

Deep Web 集成服务的不确定模式匹配

姜芳芳 孟小峰 贾琳琳

(中国人民大学信息学院 北京 100872)

摘要: 随着 Deep Web 的迅猛发展,从高度自治、异构及动态变化的 Web 数据库中,为用户提供高质量的数据逐渐成为当前 Deep Web 集成服务的一个研究热点。在大部分 Web 数据库只能通过查询接口为用户提供服务的前提下,如何建立用户请求与集成查询接口模式之间以及集成查询接口模式与 Web 数据库查询接口模式之间的匹配关系,是 Deep Web 集成服务中进行合理的用户请求转换的关键。之前的相关工作都是寻找最佳的匹配结果,回避匹配的不确定性,丢弃了可能有价值的其他匹配结果。本文首先剖析了请求转换中模式匹配的不确定性,提出了数字类型的相似度计算方法,给出了进行数字类型的模式匹配的有效的剪枝方法以及数据类型驱动的模式匹配优化方法,并在此基础上提出了一种基于相似度计算的不确定性模式匹配方法,最后通过大量的实验证明了该方法的有效性。

关键词: Deep Web; 集成服务; 相似度; 模式匹配; 不确定性

中图法分类号: TP391 **文献标识码:** A

Uncertain Schema Matching in Deep Web Integration Service

JIANG Fang-Jiao MENG Xiao-Feng JIA Lin-Lin

(School of Information, Renmin University of China, Beijing, 100872)

Abstract: With increasing of Deep Web, providing high quality data from autonomous, heterogeneous and dynamic Web databases to users is becoming a hot topic in recent research of Deep Web integration service. How to generate the reasonable schema matching between the keywords of the user request and schema of integrated interface as well as between the schema of integrated interface and that of Web database interface is essential. The related works about schema matching are generating the best schema matching which slide over its uncertainty. In this paper, we analyze the uncertainty of schema matching, and then propose a series of similarity measures. To reduce the cost of execution, we propose the type-based optimization method and schema matching pruning method of numeric data. Based on above analysis, we propose the uncertain schema matching method. The experiments prove the effectiveness and efficiency of our method.

Keywords: Deep Web; integration service; similarity; schema matching; uncertainty

本课题得到国家自然科学基金项目(60573091)、国家 863 高技术项目(2007AA01Z155)、国家基础研究与发展“语义网格”项目(2003CB317000)和新世纪优秀人才支持计划的资助。姜芳芳,女,1971年生,博士研究生,主要研究方向为 Web 数据管理与集成。Email: jiangfj@gmail.com。孟小峰,男,1964年生,教授,博士生导师,主要研究方向为 Web 数据管理、XML 数据库、移动数据管理等。贾琳琳,女,1984年生,硕士研究生,主要研究方向为 Web 数据管理与集成。

1 引言

近年来, Deep Web^[1]的发展非常迅猛, 2004 年大约有 450, 000 个 Deep Web 数据源, 这些分布自治的资源集合的数据量是 Surface Web 的 500 倍以上, 而且目前仍呈指数级的增长趋势。为了使用户快速地获得高质量的数据, Deep Web 集成服务应运而生了。Deep Web 集成服务是将 Web 上通过查询接口提供 Web 服务的结构化数据源按领域形成统一服务的过程, 其基本框架如图 1 所示。对于用户而言, 集成服务是透明的, 即用户无须再面对数百万个 Web 服务, 不需要关心所需要的数据存储在哪里, 更不需要了解如何获取这些数据。用户只需在类似于传统的基于关键字搜索引擎的查询接口的用户层提交用户请求, 由集成服务的服务层与资源层交互, 自动完成 Web 服务的发现与分类、自动进行模式匹配以完成用户请求的转换、协同各 Web 服务完成查询的处理、自动完成对结果数据的合并, 最后将高质量的结果数据返回给用户。而且集成服务能够适应 Web 服务自治性和动态性强的特点, 具有自适应和动态演化的能力。

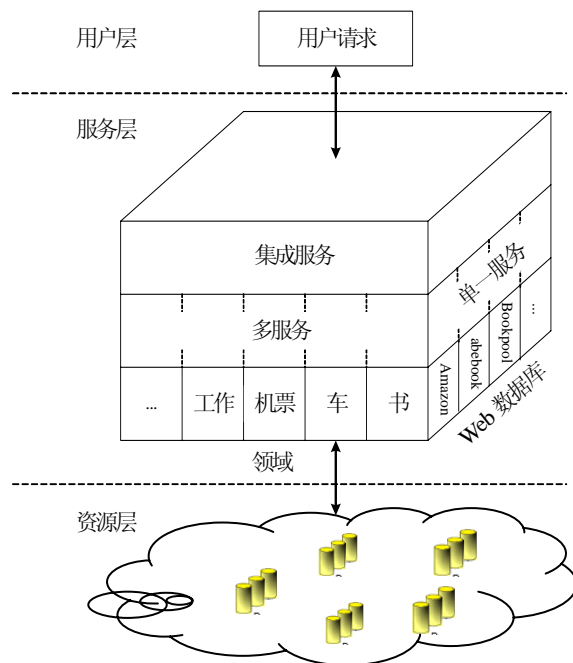


图 1 Deep Web 数据集成服务的基本框架

Deep Web 集成服务包含着很多不确定性因素, 例如: (1) 用户可能并不能明确知道要寻找什么数据; (2) 用户可能不知道应该怎样描述自己的查询; (3) 在 Deep Web 这样的大规模动态集成环境中, 必须采用自动的方法, 而这些方法在高度自治的 Web 环境中很难做出准确的判断。因此, Deep Web 集成服务的不确定性成为一个无法回避的问题。据我们所知, 目前关于不确定性的研究还只是处于初步阶段, 很多已有的研究用自动或半自动的方法在关键处选择最可能的结果, 从而抹杀了不确定性问题。例如: 在模式匹配中, 很多研究工作^{[2][3][4]}都是在寻找最佳的匹配结果。但是有时这些不确定信息是有价值的, 忽略这些信息可能使本来对于用户有用的信息却并未返回给用户。如何在不确定性存在的前提下, 返回高质量的数据, 提高用户的满意度, 是目前 Deep Web 集成服务的一个研究热点。

