第35卷 第10期 计 算 机 工 程 2009年5月
Vol.35 No.10 Computer Engineering May 2009
·软件技术与数据库· 文章编号：1000—3428(2009)10—0068—05 文献标识码：A 中图分类号：TP311.131

基于模式树的XETL过程研究

郭有限，张东站
(厦门大学计算机科学系，厦门 361005)

摘 要：XML数据与传统的关系型数据存在的差异，使得传统数据仓库的ETL方法已经不适用于XML数据，而目前也没有专门的、有效的适用于XML数据的ETL方法。针对这一问题，提出基于模式树的XML转换处理过程——XETL。从数据模型和谓词模式研究XETL模型，基于XETL模型定义ETL过程中属性选择、空置处理、聚合以及属性重命名4类主要的转换处理操作。

关键词：模式树；XML数据仓库；XETL过程

Research on XETL Process Based on Pattern Tree

GUO You-xian, ZHANG Dong-zhan
(Department of Computer Science, Xiamen University, Xiamen 361005)

【Abstract】Because of the existing differences between XML data and the traditional relational data, the traditional method of data warehouse ETL is no longer suitable for dealing with XML data. This paper proposes the XETL method which is based on pattern tree and can be applied to transfer and deal with XML data. This paper starts with the research on XETL pattern based on data model and predicate model, and defines the four main transference operations in the XETL process based on the XETL model, , which are attribute selection, null attribute operation, aggregation and attribute renamed.

【Key words】pattern tree; XML data warehouse; XETL process

1 背景介绍

1.1 传统数据仓库的局限

数据仓库是在企业管理和决策中面向主题的、集成的、与时间相关的、不可修改的数据集合[1]。数据仓库概念的提出，不但为有效地支持企业经营管理决策提供了一个全局一致的数据环境，也为历史数据、综合数据的处理提出了一种行之有效的解决方法。

随着时间的推移、企业的合并或重组，数据仓库越来越庞大，数据源也日趋多样化，并且随着数据挖掘的深入，需要解决的问题和面临的挑战也就不断地出现：(1)怎样从异构数据源中提取有用的信息；(2)怎样实现数据仓库或数据集市之间的交互或整合；(3)Web 数据挖掘(或网络数据仓库，即Web warehousing)的兴起。越来越多的组织把网络视为其商业和交流的一个重要组成部分，因此网络上的数据对于这些组织的各种决策的影响日趋重要[2]。由于网络上数据无结构化或半结构化的特性(信息以 HTML 格式存储，不包含语义元数据)，以及缺乏对数据源的控制[3]，因此怎样从这些无结构或半结构的数据中获取有用的信息成为Web数据挖掘的首要问题。

1.2 XML数据仓库

XML 是一种旨在描述和交换各种结构数据、简单易用的、独立于用户平台的通用标记语言[4]，XML 由于它的特性成为解决以上问题的最佳选择，XML 数据仓库也在此基础上提出来。将 XML 应用于数据仓库具有如下优势[5]：

(1)容易实现数据在 Web 上发布，XML 数据可以不做任何修改就和 HTML 一样在网络中传输。

(2)有利于数据集成，XML 可以解决异构数据源之间的兼容问题。

(3)支持本体数据处理，客户接收到数据后可以根据自己的需要解析数据，并做进一步编辑处理，减少网络流量，有利于信息共享。

(4)可以实现数据的独立更新，一部分数据变化后，不需要修改全部数据，也不影响数据表现形式。

1.3 ETL

抽取、转换和加载(Extract, Transformation, Loading, ETL)是建立数据仓库的必要步骤，是对原有的、陈旧的数据进行提取、转换、加载，使之成为智能信息系统的有用数据[6]。数据仓库能否对决策分析提供足够的支持，ETL 工具是关键。

传统的数据仓库的数据表示大多是关系元组，使用现有的 ETL 方法进行 XML 数据处理有如下局限：首先，与传统数据仓库数据源不同，XML 数据是结构化的数据；其次，XML 数据仓库的元数据管理与传统的数据仓库有很大的不同，因此，现有的 ETL 方法并不适用于 XML 数据仓库的 ETL 过程。

