# 基于 XML Schema 的数据匹配方法的研究

王大刚,谢荣传,彭 俊
(安徽大学 计算智能与信号处理教育部重点实验室,安徽 合肥 230039)

**摘 要**:由于 XML Schema 自身所包含的丰富的结构和语义信息,针对基于 XML schema 的文档匹配,设计了一个框架结构,把利用结构信息的基于路径的匹配和利用语义信息的宽松标识方法结合在一起,充分利用了 XML schema 中所包含的各种信息,来实现 XML schema 文档之间的匹配。实验验证对文档的匹配具有较好的精度,最后对模式匹配这个领域的研究方向做出展望。

**关键词**:匹配;路径;宽松标识;模式匹配
**中图分类号**:TP311 **文献标识码**:A **文章编号**:1673-629X(2008)06-0028-04

# Research on Data Matching Based on XML Schema

WANG Da-gang, XIE Rong-chuan, PENG Jun
(Ministry of Education Key Lab. of Intelligence Computing and Signal Processing, Anhui University, Hefei 230039, China)

**Abstract**: Due to the XML schema itself contains abundant structure and semantic information, focuses on a design of a matching frame which is based on XML schema. It mixes the match based on path match by using structure information with relaxation labeling by using semantic information, so as to achieve the match among documents. Many experiments show that this algorithm will achieve effective mathing result. Puts forward the prospect for the matching field.

**Key words**: match; path; relaxation labeling; schema matching

## 0 引 言

XML 已经成为电子商务领域数据表示和数据交换的标准。然而在广域网的分布和自治的环境下,众多的 XML 文档可能采用不同的标签和结构来表示相同含义的内容。在早期这些工作主要依靠手工完成,然而在网络的动态环境下,随着 XML 文档的越来越多,结构越来越复杂,则需要一种自动的机制来辅助用户匹配不同的 XML 文档。文中就是针对这一问题展开的研究。

XML 数据源匹配问题一般的解决办法是先用工具将 XML 数据源文件转化成树结构,然后基于树结构利用各种算法对树中的节点进行相似性匹配,最终输出关于两棵树中所有节点的相似性矩阵[1]。现在虽然已经做了很多模式匹配方法的研究,文献中已存在大量的关于异构 XML 数据源的匹配算法,它们或者从待匹配数据源的结构方面得到相似性,或者从基于待匹配数据源的大量的实例基础上得到语义上相似性[2],但是都没有很好地解决存在的一些问题。为了有效地匹配基于 XML schema 的 XML 文档,笔者参与设计一个系统框架,利用该框架,可以充分挖掘 XML 文档的结构信息和语义信息,这样在充分利用信息的基础上,可以使最终的匹配达到较好的程度。

## 1 系统框架流程图

为了有效地使用 XML schema 中的结构和语义信息,设计以下的系统框架,该设计流程充分考虑到 XML 树结构的结构信息和语义信息[3,4],将两者统一在一起。通过基于路径的匹配算法充分考虑节点的上下文信息,利用节点所在的路径进行匹配,由于 XML schema 树结构中包含一定的语义信息,可以通过形式化来确定节点间各种语义所构成的权值。然后把基于路径的匹配算法作为初始相似度,利用宽松标识法,进入迭代过程。当满足一定的迭代次数,达到系统要求的精度,退出迭代过程,最后选出满足给定阈值的匹配对,框架流程图如图 1 所示。

**收稿日期**:2007-09-01
**基金项目**:国家自然科学基金资助项目(60472065)
**作者简介**:王大刚(1982-),男,安徽肥东人,硕士研究生,研究方向数据库与 web 技术;谢荣传,副教授,硕士生导师,研究方向数据库与 web 技术。

