第 33 卷 第 7 期 计 算 机 工 程 2007 年 4 月
Vol.33 No.7 Computer Engineering April 2007
·软件技术与数据库· 文章编号：1000—3428(2007)07—0056—03 文献标识码：A 中图分类号：TP311

一种基于映射规则的冲突消解方法

傅宜生，岳丽华，蔡荣峰，金培权
(中国科学技术大学计算机科学与技术系，合肥 230027)

摘 要：针对目前基于本体的 XML 数据集成系统中，仅仅通过映射到全局模式来进行冲突消解的不足，该文提出了一种可扩展的映射规则模型。基于该模型，给出了一个冲突消解算法，可以较好地解决由于局部数据源间的冲突引起的局部数据源查询结果整合不正确和数据源间的连接操作失败等局部数据源互操作中出现的问题。

关键词：集成；映射规则；冲突消解

Mapping Rule Based Conflict Resolution Method

FU Yisheng, YUE Lihua, CAI Rongfeng, JIN Peiquan
(Department of Computer Science & Technology, University of Science & Technology of China, Hefei 230027)

【Abstract】 This paper proposes an extensible mapping rule model to make up for the deficiency of conflict resolution only by mapping to global ontology in ontology-based XML data integration systems. Based on this model, a conflict resolution algorithm is given to solve the problem of incorrect result from interoperations among local data sources, such as joint operation and unification of query results from heterogeneous data sources.

【Key words】 Integration; Mapping rule; Conflict resolution

1 概述

目前，基于本体的XML数据集成的通常解决办法是：通过集成的各个数据源，生成一个全局本体(Global Ontology)，局部数据源中的每一个元素(或属性)都对应着全局模式的一个概念，这种对应关系称之为映射规则[1~3]。应用时，系统首先分析与用户查询中的全局模式概念有关的映射规则，通过查询分解算法[1,3]将其分解成在局部数据源的子查询，最后把各个子查询的结果返回给用户。这里，系统仅仅通过到全局本体的映射进行冲突消解，而集成系统并不知道局部数据源间是否存在冲突，因此在处理查询过程中，如果需要在数据源间进行互操作，比如进行连接操作，而两个数据源中做连接的那个元素存在冲突时，连接操作就会失败。又如在整合各个数据源返回的查询结果时，由于各个数据源的异构，会返回异构的查询结果，导致数据整合结果出错[2]。下面给出一个简单的例子：有两个记录工程项目和资金的XML数据源S1和S2，利用一个本体来做集成，如图 1。

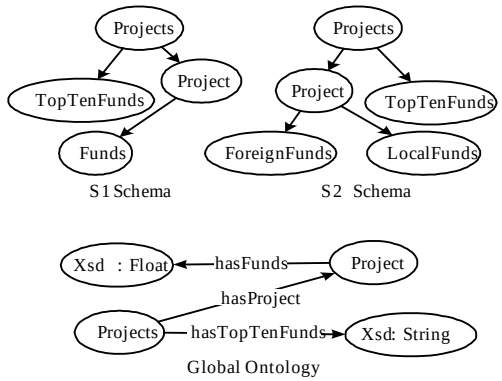

图 1 一个全局本体集成两个 XML 数据源的例子

图 1 中，S1 中Funds元素的单位是人民币，而S2 中Funds以美元为单位，这种冲突称为比例冲突[4]。另外，TopTenFunds元素都是指在这些项目中资金最多的十个项目，但S1 用的是升序，S2 用的是降序，这种冲突称之为顺序冲突[4]。由于仅仅有路径映射，现有的集成系统都无法发现并消解这类冲突，因此倘若在整合查询结果的时候没有消解冲突，整合的结果就不正确，在funds上做连接的时候单位不统一，连接操作也会失败。

针对上述问题，本文提出一种新的模型定义映射规则，使之不仅包含基本的路径映射信息，而且包含各个局部数据源间冲突信息。基于这个映射规则模型，给定一个数据源，一个全局概念，就可以方便地找出其他数据源与这个数据源在这个概念上冲突信息并进行消解。