1.4 XETL

XML 数据仓库的研究刚刚兴起，各个方面都还处于研究阶段，没有形成一个统一的标准，因此出现了各种 XML 数据处理方法：

(1)不进行处理，直接进入 XML 数据仓库。

(2)使用 XLST 等转换工具，对 XML 文档进行转换，在转换过程中完成数据的处理。

(3)把 XML 数据或半结构化的数据先转化为关系元组，

基金项目：国家自然科学基金资助项目(50604012)
作者简介：郭有限(1982－)，男，硕士研究生，主研方向：数据仓库，数据挖掘，分布式数据库；张东站，副教授、博士
收稿日期：2008-12-25 **E-mail：**guoyouxiana@163.com

在关系元组的基础上进行数据处理，数据处理完成后再使用工具转化为 XML 数据。

在(1)中，数据直接进入 XML 数据仓库，那么在 OLAP 中对数据做相应的处理，必然会降低 OLAP 的效率和速度。

在(2)中，尽管 XLST 是一种好的语言，但是用它来完成某些任务是困难的，甚至是不可能的。对于必须对输入的 XML 文档的几个元素做合并计算的这种转换是可行的，但是通常极为难写。

在(3)中，节点元素难与关系元组对应，而且节点之间的一些关系如父子关系、祖孙关系在转化成关系元组的属性列时隐藏了，要得到他们的关系必须经过复杂的连接操作；并且把数据转化为关系元组进行处理，再把处理完的关系元组重新转化为 XML 文档，计算量太大[5]。

现在比较流行的 XML 数据处理方法大多把变量与节点绑定，然后使用无节制的循环的方式操纵这些数据，这必然带来大的计算量。XML 文档的一大特点就是可以视为标记的、有序的有根树，DOM 应用就是以这种方式来处理 XML 文档的。本文根据 XML 文档的这种特性，在模式树的基础上提出了基于模式树的 XETL。

2 XETL 模型

2.1 数据模型

在 XETL 模型中，数据模型是一棵有序标记数据树或称为源数据树。对于 XML 数据树，每个节点对应于一个元素，以及代表元素属性的信息，而其子节点则代表它的子元素。在数据模型中，假设每个节点有一个特殊的单值属性，称为“tag”，“tag”的取值表明了元素的类型；节点还可以有内容属性，代表其原子值，该属性的数据类型与 XML 中的 PCDATA 类型一致。如图 1 所示为一棵数据树。

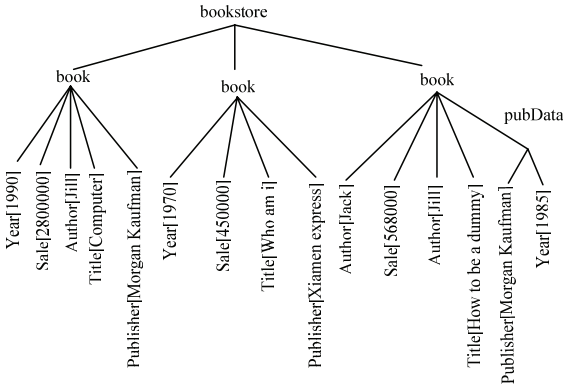


图 1　一棵数据树

2.2 谓词和模式

2.2.1　谓词[7]

对于一个节点$\$i$，任意属性 attr 和值 val，原子$\$i.attr \theta val$是合法的，其中 θ 是=,！=,>等；类似地，对于 2 个节点$\$i$ 和 $\$j$，以及属性 attr 和 attr’，原子$\$i.attr \theta \$j.attr'$是合法的；最后，谓词也可以基于一个节点在树中的位置，例如$\$i.index = first$ 表示$\$i$ 是其父亲节点的第一个儿子节点。