本文的关注点是用户请求转换的过程中模式匹配的不确定性, 即关注将用户提交的关键字查询转换到复杂的结构化集成查询接口上, 进一步提交到相关的 Web 数据库查询接口过程中的不确定模式匹配问题, 如图 2 所示。例如, 用户想买一辆二手车, 提交的关键字查询为{奔驰, 20000-30000}, 其隐含的语义存在不确定性, 当将此查询转换到结构化的集成查询接口时, 可能的匹配结果有两种, 一是已行驶里程在两万到三万公里之间的奔驰车,

二是价格在两万到三万美元之间的奔驰车。

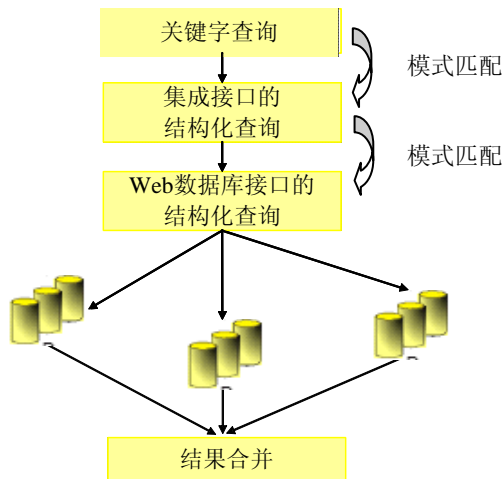


图2 用户请求转换中的模式匹配

不确定模式匹配的定义如下：源模式 S ，目标模式 T ， S 和 T 均为关系模式， S 到 T 的不确定模式匹配为一个三元组 (s, t, \mathbf{m}) ，其中 $s \in S$ ， $t \in T$ ，并且 \mathbf{m} 是 S 到 T 可能的模式匹配的集合 $\{m_i\}$ 。 m_i 满足传统的模式匹配的定义，即 $m: \forall x(\phi(x) \rightarrow \exists y \psi(x, y))$ ，其中 ϕ 是 S 上的合取查询， ψ 是 T 上的合取查询，对于 S 中的每个满足 ϕ 的实例 x ， T 存在满足 ψ 的实例 y 。

本文组织如下：第2节分析了Deep Web集成服务中用户请求转换的模式匹配的不确定性并介绍了相关工作；第3节提出了数字类型的相似度计算方法，进一步给出了模式匹配的优化方法；第4节在上述问题得到解决的基础上，提出了用户请求转换中的不确定性模式匹配方法；第5节给出了实验结果；第6节对全文做了总结。

2 不确定性模式匹配问题及相关背景介绍

从图2可知，用户请求的转换可以分为两步：首先将用户的关键字查询转换到复杂的集成查询接口，再进一步转换到各Web数据库查询接口上的过程。这两个步骤均依赖于正确的模式匹配。

2.1 用户查询转换的模式匹配问题

在用户请求转换的第一步中，需要建立用户填写关键字和集成的结构化查询接口中的相关概念之间的对应关系，其实质是利用缺失模式信息（属性名）的实例（关键字）信息和集成查询接口各概念的实例（候选值）信息，寻找关键字查询模式与集成查询接口模式之间的模式匹配关系。

进一步系统将集成接口的结构化查询转换成各Web数据库查询接口的查询，其关键是建立集成查询接口各概念与Web数据库查询接口各属性的模式对应关系，其实质是利用集成接口的模式（概念及相应的标签名集合）信息、实例（候选值）信息和Web数据库查询接口的模式（属性名）、实例（候选值）信息，寻找集成查询接口模式与Web数据库查询接口模式之间的匹配关系。

概括地说，第一步是利用实例信息进行模式匹配，第二步是利用模式信息及实例信息进行模式匹配。值得注意的是，这两个步骤的模式匹配中均存在着不确定性。

2.2 模式匹配不确定性产生的原因

一方面，用户提交的关键字查询往往只有属性值而不包含属性名，例如引言中提到的查询，{奔驰，20000-30000}，由于属性名的缺失使得属性的语义变得难以判断，当将其映射到相应的结构化集成查询接口上时，可能产生两种合理的匹配结果。

另一方面，集成接口模式与各Web数据库查询接口的模式之间的匹配也往往是不准确

的。首先，由于各 Web 数据库由不同的组织或个人在不同的时间和地点设计，自治性很强，造成了内容和形式的复杂性和多样性，这给模式匹配的准确性提出了更大的挑战；而且由于 Web 数据库一直处于动态变化中，据观察，web 数据库查询接口平均每三个月就会发生一些变化，这导致了已有的集成接口到各 Web 数据库查询接口的模式匹配经常失效。因此，Deep Web 集成服务的集成接口模式与各 Web 数据库查询接口模式之间的匹配存在很多不确定性。

2.3 相关工作

自动模式匹配方法^[5]的研究在九十年代后期有了长足的进展。一些原型系统在自动或半自动模式匹配研究方面有了新的突破。如：SEMINT^[6]，Cupid^[7]，DIKE^[8]，COMA^[9]等。但这些模式匹配的研究工作都是在寻找最佳的匹配结果，忽略模式匹配的不确定性。

