高技术通讯 2010 年 第 20 卷 第 10 期:1055 ~ 1061

doi:10.3772/j.issn.1002-0470.2010.10.012

分布式本体库存储模型的设计与优化①

郭剑锋②* ** 范玉顺* 狄小峰*

(*清华大学自动化系 北京 100084)

(**中国科学院科技政策与管理科学研究所 北京 100190)

摘 要 针对如何高效地构建用于语义交互的分布式本体库展开了研究。首先,分析了基于本体的语义交互平台的通用体系结构,进而明确分布式本体库在平台中的作用,提出了一种分布式本体库存储模型。围绕该模型,详细讨论了网络本体库与本地本体库的映射方法,语义数据的编码方法,然后详细分析了该模型中指令分发和数据演化组件的工作机制,最后通过实验验证了:在集群存储模式下,对于大数据量查询,网络数据传输对查询时间开销的影响远远小于查询本身,从而为分布式本体库存储效率的优化奠定了基础,即在本体库中采用分布式集群存储可以达到优化检索效率的目的。该存储模型已经在 STASIS 平台中得到应用。

关键词 语义交互,本体库,存储模型,数据检索

0 引 言

随着电子商务的普及和企业信息化的发展,企业间如何通过互联网进行高质量的信息交互是必须要解决的问题。研究和实践表明,要真正实现无障碍信息交互,仅仅通过消除词法的、句法的、结构的数据异质还是不够的,必须从更高的语义层面消除数据的异质[1],另外,电子商务环境下企业间信息交互的本质是 P2P 的信息交互,即参与电子商务活动的任意两个用户能够达到信息共享、信息共识、信息共用,因此,分布式环境下如何实现基于语义的 P2P 信息交互是多年来研究的最终目的。P2P 环境下,由于用户节点是完全自治的,不同用户采用不同的方式存储信息,导致企业间信息的异构,引发了互操作问题[2,3]。另一方面,P2P 环境中难以进行集中控制,这样就对每个节点独立进行信息处理的能力提出了更高的要求。针对上述问题,大多数研究的思路是如何构建网络环境下面向多用户本体创建、共享、重用的本体集成平台。

从系统架构上讲,相关研究有 Fern′andez-Breis 和 Mart′1nez-B′ejar[4]提出的面向本体集成的合作框架,语义网中分布式本体处理框架 MAFRA[5],面向本体集成的框架 OISs[6], Madhavan 提出的本体映射框架[7],面向集成高层次本体的 OntoMapO[8],Kent 提出的本体共享框架 IFF[9]等。这些框架的侧重点各有不同,其共性可以粗略描述为客户端、网络层、服务端三层架构。客户端,即本体编辑工具,让用户实现各自领域本体的创建、导入、修改以及建立与外部本体的映射;网络层指存储本体数据的网络本体库(有集中式和分布式两种),实现共享本体的存储;服务端,负责共享本体的处理(包括映射、对齐、优化和合并等)以及共享本体的演化。客户端本体编辑工具的研究主要有早期的 Ontolingua[10]、OntoSaurus[11]、WebOnto[12]等,到后来的 Protégé[13]、WebODE[14]、OilEd[15]、OntoEdit[16]以及 KAON[17]等。此外,当前在本体处理和演化方面也有较多研究[18-21]。作者尚未查找到专门讨论分布式本体数据的存储架构及优化等问题的文献,原因是基于本体的语义交互尚处于启蒙阶段,很多如本体创建、映射、匹配、对齐以及合并的基础性问题仍然存在众多难点和分歧,需要更多的研究和标准化工作来推进,因而未把分布式本体的存储作为一个单独的问题来研究。但是,针对特定的语义交互平台,其具有明确的框架和功能范围,在其推广阶段,数据量激增,可明显看出本体存储环节对整个系统性能的影响。本

① 973 计划(2006CB705407),863 计划(2009AA010308,2007AA04Z150)和国家自然科学基金(60674080)资助项目。

② 男,1976 年生,博士;研究方向:网络化制造,语义网,智能信息检索,知识工程等;联系人,E-mail: guojf@tsinghua.edu.cn

(收稿日期:2009-09-22)

文以作者实验室参加的欧盟第六框架项目周边语义交互服务软件(software for ambient semantic interoperable service, STASIS)[22]为依托,从指令操作和数据检索两个方面讨论了一种网络本体库存储模型。