2.2.2　模式树

定义 1(模式树[7]) 形式化的，一棵模式树是有序对 $P = (T,F)$，其中 $T = (V,E)$是一棵节点和边均标记过的树：

(1)V 中的每个节点都有一个唯一的整数作为标号；

(2)每条边标记为 pc(父亲-儿子)或 ad(祖先-后代)表明 2 个节点的关系；

(3)F 是一个公式，即一个谓词的布尔组合。

图 2 为一棵示例模式树。

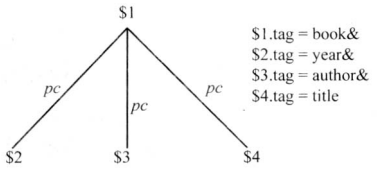


图 2　示例模式树

2.2.3　优胜树

形式化的，令 C 是一个数据树的集合，$P = (T,F)$是一棵模式树，从 P 到集合 C 的内嵌是一个完全映射 h：$P \to C$，从 T 上节点到集合 C 上的节点，有：(1)h 保留了 T 的结构，也就是，当(u, v)是 T 中的 $pc(ad)$边，则 $h(v)$是 $h(u)$的儿子(子孙)节点；(2)映射 h 下的映像满足公式 F。

定义 2(优胜树[7]) 令 C 是一个数据树的集合，$P = (T,F)$是一棵模式树，h：$P \to C$ 是一个内嵌，那么，与数据树相关的优胜树，$hC(P)$，定义如下：

(1)C 中的节点 n 出现在优胜树中，当且仅当对于模式树 P 中的某个节点 v，有 $n = h(u)$，也就是 n 匹配了模式树的某个节点；

(2)对于优胜树中的任何节点对 n, m，当 m 是 n 在 C 中最接近的祖先时，优胜树包含边(m, n)；

(3)优胜树保留了节点在 C 中的相对顺序。

3 XML 数据处理

3.1 属性选择

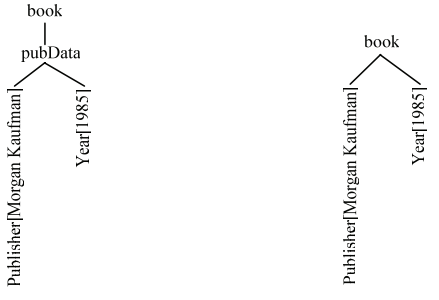
属性选择以一个集合 C 作为输入，一棵模式树作为参数，返回一个输出集合。每棵属于输出的数据树都是由 P 到 C 的内嵌产生的优胜树。在具体实现中，由于原始数据一般都是“脏数据”，有些数据可能在某些节点上缺失，或者可能存在异构，因此单纯地使用简单的模式树往往并不能得到全部所想要的信息，解决这些问题的一个有效的方法就是使用模式树的松弛(pattern tree relaxation)技术[8]。常用的模式树的松弛如下：

(1)父亲-儿子到祖先-子孙的边的概化

这种松弛已经体现在了模式树的定义中，当模式树中的一条边(u, v)标记为 ad，则表示 u, v 的关系从父子关系松弛到祖孙关系。

(2)子树提升

子树提升把一个以 n 为根的子树变为 n 的祖父节点的子树。例如在图 3 中，模式树如图 3(a)所示，通过子树提升，把子树 publisher, year 变为其祖父节点 book 的儿子节点。结果如图 3(b)所示。



(a)源数据树的一棵子树　　(b)子树提升松弛后的子树

图 3　子树提升示例

(3)叶节点删除

叶节点删除允许模式树即使在一个叶节点不存在的情况下也能被匹配。

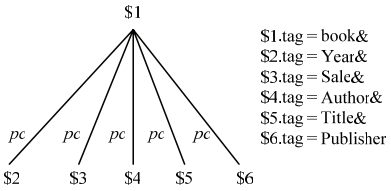
形式化地，属性选择操作 σ 以集合 C 作为输入，叶节点删除松弛列表 NDL 作为参数，输出是一个树的集合，与内嵌 $h: P \rightarrow C$ 相关的输出 σNDL，$P(C)$描述如下：

(1)输入集合 C 中的一个节点 u 属于输出，当且仅当 u 匹配了内嵌 h 下 P 中的某个节点，或者以 u 为根的子树通过松弛后(子树提升或叶节点删除)能够匹配 P 中的某个节点；