据我们所知，目前关于不确定性的研究还只是处于初步阶段^[10]，在 2003 年的数据集成综述论文[11]中还没有提及不确定数据的问题，当时主要的问题是如何区分模式之间正确的语义关系，如何去除不确定的语义关系。在另一篇论文[12]中，提及了不准确映射问题，但只是指出了可能会产生错误的匹配并需要处理不正确的结果。最近的一篇综述论文[13]已明确指出，不确定性数据的管理将是未来研究中一个非常具有挑战性的问题。

3 用户请求转换中模式匹配的准备工作的

在模式匹配的过程中，都会涉及到相似度计算的问题。字符匹配算法已经有了比较多的研究，但对于数字类型的相似度计算方法的研究还基本是个空白。我们针对 Deep Web 集成服务中的数字类型数据的特点，提出了相应的相似度计算方法。

3.1 数值型数据的相似度计算方法

大多数涉及数字类型相似度的计算都是将数字看作字符串来处理^[14]，但显然这样的方法是不适用的。例如，“1000”与“1001”的相似度大，还是“1000”与“999”的相似度大？如果采用基于字符串相似度的计算方法，无论是 Levenshtein distance, Affine gap distance, Jaro distance, Q-gram distance, 还是 WHIRL, 得到的结果都是前者的相似度远远大于后者，这显然是不合理的。因此对于数字型数据，应提出相应的相似度计算方法。

定义 1: 给定两个数字型数据 m 和 n ，其相似度为：

$$Sim(m, n) = 1 - \frac{|m - n|}{\max(m, n)} \quad (1)$$

例如： $Sim(1000, 999) = 1 - \frac{|1000 - 999|}{\max(1000, 999)} = 0.999$

定义 2: 给定两组离散的数字型数据的集合 S_1 和 S_2 , $S_1 = \{n_1, n_2, n_3, \dots\}$, $S_2 = \{m_1, m_2, m_3, \dots\}$, 两者的重复数据的多少反映了相似程度, S_1 与 S_2 的相似度为：

$$Sim(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} \quad (2)$$

例如： $S_1 = \{1, 2, 3, 4, 5, 6\}$, $S_2 = \{1, 3, 5, 7, 9, 11\}$, $Sim(S_1, S_2) = \frac{3}{9} \approx 0.333$

定义 3: 给定两组范围型的数字集合 R_1 和 R_2 , $R_1 = \{s_1, s_2, s_3, \dots\}$, $R_2 = \{t_1, t_2, t_3, \dots\}$, 两者的重叠程度反映了相似程度, R_1 与 R_2 的相似度为：

$$Sim(R_1, R_2) = \frac{(s_1 \cup s_2 \cup \dots) \cap (t_1 \cup t_2 \cup \dots)}{(s_1 \cup s_2 \cup \dots) \cup (t_1 \cup t_2 \cup \dots)} \quad (3)$$

例如: $R_1 = \{0-100, 100-200\}$, $R_2 = \{50-150, 150-300\}$, $Sim(R_1, R_2) = \frac{(50-200)}{(0-300)} = \frac{150}{300} = 0.5$

公式 (1) (2) (3) 中, $0 \leq Similarity \leq 1$, 且计算出的相似度值越大, 两个数据的相似度越大。

3.2 基于数字实例的模式匹配的剪枝方法

在利用实例信息进行模式匹配时, 通常的方法是计算实例之间的相似度, 再判断模式匹配结果。而据我们的分析, 对于数字型数据, 通过进行一些覆盖程度和数值依赖分析, 有些匹配可以直接判断, 其余的再通过相似度计算并判断模式匹配结果, 这样将大大减少计算的复杂性。

3.2.1 基于数字实例的模式匹配剪枝方法

下面介绍 2-维查询空间与 2-维概念空间的模式匹配的剪枝方法。假设集成接口上的两个的概念 C_i 和 C_j ($i < j$), C_i 与 C_j 的候选值集合分别为 V_i 和 V_j (均为数字型), 用户查询包含数字型关键字 Key_1 和 Key_2 , 且 Key_1 和 Key_2 必与 C_i 与 C_j 中的一个匹配。另若 A 与 B 匹配, 记作 {A,B}:

1、无覆盖:

$V_i \cap V_j = \Phi$, 即概念 C_i 与概念 C_j 的候选值之间不存在任何覆盖的关系, 如图 3 (a)。关键字 $Key_1 < Key_2$, 且 $Key_1, Key_2 \in (V_i \text{ 或 } V_j)$, 那么根据关键字的值的大小可以准确判断与概念 C_i 和 C_j 的匹配关系为下面的情况之一:

(1) 若 $Key_1 \in V_i, Key_2 \in V_j$, 则匹配结果为: { Key_1, C_i }, { Key_2, C_j };

(2) 若 $Key_1 \in V_j, Key_2 \in V_i$, 则匹配结果为: { Key_1, C_j }, { Key_2, C_i };

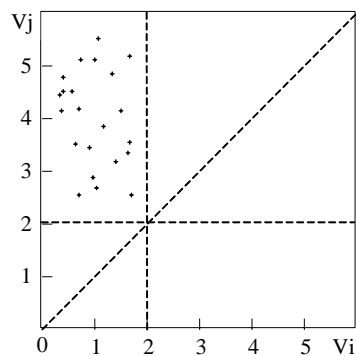


图 3(a)

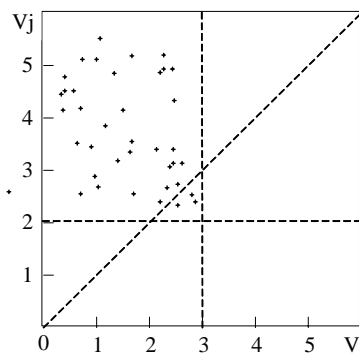


图 3(b)

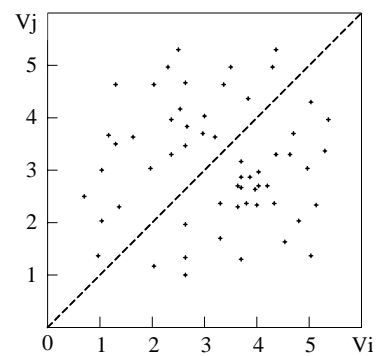


图 3(c)