2 板映射规则

2.1 映射规则模型

在基于本体的XML集成系统中，每个局部数据源都有一组映射规则，每一条映射规则描述了本体中的概念到该数据源每个元素的映射信息，为全局模式到局部数据源的查询分解算法提供了必要的查询转换信息[1,3]。但是，由于此类映射规则并不记录各局部数据源间的冲突，因此当局部数据源间存在冲突时，系统在处理局部数据源数据的连接操作或者返回结果数据整合时，将导致连接失败或者整合结果错误。为了解决这个问题，在传统的映射规则基础上进行扩展，使得每一条映射规则不仅描述本体中的概念到该数据源每个元素

作者简介：傅宜生(1981－)，男，硕士生，主研方向：数据库应用，信息集成，Semantic Web；岳丽华，教授、博导；蔡荣峰，博士生；金培权，副教授

收稿日期：2006-05-16 **E-mail：**boysen@ustc.edu

的映射信息，还描述在这个概念上该数据源与其他数据源的冲突信息。

RDF(资源描述框架)[5]是W3C的推荐标准，它是一种可以无限扩展的框架。RDF允许用户定义元数据集来描述特定的资源。元数据集被称作词汇集(Vocabulary)，也是一种资源，可以用URI来唯一标识。因此用RDF来描述映射规则，并定义了一个基本的词汇集(表 1)。在以下的映射规则举例中，表 1 的词汇集的URI 是http://www.mycompany.com/MappingRule，名称空间是MR。随着集成的发展和应用，可以扩展该词汇集，增加更多词汇以描述新的冲突。

表 1 基本词汇集

词汇	语义
Rule	映射规则
GlobalConcept	全局概念
LocalPath	局部数据源路径
Source	局部数据源
OrderConflict	顺序冲突
ScaleConflict	比例冲突
TransformFunction	转换函数

本文的映射规则模型是基于Path-to-Path的[3]，一条规则由Rule定义，其他词汇用于定义规则的属性。GlobalConcept和LocalPath是Rule必需的属性，它们分别定义该规则描述的全局概念和局部数据源的路径。Rule还可以有描述冲突的属性，如：OrderConflict，ScaleConflict。它们分别用于描述顺序冲突，比例冲突。每种冲突至少要有两个属性：Source和TransformationFunction，分别用来描述局部数据源的名称以及把冲突数据源的对应元素转换到所在数据源模式的函数名称。

使用该模型，S1 的/Projects 元素到全局本体 Projects 概念的映射规则描述如下：

```
<rdf :RDF
xmlns :rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#
xmlns:MR=”http://www.mycompany.com/MappingRule#>
        …
  <MR:rule>
     <MR:GlobalConcept> Projects</ MR:SourceConcept >
     <MR:LocalPath>/Projects</ MR:LocalPath >
   </MR:rule>
        …
</rdf:RDF>
```

如果 S1 在这个概念上与其他数据源有冲突，便可以对应的描述冲突的词汇来描述。下面说明如何用该映射规则模型描述的上述例子中的顺序冲突和比例冲突。这个规则如图 2。

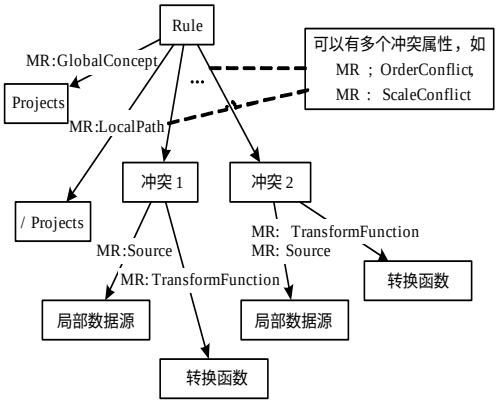

图 2 S1 中/Projects 元素映射到 Projects 概念

2.2 语义冲突描述举例

(1)顺序冲突

顺序冲突是指，两个元素的排列方式不同，当对这两个元素进行操作时(如连接，合并等)，就会产生冲突。上述例子中的顺序冲突可以描述如下：

数据源 S2 的映射规则文档：

```
<MR:Rule>
  <MR:GlobalConcept>hasTopTenFunds<MR:GlobalConcept>
  <MR:LocalPath>/Projects/TopTenFunds<MR:LocalPath>
  <MR:OrderConflict>
    <MR:Source>S1</MR:Source>
    <MR:TransformFunction>hasTopTenFunds_S1_To_S2
    </MR:TransformFunction>
  </MR:OrderConflict>
</MR:Rule>
```