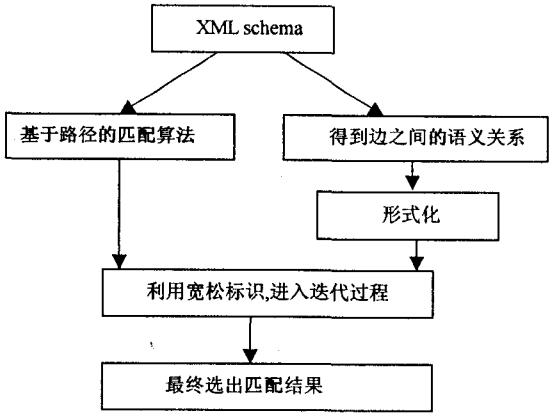


图 1　XML 匹配框架流程图

## 2　基于路径的 XML 匹配算法

XML schema 是一种描述 XML 文档的结构等信息的一种文档,同时含有丰富的语义。这里将 XML schema 文档树看成是由从根节点开始到叶节点的众多的一条条的路径组成,路径匹配算法是基于以下两个角度来思考问题:

(1)两条路径:author－name－firstName－name－firstName。这两条路径的相似性是根据两条路径所含的相似节点考虑的,因为它们有两个相似的节点 name 和 firstName。从而可以认为这两条路径具有一定的相似性。

(2)节点之间的相似性:把待匹配节点作为根节点,根节点下所包含的所有路径间的相似性作为节点之间相似性的影响条件,如路径 author－name－firstName 和 au－name－firstName 。在考察 author 和 au 两个节点的相似性时,因为它们路径间具有相似性(路径上有两个完全一样的节点,在这里假设它们的属性是相同的,实际上不一定)。所以从路径之间的相似性推导出 author 和 au 之间具有一定的相似性。这样就可以增加 author 和 au 这两个词是缩写词的可能性。为进一步描述算法需要定义如下几个概念:基本相似度、路径匹配、节点匹配。

首先给出节点之间的基本相似度的定义,由下面的等式给出其定义:

$\text{sim}_{\text{基本相似度}}(s,t) = w1 * \text{Lmatch}(s,t) + w2 * \text{Pmatch}(s,t)$ 由两部分构成:中文名称相似度(Lmatch)、属性相似度(Pmatch)。然后分别乘上一定的权值 $w1$,$w2$ 相加构成,中文名称相似度考察两个节点名称是否是相同、相似关系,是否是简写、缩写、领域内专用名字等名称之间的相互关系,对于每一种关系,给它对应一种权重,来标识它们之间的相似性,同时还可以利用文本相似性中向量之间的夹角余弦来判断向量之间的相似性大小。属性相似度考察 XML 中带有属性的节点间的相似性,属性一般起到对节点的说明和解释的作用,即使是名称完全一样的两个节点(标签),如果所具有的属性不同,那么将导致节点间是有区别的,就像数据类型的定义一样,两个数值一样的数据因为数据类型不一样,它们就表示不相同的数据。通常一个节点会包含多个属性(属性集),节点间属性匹配要对属性集之间所有属性同时进行匹配,属性可以由值和类型组成,如果属性包含类型,需要从三方面去考虑:

(1)类型是不是相等的;

(2)类型是否是可转化的(如 int 型和 double 型);

(3)属性的值是不是相等的。

接下来定义路径匹配和树匹配的概念。

路径匹配:两个路径的相似性从以下两个角度来衡量。即影响路径匹配有两个因素:

1)路径的长度;

2)路径上相似节点的个数。

树匹配:XML 文档被转化成树形结构进行匹配,两个节点下面如果有多个孩子节点,它们的相似性就用树匹配算法描述,即两个节点下所包含的所有路径的之间的相似性。它们的相似性也是从两个角度来衡量:

1)相似路径的个数(路径匹配的结果大于给定阈值的路径);

2)相似路径匹配的质量。

## 3　语义关系及利用宽松标识优化匹配

### 3.1　XML Schema 标签之间的语义关系

在 XML Schema 中,元素之间的语义关系类型可归纳为四种[5]:

generalization, association, aggregation, of－property,分别用符号{g,s,a,p}来标识。

对于两个元素之间的语义关系的定义有:

边(edge)[$e(v_1,v_2)$]:如果元素 $v_1$,$v_2$ 通过一条直接的连线相互连接,则 $v_1$ 是 $v_2$ 的父元素,$v_1$ = parent($v_2$),$v_2$ 是 $v_1$ 的子元素,$v_2$ = children($v_1$)。如果从 $v_i$ 到 $v_j$ 存在直接连线,则定义 $e(v_i,v_j)=1$,否则 $e(v_i,v_j)=0$。