1 网络本体库存储模型

1.1 本体集成平台的体系结构

本体集成平台的简化框架如图 1 所示。本体库是本体集成平台的一部分,因此对于本体库的研究需要有本体集成平台作为背景。本文以 STASIS 平台为背景,STASIS 系统框架如图 2 所示,其可以看作图 1 所示简化框架的一种细化。其中客户端包括:STASIS 语义实体(STASIS semantic entity, SSE)编辑器,编辑最小单位的用户语义实体;专用 STASIS 语义实体(terminological STASIS semantic entity, SSEt)编辑器,编辑以类为单位的用户语义实体或共享语义实体。服务端包括:过滤器组件,负责发现逻辑错误和清楚垃圾数据;比较器组件,通过匹配和自动地发现潜在的各种语义实体间的映射,其它如本体对齐、合并、进化等动作没有显性的组件,以 Web Sevice 接口的形式通过网络被调用。图中STASIS链接规范

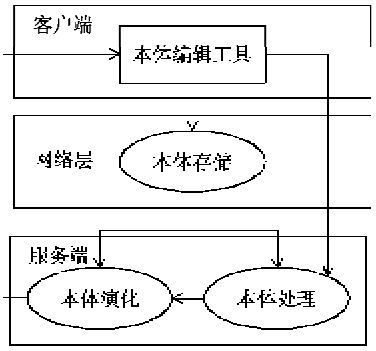

图 1 本体集成平台简化框架

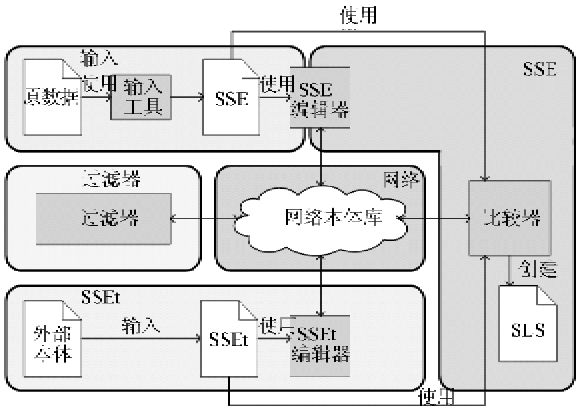

图 2 STASIS 系统架构[2,3]

(STASIS link specification, SLS)记录语义实体间的映射关系的语义实体,其不同于本体内部概念间的关系。网络本体库是整个系统枢纽,负责各种功能指令的导向和数据的存储与优化,这是本文的研究内容。语义实体(semantic entity, SE)包括 SSE、SSEt、SLS。

1.2 分布式本体库存储模型

本文提出和讨论了一种分布式本体库存储模型,分布式包括功能指令在服务端分布执行,以及共享本体的分布存储两方面。分布式本体库存储模型(storage model for distributed ontologies repository, SM4DOR)的组成及其架构如图 3 所示。用户通过 STASIS 客户端工具(STASIS Workbench, 可以在 http://www.stasis-project.net/intranet/home.cfm 下载)完成与其它用户的语义交互。其主流程是:(1)用户领域本体的创建,通过新建或其他数据格式的导入完成,输出结果是各种 SE 的集合;(2)编辑调整 SE 及其属性;(3)将用户本体以 Web 本体语言(OWL)形式保存到本地本体库或网络本体库;(4)在共享本体库中检索需要的 SE;(5)在比较器的帮助下,自动或半自动地建立 SE 间的映射;(6)映射结果以 OWL 形式保存到网络本体库。从安全角度,保存入网络本体库的数据会同时保存到本地本体库,反之不真。