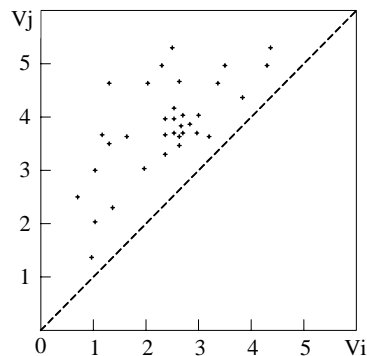


图 3(d)

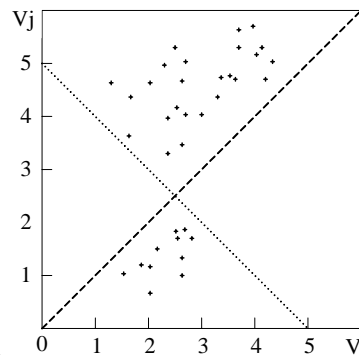


图 3(e)

2、松散部分覆盖:

$V_i \cap V_j \not\subset \Phi$, 即概念 C_i 与概念 C_j 的候选值之间存在互相覆盖的关系, V_i 和 V_j 之间无约束关系或者很难捕捉到其约束关系, 但是覆盖程度比较小, 如图 3(b)。一般地, 对于关键字 Key_1 和 Key_2 :

(1) 若 $Key_1 \notin (V_i \cap V_j)$ 且 $Key_2 \notin (V_i \cap V_j)$, 匹配方法同 1;

(2) 若 $Key_1 \in (V_i \cap V_j)$ 且 $Key_2 \in (V_j - V_i)$, 根据匹配的互斥性, 得到匹配结果 $\{Key_1, C_i\}$, $\{Key_2, C_j\}$; 反之亦然;

(3) 若 $Key_1 \in (V_i \cap V_j)$ 且 $Key_2 \in (V_i \cap V_j)$, 则不能判断匹配关系, 需要进一步计算两者的相似度。

3、松散覆盖:

$V_i \cap V_j \not\subset \Phi$, 即概念 C_i 与概念 C_j 的候选值之间存在互相覆盖的关系, V_i 和 V_j 之间无约束关系或者很难捕捉到其约束关系, 而且覆盖程度比较大, 如图 3(c)。则不能判断匹配关系, 需要进一步计算两者的相似度。

4、单一约束覆盖:

$V_i \cap V_j \not\subset \Phi$, 即概念 C_i 与概念 C_j 的候选值之间存在互相覆盖的关系, 且 V_i 和 V_j 之间的有简单的约束关系, 例如, 对于概念 C_i 与概念 C_j 的一对取值 $\langle V_i, V_j \rangle$, V_i 总是小于 V_j , 如图 3(d)。如果 $Key_1 < Key_2$, 则相应的匹配为 $\{Key_1, C_i\}$, $\{Key_2, C_j\}$ 。一般地, 若概念 C_i 与概念 C_j 的任意一对取值 $\langle V_i, V_j \rangle$:

(1) 若 $V_i < V_j$ 且 $Key_1, Key_2 \in V_i, V_j$, 若 $Key_1 < Key_2$, 则匹配结果为: $\{Key_1, C_i\}$, $\{Key_2, C_j\}$;

(2) 若 $V_i > V_j$ 且 $Key_1, Key_2 \in V_i, V_j$, 若 $Key_1 < Key_2$, 则匹配结果为: $\{Key_1, C_j\}$, $\{Key_2, C_i\}$;

其它情况, 依此类推。

5、复杂约束覆盖:

$V_i \cap V_j \not\subset \Phi$, 即概念 C_i 与概念 C_j 的候选值之间存在互相覆盖的关系, 且 V_i 和 V_j 之间的有复杂的约束关系, 例如, 对于概念 C_i 与概念 C_j 的一对取值 $\langle V_i, V_j \rangle$, 即 V_i 与 V_j 满足的约束关系不是单一的, 但 V_i 与 V_j 在分段上满足单一约束覆盖 (第 4 种情况), 即在每一段上满足单一约束覆盖, 如图 3(e)。我们可以先分段, 再根据对上述第 4 种方法分段判断关键字与概念的匹配关系。一般地, 概念 C_i 与概念 C_j 的任意一对取值 $\langle V_i, V_j \rangle$:

(1) 若 $\langle V_i, V_j \rangle \in \text{Segment}1$, $V_i < V_j$ 且 $Key_1 < Key_2$, 则匹配结果为: $\{Key_1, C_i\}$, $\{Key_2, C_j\}$;

(2) 若 $\langle V_i, V_j \rangle \in \text{Segment}2$, $V_i > V_j$ 且 $Key_1 < Key_2$, 则匹配结果为: $\{Key_1, C_j\}$, $\{Key_2, C_i\}$;

其它情况, 依此类推。

3.2.2 剪枝方法扩展——n-维查询空间与 m-维概念空间

3.2.1 节介绍了 2-维查询空间与 2-维概念空间的模式匹配的剪枝方法。下面对此做 n-维查询空间与 m-维概念空间的扩展。

查询空间的维数不同直接影响匹配的结果。考虑图 3(d)的情况, 若查询只包括一个关键字 Key_1 , 其他条件不变, $V_i < V_j$ 且 $Key_1 \in V_i, V_j$, 则匹配结果却并不唯一, 匹配 $\{Key_1, C_i\}$ 和 $\{Key_1, C_j\}$ 的概率各为 50%。另一方面, 概念空间的维数越高, 候选概念子空间越多, 因此可能的匹配结果越多。例如, 有 2-维查询 $Q = \{Key_1, Key_2\}$ 和 3-维概念 $\{C_1, C_2, C_3\}$, 候选概念子空间为 $\{C_1, C_2\}$, $\{C_2, C_3\}$, $\{C_1, C_3\}$, 与 2-维概念 $\{C_1, C_2\}$ 相比, 候选概念子空间增加, 可能的匹配结果也相应地增加。

