第34卷 第20期 计 算 机 工 程 2008年10月
Vol.34 No.20 Computer Engineering October 2008
·博士论文· 文章编号：1000—3428(2008)20—0010—03 文献标识码：A 中图分类号：TP311

基于上下文仲裁的语义异构解决方案

周建芳，徐海银，卢正鼎
(华中科技大学计算机学院，武汉 430074)

摘 要：基于本体的语义信息集成主要解决分布异构的数据源之间的模式级异构和部分数据异构(包括同义字和同音异义字)。在基于本体的语义信息集成的基础上引入上下文机制来全面解决异构数据源之间的语义异构，弥补了基于模式映射的语义信息集成的不足，具有较好的适应性和扩展性。

关键词：语义信息集成；模式语义；属性级上下文异构；上下文仲裁

Semantic Heterogeneity Solution Based on Context Mediation

ZHOU Jian-fang, XU Hai-yin, LU Zheng-ding
(School of Computer, Huazhong University of Science and Technology, Wuhan 430074)

【Abstract】 Ontology-based semantic information integration resolves the schema level heterogeneity and part of data level heterogeneity between distributed data sources. Ontology-based semantic information integration can't resolve this context heterogeneity. In this paper, a context mediation based information integration frame is put forward to resolve the attribe-level context heterogeneity, hence the complete resolution of semantic heterogeneity is formed and the resolution is adaptive and extensive.

【Key words】 semantic information integration; schema semantics; attribe-level context heterogeneity; context mediation

1 概述

随着网络通信技术的发展，分布异构的信息系统之间的物理连接(交换比特和字节的能力)已经广泛地建立起来，但它们之间的逻辑连接(交换数据语义的能力，也称为互操作)还远没有建立[1]。究其原因，主要是传统的信息语义往往通过编码写入应用程序中，只能被应用程序和应用程序设计者、维护者理解，其他应用程序无法理解和处理，从而造成了互相隔离的信息孤岛彼此之间无法通信，也无法进行知识的共享、重用和信息系统之间的互操作，导致信息无法得到高效的利用。解决的办法是对异构分布的自治数据源进行信息集成，其核心问题是解决数据源之间的异构情况。现在，信息集成已经成为企业关注最多、投资持续增长最热门的信息技术之一。

2 基于本体的信息集成的原理和存在的问题

基于本体的信息集成通过在需要集成的数据源和用户之间构建一个逻辑层用于进行语义异构的检测和消解，从而解决模式级语义异构的问题。该逻辑层需要包括以下组件[2]：本体管理组件，元数据管理组件，查询引擎。

由于本体仅仅定义了领域知识(类、属性、关系以及公理)，真正的数据存储在各个局部数据源中，通过在各个分布异构的数据源与全局本体之间建立语义映射，由查询引擎将综合查询分解为与数据源相关内容和格式的子查询，这些子查询在数据源本地执行后将子结构由查询引擎综合后返回使用。可见查询引擎是信息集成的核心组件，在查询重写的过程中解决了数据源之间的模式级语义冲突。但是，对于不同数据源中具有相同模式语义的信息，仍然无法解决由于不同数据源中采用不同的缺省假设导致其具有不同解释的数据级异构问题(在本文中称为上下文异构)。

3 例子

为了论述清楚，现举例如下：

有 2 个不同的数据源 DS1 和 DS2(为简单起见假定都是关系数据库，实际上也可以是半结构化数据)，其模式与数据如表 1、表 2 所示。

表 1 数据源 DS1 中 company 模式定义与数据

名称(Name)	地址(Address)	注册日期(RegDate)	注册资金/万元(RegMoney)	负责人(Owner)
三元贸易有限公司	武汉市胜利街 564 号	05/03/09	100	李三元
好宝宝玩具用品公司	武汉市黄陂路 334 号	01/04/06	50	黄先河
小四川餐饮服务公司	武汉市中华路 92 号	01/05/08	80	刘义

表 2 数据源 DS2 中 enterprise 模式定义与数据

名称(Name)	地址(Area)	注册日期(RegisterDate)	注册资金/元(Acount)	负责人(Leader)
友好商贸	黄石市民族大道 222#	05/06/01	650 000	朱友才
凯琳化妆品有限公司	鄂州市中山路 546#	02/15/04	780 000	张凯琳
白马服装批发公司	黄冈市九龙路特 1#	04/12/01	2 000 000	马自强