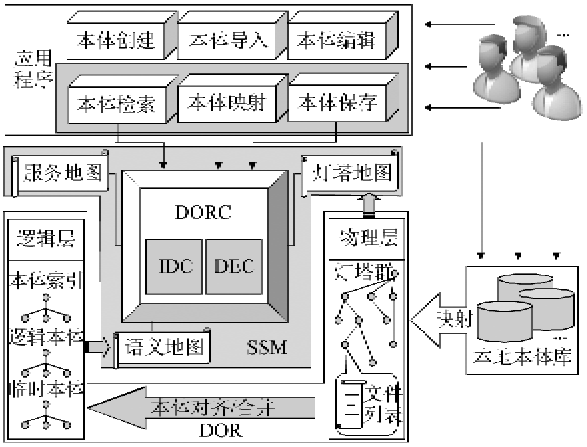

图 3 SM4DOR 的涵盖与架构

1.2.1 网络本体库与本地本体库的映射

本地本体库与网络本体库的映射的目的是:建立共享本体(以 OWL 文件记录和存储)在本地本体库和网络本体库之间的关联。一个共享本体在本地本体库和网络本体库中同时存在,映射的结构关系如图 4 所示。存储灯塔(beacon):网络本体库的一个存储 Web Service 实例,用 Web Service URL 向用户指出其入口,可能运行在不同的计算机上,具有存储空

间和运算能力,以下简称灯塔。分布式本体库存在多个灯塔,它们按照一定的关系组织关联在一起,并行有机地协同完成任务,组织在一起的多个灯塔集合称为存储集群(flock),以下简称集群。

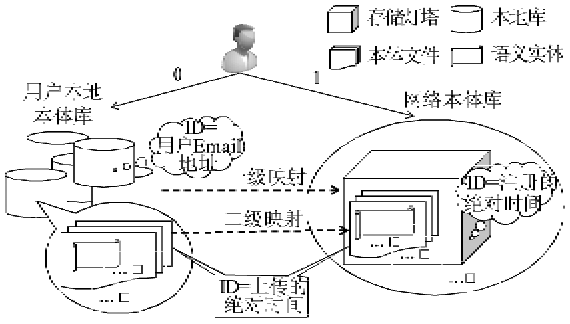

图 4　本地本体库与网络本体库的映射

映射过程中的两个问题:(1)如图 4,映射过程分为两级:第一级是本地本体库与灯塔绑定,当某用户通过客户端工具设置灯塔 Web Service 入口后;第二级是本地本体库中的部分本体文件与绑定灯塔中本体文件绑定。(2)存储模型中元素编码的作用:在各自空间中为各元素生成一个唯一的 ID。各种元素的具体编码方法见图 4,不累述。

1.2.2　SM4DOR 的组成

如图 3 所示,SM4DOR 由三个部分组成:分布式本体库控制器(distributed ontology repository controller, DORC)、存储服务地图(storage service map, SSM)和分布式本体库(distributed ontology repository, DOR)。其中,DORC 由指令分发控制器(instruction dissemination controller, IDC)和数据演化控制器(data evolution controller, DEC)组成。

DOR 分别从物理层和逻辑层存储共享本体的数据内容。从物理层角度,DOR把一个SE表达为某灯塔中的某文件的片段 OWL 描述;从逻辑层角度,DOR 一个 SE 表达为符合语义限定的一个实体。从物理角度,DOR 中存储的是用户发布的原始本体文件集合。从逻辑角度,DOR 存储的数据有本体的索引数据、用户发布的原始本体经过对齐或合并后的到新本体集合以及按交互需求产生的临时本体集合。因此,DOR 中也提供了物理的和逻辑的两种检索和操作 SE 的途径,前者一般用于后台操作与维护,后者面向用户。

SSM 为 DORC 和 IDC 的运行提供数据支持,其中服务地图存储的是操作指令与操作对象的关联信息。语义地图即 DOR 中逻辑数据的概览和索引信息,其帮助完成基于语义的各种操作,本文不进行研究。灯塔地图记录集群的概况以及集群中灯塔的相互关联。

DORC 是 SM4DOR 的核心,下面两节分别讨论其中的 IDC 和 DEC。

2　指令分发控制器——IDC

DOR 是网络环境下的语义交互平台的数据支持,因此,IDC 要在交互平台操作多变、DOR 中数据种类多样以及大数据量的情况下,保证交互平台中各项操作能准确定位到正确的操作对象,实现交互系统的安全验证机制。IDC 的工作依赖于服务地图中的指令导航信息。服务地图与 IDC 工作流程如图 5所示。可以看出,语义地图中的指令面向的是各种元素的物理操作,原因是:(1)IDC 是 SM4DOR 的组成部分,不参与数据的逻辑操作;(2)任何逻辑操作,如果其修改了 DOR 中的数据,都可以分解为相应的物理指令。

