基于用户日志的本体进化

郭 臻,徐德智,汪智勇
GUO Zhen,XU De-zhi,WANG Zhi-yong

中南大学 信息工程学院,长沙 410083
College of Information Engineering,Central South University,Changsha 410083,China
E-mail:guozhencs@163.com

GUO Zhen,XU De-zhi,WANG Zhi-yong.Ontology evolution based on userlog.Computer Engineering and Applications, 2008,44(5):191-195.

Abstract: Ontology is the cornerstone of the semantic Web.It plays the more and more important role in the semantic Web.The purpose of ontology evolution is dynamically to adjust itself in the changing environment.The thesis start from the users log on domain ontology,used Aporior mining algorithms to get frequent and frequent users of the operating path.Through the analysis of frequent path,and get the advice of the ontology evolution and realize the method in experiments.The experiments prove the method is effective.

Key words: ontology;userlog;sequent mine;ontology evolution

摘 要:本体作为语义网的基石,发挥着越来越重要的作用。本体进化的目的在于动态调整本体以适应环境的变化。从基于领域本体的用户日志出发,使用 Aporior 算法挖掘用户操作的频繁项集和频繁路径。通过对频繁项集和频繁路径的分析,得到了本体进化的辅助建议,并在实验中对该方法进行了实现和测试,实验结果证明该方法能有效地实现本体进化。

关键词:本体;用户日志;序列挖掘;本体进化
文章编号:1002-8331(2008)05-0191-05 **文献标识码:**A **中图分类号:**TP331

1 前言

伴随着 Internet 的飞速发展,Web 上的信息不断增加,几乎成为了可以包含一切内容的信息库。网络资源的丰富和共享给社会生活带来了极大的方便,但是目前网络上的数据通常是适合于人阅读,它们的语义却不能被机器理解,不便于机器对信息的自动处理, 也不能够使人们按内容的语义表达需求,迅速准确地从成千上万的网页中过滤出自己感兴趣的内容。为了解决以上问题万维网之父 Tim Berners-Lee 提出了下一代万维网——语义 Web[1]的理念。本体(Ontology)位于语义 Web 体系结构中的第四层, 它是解决语义层次上 Web 信息共享和交换的基础。

在本体信息系统中,终端用户在系统的交互界面执行基于领域本体的查询和浏览操作。领域本体对于领域内的信息而言是一个概念结构,但是对于基于它的操作而言,它同时也是数据。每当用户执行基于领域本体的相关操作时,所访问的本体实体、操作时间、用户 ID 等信息,在用户日志中都有相应的记录。通过分析用户日志,能够发现用户操作本体的共同行为,进而优化本体的拓扑结构,为本体进化提供参考建议,同时加深对领域相关概念的认识。

本文通过对领域本体用户日志的分析,研究用户操作的上下文关系与用户需求的关系,通过对关联规则和序列的处理挖掘出用户操作中对本体修改的隐含需求,并根据这些用户的隐含需求对本体进行进化。

2 基本概念

2.1 本体模型

定义 1 一个领域本体就是一个具有如下形式的有向图:$G=(C,I,R)$, 其中 C 表示领域中的概念,I 表示领域中的实例,R 表示概念或实例之间的关联,R 通过三元组$(c1,c2,r)$或$(i1,i2,r)$来表示,其中,$c1\in C,c2\in C$,r 表示 $c1$ 和 $c2$ 两个概念之间的关系名称, 或 $i1\in I,i2\in I$,r 表示 $i1$ 和 $i2$ 两个实例之间的关系名称。

2.2 用户日志结构

日志文件的具体格式如表 1 所示。

表 1 用户操作日志

用户名	操作类型	操作内容(含有多个结果)	操作时间
WQT	query	Select ?x where ?x rdf:type :ibm	2007-03-14 10:13:04
JD	browse	?x=IBM_ThinkPad_T40p501750	2007-03-14 10:13:28
WZY	query	select ?x where ?x hasprice"$800"	2007-03-14 10:13:56

基金项目:湖南省自然科学基金(the Natural Science Foundation of Hunan Province of China under Grant No.06JJ50142)。
作者简介:郭臻(1976-),男,硕士,研究方向语义网;徐德智(1963-),男,博士,硕士生导师,研究方向为语义网;汪智勇,男,硕士,研究方向语义网。
收稿日期:2007-06-04 **修回日期:**2007-08-02