用于定义该领域知识的本体中描述的“公司”概念以及与数据源中实体对应关系如图 1 所示。在本文中为简化起见，

基金项目：国家自然科学基金资助项目(50305007)
作者简介：周建芳(1972－)，女，讲师、在职博士研究生，主研方向：信息集成，分布式数据库，语义Web；徐海银，副教授、博士；卢正鼎，教授、博士生导师
收稿日期：2007-11-11 **E-mail：**zhoujf@whpu.edu.cn

模式映射为简单的一一映射，实际上数据源和本体之间的模式映射是十分复杂的[3]。

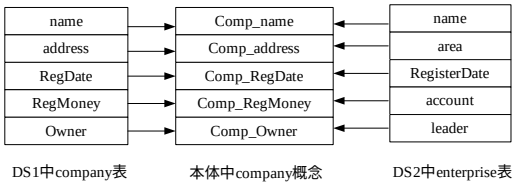

图 1　数据源中关系模式、本体及模式映射关系

当用户发出一个综合查询用来列出“所有在 02 年 3 月 1 日以前注册，注册资金在 700 000 元以上的公司”，该查询返回结果为空。因为用户假设注册资金以元为单位，日期格式是“yy/mm/dd”。但在 DS1 中，注册资金的单位是“万元”，日期格式是“yy/mm/dd”，DS2 中注册资金的单位是“元”，日期格式是“mm/dd/yy”。因此返回结果应为：

小四川餐饮服务公司	武汉市中华路 92 号	01/05/08	800 000	刘义
白马服装批发公司	黄冈市九龙路特 1#	01/04/12	2000 000	马自强

通过上面的例子可以看出，即使是模式语义相同的信息，也会由于不同数据源中对模式的一些缺省假设而造成信息语义上的模糊性。如果不能解决这种问题，则集成系统集成的结果会不正确或引起用户的迷惑，甚至得出相反的结论。本文讨论的正是这种情况下的语义异构问题的解决方法。

4　上下文描述

上下文在知识表示中起着十分重要的作用，因为任何一个声明语句都需要依靠其所在的上下文来确定真假。上下文在信息集成中的应用以麻省理工学院的COIN项目[4-5]为代表。结合已有文献和本文研究重点对上下文定义如下：

上下文——一套假设集合，据此假设可以使信息暗含的语义明确化。

每个数据源和进行综合查询的用户都在自己的上下文下理解信息的语义，当这些上下文不完全一致时就会产生混淆甚至完全相反的结果。但是在分布异构的环境中，强制要求所有的用户遵循相同的上下文是不现实的，因此必须在信息集成过程中不仅考虑到模式的异构，还要考虑上下文异构的情况。这就需要对上下文进行形式化的描述使得信息集成系统可以自动地检测并解决上下文异构的问题。

本文的前提假设是在集成系统范围内，维持着公共认可的上下文类型和取值的词汇。在此前提下，一个完整的可以由系统自动处理的上下文描述可以表示为四元组：

$C = \{CS,T,V,O\}$

其中，CS 表示集成系统中的上下文空间的集合，对于同一个数据源的所有上下文描述的集合称为一个数据源的上下文空间，可以用唯一的标识命名。例如，数据源 DS1 的上下文空间可以命名为 C_DS1；T 表示上下文类型集合，将可以采用同样方法处理的上下文异构归为同一上下文类型。对于上下文类型，用“currency”表示货币单位类型，“weight”表示重量单位，“dateformat”表示日期格式类型等；V 表示上下文类型的取值集合。同一上下文类型在不同数据源中具有不同的上下文取值，这正是导致具有相同模式语义的数据具有不同解释的原因。对于上下文取值，长度单位中统一用“m”表示“米”，或者在重量单位中用“g”表示“克”等；O 表示一组上下文操作，包括：

(1)上下文类型绑定：为具有上下文异构的属性绑定上下文类型。可以采用的形式如下：

bind(上下文空间标识，对象，属性)=上下文类型标识

其中，对于同一个属性，可以有不同类型的上下文绑定，例如：对于“注册资金”属性，可以有不同货币单位类型的上下文异构，也可以有不同的度量类型的上下文异构。

bind(C_DS1, company, RegMoney) = scale
bind(C_DS1, company, RegMoney) = currency
bind(C_DS1, company, RegDate) = dateformat
bind(C_DS2, enterprise, acount) = scale
bind(C_DS2, enterprise, acount) = currency
bind(C_DS2, enterprise, RegisterDate) = dateformat