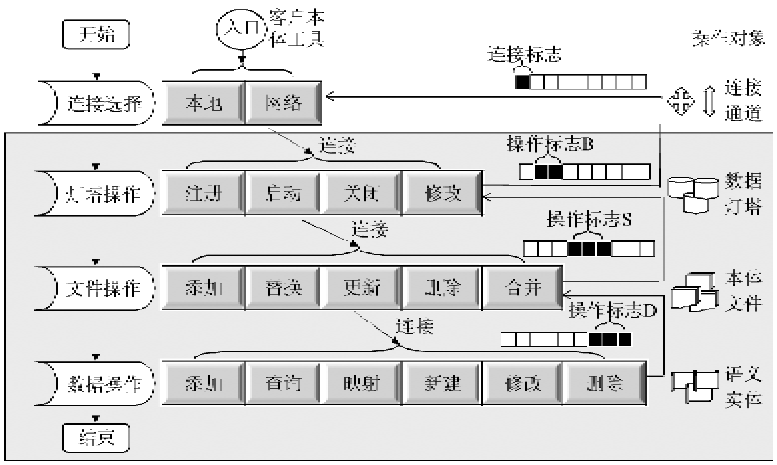

图 5　服务地图与 IDC 工作流程

3 数据演化控制器——DEC

3.1 DEC 的体系结构

DEC 是 DORC 的组成之一,从形式上讲,DEC 是一个独立的、对用户透明的 Web Service 实例,其挂载在集群中所有灯塔之前,负责与数据操作有关的指令的分解、分发,存储所有 SE 的映射数据——SLS,以及本体数据的统计、合并,本体数据存储的优化。DEC 的体系结构如图 6 所示。集群中所有的灯塔由灯塔转发服务、灯塔内部服务和数据内容三部分组成。每个用户应用程序和一个灯塔绑定,用户数据操作的指令发向与其绑定的灯塔,灯塔转发服务把用户指令转发到 DEC 公共服务,DEC 公共服务再将分解和分析后的指令集发到相关的灯塔,调用相应的灯塔内部服务。

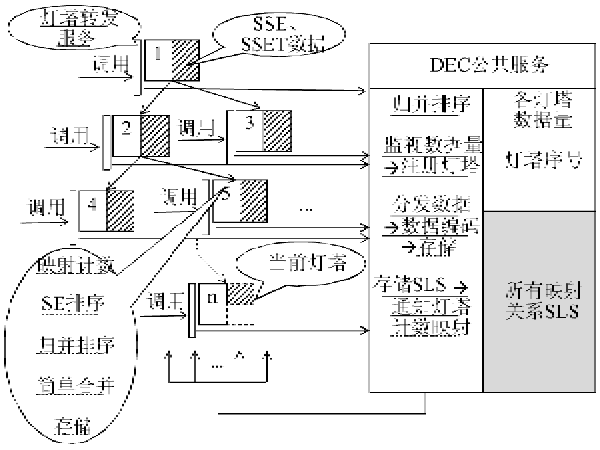

图 6 DEC 体系结构

(1) 灯塔内部服务

灯塔内部服务包括:(1)映射计数,负责随时计数和更新存储在当前灯塔中的 SE 拥有的映射数目;(2)SE 排序,对当前灯塔中的 SE 按照其拥有的映射数目排序;(3)归并排序,对汇集到当前灯塔中的所有 SE 按照其拥有的映射数目排序,即采用归并排序算法合并当前灯塔与其向下一层的子灯塔的 SE 排序结果;(4)简单合并,将用户保存到当前灯塔中的本体文件中的 SE 合并入已有本体集合,作为排序的输入。

(2) DEC 公共服务

DEC 公共服务包括:(1)归并排序,对所有 SE 按照其拥有的映射数目排序,即采用归并排序算法合并当前灯塔与其向下一层的子灯塔的 SE 排序结果;(2)监视当前灯塔中的数据量,即 SE 的数目,如果超出阈值则新建一个灯塔;(3)分发数据,根据编码规则对本体数据进行编码,将新建、更新和编辑过的数据存储到相应的灯塔,其中新建的本体数据存储入未达阈值的当前灯塔;(4)存储服务存储 SE 相互之间映射的 SLS,存储在 DEC 公共服务的数据区,调用 SLS 源节点和目标节点 SE 所在灯塔的“映射计数”服务。