第 1 部分包含有关访问服务器的用户信息,从第一条记录可以知道,执行用户操作的用户名为 WQT。

第 2 部分为用户操作的类型,query 表示查询,browse 表示浏览。

第 3 部分为用户操作的内容,当用户查询时,该内容为用户所使用的 SparQL 或 RDQL 语句;用户浏览时,该内容为用户浏览的节点名。

第 4 部分指明了操作的日期和时间,这次访问操作在 2007 年 4 月 17 日上午 10 点 13 分 04 秒。

2.3　用户操作的描述

定义 2 操作节点是二元组 (ID,P),ID 唯一标志了一个操作节点,P 是属性的集合,即 P={属性 P_j,j=1,2,…}。

操作节点代表了本体信息系统中终端用户与系统典型的交互操作,用{〈操作节点 ID〉,〈属性名〉}表示(以下 ID 用 N 或 $A,B,C,\cdots$表示)。操作节点分为两类:一类是查询节点(简写为 Q);另一类是浏览节点(简写为 B)。属性集合 P 描述了该节点各种属性。主要属性有以下几种(括号内表示属性名):

(1)用户名(username),表示该操作节点的操作者;

(2)操作类型(op),$op\in\{0,1\}$,op=0 表示该次操作是查询,op=1 表示该次操作是浏览;

(3)后继链接(next),指向下一操作节点;

(4)操作内容(content),当 op=0,content 为查询 SparQL 语句,当 op=1,content 为浏览信息;

(5)操作的时间戳(time),代表用户操作的时间。

例如,对于查询节点 $Q1$,其所使用的 SparQL 语句见图 1。

```
select ?x ? y where{?x rdf:type :laptop,?y rdf:subclass ?x,
?y hasprice "$800"}
```

图 1　查询语句

对于图 1 中的查询语句,将该查询语句分解为多个查询三元组,并由这些查询三元组构成查询图,如图 2 所示。

SparQL 的查询过程就是对该查询语句匹配图与源本体文件的图匹配过程。对于浏览节点 $B1$,它访问了领域本体中的若干实例,如图 3 所示。

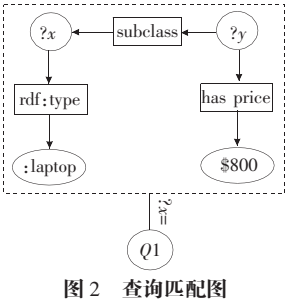

图 2　查询匹配图

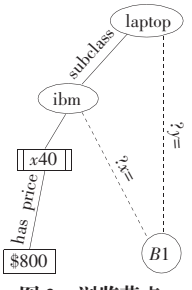

图 3　浏览节点

领域本体的拓扑结构是一个有向图,用户的操作行为就蕴涵在本体的拓扑结构中,即为它的一个子图。

在本体信息系统中,对于一个特定的领域本体来说,其拓扑结构是已知的。虽然不同的用户在不同的时期可能会有不同的操作模式,但其长期趋势应该是稳定的。因此,通过分析一定时期内用户的操作信息便可以发现该领域本体的相关内容和频繁操作路径。

主题是指同一个用户连续进行基于同一目的的一组操作。用户一个序列操作包含若干主题,每一主题由若干节点组成。操作节点之间的内容相关性随着距离的增大而减弱,主题也随着操作节点的迁移而发生变化,如图 4 所示。

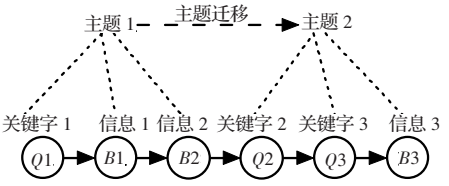

图 4　用户操作中的主题迁移

用户序列操作具有以下性质:

性质 1 终端用户的序列操作由多个主题组成,每一主题由一个或多个操作节点组成。

性质 2 同一主题的所有操作节点必然是相邻的,例如在图 4 中,$Q1$ 与 $B1$、$Q2$ 与 $Q3$ 是属于同一主题。

性质 3 相邻的操作节点未必是属于同一主题的,例如在图 4 中,$B2$ 与 $Q2$ 分别属于不同的主题。

考虑以某个操作节点为核心,则该操作节点的相邻节点可能与该节点是同一主题。

为了判断相邻的两个操作节点是否是同一主题,构造一个评价函数 $F(x)$,用于判断相邻的操作节点是否是同一主题。

评价函数 $F(x)$是从相邻节点到非负实数的映射,其取值为相邻两节点的相关度。为了构造评价函数 $F(x)$,从用户日志中可以得到如下线索:

(1)用户操作的次数;

(2)用户操作的时间;

(3)用户操作的内容。

这些线索是确定相邻操作节点是否是同一主题的关键,将在 4.3 节中详细介绍评价函数 $F(x)$。

2.4　语义相似与语义相关

在终端用户操作时,属于同一主题的所有操作节点具有紧密的语义关系。为了更好地了解终端用户操作的隐含需求,对这种语义关系进行了分析。

终端用户操作时,操作节点的语义关系分为两种,语义相似和语义相关。例如,“车”与“汽车”是语义相似的,“汽车”与“有引擎”是语义相关的。

终端用户在操作时,相继使用“车”和“汽车”来查询,或者使用“汽车”与“有引擎”进行查询。同样的情形也在浏览时出现,即用户浏览“汽车”后,继续浏览“引擎”。

终端用户操作节点的语义关系反映了操作节点对应的领域本体中的相关实体的相互关系,操作节点之间语义相似是因为这些操作节点所对应的领域本体中的若干个相关实体之间有层次关系;而操作节点之间语义相关是因为这些操作节点所对应的领域本体中的若干个相关实体之间有对象属性关系。

3　调整领域本体

3.1　简单调整

将对本体简单调整大致分为下述 6 种操作:

AddConcept(fatherconcept,concept);

DelConcept(concept);

AddProperty(property);

DelProperty(property);

AddIndividual(Concept,individual);

DelIndividual(individual);

对领域本体的调整基于以下原则:

(1)很少使用的概念将被删除;

(2)很少使用的属性将被删除;

(3)很少使用的实例将被删除;

(4)经常使用的新概念将被插入;

(5)经常使用的新属性将被插入;

(6)概念、属性和实例的关系根据用户使用情况进行调整。

当出现新概念、属性和实例时,其插入位置为上下文信息的最大值。即与该新概念、新属性和新实例共同出现次数最多的领域本体中概念、属性和实例的位置作为新概念插入位置。

3.2　复合调整

复合调整为简单调整的组合。例如:将某实例 i 由概念 A 移到概念 B 下,对应的操作为:

DelIndividual(i);

AddIndividual(B,i);

4　本体进化系统

提出一种本体进化方法支持本体管理者根据终端用户的需求处理和优化本体。其中一项关键任务是检查本体如何履行终端用户的需要。因为应用是在本体之上被处理的,得到详细的对本体和基于本体的应用终端用户视图。通过跟踪终端用户和应用程序交互操作,收集用于估计什么是终端用户主要兴趣的有用信息。所有用户交互操作在用户日志中存放。关注用户的两类活动:查询和浏览。

本体进化功能结构图如图 5 所示。

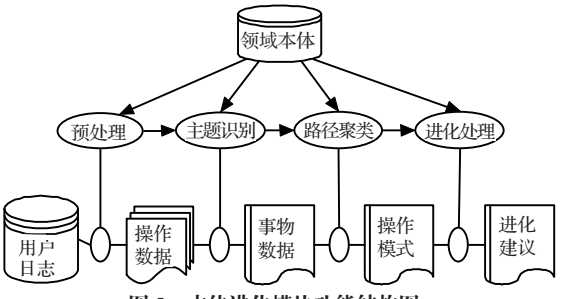

图 5　本体进化模块功能结构图

本体进化系统有 4 大功能模块组成,分别是预处理、主题识别、路径聚类和进化处理。核心功能模块为进化处理。

4.1　预处理

原始的用户操作日志不适合于直接提供给本体进化使用,所以进化前必须对用户操作日志进行预处理。

无用的记录不仅会降低系统的运行速度,而且影响挖掘信息的正确率。

在预处理中处理 3 类数据:

(1)使用机器爬虫进行查询的记录。使用机器爬虫查询的纪录在短时间内产生大量的日志记录;

(2)露宿者数据。露宿者是偶尔访问本体且逗留时间极短的用户,这些数据对挖掘信息没有什么作用,也应删除;

(3)查询语句为空的情况。这属于明显的错误操作,查询不能返回任何结果。用户输入的查询语句可能含有些拼写错误,在大量查询的情况下,所占的比例很小,基本上属于小概率事件;