(2)上下文类型赋值：同一类型的上下文异构在不同数据源中具有不同取值，这是引起混淆和错误的原因，因此对于各个数据源中存在异构的上下文类型必须指定确切取值，以便使模糊的语义明确化。在同一个数据源中，上下文异构类型的取值是一致的，对于上下文类型的赋值可以定义如下：

value(上下文空间标识，上下文类型) =上下文取值

例如，在前面所举的例子中，上下文赋值情况如下：

value(C_DS1,scale) = 10 000
value(C_DS2,scale) = 1
value(C_DS2, dateformat) = “mm/dd/yy”
value(C_DS1, dateformat) = “yy/mm/dd”
value(C_DS1,currency) = “RMB”
value(C_DS2,currency) = “RMB”
value(C_targer,scale) = 1
value(C_target,dateformat) = “yy/mm/dd”
value(C_target,currency) = “RMB”

(3)转换函数：为了消除具有相同模式语义而具有不同上下文语义的信息之间的语义异构，需要在同一类型的不同上下文取值之间进行转换。这种转换是通过在同一上下文类型内部不同上下文取值之间建立转换函数实现的。

5　基于上下文仲裁的查询处理过程

5.1　体系结构

通过在基于本体的语义信息集成的基础上引入上下文仲裁机制之后，既能够解决模式语义的冲突，又可以解决上下文语义异构，为信息的共享和重用以及信息系统互操作提供了完整的语义冲突解决方案。

基于上下文仲裁的信息集成体系结构如图 2 所示。

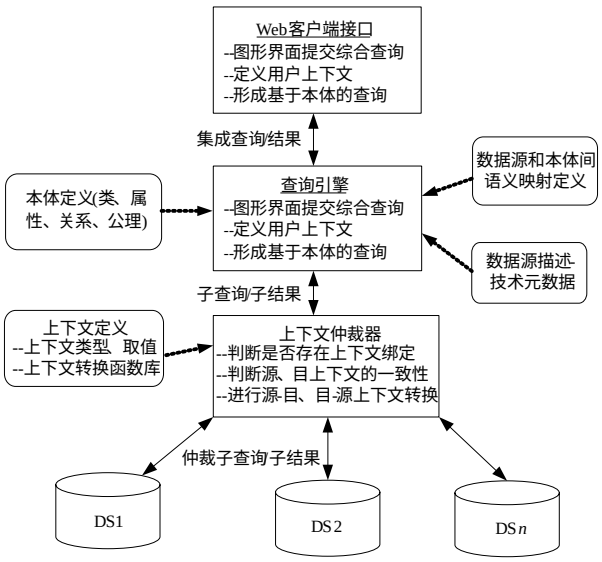

图 2　基于上下文仲裁的信息集成体系结构

在基于本体的语义信息集成的基础上，增加了一个组件：上下文仲裁器，同时在业务元数据管理组件中增加了如上所述的上下文描述。

用户在通过信息集成系统进行综合查询之前，先根据接口提示的上下文支持情况进行上下文设定，以满足不同用户进行不同上下文假设的要求。在查询处理过程中，将用户设定的上下文(称为用户上下文或目标上下文)作为参数传递给上下文仲裁器。

5.2 查询处理及仲裁过程

信息集成系统的查询处理过程如下：

(1)用户通过集成接口发出基于本体的综合查询，由查询引擎通过搜索模式映射规则将基于本体的综合查询分解成与各个数据源相关的子查询，并将子查询传递给上下文仲裁器。这一步已经根据模式映射规则解决了模式语义异构。

对于例子中的综合查询，经查询引擎搜索数据源与本体间的语义映射规则和技术元数据之后，生成的 2 个子查询分别为：

SubQuery to DS1:

Select name, address, RegDate, RegMoney, owner from company

where RegDate <= “02/03/01” and RegMoney >= 700 000

SubQuery to DS2:

Select name, allocate, RegisterDate, account, leader from enterprise

where RegisterDate <= “02/03/01” and account >= 700 000

(2)上下文仲裁器对于每一个子查询，搜索各个数据源的上下文描述，确定子查询中涉及到的具有上下文异构的属性，确定异构类型及该类型在指定上下文空间中的取值(称为源上下文)。比较每一个源上下文取值和目标上下文取值，如果不同，则表示存在上下文异构，由上下文仲裁器根据上下文类型去转换函数库中调用相应的转换函数来实现目标上下文到源上下文的转换。这一步解决了具有相同模式语义的信息之间的上下文异构。上下文仲裁器将进行了上下文转换的子查询传递给各个数据源。