(3) SE 排序的作用

对 SE 排序的目的是评价其重要程度,可以简单认为某个 SE 拥有的较多的映射,其重要性也越大。如果新加入的 SE 与重要性高的 SE 建立映射,其连接到其他 SE 的平均路径将缩短,从而提高交互过程中的检索效率。在单个 SE 能够拥有的映射数没有限制的情况下,通过一段时间的自然选择和演化,将形成以重点用户(如声誉高的企业)的 SE 为中心集合的聚集度很高的语义关联网络,这既是现实情况的反映,也有利于提高整个系统的交互效率。

3.2 存储集群的优化

从以上分析可知,集群的结构直接关系到语义信息的检索效率。在硬件设施均等的条件下,网络环境下集群对检索效率的影响因素有单个灯塔存储数据量的阈值、网络数据传输效率以及集群的组织结构。单个灯塔存储数据量的阈值决定同样数据总量条件下集群中灯塔的数量。集群存储的本质是变单一存储服务的单线程查询为多线程查询,从而缩短查询时间,但另一方面,集群存储必然在网络数据传输以及结果合并上耗费额外的时间,这是一对矛盾。集群中灯塔的组织结构可以是任何形式,其对检索效率有一定的影响,但无论存储集群采取何种组织结构,其本质都是上述矛盾不同均衡模式的体现。因此定量分析多线程查询和网络开销的关系是至关重要的。由于用户在交互过程中对数据的检索是完全随机的,没有固定模式,因此,我们采用实验的方法进行分析。

实验目的:对比单一存储服务和建立集群后数据检索需要的时间。

实验内容:分别在单一存储服务和集群(分布检索)条件下,测试客户端检索 10 到 100 万条 SE 记录需要的时间。

实验条件:集群中的组织为平衡二叉树;单个灯塔节点中的数量阈值设为 10 万条 SE;实验数据为相同的 100 万条 SSE 测试记录;测试服务入口为 http://84.242.131.209:9090/axis2/services/RepositoryService。

每次检索的 Sparql 语句如下:

PREFIX rdf… < http…//www.w3.org/1999/02/22-rdf-syntax-ns〕> PREFIX cdm… < http…//stasisproject.net/cdm〕> PREFIX pre… < http…//stasisproject.net/model〕> SELECT ? subject ? ID ? SeemID ? Name ? CreationDate ? Rating ? Owner ? Domain ? LastUsageDate? PurificationDate ? Description WHERE { ? subject rdf…type pre…SSE OPTIONAL {? subject pre…ID ? ID.} OPTIONAL {? subject pre…SeemID ? SeemID.} OPTIONAL {? subject pre…Name ? Name.} OPTIONAL {? subject pre…creationDate ? CreationDate.} OPTIONAL {? subject pre…Rating ? Rating.} OPTIONAL {? subject pre…ownerID ? Owner.} OPTIONAL {? subject pre…Domain ? Domain.} OPTIONAL {? subject pre…LastUsageDate ? LastUsageDate.} OPTIONAL {? subject pre…PurificationDate ? PurificationDate.} OPTIONAL {? subject pre…Description ? Description.} }