(4)查询语句中,对于那些仅能影响查询显示结果的三元组,及与查询参数没有直接关系的三元组,因为它们对查询过程只有间接的影响,所以在预处理阶段将它们过滤掉。

4.2　用户识别

用户识别即识别出每一个用户,并将日志文件按不同的用户划分成若干个相互独立的用户访问集合。通过对日志文件进行分割,可以得到每一个用户的访问记录集。

识别用户最简单有效的方法是使用用户注册信息。采用以下两种方法来识别用户:(1)不同的 ID 属于不同的用户;(2)一旦发现用户客户端软件或操作系统发生改变,则它为新用户。

4.3　主题识别

不同用户进行的操作属于不同的主题,如果同一用户进行的操作跨越时间较长,一般认为用户的操作不止一次。最简单的方法是使用时间戳 time, 如果用户操作之间时间差超过了 time,则认为用户开始了一个新的主题。

在这里,将 2.3 节提到的 $F(x)$ 具体化为以下的形式:

$$F(x)=\max_{\substack{i=1,\cdots,m\\ j=1,\cdots,n}}\{NR(Ni,Nj)\}\wedge\min_{\substack{i=1,\cdots,m\\ j=1,\cdots,n}}\{T(Ni)-T(Nj)\}$$

该公式表示,在终端用户操作中,共同出现次数最多的操作节点,同时操作之间时间差最小的节点是属于同一主题。

4.4　关联规则与序列挖掘

数据库中发现序列模式的问题由 Agrawal 等人最先提出。一个序列的支持计数 $count(S)$ 记为所有包含某序列的数据序列的总数。一个序列的支持度 $supp(s)$ 记为所有包含某序列的数据序列的总数与 DB 中的数据序列总数之比。也称 $count(s)$ 为 s 的绝对支持度,$supp(S)$ 为 s 的相对支持度。最小支持度 $minSupp$ 是一个阈值,一般由用户指定,满足 $supp(s)>minSupp$ 称之为频繁序列或序列模式。

用户操作日志中有 m 个用户主题 $T=\{t1,t2,\cdots,tm\}$。对同一主题中的查询三元组进行关联规则挖掘, 可以使用 Apriori[2] 算法, 此算法首先产生满足支持度约束的频繁项目集 $T=\{T1,T2,\cdots,Tk\}$,其中 Ti 为查询三元组。

定义 3　对于每一个用户查询模式,其支持度(sup)和可信度($conf$)定义如下:

Ti 的支持度为

$$sup(Ti)=\frac{\sum_{i=1}^{k}P(Ti)}{|T|}$$

Ti 的可信度为

$conf(Ti)=50\%$

然后由 T 产生关联规则 $Xsup,confY$, 其中,sup 为 $X\cup Y$ 的支持度,$conf=sup(X\cup Y)/sup(X)$ 为规则的信任度。

算法 1　挖掘关联规则算法

输入:用户日志中的频繁三元组;最小支持度 min_sup

输出:频繁三元组集

开始:

生成 1 项频繁集 L_1;

For($k=2;L_{k-1}!=$null;k++)

对 L_{k-1} 做连接 C_k;
If C_k 的子集不是频繁集 then
删除 C_k;
Else
将 C_k 加入 L_k 中;
End If
End For
结束

定义 4 用户操作路径 $p=\{N1,N2,\cdots,Ni,\cdots,Nn\}$的频度 fp 定义为该路径的出现次数与总路径次数的比值。即

$$fp=\frac{\text{该路径的出现次数}}{\text{总路径次数}}$$

频繁三元组与频繁访问路径之间的区别在于:在频繁操作路径中,操作节点必须形成一个连续的序列,而频繁三元组是一个主题中项的集合,没有顺序关系。

算法 2 挖掘频繁路径算法
输入:用户日志中;最小支持度 *min_sup*
输出:频繁操作路径集
开始:
生成长度为 1 的路径 P_1;
For(k=2;L_{k-1}! =null;k++)
将 P_{k-1} 插入一后继节点使之成为 P_k;
If P_k 小于最小支持度 *min_sup* then
删除 P_k;
Else
将 P_k 加入 L_k 中;
End If
End For
结束

构造频繁路径的算法主要是基于候选路径的选择,首先找出长度为 k 的候选路径$\{xj,\cdots,xj+k-1\}$,然后计算它在用户日志中的支持度。支持度最大的 M 个路径组成的集合就是频繁操作路径,首先处理路径长度 p–*len*=1 的情况。然后从 p–*len*=2 直到 p–*len*=k 循环调用算法,每一次循环都可以利用上一次循环结果中的支持度。

