第35卷 第19期 计 算 机 工 程 2009年10月
Vol.35 No.19 Computer Engineering October 2009
·博士论文· 文章编号：1000—3428(2009)19—0017—03 文献标识码：A 中图分类号：TP301.6

RDF(S)三元组的推理控制算法

王进鹏，张亚非，苗 壮
(解放军理工大学指挥自动化学院，南京 210007)

摘 要：为实现资源描述框架(RDF)数据的访问控制，提出一种RDF(S)三元组的推理控制算法，通过计算推理依赖图得到三元组的逻辑表达式，对敏感三元组的逻辑表达式求析取范式进而得到推理控制问题的候选解和最优解。实验表明，该算法能够有效地阻止非法推理，合理控制语义信息的丢失。

关键词：资源描述框架；访问控制；推理控制

RDF(S) Triple-level Inference Control Algorithm

WANG Jin-peng, ZHANG Ya-fei, MIAO Zhuang
(Institute of Command Automation, PLA University of Science and Technology, Nanjing 210007)

【Abstract】In order to implement access control for Resource Description Framework(RDF) data, a RDF(S) triple-level inference control algorithm is proposed. The inference dependence graph is defined to get the logic expressions of sensitive RDF(S) triples. The candidate and optimal answers of the inference control problem are obtained by translating the logic expressions into the disjunctive normal form. Experiments show that this algorithm can prevent illegal inference and minimize the loss of semantic effectively.

【Key words】 Resource Description Framework(RDF); access control; inference control

随着语义网技术的广泛应用，国内外学者对资源描述框架(Resource Description Framework, RDF)数据的安全问题给予了越来越多的关注。强制访问控制虽然能够控制对敏感信息的直接访问，但用户可以根据RDF(S)推理规则，从被授权的数据出发推得未被授权的数据，从而以间接方式访问敏感信息。因此，需要研究针对RDF(S)数据源的推理控制算法。文献[1]提出了安全数据库上基于视图的推理控制方法。文献[2]提出了基于逻辑的XML查询推理检测方法，这种方法可以检测许多公共的推理通道。但上面提及的方法都无法在RDF(S)的数据源上得到有效的应用，本文提出一种RDF(S)三元组的推理控制算法，是文献[3]工作的进一步扩展。

1 RDF(S)推理依赖图的定义

RDF(S)三元组由主体、谓词和客体3个部分组成，表示主体和客体之间存在谓词指示的关系。RDF(S)数据源可以看作一个三元组的集合，根据RDF(S)推理规则，可以在已有数据源上推出新的三元组。表1列出5条RDF(S)推理规则。其中，a代表rdf: type；r代表rdfs: Resource；sp代表rdfs: subPropertyOf；sc代表rdfs: subClassOf。

表1 RDF(S)推理规则

规则	前提	结论
规则4	<s p o>	<s a r>, <o a r>
规则5	<p1 sp p2>, <p2 sp p3>	<p1 sp p3>
规则7	< p1 sp p2>, <s p1 o>	<s p2 o>
规则9	< c1 sc c2>, <s a c1>	< s a c2>
规则11	<c1 sc c2>, < c2 sc c3>	<c1 sc c3>

以tap-cmu.rdf和tap-cmu-schema.rdf 2个数据源(http://tap.stanford.edu)为例，关于Wilson Harvey这个人，有如下这样一段描述：