对于(1)中生成的 2 个子查询，上下文仲裁器通过搜索涉及到的数据源的上下文描述，发现 DS1 中 company 对象的 RegDate 和 Regmoney，DS2 中 enterprise 对象的 RegisterDate 和 acount 也具有上下文描述。而且源上下文描述与目标上下文有差异，通过调用上下文转换函数将子查询分别转换为：

Mediated SubQuery to DS1:

Select name, address, RegDate, RegMoney, owner from company

where RegDate <= “02/03/01” and RegMoney >= 70

Mediated SubQuery to DS2:

Select name, allocate, RegisterDate, account, leader from enterprise

Where RegisterDate <= “03/01/02” and account >= 700 000

(3)各个数据源进行局部查询后就将子查询结果返回给查询引擎，查询引擎根据上下文异构的情况，将查询结果进行源上下文到目标上下文的转换，合并计算后，执行格式转换，最后将转换后的综合结果按照用户上下文的设定返回给用户。

6 分析

本文提出的基于上下文仲裁的信息集成框架可以很好地解决属性级上下文异构问题，而且在本方案中，由于数据源和本体是各自独立发展的，本体与数据源之间的映射以及上下文描述都是在中间逻辑层采用声明式定义，因此数据源的变化和本体的进化都不会互相影响，只需要调整中间逻辑层即可。该解决方案具有很好的适应性、扩展性并易于维护。

适应性 :指当上下文发生变化后系统仍然能够自动处理，较少需要人工干预。当上下文(源上下文)的取值发生变化时(该变化了的取值为已经存在的取值)，只需要修改相应属性的上下文赋值即可。上下文仲裁器在生成仲裁子查询时调用不同的转换函数来解决上下文异构。

扩展性 :即当上下文有所扩展(增加新的上下文类型或取值)时对集成系统(包括本体和数据源以及查询引擎和仲裁器)所做的修改应尽量最小。根据语义异构解决原理，上下文仲裁器能够解决的上下文语义异构类型需要事先预定。因此，当信息集成系统中上下文异构有所扩展时只需要在元数据管理模块中注册新增上下文的描述(包括上下文类型、上下文赋值和新增上下文取值与已有取值之间的转换函数)，如此，上下文仲裁器就可以自动检测和解决新增上下文异构情况。

7 结束语

对分布异构的数据源进行集成以进行信息的交换和共享以及提供信息系统之间的互操作在当前的信息社会变得尤为紧迫。已有的基于本体的语义信息集成有效地解决了异构数据源之间的模式语义异构和实体级上下文异构(同义字和同音异义字)，但对于普遍存在的具有相同模式语义但数据语义随上下文不同而有不同解释的属性级上下文异构无能为力。

本方案在基于本体的语义信息集成的基础上引入上下文描述和上下文仲裁机制，提供了一种灵活的方式动态地解决分布异构环境下普遍存在的上下文异构问题，对于简单的基于模式匹配的语义异构解决是一种很好的补充。它弥补了基于本体的语义信息集成的不足，两者共同形成了完整的语义异构解决方案。

但是在上下文描述中，为了能够解决下文异构的问题，必须定义同一上下文类型内部不同上下文取值之间的双向转换函数。当上下文类型和取值的数量很大时，需要定义的转换函数的数量相当大，这些函数的手工维护难度也很大。如何给出一个需要较少转换的转换函数，并能自动推理和自动维护上下文仲裁时使用的转换函数的处理机制是笔者下一步需要重点研究的问题。

参考文献

[1] Goh C H. Context Interchange: New Features and Formalisms for the Intelligent Integration of Information[J]. ACM Transaction on Information Systems, 1999, 17(3): 270-293.

[2] Cui Zhan, Jones D, Brien O P. Issues in Ontology-based Information Integration[C]//Proc. of the IJCAI-01 Workshop on Ontologies and Information Sharing. Seattle, USA: [s. n.], 2001.

[3] An Yuan, Borgida A, Mylopoulos J. Inferring Complex Semantic Mappings Between Relational Tables and Ontologies from Simple Correspondences[C]//Proc. of Int. Conf. on Ontologies, Databases and Applications of Semantics. Agia Napa, Cyprus: [s. n.], 2005.

[4] Firat A. Information Integration Using Contextual Knowledge and Ontology Merging[D]. Cambridge, Massachusetts, USA: MIT Sloan School of Management, 2003.

[5] Zhu Hongwei, Madnick S E. Context Interchange as a Scalable Solution to Interoperating Amongst Heterogeneous Dynamic Services[C]//Proc. of the 3rd Workshop on eBusiness. Washington, D. C., USA: [s. n.], 2004.