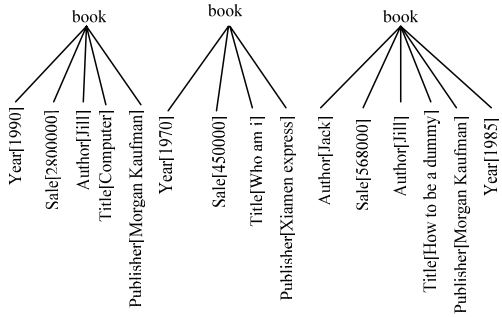
(2)当 u, v 同时属于输出，而 u 是 v 在输入中的最接近的祖先，则输出包含边(u, v)；

(3)输出集合中的节点保持其在输入集合的相对位置不变。

例 1 图 4(a)是一模式树，叶节点删除松弛列表 NDL 为{$3}，即 book 的作者可以为空，图 4(b)是把该模式树应用到图 1 的数据树的结果。



(a)一棵模式树



(b)模式树应用到图 1 数据树的结果

图 4　叶节点删除示例

图 1 中的第 2 棵 book 子树缺少 author 节点，第 3 棵 book 子树没有 Publisher 和 Year 2 个子节点，并不匹配模式树 P。对第 2 棵子树使用叶节点删除的松弛技术，对第 3 棵子树使用子树提升的松弛技术，则 2 棵子树都能匹配模式树 P，其信息都能进入到结果集中。

3.2　空值处理

定义 3(空缺值节点) 令 P 是一模式树，则 P 中可能为空值的节点称为空缺值节点。

定义 4(载体节点) 令 P 是一模式树，则 P 中的空缺值节点的父节点称为载体节点。

顾名思义，空缺值节点就是所要处理的可能缺失、或者值可能为空的节点，而载体节点就是空缺值节点的父节点。

空缺值处理操作 n 以集合 C 作为输入，模式树 P、空缺值节点列表 NL 以及空缺值处理函数 f 作为参数，返回一个输出集合。形式化地，空值处理操作 n 的输出 $nP, NL, f(C)$是一个树的集合，与内嵌 h：$P \rightarrow C$ 相关的输出描述如下：

(1)对于集合 C 中的每棵树，$nP, NL, f(C)$包含一棵与其同构的树，该树中的每个节点的每个属性与 C 中相应节点一致。

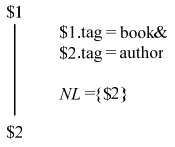
(2)当输出集合 $nP, NL, f(C)$的节点匹配了模式树中的载体节点，则对该节点执行如下操作：

1)若该节点已经包含与之对应的空缺值节点，且取值不为空，则输出该节点；

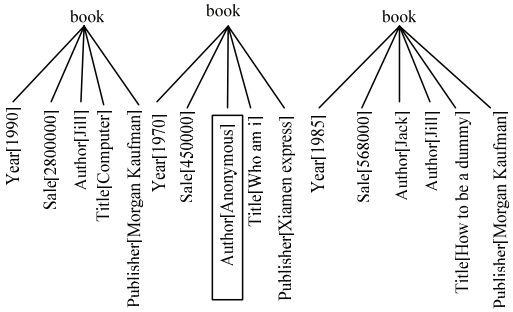
2)若该节点已经包含与之对应的空缺值节点，但取值为空，则为该空缺值节点赋一新值，该值由空值处理函数 f 决定；

3)若该节点不包含与之对应的空缺值节点，则为该节点创建一个新的儿子节点，新创建的节点的名称即为该空缺值节点名，其值由空缺值处理函数 f 决定。

例 2 以例 1 的属性选择结果(图 4(b))作为输入，空值列表为 NL={$2}，则载体节点为{$1}，如图 5(a)所示。空值处理函数为 f: $2.value = 'Anonymous'，则空值处理操作 $nP, NL, f(C)$的结果如图 5(b)所示，方框内即为经过空值处理操作的节点。