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    …
    <tap:CMU_RAD rdf:ID="http://tap.stanford.edu/data/CMUSpecialFacultyHarvey,_Wilson">
      <tap:homePage>
        http://people.cs.cmu.edu/person/7018.html
      </tap:homePage>
      <rdfs:label xml:lang="en">Harvey, Wilson</rdfs:label>
      <tap:hasEmailAddress>wah@cs.cmu.edu</tap:hasEmailAddress>
      <tap:worksFor rdf:resource="http://tap.stanford.edu/data/CMU_ComputerScienceDepartment"/>
      <tap:hasPosition>ProjectScientist</tap:hasPosition>
    </tap:CMU_RAD>
  …
```

从这段描述中可以获取Wilson Harvey的个人主页、电子邮箱、工作单位和职务等信息，同时还知道他是CMU_RAD的实例。根据表1中的规则9，可以推出Wilson Harvey也是CMUPerson, Employee和Person的实例。

定义1 RDF(S)数据源表示的三元组集合记为S，根据推理规则，所有新推出三元组集合与S的并叫作RDF(S)的闭包，记为S^*。

定义2 推理依赖图是一个有向标记图，图中的每个节点代表一个三元组，有向边代表三元组之间的推出关系，从条件指向结果。每一条有向边都对应于一个条件，单个条件用

基金项目：国家“863”计划基金资助项目(2007AA01Z126)
作者简介：王进鹏(1982－)，男，博士研究生，主研方向：数据集成；张亚非，教授、博士生导师；苗 壮，讲师、博士
收稿日期：2009-06-10 **E-mail：**wangjinpeng1982@gmail.com

该条件自身标记；2 个条件之一的有向边用另一个条件来标记。

为使表达过程简洁，将三元组编号，如表 2 所示，表中最后一列指出该三元组是否出现在原来的 RDF(S)数据源中。依然以前面的数据源为例，其推理依赖图如图 1 所示，描述了三元组间的推理关系，虚节点表示该三元组由推理得到。

表 2　三元组及其标号的对应关系

三元组			编号	$\in S$
主体	谓词	客体		
Harvey,_Wilson	hasmailAddress	wah@cs.cmu.edu	1	是
Harvey,_Wilson	hasPosition	ProjectScientist	2	是
Harvey,_Wilson	homePage	http://people.cs.cmu.edu/person/7018.html	3	是
Harvey,_Wilson	worksFor	CMU_ComputerScience Department	4	是
Harvey,_Wilson	type	CMU_RAD	5	是
Harvey,_Wilson	label	Harvey, Wilson	6	是
Harvey,_Wilson	type	CMUPerson	7	否
Harvey,_Wilson	type	Person	8	否
Harvey,_Wilson	type	Employee	9	否
CMU_RAD	subClassOf	CMUPerson	10	是
CMU_RAD	subClassOf	Person	11	是
CMU_RAD	subClassOf	Employee	12	否
CMUPerson	subClassOf	Person	13	是
CMUPerson	subClassOf	Employee	14	是
Harvey,_Wilson	type	rdfs:Resource	15	否

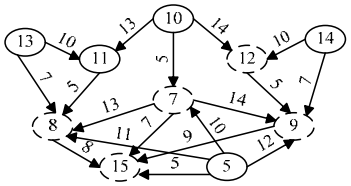

图 1　RDF(S)推理依赖

2　RDF(S)三元组的推理控制算法

本文提出的算法通过删除 S 中的部分三元组来实现推理控制。已知一个 RDF(S)数据源，其对应的三元组集合为 S，在 S 上使用推理规则得到的闭包为 S^*。根据访问权限的不同，不同用户对应不同的禁止访问的三元组集合，称这种集合为敏感三元组集合，记为 $S_n=\{s_1,s_2,\cdots,s_k\}\subset S^*$。推理控制的目标是：针对不同用户，考察三元组之间的推理关系，确定 S 的一个子集 S'，满足$(S')^*\cap S_n=\phi$，从而保证用户不能通过推理以间接的方式访问 S_n 中的三元组。

定义 3 三元组 s 的直接条件 R 是一个三元组$<s,r_1,r_2>$。其中，r_1 和 r_2 满足下列条件之一：

(1)三元组 s 可以由三元组 r_1 和 r_2 推出；

(2)三元组 s 可以由 r_1 单独推出，令 $r_2=\phi$；

(3)三元组 s 出现在原来的 RDF(S)数据源中，令 $r_1=r_2=\phi$。

三元组 s 的所有直接条件的集合称为 s 的直接条件集，记为 $R(s)$。

定义 4 三元组 s 的推出条件 C 是一个同时满足下面 3 个条件的三元组的集合：

(1)$C\subseteq S$；

(2)对 C 应用推理规则可以推出 s；

(3)对 C 的任何一个真子集应用推理规则都不能推出 s。

三元组 s 的所有推出条件的集合称为 s 的推出条件集，记为 $C(s)$。

显然，三元组的每个推出条件对应于一个推理通道。对于敏感三元组 s，如果可以求出它的推出条件集 $C(s)$，那么只要保证 $C(s)$中的每一个推出条件 C 都不成立，s 就不可能由其他三元组推出。为了求出敏感三元组 s 的推出条件集，需要定义三元组逻辑表达式，将求三元组推出条件集的问题转化为逻辑表达式的运算。