频繁项集和频繁路径是一种非常可贵有用的上下文信息。真实世界的变化,用户要求上的变化,领域本体最初设计的缺点等,所有这类的信息都可以通过对上下文信息地分析中反映出来,它反映了详细的终端用户视图。

4.5 节点进化

终端用户在本体之上进行各种操作,用户的操作行为一定程度上表达了用户对本体的需求。基于这些频繁项集和频繁路径的有效信息,本体工程师可以准确了解和掌握本体中概念和实例的使用情况,从而可以有效地对本体进行调整,以使本体能更好地适应终端用户的需要。

4.5.1 概念调整

领域本体中有 n 个概念, 则每个概念的平均查询概率应为:1/n。

设定阈值 a 和 b,插入临界值 *insertnumber*=a* 平均查询概率;删除临界值 *delnumber*=b* 平均查询概率。

对于概念 $C1$,其被查询的概率为 $p(C1)$,若 $p(C1)$小于删除临界值,则说明该概念很少被用户所使用,则应将该概念删除;若 $p(C1)$大于插入临界值,则说明概念 $C1$ 经常被用户所使用若本体中没有 $C1$ 这个概念,则应该把 $C1$ 添加到本体中,在不能确定插入位置的情况下,将其插到根节点下。

算法 3 概念进化算法
输入:用户日志中全部概念被查询的概率,即 $P=(P(C1),P(C2),\cdots,P(Ck))$
输出:概念进化建议
开始:
For 所有概念
If $P(Ci)$<删除临界值 then
删除概念 Ci
Else
If $P(Ci)$>插入临界值 and Ci 不在领域本体中 then
将 Ci 插入领域本体中
End If
End For

4.5.2 属性调整

若领域本体中某属性被查询的概率小于删除临界值,则将其删除;反之,若其被查询的概率大于插入临界值,则将其加入到领域本体中。

4.5.3 实例调整

若领域本体中某实例被浏览的概率小于删除临界值,则将其删除;反之,若其被浏览的概率大于插入临界值,则将其加入到领域本体中。

对本体的调整和优化本身就是一个矛盾,保存大量很少使用的概念、实例和关系可以提高本体的查全率,但是增加了本体文件的大小从而降低了本体的效率。

同样,将新概念、实例和关系加入本体中,虽然能够提高本体查全率,但同时降低了本体的效率。

本文通过设定最小支持度,在这两者之间寻求一种平衡。

4.6 基于频繁项集和序列路径的本体结构调整

对于挖掘到的关联规则和序列,被认为是最终用户对领域和本体的隐含的修改需求,将其转化为概念子图,并将其与领域本体的相应实体进行比较,并根据该概念子图调整领域本体的相应节点。

算法 4 基于关联规则的领域本体调整算法
输入:挖掘到的所有关联规则和序列
输出:概念进化建议
开始:
For 每一关联规则和序列
将它转化为概念子图
If 领域本体中的相应节点与该概念子图不一致 then 根据该概念子图调整领域本体结构
End If
End For
结束

5 实验及分析

5.1 实验设计

运行环境:Jbuilder 2006,XML 解析器为 Xerces 2.9,本体解析器为 Protégé 3.2,操作系统:Windows 2003 SP2。所有的实验运行在 1GRAM 的 AMD X64 3600 个人计算机上。测试本体

为作者从 Stanford 大学 DAML+OIL 组下载了计算机本体(http://www.daml.org/2001/03/daml+oil/computer.owl),该本体共有 10 个概念,54 个实例。组织 16 个用户在 SNAX 环境中对该本体进行操作。共获得 1 290 次用户操作,其中(455 次查询,835 次浏览)。

5.2 结果及分析

对照 KAON 系统中所使用的本体进化方法,绘出本体进化准确率图,如图 6 所示。图中纵坐标表示进化建议的准确率,横坐标表示不同的本体进化方法。

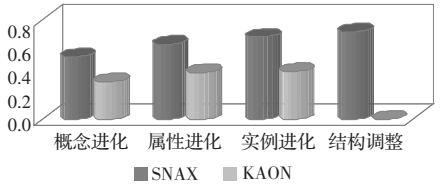

图 6　进化建议准确率

由图 6 可知,左边 SNAX 为本文所使用的本体进化方法的进化准确性,右边为 KAON 系统中本体进化方法的进化准确性。采用本文的本体进化算法,在概念进化方面要比 KAON 系统高出 20%,在属性进化方面要比 KAON 系统高出 23%,在实例进化方面要比 KAON 系统高出 28%,并且本文所提供本体结构调整方法是 KAON 系统所不具备的。