实验结果见表 1,其中 t_1 为单一存储服务条件下的测试结果,t_2 为集群(分布检索)条件下的测试结果。对实验结果拟合结果见图 7。

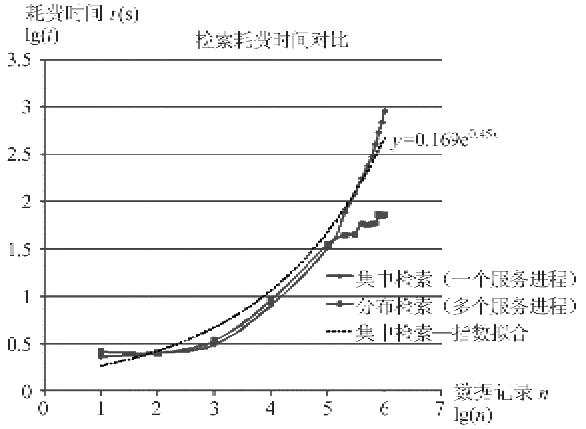

图 7 不同模式下检索耗时对比

表 1 实验结果

数据记录 n	$\lg(n)$	t_1(ms)	$t_1/1000$	$\lg(t_1/1000)$	t_2(ms)	$t_2/1000$	$\lg(t_2/1000)$
10	1	2312	2.312	0.363988	2603	2.603	0.415474
100	2	2536	2.536	0.404149	2531	2.531	0.403292
1000	3	3120	3.12	0.494155	3407	3.407	0.532372
10000	4	8102	8.102	0.908592	9090	9.09	0.958564
100000	5	32574	32.574	1.512871	35666	35.666	1.552254
200000	5.30103	76251	76.251	1.882246	44422	44.422	1.647598
300000	5.477121	121155	121.155	2.083341	45600	45.6	1.658965
400000	5.60206	175684	175.684	2.244732	58547	58.547	1.767505
500000	5.69897	239651	239.651	2.379579	56890	56.89	1.755036
600000	5.778151	301259	301.259	2.47894	58511	58.511	1.767238
700000	5.845098	402222	402.222	2.604466	59041	59.041	1.771154
800000	5.90309	534885	534.885	2.72826	73444	73.444	1.865956
900000	5.954243	691586	691.586	2.839846	70565	70.565	1.848589
1000000	6	905743	905.743	2.957005	72006	72.006	1.857369

从图 7 可以看出,单一存储服务条件下,随着检索数据量的增加,检索耗费的时间成指数增长,当检索数据量为 100 万时,耗时为 15min 左右;而集群(分布检索)条件下,当检索数据量为 100 万时,耗时为 1.2min 左右,而且从 70 万开始随着数据量的增加,集群条件下检索耗费时间的变化率明显减小,符合使用的要求。

对实验结果的分析和解释:(1)单一存储服务条件下,检索耗费的时间主要决定于用 Sparql 查询耗费的时间;(2)集群条件下,检索耗费的时间决定于用 Sparql 查询耗费的时间和数据在节点间传输的时间;(3)随着一个节点内查询数据量的增加,Sparql 的查询时间成指数上升;(4)集群中的节点数增加,导致灯塔构成的平衡二叉树深度增加,随之数据在节点间传输的时间增加;(5)随着数据总量的增加,数据在节点间传输的时间虽然有所增加,但相对数据量的增加变化非常缓慢,使得检索耗费时间变化率大幅降低。

4 SM4DOR 在 STASIS 中的应用

SM4DOR 在 STASIS 中的实现是数据存储和操作的一系列组件。用户在 STASIS 工具中任何与其他用户本体数据有关的操作都要通过连接 DOR 才

能完成,以某用户通过 STASIS 工具完成一个自己的本体文件与其他用户的一个本体文件的映射为例,过程如图 8 所示。第一步,用户需要在 STASIS 工具中建立与 DOL 的连接,即设置入口灯塔的服务地址、用户和密码就可以建立与 SM4DOR 的连接;第二步,建立与 DOR 的连接后,可以通过启动 STASIS 工具中的比较器,比较器首先会通过对 DOL 的检索,列出所有 DOR 中所有可以用于比较的本体文件(所有用户);第三步,从列出的文件列表中选择两个作为比较对象(其中一个为自己的本体文件);第四步,比较器通过计算得到比较结果;第五步,用户根据比较器提供的比较结果,进行接受、修改和拒绝等操作完成两个本体文件的映射;第六步,将映射后的结果保存回 DOR 中。

以上描述了一个 STASIS 工具中基础和常用的操作,从中可见 DOR 是 STASIS 平台功能得以实现的重要基础之一,其结构、效率和稳定性等的优劣直接影响整个系统的使用效率。