(a)模式树和空值列表



(b)空值处理结果

图 5　空值处理示例

3.3　聚合

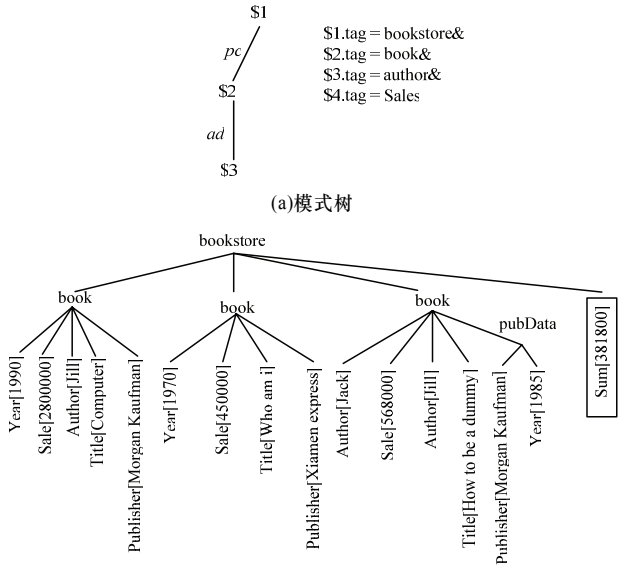
聚合的目的就是把集合的值映射到聚合或总计(summary)值。通常的聚合函数有 MIN, MAX, COUNT, SUM 等。当聚合值生成时，必须指定计算出的值应该插入到哪里，在哪个 tag(标签)或作为哪个属性的值。

形式化地，聚合操作 A 以集合 C 作为输入，以模式树 P、聚合函数 f 和更新实现 US 作为参数。其中，更新实现指出新计算的值应该插入到输出树的哪个地方，具体插入到哪个位置由更新实现指定，比如说，插入到特定节点的最后一个儿子节点后面、或者插入到某个特定节点的前面(或后面)。

聚合操作 A 的语义描述如下：

对于每棵输入树，有一棵对应的输出树，输出树与输入树多了一个节点，该节点的值由聚合函数 f 决定，位置由更新实现 US 指定。除了这点不同外，输出树与输入树保持一致。

例 3 计算书店中所有的书的销售量。销售量保存在节点 sum 中，并把该节点作为 bookstore 节点的最右儿子节点插入到数据中，所以更新实现 US 为{after lastChild($1)}，聚合函数为 sum.value = SUM($3.value)，模式树如图 6(a)所示，聚合结果如图 6(b)所示。



(a)模式树

(b)聚合操作结果

图 6　聚合操作实例

3.4　属性重命名

属性重命名操作改变输入树的特定节点的特定属性的名称。以如下概念来定义重命名操作:

令 P 是一模式树，则一个重命名实现(RS)是一个$\$i$: $oldname$←$newname$ 形式的表达式，其中，$\$i$ 是 P 中出现的节点；$newname$ 是任意标志符。

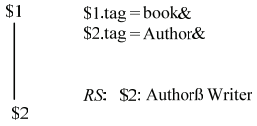
重命名以一个集合 C 作为输入，以模式树 P 和一个重命名实现作为参数，并以如下方式生成输出集合 σP，$RS(C)$作为输出:

(1)C 中每个匹配了 P 中的某个节点$\$i$ 的节点都标记为 i。

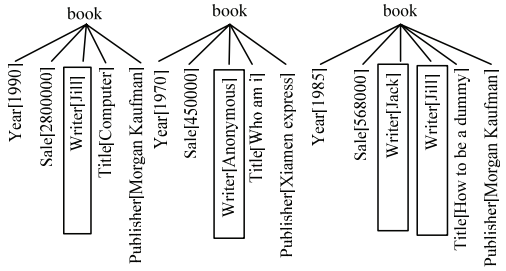
(2)对于集合 C 中的每棵树，σP, $RS(C)$包含一棵与其同构的树，该树中的每个节点的每个属性与 C 中相应节点一致。