边类型 $r(v_i,v_j)$:如果任意两个元素 $v_i$,$v_j$ 有边相连,构造函数的类型是 $k(k\{g,s,a,p\})$,则定义 $r(v_i,v_j)=k$;如果不存在,则 $r(v_i,v_j)$ = null。

连接两个元素的边代表的是元素之间的语义关系,它的标识则代表了语义关系的类型。

路径(path)[$e\sim(v_1,v_n)$]：如果有一系列元素 $v_1,v_2,\cdots,v_n$，其中任意相邻两个元素 $v_i,v_{i+1}(1<i<n-1)$ 之间都有直接连线，即 $e(v_i,v_{i+1})=1$，则定义 $e\sim(v_1,v_n)=1$，否则 $e\sim(v_1,v_n)=0$。路径 $e\sim(v_i,v_j)$ 的深度即是这条路径所拥有的边的数量。

路径类型 $r\sim(v_1,v_n)$：如果任意两个元素 $v_1,v_n$ 之间存在一条路径，假设这条路径的边为 $e_1,e_2,\cdots,e_{n-1}$，它们相对应的边类型为 $r_1,r_2,\cdots,r_{n-1}$。如果 $r_1=r_2=\cdots=r_{n-1}=k$，则定义路径类型为 $r\sim(v_1,v_n)=k$，如果不完全相同，则 $r\sim(v_1,v_n)=$ null。

类型相似度 $\mathrm{sim}(r_1,r_2)$：如果两个边类型 $r_1$ 和 $r_2$，或一个边类型 $r$ 和一个路径类型 $r\sim$，或两个路径类型 $r_1\sim$ 和 $r_2\sim$ 有相同的语义关系，则定义 $\mathrm{sim}(r_1,r_2)=1$，$\mathrm{sim}(r,r\sim)=1$ 或 $\mathrm{sim}(r_1\sim,r_2\sim)=1$；否则 $\mathrm{sim}(r_1,r_2)=0$，$\mathrm{sim}(r,r\sim)=0$ 或 $\mathrm{sim}(r_1\sim,r_2\sim)=0$。

在 XML Schema 的语义模型中，元素之间的约束，即语义关系是通过边 $e(v_1,v_2)$，边类型 $r(v_i,v_j)$，路径 $e\sim(v_1,v_n)$，路径类型 $r(v_1,v_n)$ 和类型相似度 $\mathrm{sim}(r_1,r_2)$ 来表示的。

### 3.2 利用宽松标识

对模式中的约束运用宽松标识来确定元素之间的一致性。

给定在模式 $C_1$ 中的元素 $v_i,v_j$，模式 $C_2$ 中的元素 $v_k,v_l$，一致性系数 $r_{ij}(k,l)$ 表示元素 $i$ 同元素 $k$，元素 $j$ 同元素 $l$ 在语义关系上的一致性程度，$(i,j)$ 属于模式 $C_1$，而 $(k,l)$ 属于模式 $C_2$，元素 $i$ 和元素 $k$ 的匹配程度可以通过 $p_i(k)$ 来衡量，定义下列函数来量化 $p_i(k)$ 和一致性系数之间的一致性程度。

$$q_i(k)=\sum_{j=1}^{m}\sum_{i=1}^{n}r_{ij}(k,l)p_j(l)$$

将经过路径匹配得到的相似性作为 $p_i^{(0)}(k)$ 代入下列函数进行迭代，即 $p_i^{(k+1)}(k)=f(p_i^{(k)}(k),q_i^{(k)}(k))$，用 $t$ 反复调整迭代次数，函数 $f$ 可表示成 $p_i^{(t+1)}(k)=\dfrac{p_i^{(t)}(k)q_i^{(t)}(k)}{\sum_{l=1}^{n}p_i^{(t)}(l)q_i^{(t)}(l)}$。把最终的 $p_i^{(k+1)}(k)$ 作为元素 $i$ 和元素 $k$ 的相似结果。