因此查询空间的维数与概念空间维数都将影响到最后的匹配结果。对于 n-维查询空间与 m-维概念空间, 且 $n \leq m$, 则可能的匹配个数为 $N = P_m^n$ 。

在可能的匹配中，利用 3.2.1 中探讨各坐标分量（概念的候选值）的覆盖关系，约束关系以及关键字的值的的大小，是否属于各分量等特征，对模式匹配进行剪枝。

3.3 基于数据类型的匹配优化

我们观察到：数据类型（主要包括文本型、数字型和时间型）是否相同其实是进行关键字和集成查询接口的概念之间或者集成查询接口的概念与 Web 数据库查询接口的属性之间是否匹配的先决条件，即关键字往往与其数据类型相同的概念匹配，概念也只与数据类型相同的属性匹配。例如，文本型的关键字只可能与文本类型的概念相匹配。下面以关键字与集成查询接口的概念之间的匹配为例，说明如何基于数据类型进行匹配的优化。

如果不考虑数据类型，将查询的关键字 $\{Key_1, Key_2, \dots, Key_k\}$ 与集成查询接口的概念 $\{C_1, C_2, \dots, C_j\}$ 一一匹配，那么可能的匹配结果数量为 P_j^k 。其中 j 是复杂查询接口的概念个数， k 是用户提交的关键字个数，因为通常复杂的集成查询接口的概念一般在 50 个左右，计算量是比较大的。

如果只进行同数据类型的关键字与概念之间的一一匹配，则可能的匹配结果数量为：

$$P_{j_1}^{k_1} + P_{j_2}^{k_2} + P_{j_3}^{k_3}, \quad (k_1+k_2+k_3=k, j_1+j_2+j_3=j)$$

其中， k_1, k_2 和 k_3 分别是关键字中包含文本型、数字型和时间型数据的个数， j_1, j_2 和 j_3 分别是集成查询接口的概念中包含文本型、数字型和时间型数据的个数。由于：

$$P_{j_1}^{k_1} + P_{j_2}^{k_2} + P_{j_3}^{k_3} \ll P_j^k$$

因此基于数据类型的匹配得到的可能匹配结果数量远远小于原来进行一一匹配的结果数量。如图 4 所示。

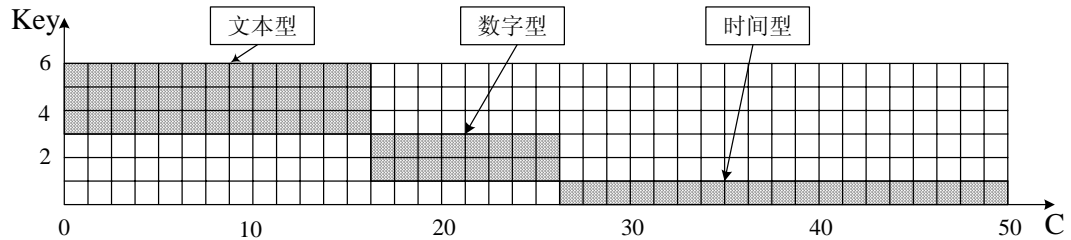


图 4 基于数据类型的可能模式匹配

4 用户请求转换的不确定性模式匹配方法：

现有的自动模式匹配工具都是产生一个最佳的匹配结果。但是大量二义性和异构性的概念描述使得模式匹配存在很多不确定性，很有可能在保留系统认为最好的匹配结果的同时，将本来有意义的较好的匹配结果丢失了。我们的方案是考虑不确定性模式匹配，尽可能使系统保留合理的模式匹配结果并按匹配程度排序。

4.1 查询模型与接口模型

用户提交的关键字查询往往没有标签说明其语义信息，可以将查询的关键字看作是缺乏模式信息的实例信息。复杂的结构化集成接口中则包含了丰富的模式信息和实例信息，其中的每一个概念是对各 Web 数据库查询接口相同语义的属性名的概括和总结，而且其不仅保留了表示此概念的各查询接口属性的标签名，还保留了表示此概念的各查询接口所包含的属性的候选值。各 Web 数据库查询接口包含了一定的模式信息和实例信息，每个属性都有相应的标签名，一些属性有相应的候选值。

定义 4：关键字查询模型：用户提交的查询 Q 由一组关键字组成，

$$Q = \{key_i \mid key_i \in Key, 1 \leq i \leq k\}$$

其中, Key 为用户查询的关键字集合。

定义5: 集成查询接口模型: 集成查询接口 II (Integrated Interface) 包括一组概念、与概念同义的标签、概念所属数据类型以及与此概念相应的候选值。

$II = \{ \langle C_i, CL_i, CT_i, CV_i \rangle \mid C_i \in C, CL_i \in CL, CT_i \in CT, CV_i \in CV, 1 \leq i \leq k \}$ 其中

- 1) C : 表示集成接口上概念的集合。
- 2) CL : 各查询接口上表示此概念的属性的标签名使用的词汇集合。
- 3) CT : 表示此概念的数据类型, 包括文本型、数字型和时间型。
- 4) CV : 表示此概念的各查询接口的候选值的集合。

定义6: Web 数据库的查询接口模型: Web 数据库的查询接口 LI (Local Interface) 包括一组属性的标签、所属数据类型以及与此相应的候选值。 LI 的定义如下:

$LI = \{ \langle A_i, AT_i, AV_i \rangle \mid A_i \in A, AT_i \in AT, AV_i \in AV, 1 \leq i \leq k \}$

其中:

- 1) A : 表示 Web 数据库查询接口上属性的标签名的集合。
- 2) AT : 表示此属性的数据类型, 包括文本型、数字型和时间型。
- 3) AV : 表示此属性的候选值的集合。

4.2 不确定性模式匹配方法:

4.2.1 相关概念

定义7: 属性概念匹配 ACM (Attribute-Concept Matching): 是指某 web 数据库查询接口的某个属性与复杂集成接口上的一个概念之间建立的匹配。

给定 Web 数据库查询接口的属性 A 和复杂接口概念 C , 他们之间的 ACM 可以记为 $ACM(A, C)$ 。那么 i 个属性与 j 个概念可以构成 $i*j$ 个不同的 ACM , 然后在这 $i*j$ 个 ACM 中用排列组合的方法找到所有的可能匹配组合。

相应地, 关键字概念匹配 KCM (Keyword-Concept Matching), 是指在一个关键字和复杂接口上的一个概念之间建立的匹配。

定义8 属性概念匹配度 $ACMD$ (Attribute-Concept Matching Degree): 是指某 web 数据库查询接口的一个属性与复杂集成接口上的一个概念之间匹配的程度。

匹配的特征表现在如下几个方面: 标签名、数据类型、实例的值 (文本枚举值、最小值、最大值或平均值)。我们用属性与概念的相似度, 包括模式信息的相似度和实例信息的相似度, 来表示两者之间的匹配程度。

相应地, $KCMD$ (Keyword-Concept Matching Degree) 是关键字概念匹配度。

$KCMD$ 与 $ACMD$ 的不同之处在于, 在关键字与概念的匹配度计算中, 主要是计算关键字与概念的候选值集合之间的匹配关系, 是通过计算两者实例的相似性得到。而概念与属性的匹配中, 概念由其标签集合和候选值集合两部分组成, 它们可以分别与属性的标签和候选值集合进行匹配计算。因此, 在进行概念与属性比较的时候, 仅仅用一个匹配器是远远不够的, 例如: 假设 web 数据库查询接口的一个属性与复杂集成接口上的一个概念有很多相同的实例, 实例匹配器判断两者是匹配的; 但是由于这两个属性在属性名上差异很大, 属性名匹配器判断两者是不匹配的。那么这两个属性是否匹配? 需要综合考虑属性的各方面特征, 由多个匹配器共同决定两者之间的匹配关系。本文采用基于模式的匹配器和基于实例的匹配器混合匹配方法, 如公式 (4)。

$$ACMD(A_i, C_j) = w_1 * Sim_L(A_i, C_j) + w_2 * Sim_V(A_i, C_j) \quad (4)$$

其中: Sim_L , Sim_V 分别是属性和概念的标签相似度和实例相似度。 w_1 和 w_2 分别为模式

相似度和实例相似度的相应权重。为了避免用经验值确定 w_1 和 w_2 的主观性，我们引入了遗传算法，为训练集选择最佳的权重组合，得到的权重 w_1 和 w_2 近似为 0.5。

而关键字概念匹配度的计算是基于实例的匹配，如公式 (5)

$$KCMD(Key_i, C_j) = Sim(Key_i, C_j) \quad (5)$$

其中， Sim 是关键字和概念的候选值之间的实例相似度。

4.2.2 相似度计算

3.3 节已经分析过，查询转换只在相同数据类型之间进行，因此模式匹配问题也限定在相同数据类型的关键字、概念或属性之间（不同数据类型关键字、概念或属性之间的相似度为 0）。而且对于 3.2 节已经可以判断出的匹配，则不必再计算其相似度（其相似度为 1）。

1. 关键字与概念的相似度计算

在关键字与集成接口概念的匹配中，主要是计算关键字与概念的候选值集合之间的匹配关系，通过计算两者实例的相似度得到。

如果关键字 Key_i 和概念 C_j 同属于字符类型，则将 Key_i 与概念 C_j 的候选值集合 CV_j 中的每一个值相比较，选择相似度值最大的作为关键字 Key_i 和概念 C_j 的相似度，其中隐含了这样一个原则：如果概念 C_j 的候选值集合中包含与关键字 Key_i 类似的字符串，越类似则 Key_i 与 C_j 越相似，相似度计算如公式 (6)：

$$Sim(Key_i, C_j) = \max(1 - \frac{edit\ distance(Key_i, CV_{j_k})}{\max(|Key_i|, |CV_{j_k}|)}) \quad (6)$$

如果关键字 Key_i 和概念 C_j 同属于非字符类型，即数字类型或时间类型，且无法用 3.2 节的剪枝方法准确判断匹配结果的，则需要计算关键字 Key_i 和概念 C_j 之间的相似度。将 Key_i 与概念 C_j 的候选值集合 CV_j 中的每一个值（可以是离散值，也可以是范围连续值）相比较，选择相似度值最大的作为关键字 Key_i 和概念 C_j 的相似度。相似度计算如公式 (7)：

$$Sim(Key_i, C_j) = \max(1 - \frac{|Key_i - CV_{j_k}|}{\max(Key_i, CV_{j_k})}) \quad (7)$$

2. 概念与属性的相似度计算

在集成接口概念与各 Web 数据库查询接口的匹配中，主要是计算概念与属性之间的匹配关系，由于相应的模式信息和实例信息比较丰富，因此通过计算两者模式相似度和实例的相似度的加权求和得到，如前公式 (4)。

$$Sim(A_i, C_j) = w_1 * Sim_L(A_i, C_j) + w_2 * Sim_V(A_i, C_j)$$

(1) 计算 $Sim_L(A_i, C_j)$

Web 数据库的任一属性名与集成接口的任一概念的标签匹配度计算，如公式 (8)：

$$Sim_L(A_i, C_j) = \max(1 - \frac{edit\ distance(A_i, CL_{j_k})}{\max(|A_i|, |CL_{j_k}|)}) \quad (8)$$