(3)当输出中的一个节点 u 与输入树中标记为 i 的节点以及模式树中标记为$\$i$ 的节点相对应，且 RS 包含: $\$i$: $oldname$←$newname$，则: 1)如果 u 的“tag”为 $oldname$，则改为 $newname$；2)如果 u 包含一个 $oldname$ 的属性，则该属性更名为 $newname$。

例 4 图 7(a)是一棵模式树和重命名实现 RS，图 7(b)是把图 7(a)应用到图 5(b)的结果，处理过的节点都用方框标出。



(a)模式树和重命名实现RS



(b)属性重命名结果

图 7　属性重命名实例

3.5　其他操作

其他数据处理过程如噪声数据处理、数据规约、离散化等，都可以根据模型推导出来，篇幅所限，本文只讨论以上 4 种数据处理过程。

4　应用

本文提出的 XML 数据处理模型已经应用于一个基于 XML 的信息集成系统。该系统从 Oracle XML DB 抽取数据，经过 XETL 过程处理，最后把数据集成到信息系统中。下面是该系统使用的一个实例。

BookSale.xml 是 Oracle XML DB 中的一个文件:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Bookstore>
 <Book>
      <Title>Contemporary American Culture and Society
</Title>
      <Author>JingQiong Zhou</Author>
      <Publisher>Shanghai Foreign Language Press</Publisher>
      <Year>2003</Year>
      <Price>28.00￥</Price>
      <Sale>2,540,433</Sale>
 </Book>
 <Book>
      <Title>Reading Plus</Title>
      <Author>Anonymous</Author>
      <Publisher>McGraw-Hill</Publisher>
      <Year>1998</Year>
      <Price>10$</Price>
      <Sale>50,749</Sale>
 </Book>
 <Book>
      <Title>The Bedford Guide for College Writers</Title>
      <Author>X.J.Kennedy</Author>
      <Athor>Dorothy M.Kennedy</Author>
      <PubData>
          <Publisher>Bedford Books of St.Martin's Press
</Publisher>
          <Year>1996</Year>
      </PubData>
      <Price>20$</Price>
      <Sale>30,500</Sale>
 </Book>
 <Book>
      <Title>希腊的神话和传说</Title>
      <Author>斯威石</Author>
      <Publisher>人民文学出版社</Publisher>
      <Year>2002</Year>
      <Price>38.00￥</Price>
      <Sale>104,500</Sale>
 </Book>
 <Book>
      <Title>Social Problems</Title>
      <Author>D.Stanley Eitzen</Author>
      <Publisher>Allyn&Bacon</Publisher>
      <Year>1997</Year>
      <Price>25$<Price>
      <Sale>380,400</Sale>
 </Book>
</Bookstore>
```

在示例中，对该文件作如下处理：抽取每本书籍的书名(Title)、作者(Author)、出版社(Publisher)、售价(Price)、销量(Sale)等信息；在所抽取的信息中，书的价格(Price)有人民币和美元 2 种货币单位，在进一步的处理中将它们统一为以人民币为货币单位；对丢失的信息作进一步的处理；统计所有书的销量。

经过处理后的文件保存为 BookSaleForDW.xml，并加载进数据仓库中。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Bookstore>
  <Book>
        <Title>Contemporary American Culture and Society </Title>
        <Author>JingQiong Zhou</Author>
        <Publisher>Shanghai Foreign Language Press</Publisher>
        <Price>28.00￥</Price>
        <Sale>2,540,433</Sale>
  </Book>
  <Book>
        <Title>Reading Plus</Title>
        <Publisher>McGraw-Hill</Publisher>
        <Price>80￥</Price>
        <Sale>50,749</Sale>
  </Book>
  <Book>
        <Title>The Bedford Guide for College Writers</Title>
        <Author>X.J.Kennedy</Author>
        <AUthor>Dorothy M.Kennedy</Author>
        <Publisher>Bedford Books of St.Martin's Press </Publisher>
        <Price>160￥</Price>
        <Sale>30,500</Sale>
  </Book>
  <Book>
        <Title>希腊的神话和传说</Title>
        <Author>斯威石</Author>
        <Publisher>人民文学出版社</Publisher>
        <Price>38.00￥</Price>
        <Sale>104,500</Sale>
  </Book>
  <Book>
        <Title>Social Problems</Title>
        <Author>D.Stanley Eitzen</Author>
        <Publisher>Allyn&Bacon</Publisher>
        <Price>200￥<Price>
        <Sale>380,400</Sale>
  </Book>
  <Total Sale>3,106,582</Total Sale>
</Bookstore>
```