定义 5 设三元组 s 的直接条件集 $R(s)$为$\{R_1,R_2,\cdots,R_n\}$，则 s 的逻辑表达式定义为 $E(s)=E(R_1)\vee E(R_2)\vee\cdots\vee E(R_n)$。$E(s)$为真表示 s 可以由对应的直接条件推出。

定义 6 三元组 s 的直接条件 $R=<s,r_1,r_2>$的逻辑表达式定义为

$$E(R)=\begin{cases}E(r_1)\wedge E(r_2) & r_1\neq\phi,r_2\neq\phi\\ E(r_1) & r_1\neq\phi,r_2=\phi\\ s & r_1=r_2=\phi\end{cases}$$

三元组逻辑表达式的析取范式对应于该三元组的推出条件集，每一个析取项对应于该三元组的一个推出条件。因此，只要满足三元组逻辑表达式的每个析取项都为假，就可以保证该三元组不在 S 中而且也不能由其他 S 中的三元组推出。

多个敏感三元组的情况也可以类似处理，假设有敏感三元组集 $S_n=\{s_1,s_2,\cdots,s_k\}$，推理控制的目标是保证 S_n 中的每个三元组都不能被用户推出，即保证逻辑表达式

$$\overline{E(s_1)}\wedge\overline{E(s_2)}\wedge\cdots\wedge\overline{E(s_k)}=1$$

将该表达式化为形如 $E_1\vee E_2\vee\cdots\vee E_n$ 的析取范式，那么只要任何一个析取项 E_j 为真，就能实现推理控制的目标，因此，$E_1,E_2,\cdots,E_n$ 对应于推理控制算法的所有候选解。由于上面的三元组逻辑表达式的定义中没有引入三元组的非，只是在计算 $\overline{E(s_1)}\wedge\overline{E(s_2)}\wedge\cdots\wedge\overline{E(s_k)}$ 时引入了非运算，因此根据德摩根律，析取范式中的每个析取项 $E_j(j=1,2,\cdots,n)$都是三元组非的合取子句，形如 $\overline{t_1}\wedge\overline{t_2}\wedge\cdots\wedge\overline{t_m}$。析取项 E_j 为真，其含义是应当将三元组集$\{t_1,t_2,\cdots,t_m\}$从 S 中删除，所以$\{t_1,t_2,\cdots,t_m\}$是一个候选解。基于上述分析，给出 RDF(S)三元组的推理控制算法的描述。

输入 RDF(S)数据源 S，敏感三元组集合 S_n

输出 S 的一个子集 S'，满足$(S')^*\cap S_n=\phi$

算法步骤如下：

(1)应用前向链消解闭包算法[4]计算 S 的闭包 S^*，计算过程中考察并记录三元组之间的推出关系，求出 RDF(S)推理依赖图，供后面的计算使用。

(2)对 $S_n=\{s_1,s_2,\cdots,s_k\}$中的每一个三元组 s_i，根据(1)中得到的 RDF(S)推理依赖图中的直接条件，计算其逻辑表达式 $E(s_i)$。计算三元组和直接条件的逻辑表达式算法分别如下：

计算三元组 s 的逻辑表达式 $E(s)$：

输入 三元组 s

输出 该三元组的逻辑表达式 $E(s)$

{确定 s 的直接条件集 R(s);

for(s 的每一个直接条件 R_i) (i=1~n)

利用计算直接条件的逻辑表达式中的算法，计算 $E(R_i)$;

求出 $\vee_{i=1}^{n}E(R_i)$,作为结果返回;}

计算直接条件 R 的逻辑表达式 $E(R)$：

输入 某三元组 s 的一个直接条件 R

输出 该直接条件的逻辑表达式 $E(R)$

{若 R 的 2 个条件都是空：

返回 s;

若 R 的一个条件为空,另一个条件不空：

(设这个不空的条件是 r)

返回 E(r);

若 R 的 2 个条件都不空：

(设一个条件为 r_1,另一个为 r_2)

返回 $E(r_1)\wedge E(r_2)$;}

(3)根据(2)中计算得到的三元组 s_i 的逻辑表达式 $E(s_i)(i=1,2,\cdots,k)$，计算 $\overline{E(s_1)}\wedge\overline{E(s_2)}\wedge\cdots\wedge\overline{E(s_k)}$。

(4)将(3)中计算得到的 $\overline{E(s_1)}\wedge\overline{E(s_2)}\wedge\cdots\wedge\overline{E(s_k)}$ 化为形如 $E_1\vee E_2\vee\cdots\vee E_n$ 的析取范式。

(5)从 $E_1,E_2,\cdots,E_n$ 中选出一个最优的 E_i，将其中包含的三元组集从 S 中删除得到 S'。在这一步中，通过计算 S^*与$(S')^*$所包含三元组的数目差来衡量 E_i 的优劣，相差越少，说明 S^*中的信息丢失得越少。

(6)标记 S^*中的三元组，凡是在$(S')^*$中出现的三元组都标记为该用户可以访问。

3 实验分析

实验使用的 RDF(S)数据源为 tap-cmu.rdf 和 tap-cmu-schema.rdf，以 J2RE1.4.2 作为平台，使用 Jena 提供的 API (http://jena.sourceforge.net)处理 RDF(S)数据，数据库端使用 MySQL4.1，实验中用到的 RDF(S)三元组如表 2 所示。使用几组不同的 S_n，得到不同的候选解和最优解，实验结果如表 3 所示。

表 3 实验数据

S_n	候选解/(S')*与 S^*所包含三元组的数目差	最优解
{1,2,3,4,7}	{1,2,3,4,10}/304 {1,2,3,4,5}/9	{1,2,3,4,5}
{1,2,3,4,5}	{1,2,3,4,5}/9	{1,2,3,4,5}