其中， CL_{j_k} 是概念 C_j 的任一标签名。

(2) 计算 $Sim_V(A_i, C_j)$

Web 数据库的任一属性名与集成接口的任一概念的候选值匹配度计算：

①若候选值为字符型，则二者的相似度为此属性任一候选值与此概念任一候选值的最大相似度的平均值，如公式（10）：

$$Sim_V(A_i, C_j) = avg(\max(1 - \frac{edit\ distance(AV_{i_p}, CV_{j_p})}{\max(|AV_{i_p}|, |CV_{j_p}|)}) \quad (9)$$

②若候选值为数字型，则二者的相似度为候选值的重复程度（离散值），如公式（10）；或覆盖程度（连续范围值），如公式（11）：

$$Sim_V(A_i, C_j) = \frac{|AV_i \cap CV_j|}{|AV_i \cup CV_j|} \quad (10)$$

$$Sim_V(A_i, C_j) = \frac{(AV_{i_1} \cup \dots \cup AV_{i_p}) \cap (CV_{j_1} \cup \dots \cup CV_{j_k})}{(AV_{i_1} \cup \dots \cup AV_{i_p}) \cup (CV_{j_1} \cup \dots \cup CV_{j_k})} \quad (11)$$

其中， AV_{i_p} 是属性 A_i 的任一候选值， CV_{j_p} 是概念 C_j 的任一候选值。

4.2.3 不确定模式匹配的选择

定义9 可能模式匹配 PSM (Possible Schema Matching): 是指可以产生的可能的模式匹配方案， $ACM(A_1, C_1') \wedge ACM(A_2, C_2') \wedge \dots \wedge ACM(A_k, C_k')$ 是 web 数据库查询接口属性与集成查询接口概念的一个可能的模式匹配方案， C_i' 表示 A_i 对应的概念，可以为 nil。 $KCM(Key_1, C_1') \wedge ACM(Key_2, C_2') \wedge \dots \wedge ACM(Key_j, C_j')$ ，是关键字与集成查询接口概念的一个可能的模式匹配方案， C_j' 表示 Key_j 对应的概念，可以为 nil。

PSM 必须遵循的两个原则：

(1) 存在性原则：

对于每个关键字 Key_i ， $\exists KCM(Key_i, C_i')$ ， $C_i' = C_j$ 或者 nil。即每个关键字都有一个匹配。类似地，对于 Web 数据库查询接口的每个属性 A_i ， $\exists ACM(A_i, C_i')$ ， $C_i' = C_j$ 或者 nil。

(2) 唯一性原则：

对于任意一个概念 C_j ， $\neg \exists (KCM(Key_i, C_j) \wedge (KCM(Key_k, C_j))) (i < k)$ 。即不存在某个概念同时对应多个关键字。类似地，对于任意一个概念 C_j ， $\neg \exists (ACM(A_i, C_j) \wedge (ACM(A_k, C_j))) (i < k)$ 。即不存在某个概念同时对应 Web 数据库查询接口的多个属性。因为前面我们提到，集成查询接口的概念是重要的而且是最详细的。

定义10 可能模式匹配的可信度 PSMC (Possible Schema Matching Conviction): 综合考虑可能模式匹配 $KCM(Key_1, C_1') \wedge ACM(Key_2, C_2') \wedge \dots \wedge ACM(Key_j, C_j')$ 或 $ACM(A_1, C_1') \wedge ACM(A_2, C_2') \wedge \dots \wedge ACM(A_k, C_k')$ 中各关键字概念或属性概念匹配程度后，得到的评价可能模式匹配优劣的标准。

它与各 KCMD 或 ACMD 的值密切相关。直观地，KCMD 或 ACMD 的值越高，代表匹配越合理，那么包含这样的 KCM 或 ACM 的 PSM 就可能越好。考虑到各概念在复杂查询接口上的重要程度的不同，我们赋予其的权重也有差异。因此可能模式匹配的可信度是相关 KCMD 或 ACMD 值的加权求和：

关键字与集成查询接口模式的可能模式匹配的可信度，如公式（12）：

$$PSMC = \sum_{i=1}^n w_i * KCMD(key_i, C_j) \quad (12)$$

类似地，Web 数据库接口属性与集成接口概念的可能模式匹配的可信度，如公式（13）：

$$PSMC = \sum_{i=1}^n w_i * ACMD(A_i, C_j) \quad (13)$$

其中，各概念的权重是各 Web 数据库查询接口上： $w_i = if_i / \sum_{k=1}^n if_k$ ， if_i 是各个 web 数

据库查询接口上表示概念 C_i 的属性出现的频率。 $\sum_{k=1}^n if_k$ 是各个 web 数据库查询接口上表示

概念 C_k 的属性出现的频率的总和。

将所有的 PSM 按照其 PSMC 值排序， $PSMC_1 > PSMC_2 > PSMC_3 > \dots > PSMC_i$ ，如果第 k 个可能模式匹配的可信度与第 $k+1$ 个可能模式匹配的可信度的差值远大于其与第 $k-1$ 个可能模式匹配的差值，如公式 (14)，则选择可信值前 k 个 PSM 作为最后的模式匹配结果。

$$|PSMC_k - PSMC_{k+1}| > \tau |PSMC_{k-1} - PSMC_k|, \quad (\text{设 } \tau \text{ 为 } 2, \text{ 实验结果比较理想}) \quad (14)$$

5 实验

本节我们首先介绍用于实验的数据集，然后给出各项实验结果以及对实验结果的分析。

5.1 数据集

我们选取工作、车、机票和书四个领域的集成服务以及大量用户查询作为实验的数据集。

1. 招聘集成

JobTong 是由 WAMDM 实验室自行开发的招聘信息的集成系统，它集成了当前国内流行的招聘网站（智联招聘、中华人才网、前程无忧网、易才等几十个网站）的招聘信息。