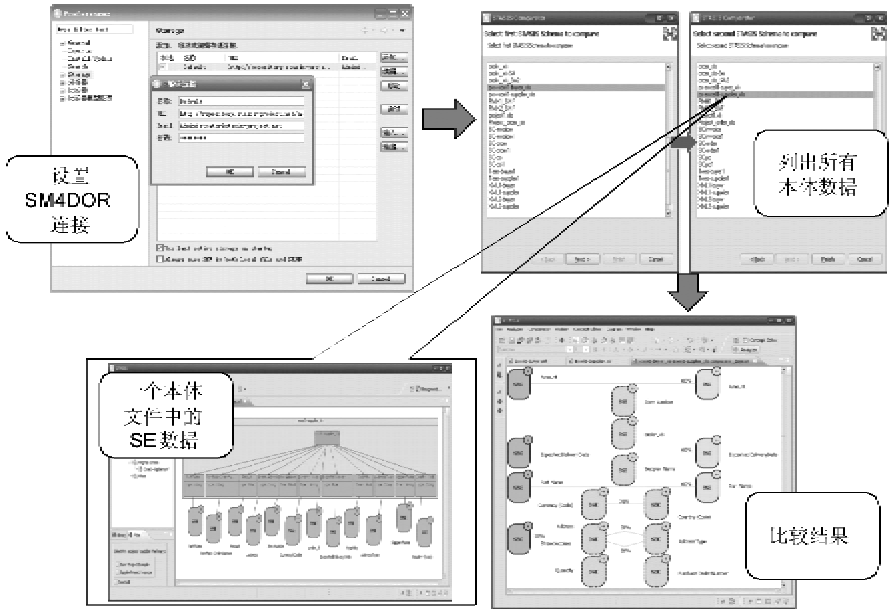

图 8 STASIS Workbench 通过 SM4DOR 完成本体比较

5 结 论

在分布式条件下构建面向多用户的本体集成平台时,本体数据的网络存储机制关系到平台中本体数据检索的效率、用户交互的实时性以及操作指令的优化等方面。以作者实验室参加的欧盟第六框架项目 STASIS 为背景,从指令操作和数据检索两个方面讨论了一种网络本体库存储模型,得到如下结论:(1)集中式本体数据库不能满足分布式条件下本体数据存储和交互的需求,随着数据量的增大,很难保证实时性。因此,必须采用具备某种组织结构的分布式集群才能满足需求。(2)合理的分布式存储模型有利于设置和保护用户数据的安全,保证语义实体标识编码的唯一性,快速地定位语义实体,高效地执行数据操作指令。(3)随着交互平台中总数据量的增加,数据传输对检索时间的影响远远小于检索本身的影响。(4)分布式存储结构对于用户不可见,从而保证用户使用的便利性,但必须保证任何用户从任意一个入口连接分布式本体库后,执行所有操作的效率一致。(5)分布式本体库中的数据演化后,保证对于所有用户的平均使用效率最优,但不能保证对某个用户是最高效的。

本文以项目为背景介绍了一种分布式本体存储结构,指明了设计分布式本体库时可能涉及的各个方面,并给出其中一种解决方案,但并不是唯一的解决方案,作者认为在设计分布式本体库时应进一步进行以下两方面研究:

(1)本文中的分布式存储集群采用平衡二叉树结构,这是一种比较简单的结构类型,而且是静态的。但实际上,我们知道发生交互的用户间一般有着实际的联系,如,属于共同的行业或区域,所以,应该根据用户群的属性对其分类,并使之与集群的组织结构动态关联,这必将进一步提高交互平台的工作效率。(2)结合已经建立的分布式本体库存储模型,进一步研究分布式环境下语义交互系统的安全验证机制。

参考文献:

[1] Yin X, Han J, Yu P S. Linkclus: Efficient clustering via heterogeneous semantic links. In: Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, 2006. 427-438

[2] Euzenat J. An infrastructure for formally ensuring interoperability in a heterogeneous semantic Web. In: Proceedings of the 1st Semantic Web Working Symposium, Palo Alto, USA, 2001.345-360

[3] Castano S, Ferrara A, Montanelli S, et al. Matching techniques for resource discovery in distributed systems using heterogeneous ontology descriptions. In: Proceedings of International Conference on Coding and Computing, Las Vegas, USA, 2004. 360-366

[4] Fern′andez-Breis J, Mart′1nez-B′ejar R. A cooperative framework for integrating ontologies. *International Journal of Human-Computer Studies*, 2002,56: 665-720

[5] Maedche A, Staab S. Semi-automatic engineering of ontologies from texts. In: Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering, Chicago, USA, 2000. 231-239

[6] Calvanese D, Giacomo G, Lenzerini M. Ontology of integration and integration of ontologies. In: Proceedings of the 9th International Conference on Conceptual Structures, Stanford, USA, 2001

[7] Madhavan J, Bernstein P A, Domingos P, et al. Representing and reasoning about mappings between domain models. In: Proceedings of the 18th National Conference on Artificial Intelligence, Edmonton, Canada, 2002. 80-86