实验证明，基于模式树的 XETL 处理模型可以很好地支持各种 XML 的数据预处理操作。与现有的处理方法相比，该模型能完整地保留 XML 文件的树结构，处理过程中的计算量较小。此外该模型还具有平台无关性、可扩展性等优点。

5 结束语

本文以模式树为基础，提出一种 XML 数据处理模型，并定义相关的数据处理操作。该数据处理模型已应用于一个基于 XML 的信息集成系统中。事实表明，它能够有效地支持 XML 数据处理的各种操作。本文是从逻辑层面对 XML 数据进行处理，进一步的工作可以根据 XML 的物理存储结构，例如索引，来改善模式树的匹配过程，从而达到更高效的处理效果。

参考文献

[1] Inmon W H. Building the Data Warehouse[M]. Beijing, China: China Machine Press, 2000.

[2] Golfarelli M, Rizzi S, Vrdoljak B. Data Warehouse Design from XML Sources[C]//Proc. of the 3rd ACM International Workshop on Data Warehousing and OLAP. Atlanta, USA: [s. n.], 2001: 40-47.

[3] Rizzi S, Abell A, Lechtenborger J, et al. Research in Data Warehouse Modeling and Design: Dead or Alive?[C]//Proc. of the 9th ACM International Workshop on Data Warehousing and OLAP. Arlington, Virginia, USA: [s. n.], 2006: 3-10.

[4] Roy J, Ramanujan A. XML: Data's Universal Language[J]. IEEE IT Professional, 2000, 2(3): 32-36.

[5] 仇丽青, 赵庆祯. 基于 XML 的数据仓库系统[J]. 计算机系统应用, 2004, 13(2): 12-14.

[6] 缪嘉嘉, 邓　苏, 刘青宝. ETL 综述[J]. 计算机工程, 2004, 30(3): 4-5, 21.

[7] Jagadish H, Lakshmanan L, Srivastava D, et al. TAX: A Tree Algebra for XML[C]//Proc. of the 8th International Workshop on Databases and Programming Languages. Rome, Italy: [s. n.], 2001.

[8] Hümmer W, Bauer A, Harde G. XCube: XML for Data Warehouses[C]//Proc. of the 6th ACM International Workshop on Data Warehousing and OLAP. New Orleans, Louisiana, USA: [s. n.], 2003: 33-40.

编辑　任吉慧

(上接第 67 页)

参考文献

[1] Benatallah E, Sheng Quanzheng, Dumas M. The Self-serv Environment for Web Services Composition[J]. IEEE Internet Computing, 2003, 7(1): 40-48.

[2] 王创伟, 钱雪忠. 蚁群算法在 Web 服务组合问题中的应用研究[J]. 计算机工程与设计, 2007, 28(24): 5912-5914.

[3] Menasce D A. QoS Issues in Web Services[J]. IEEE Internet Computing, 2002, 6(6): 72-75.

[4] 陈彦萍, 李增智. 一种满足马尔可夫性质的不完全信息下的 Web 服务组合方法[J]. 计算机学报, 2006, 29(7): 1076-1082.

[5] Zeng Liangzhao, Benatallah B, Dumas M, et al. Quality Driven Web Services Composition[C]//Proceedings of the 12th International Conference on World Wide Web. Budapest, Hungary: [s. n.], 2003.

[6] 廖　渊, 唐　磊, 李明书. 一种基于 QoS 的服务构件组合方法[J]. 计算机学报, 2005, 28(4): 627-634.

[7] 倪晚成, 刘连臣, 吴　澄. Web 服务组合方法综述[J]. 计算机工程, 2008, 34(4): 79-81.

编辑　顾姣健