM SIGMOD 2002 Proceedings of the ACM SIGMOD International Conference on Managment of Data, Madison, WI, United States: Association for Computing Machinery, June 2002

[37] Jiang H, Wang W, Lu H, et al. Holistic twig joins on indexed XML documents[C]// VLDB '2003. VLDB Endowment, 2003

[38] Lu J, Chen T, Ling T W. Efficient processing of XML twig patterns with parent child edges: A look-ahead approach[C]// Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management(CIKM 2004). Association for Computing Machinery, November 2004

[39] Lu J, Chen T, Ling T W. TJFast: Effective processing of XML twig pattern matching[C]// 14th International World Wide Web Conference(WWW2005). Chiba, Japan: Association for Computing Machinery, May 2005

[40] Chen T, Lu J, Ling T W. On boosting holism in XML twig pattern matching using structural indexing techniques[C]// SIGMOD '05. New York, NY, USA: ACM, 2005

[41] Chen S, Li H, Tatemura J, et al. Twig2Stack: bottom-up processing of generalized-tree-pattern queries over XML documents [C]// Proceedings of the 32nd International Conference on Very Large Data Bases. Seoul, Korea: VLDB Endowment, 2006

[42] Qin L, Yu J X, Ding B. TwigList: Make twig pattern matching fast[C]// 12th International Conference on Database Systems for Advanced Applications (DASFAA 2007). Bangkok, Thailand: Springer Verlag, April 2007

[43] Wu X, Liu G Q. XML twig pattern matching using version tree [J]. Data & Knowledge Engineering, 2008, 64(3): 580-599

[44] Jiang H, Lu H, Wang W. Efficient processing of XML twig queries with OR-predicates[C]// SIGMOD '04. New York, NY, USA: ACM, 2004

[45] Lu J H, Ling T W, Bao Z F, et al. Extended XML Tree Pattern Matching: Theories and Algorithms[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(3): 402-416