[8] Kiryakov A, Simov K I, Dimitrov M. OntoMap: portal for upper-level ontologies. In: Proceedings of the 2nd International Conference on Formal Ontology in Information Systems, New York, USA, 2001.47-58

[9] Kent R E. The information flow foundation for conceptual knowledge organization. *Advances in Knowledge Organization*. 2000, 7: 111-117

[10] Farquhar A, Fikes R, Rice J. The Ontolingua server: A tool for collaborative ontology construction. *Int' l Journal of Human-Computer Studies*, 1997,46(6):707-727

[11] Swartout B, Ramesh P, Knight K, et al. Toward distributed use QDof large-scale ontologies. In: Proceedings of the AAAI Symposium Series Workshop on Ontological Engineering, Providence, USA, 1997.138-148

[12] Duineveld A J, Stoter R, Weiden M R, et al. Wonder tools? A comparative study of ontological engineering tools. *Int' l Journal of Human-Computer Studies*,2000,52(6):1111-1133

[13] Noy N F, Fergerson R W, Musen M A. The knowledge model of protégé-2000: Combining interoperability and flexibility. In: Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management, Juanles-Pins, France, 2000. 17-32

[14] Arpirez J C, Corcho O, Fernandez-Lopez M, et al. WebODE: A scalable ontological engineering workbench. In: Proceedings of the 1st International Conference on Knowledge Capture, Victoria, B.C., Canada, 2001.6-13

[15] Bechhofer S, Horrocks I, Goble C, et al. OilEd: A reasonable ontology editor for the semantic Web. In: Proceedings of the KI2001, Joint German/Austrian Conference on Artificial Intelligence, Vienna, Austria, 2001. Lecture Notes in Computer Science, 2174.396-408

[16] Sure Y, Angele J, Erdmann M, et al. OntoEdit: Collaborative ontology engineering for the semantic Web. In: Proceedings of the International Semantic Web Conference 2002, Sardinia, Italia, 2002.221-235

[17] Bozsak E, Ehrig M, Handschuh S, et al. KAON-Towards a large scale semantic web. In: Proceedings of the 3rd International Conference, Ec-Web 2002, Aixen-Provence, France, 2002.304-313

[18] 杜小勇,李曼,王珊. 本体学习研究综述. 软件学报, 2006, 17(9): 1837-1847

[19] 于娟,党延忠. 本体集成研究综述. 计算机科学,2008, 35(7): 9-13

[20] 仲新宇. 基于结构相似的本体匹配方法综述. 信息技术与标准化,2008,(12): 43-45

[21] 韩婕,向阳. 本体构建研究综述. 计算机应用与软件, 2007, 24(9): 21-23

[22] STASIS project, SofTware for Ambient Semantic Interoperable Services [OL], http://www.stasisproject.net, 2008

[23] Bhullar G, Núñez MJ, Roca de Togeras A, et al. STASIS D2.3.4 Architecture [OL]. Available from: http://www.stasisproject.net/deliverables/. 2008.01.04

The design and optimization of a storage model for distributed ontology repositories

Guo Jianfeng[*,**], Fan Yushun[*], Di Xiaofeng[*]

([*] Department of Automation, Tsinghua University, Beijing 100084)

([**] Institute of Policy and Management, Chinese Academy of Sciences, Beijing 100190)

Abstract

Oriented to how to establish distributed ontology repositories efficiently to support semantic interaction among different users though the Internet, at the beginning, the paper reviews some semantic interoperation platforms based on ontology repositories to extract the common architecture of this kind of system, which helps to define the role of distributed ontology repositories in the whole system. A storage model for distributed ontology repositories is presented. According to the model, the method for making mapping between network ontology repositories and local repositories is discussed, and the encoding mechanism for semantic entities is explained. Then, the principles of the instruction dissemination controller and the data evolution controller are analyzed in detail. Finally, based on the experiment, it is validated that data transfer in network contributes much less to time cost than query itself in the cluster storage pattern, which presents the base for optimizing the retrieval efficiency in distributed ontologies repositories, namely, the optimization for the efficiency of data retrieval in ontologies repositories can be achieved by adopting the distributed cluster storage. The storage model has been applied to the project of the software for ambient semantic interoperable service (STASIS).

Key words: semantic interoperability, ontology repository, storage model, data retrieval