从上述对实验结果的分析可以得出,所使用的本体进化方法在进化准确性上大大优于 KAON 系统中所使用的本体进化方法。

6 相关工作

在 Prompt 项目中,Natalya F.Noy 开发了一组支持集成合作编辑状态的 Protégé 变化管理插件用于跟踪变化。在集成合作编辑环境,Natalya F.Noy 研究了不同用户版本的变化及其导致的本体进化[3]。

A.Maedche 提出一种分布式本体映射框架 MAFRA[4],MAFRA 为本体映射提供一个通用方法和概念性框架。在 MAFRA 中注册现有本体后,A.Maedche 通过现有本体与 WordNet 的比较,研究了本体重用和进化问题[5]。而 Ljiljana Stojanovic 首先研究知识管理系统中基于本体的元数据进化问题[6],然后分析了本体中用户操作对本体进化的影响[7],提出了变化获取、变化表示、语义变化、变化传播、变化实施和变化检验 6 阶段的本体进化框架[8]。Peter Haase 基于 OWL 描述的本体,提出一个变化语义模型,研究本体的结构,逻辑和用户定义的一致性问题[9],并提出一个处理本体变化后不一致的框架结构,该结构有效地处理了本体变化后的不一致性[10],Peter Haase 在用户对本体用法的基础上研究本体个性化进化的问题[11]。

7 总结

本文基于用户操作日志,提出了一套本体进化算法,该算法通过挖掘用户对本体浏览和查询操作的频繁集来调整和改进本体。对本体管理员提出对本体的连续修改的建议,目标是发现在本体里"最弱的地方",即不适应终端用户的需要的那些部分。

作者通过实验验证了该算法是可行和有效的,能够为本体进化相关研究提供一定的参考,下一步的工作将是进一步完善本体进化的理论框架,同时将其实际应用到本体库管理系统中。

参考文献:

[1] Berners-Lee T,Hendler J,Lassila O.The semantic Web[J].Scientific American,2001,284(5):34-43.

[2] Agrawal R,Imielinski T,Swami A.Mining associations between sets of items in large databases[C]//International Conference on Management of Data,1993:207-216.

[3] Noy N F,Chugh A,Liu W,et al.A framework for ontology evolution in collaborative environments[C]//International Semantic Web Conference,2006:544-558.

[4] Maedche A,Motik B,Stojanovic L,et al.An infrastructure for searching,reusing and evolving distributed ontologies[C]//Proceedings of the 12th International World Wide Web Conference(WWW'03),ACM,2003:439-448.

[5] Maedche A,Motik B,Stojanovic L.Managing multiple and distributed ontologies in the semantic Web[J].VLDB J,2003,12(4):286-302.

[6] Stojanovic L.Methods and tools for ontology evolution[D].University of Karlsruhe,2004.

[7] Stojanovic L,Maedche A,Motik B,et al.User-drivenontology evolution management[C]//LNCS 2473:Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management:Ontologies and the Semantic Web(EKAW'02).[S.l.]:Springer,2002:285-300.

[8] Stojanovic L,Maedche A,Stojanovic N,et al.Ontology evolution as reconfiguration-design problem solving[C]//Proceedings of the 2nd International Conference on Knowledge Capture(K-CAP'03),ACM,2003:162-171.

[9] Haase P,Stojanovic L.Consistent evolution of OWL ontologies[C]//LNCS 3532:Proceedings of the 2nd European Semantic Web Conference(ESWC'05).[S.l.]:Springer,2005:182-197.

[10] Haase P,van Harmelen F,Huang Z,et al.A framework for handling inconsistency in changing ontologies[C]//LNCS 3729:Proceedings of the 4th International Semantic Web Conference(ISWC'05).[S.l.]:Springer,2005:353-367.

[11] Haase P,Hotho A,Schmidt-Thieme L,et al.Collaborative and usage-driven evolution of personal ontologies[C]//LNCS 3532:Proceedings of the 2nd European Semantic Web Conference(ESWC'05).[S.l.]:Springer,2005:486-499.