{1,2,3,4,7,9}	{1,2,3,4,10}/304 {1,2,3,4,5}/9	{1,2,3,4,5}
{1,2,3,4,5,7,9}	{1,2,3,4,5}/9	{1,2,3,4,5}
{1,2,3,4,8}	{1,2,3,4,10,11}/454 {1,2,3,4,11,13}/492 {1,2,3,4,5}/9	{1,2,3,4,5}

4 结束语

本文深入研究了 RDF 数据的推理控制，提出了一种 RDF(S)三元组的推理控制算法。实验表明，该算法能够有效阻止非法推理，同时控制语义信息的丢失。在上述算法中，比较候选解的优劣只是简单地比较$(S')^*$与 S^*所包含三元组的数目差，没有考虑到三元组表示的语义信息往往具有不同的重要性这一事实。因此，根据三元组的语义分配权值，设计一种比较候选解优劣的加权算法是需要进一步研究的内容。另外，在实际应用中，推理控制的需求并非是静态的，需要经常更新。如何结合 RDF(S)推理依赖图设计一种 RDF(S)推理控制的增量更新算法，提高计算效率，也需要进一步的研究。

参考文献

[1] 严和平, 汪 卫, 施伯乐. 安全数据库的推理控制[J]. 软件学报, 2006, 17(4): 750-758.

[2] Xu Baowen, Jiang Jixiang, Lu Jianjiang, et al. Logic-based Inference Detection in Querying XML Document[C]//Proc. of the International Conference on Information and Knowledge Engineering. Nevada, USA: [s. n.], 2005: 105-112.

[3] Lu Jianjiang, Wang Jinpeng, Zhang Yafei, et al. An Inference Control Algorithm for RDF(S) Repository[C]//Proc. of the Pacific Asia Workshop on Intelligence and Security Informatics. Chengdu, China: [s. n.], 2007: 262-268.

[4] Broekstra J. Storage, Querying and Inferencing for Semantic Web Languages[D]. Amsterdam, Netherlands: Department of Mathematics and Computer Science, Vrije Universiteit, 2005.

编辑 顾逸斐

（上接第 16 页）

URL，数目与前者比值约为 82%，差额原因是部分 URL 没有通过重复性检测，可以看出，系统能对所有抽取出的 URL 请求进行分析，不会出现大量延迟。

6 结束语

主题网络爬虫可以解决通用爬虫的问题，是目前网络信息挖掘方面的热点。本文的 Broker 分布式爬虫系统采用基于分类标注的多主题策略实现了同一爬虫系统内多主题自适应扩展，并设计了二级加权传递的 URL 队列排序算法和加权最小连接调度算法，联合解决了分布式爬虫中基于负载情况导向性的任务分割问题。该系统增强了系统可扩展性，实现了基于 Trie 结构的 URL 存储策略，高效地支持了 URL 查询、插入和重复性检测。实验表明，Broker 分布式爬虫系统可以高效高质量地运行多主题抓取任务。

参考文献

[1] Rungsawang A, Angkawattanawit N. Learnable Topic-specific Web Crawler[J]. Journal of Network and Computer Applications, 2005, 28(2): 97-114.

[2] Chakrabhik S, Vandenburg M, Dom B. Focused Crawling：A New Approach to Topic-specific Web Resource Discovery[C]// Proceedings of the 8th International World-Wide Web Conference. Toronto, Canada: [s. n.], 1999.

[3] Liu Hongyu, MIuOS E, Janssen J. Probabilistic Models for Focused Web Crawling[C]//Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management. New York, USA: ACM Press, 2004.

[4] 刘金红, 陆余良. 主题网络爬虫研究综述[J]. 计算机应用研究, 2007, 24(10): 26-29.

[5] Florescu D, Levy A, Mendelzon A. Database Techniques for the World-Wide Web: A Survey[J]. SIGMOD Record, 1998, 27(3): 59-74.

[6] Wei Jiying, Wen Jirong. Instance-based Schema Matching for Web Databases by Domain-specific Query Probing[C]//Proceedings of the 30th International Conference on VLDB. Toronto, Canada: [s. n.], 2004.

[7] 叶允明, 于 水. 分布式 Web Crawler 的研究: 结构、算法和策略[J]. 电子学报, 2002, 30(Z1): 26-29.

[8] 钱 镕, 徐新华, 郑 莹, 等. 智能专题化信息搜索 Crawler[J]. 计算机工程, 2006, 32(3): 57-59.

编辑 张正兴