对于一致性系数 $r_{ij}(k,l)$ 的定义为：

1) 如果 $e(i,j)=1$ 且 $e(k,l)=1$ 且 $r(i,j)=r(k,l)$，则 $r_{ij}(k,l)=w1$；

2) 如果 $e(i,j)=1$ 且 $e(k,l)=1$ 且 $\mathrm{sim}(r(i,j),r(k,l))=1$，则 $r_{ij}(k,l)=w2$；

3) 如果 $e(i,j)=1$ 且 $e\sim(k,l)=1$ 且 $r(i,j)=r\sim(k,l)$ 或者 $e\sim(i,j)=1$ 且 $e(k,l)=1$ 且 $r\sim(i,j)=r(k,l)$ 或者 $e\sim(i,j)=1$ 且 $e\sim(k,l)=1$ 且 $r\sim(i,j)=r\sim(k,l)$，则 $r_{ij}(k,l)=w1/(d_{ij}+d_{kl})$；

4) 如果 $e(i,j)=1$ 且 $e\sim(k,l)=1$ 且 $\mathrm{sim}(r(i,j),r\sim(k,l))=1$ 或者 $e\sim(i,j)=1$ 且 $e(k,l)=1,\mathrm{sim}(r\sim(i,j),r(k,l))=1$ 或者 $e\sim(i,j)=1$ 且 $e\sim(k,l)=1$ 且 $\mathrm{sim}(r\sim(i,j),r\sim(k,l))=1$，则 $r_{ij}(k,l)=w2/(d_{ij}+d_{kl})$；

5) 其他情况下，$r_{ij}(k,l)=w3$。

实验中将其中权重 $w1,w2,w3$ 分别取值为：$w1=1.0,w2=0.8,w3=0.2$。

## 4 算法过程中部分主要形式化与算法描述

利用松散标签来加入语义优化匹配结果对上文提到的路径匹配、节点匹配的定义经过形式化，用公式描述如下：

$\mathrm{SimPath}(s,t)=w1*\mathrm{Lmatch}(s,t)+w2*\mathrm{Pmatch}(s,t)+w3*\{\mathrm{Match}(cs,ct)-w4*(\mathrm{pathLength1}-\mathrm{pathLength2})\}$；节点 $(cs,ct)$：$cs$ 取自 $s$ 节点下的直接孩子节点，$ct$ 是以 $t$ 作为根节点的下面的所有的孩子节点。

该公式的含义是考察树 $s,t$ 中路径 $s\rightarrow\ldots\rightarrow cs$ 和路径 $t\rightarrow\ldots\rightarrow ct$ 的相似性，最后节点的相似性可以用树匹配来衡量。

树匹配定义：

$$\mathrm{sim\ Tree}(s,t)=\frac{\sum\mathrm{simPath}(s,t)+n\mathrm{Matchs}}{2*|\mathrm{child}(s)|}$$

该公式反映以 $(s,t)$ 为根的两棵树，用它们的有效匹配路径的个数(路径匹配大于给定阈值，认为是有效匹配路径的个数)和有效路径的匹配的质量(有效路径匹配的具体的值)。

用这两个因素来反映两个节点 $(s,t)$ 之间的相似性。

算法描述如下：Match()函数计算节点相似性。

```
Match(tree s,tree t)//用路径匹配算法计算节点之间的相似性
 {s=post-order(s); t=post-order(t);//s中保存后序遍历从叶节点到s的所有节点
  For each s1∈s
    For each t1∈t
       Lmatch(s1,t1);//计算名称相似度
       Pmatch(s1,t1);//计算属性相似度
     If (isleaf(s1) and isleaf(t1))
     Sim=w1*Lmatch(s1,t1)+w2*Pmatch(s1,t1)+w3;
     If((isleaf(s1) and ! isleaf(t1)) or(! isleaf(s1) and isleaf(t1))
       Sim=w1* Lmatch(s1,t1)+w2*Pmatch(s1,t1)+0;
```

```
    Else//下面调用函数 childMatch()
    Sim= w1 * Lmatch(s1,t1) + w2 * Pmatch(s1,t1) + w3 *
childMatch(s1,t1)}
childMatch (tree s, tree t)//计算节点(s,t)下面所有孩子节点的
相似性,其中用到函数 Match(),两个函数间相互调用
{ nMatch=0;//匹配的路径数
  Max=0;
    For each dirct cs of s
      For each nodes ts of t
        Weight1 = Match(ts,cs);//递归调用
        If Weight > threshold
        Max= Weight:
      If Max>0
      pathLength1 = pathLength(s,cs);//求出从 s 到 cs 路径长
度
      pathLength2 = pathLength(t,ct);
      Weight = Weight1 - w4 * |pathLength1 - pathLength|;
      nMatch++;
      //递归调用结束,下面计算最后匹配结果
    w1 = Weight/|child(s)|;
    w2 = nMatch/|child(t)|;
    W = (w1 + w2)/2;
    Return W;//返回最后匹配结果
}
```