数据源 S1 的映射规则文档：

```
<MR:Rule>
  <MR:GlobalConcept>hasTopTenFunds<MR:GlobalConcept>
  <MR:LocalPath>/Projects/TopTenFunds<MR:LocalPath>
  <MR:OrderConflict>
    <MR:Source>S2</MR:Source>
    <MR:TransformFunction>hasTopTenFunds_S2_To_S1
    </MR:TransformFunction>
  </MR:OrderConflict>
</MR:Rule>
```

这里用<MR:OrderConflict>定义了该顺序冲突，以及相应的转换函数名称 TopTenFunds_S2_To_S1 和 TopTenFunds_S1_To_S2，其中，TopTenFunds_S2_To_S1 的功能是把 S2 数据源中对应 hasTopTenFunds 这个概念的元素转换成与 S1 没有顺序冲突的元素，而 TopTenFunds_S1_To_S2 则是从 S1 到 S2 的转换函数。

为节省篇幅，以下的映射规则都省略了描述路径映射的语句，只给出描述冲突的语句。

(2)比例冲突

比例冲突是指，两个元素的单位不一样，在对这两个元素进行操作时，也会产生冲突。

数据源 S1 和 S2 中的比例冲突可以描述如下：

数据源 S2 的映射规则文档：

```
  …
<MR:ScaleConflict>
   <MR:Source>S1</MR:Source>
   <MR:TransformFunction>hasFunds_S1_To_S2
   </MR:transformFunction>
</MR:ScaleConflict>
  …
```

数据源 S1 的映射规则文档：

```
  …
<MR:ScaleConflict>
  <MR:Source>S2<MR:Source>
  <MR:TransformFunction>hasFunds_S2_To_S1
  </MR:transformFunction>
</MR:ScaleConflict>
  …
```

3 基于映射规则的冲突消解算法

映射规则给出了异构数据源间的冲突描述，将该映射规则用于数据集成还需要有能理解该冲突描述并进行冲突消解的算法。在系统进行互操作前，调用算法进行冲突消解，在进行互操作时，结果正确。

首先，定义查询 Q 的互操作概念集￥(Q)，它是查询 Q 可能与其他数据源进行互操作的概念的集合。下面给出一个例子：

例 基于 OQL[6]的查询 Query1

Select x1,x2 from Projects ps, p.hasTopTenFunds x1, ps.hasProject p, p.hasFunds x2

由于 TopTenFunds, Funds 都是是返回结果对应全局本体中的概念，返回结果整合需要和其他数据源进行互操作，除此之外，Query1 没有其他概念需要与其他数据源进行互操作，因此￥(Query1)={TopTenFunds, Funds}。

另外，用.操作符来表示规则定义中的属性操作，比如，假设 rule 是<Rule>元素，则 rule.GlobalConcept 表示该 rule 的 GlobalConcept 属性，而 rule.OrderConflict.Source 表示 rule 的 OrderConflict 属性的 Source 属性。

本文给出的冲突消解算法的描述算法如下。具体流程是：

(1)任意选择一个子查询作为标准查询；

(2)得到它的互操作概念集；

(3)对每一个概念，查找该子查询所在数据源的映射规则文档，如果这个概念对应的规则包含冲突属性，则遍历每个冲突属性，找出冲突的数据源和转换函数，修改子查询。知道概念集中的所有概念都处理完算法结束。

```
算法 DoConflictResolution(Q)
输入 子查询集合 Q
输出 冲突消解后的子查询集合 Q
Begin
  q1:= Q 中的任意一个子查询
  C=￥(q1);
  For each c in C
     For each rule in q1 所在数据源的映射规则文档
       If rule.GlobalConcept = c
        Begin
          Conflicts = rule 所有的 Conflict 属性
          For each conflict in Conflicts
            If rule.conflict.Source ∈ Q
              Begin
                function = rule.conflict.TransformFunction;
                Replace (rule.conflict.Source, c, function(c));
              End
            Break;
        End
  Return Q;
End
```