最后把基于路径匹配算法得到的结果 $W$ 作为输入代入以下算法:

Input $W$; $t = 0$;begin $p_i^{(0)}(k) = W$;

While($|p_i^{(t+1)}(k) - p_i^{(t)}(k)| > \sigma$)do//$\sigma$ 为给定的满足精度的阈值

Evaluate $qi(k) = \sum_{j=1}^{m}\sum_{l=1}^{n} r_{ij}(k,l)p_j(l)$;

$$p_i^{(t+1)}(k) = \frac{p_i^{(t)}(k)q_i^{(t)}(k)}{\sum_{l=1}^{n} p_i^{(t)}(l)q_i^{(t)}(l)};$$

$t = t + 1$;

end while

output $p_i^{(t)}(k)$;

end//算法结束,输出为节点 $i$ 和节点 $k$ 的最终的相似性。

## 5　实验验证与分析

利用实验来验证上述算法的有效性,由于 XML schema 也是一种 XML 格式文件,在 C#.NET 开发环境下将 XML schema 作为程序的数据源输入。利用.NET 提供的对 XML 文件的处理对象 XmlNodeReader 来读取树结构数据,然后使用基于路径的 XML 数据匹配算法对数据进行匹配。实验中选取关于课程的两个 XML 数据源:course1.xsd 和 course2.xsd,算法经过 4 次迭代($k = 4$)得到的部分标签的相似度的匹配结果如表 1 所示。

表 1　部分相似性匹配结果

| course1 | course2 | similarity |
|---|---|---|
| title | name | 0.91 |
| time | time | 0.92 |
| course | course | 0.92 |
| endtime | end | 0.80 |
| credits | crse | 0.48 |

从表 1 的匹配结果可以看到该算法对异名同义,同名同义现象都能得到较好的匹配,通过实验的结果证明了算法的有效性,当然文中的算法框架还可以融入其他元素,例如如果可以获得大量的可以用于训练的实例数据,可以对实例进行训练,得到它们基于统计层次的语义相似性。这样可以得到更好的匹配精度。

## 6　结束语

设计了一个框架,采用将基于路径的匹配算法与基于语义的松散标签法结合起来对 XML schema 进行匹配,实验验证能够对基于 XML schema 的 XML 文档进行有效的匹配。

当然文中提出的框架中还可加入诸如基于实例训练的匹配方法,这样将会使最终的匹配效果更加精确,这个领域还有很多问题需要解决,例如如何对 XML 标签的 $m:n$ 匹配(即多对多关系)的发现,这些都是需要进一步做的工作。

**参考文献:**


[1] Rahm E, Bernstein P A. A survey of approaches to automatic schema matching[J]. The VLDB Journal, 2001, 10: 334 - 350.

[2] 强保华,陈　凌,余建桥,等.基于 BP 神经网络的属性匹配方法研究[J].计算机科学,2005,33(1):249 - 259.

[3] Li Wen Syan, Clifton C. SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks[J]. DKE, 2000, 33(1): 49 - 84.

[4] 韩　恺,岳丽华,龚育昌.基于上下文的异构文档类型定义匹配[J].小型微型计算机系统,2005,26(2):256 - 260.

[5] Yi Shanzhen. XML application schema matching using similarity measure and relaxation labeling[J]. Information Sciences, 2005, 169: 27 - 46.