其中，函数 Replace (x,y,z)的功能是，在 x 数据源上的子查询 q 中，用 z 替换 q 中出现的所有概念 y (其中，y ∈￥(q))。

例子中的全局查询Query1 经过查询分解，得到在S1 和S2 上的两个子查询QueryS1 和QueryS2， 由于S1 和S2 都能完全满足全局查询(全绑定)[1,3]，因此QueryS1 和QueryS2 与Query1 完全一样。然后调用DoConflictResolution({QueryS1,QueryS2})可以得到冲突消解后的子查询QueryS1’和QueryS2’(其中，QueryS1 被选为标准模式，保持不变，修改QueryS2 以消解与 QueryS1 的顺序，比例冲突)，最后，进入查询重写算法，将其重写成 XML 的查询语言然后执行。这样，在 QueryS2’的返回结果中，S2 中 Funds 由 Funds_S2_To_S1 函数转换成以人民币为单位，而 TopTenFunds 元素则转换成升序排列。该过程如图 3。

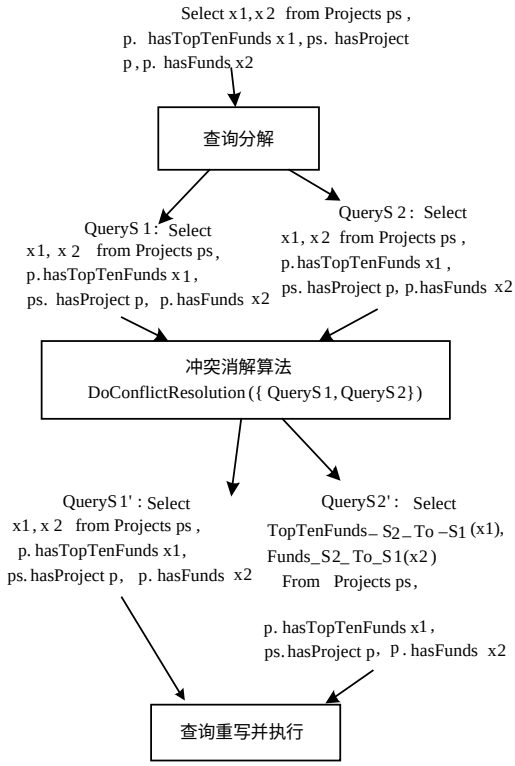

图 3 Query1 的处理流程

4 总结

在数据集成系统中，消解各个数据源间的异构而引起的各种冲突是一个关键问题，而目前基于本体的 XML 数据集成系统中，仅仅通过路径映射到全局模式来进行冲突消解，而如果局部数据源间存在冲突，则集成系统在处理局部数据源间的互操作就会出错，如数据源之间的连接操作以及各个数据源返回结果的整合。本文提出了一种基于扩展映射规则的冲突消解方法，定义了描述这种映射规则的基本词汇集，举例用这种映射规则模型描述了顺序冲突和比例冲突，基于上述工作，给出了消解冲突的算法。

参考文献

1 Amann B, Beeri C, Fundulaki I, et al. Ontology-based Integration of XML Web Resources[C]//Proceedings of the 1st International Semantic Web Conference. 2002: 117–131.

2 Cruz I F, Xiao Huiyong, Hsu Feihong. An Ontology-based Framework for XML Semantic Integration[C]//Proc. of International Database Engineering & Application Symposium. 2004: 217-226.

3 杨 洋, 岳丽华, 韩 凯, 等. 一种语义集成框架的研究和实现[J]. 中国科学技术大学学报, 2006, 36(2).

4 Li Changqing, Tok Wang Ling. OWL-based Semantic Conflicts Detection and Resolution for Data Interoperability[C]//Proc. of International Conference on Entity Relational Approach Workshops. 2004: 266–277.

5 XML. RDF[Z]. 2006-02. http://www.w3c.org.

6 OQL[Z]. 2006-02. http://www.odmg.org.