2. 车、机票和书的集成

车、机票和书这三个领域的 web 数据库资源来自 UIUC 提供的 Deep Web 数据集¹，我们采用了 Wise-integrator^[15]的 Demo 系统生成复杂的结构化集成查询接口，并利用 WordNet 将集成接口概念的标签名集合 CL 中所有标签名的同义词、概念的候选值集合 CV 中所有候选值的同义词和相应的下位词补充到集合中。

3. 用户查询

用户提交的查询由一组关键字组成，关键字之间用空格隔开。我们请 20 名学生提供在这四个不同领域上的关键字查询，并尽量使得这些学生有不同专业背景，并来自不同的年级。每个学生在每个集成系统上提交 15 个查询，在招聘集成系统和车、机票和书集成系统中分别有 300 个（共 1200 个）用户关键字查询。

5.2 衡量标准

我们采用模式匹配的准确率和召回率两个度量标准来衡量匹配结果的有效性。对于用户提交的关键字查询，假设在将其匹配到集成查询接口，进一步匹配到 Web 数据库查询接口的过程中，获得了相应的模式匹配结果，则准确率和召回率的定义如下：

$$\text{准确率} = \frac{\text{获得的正确的模式匹配个数}}{\text{获得的模式匹配的总个数}}$$

$$\text{召回率} = \frac{\text{获得的正确的模式匹配个数}}{\text{正确的模式匹配的总个数}}$$

5.3 实验结果

对于用户提交的关键字查询，图 5 (a) 给出了在各个领域中，关键字与集成查询接口的模式匹配的准确率和召回率。在工作、车、机票和书这四个领域，模式匹配的准确率分别达到了 88.3%、92.5%、90.2% 和 83.8%，召回率分别达到了 96.2%、97.8%、96.9% 和 94.5%。

¹ The UIUC Web Integration Repository. <http://metaquerier.cs.uiuc.edu/repository/>

对于用户提交的大部分关键字查询，都可以成功地匹配到集成查询接口上，其中只有书的领域匹配结果不太理想，这主要是由于此领域的一些重要属性，如“Title”、“Author”，后接的是文本框，要预先知道其值域范围是非常困难的。图 5 (b) 给出了在各个领域中，集成查询接口与 Web 数据库查询接口的模式匹配的准确率和召回率，准确率分别是 90.5%、93.3%、91.8%和 85.6%，召回率分别达到了 97.2%、98.6%、98.3%和 94.9%。图 5 (a) 与图 5 (b) 相比，我们可以看出，查询关键字与集成查询接口的模式匹配在准确率和召回率上都略低于集成查询接口与 Web 数据库查询接口的模式匹配，这主要是因为前者只能根据实例信息进行模式匹配，而后者不仅可以根据实例信息还可以根据模式信息进行模式匹配。

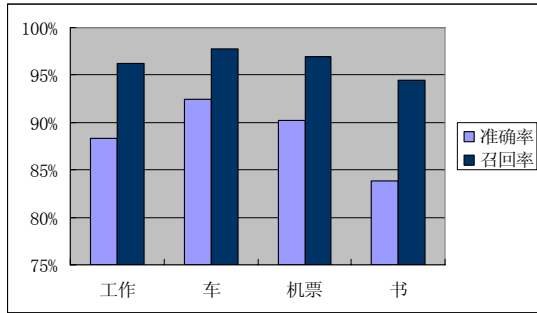


图 5(a) 查询关键字-集成查询接口的模式匹配

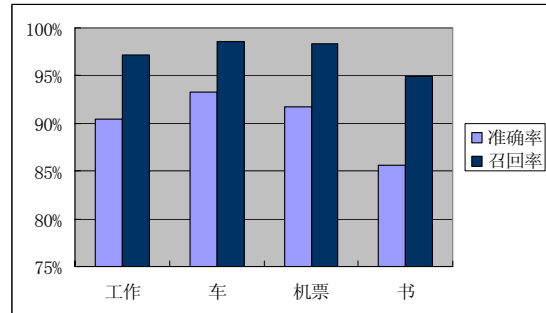


图 5(b) 集成查询接口-Web 数据库查询接口的模式匹配

而且我们的实验结果表明，与传统的模式匹配，即与不考虑模式匹配的不确定性相比，在匹配的准确率，尤其是召回率上有了明显的提高，如图 5(c)- 图 5 (f)。

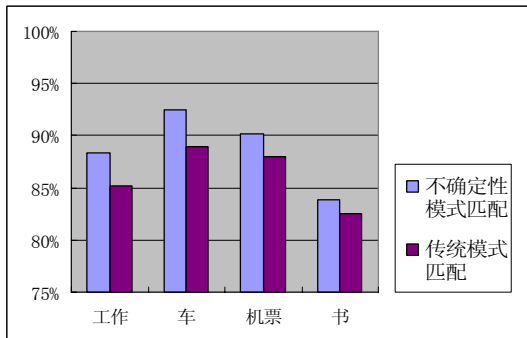


图 5(c) 不确定性模式匹配与传统模式匹配的准确率比较 (查询关键字-集成查询接口)

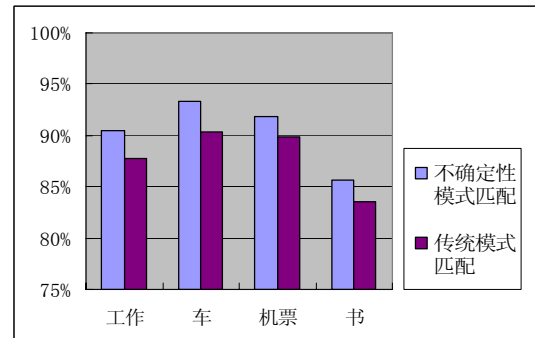


图 5(d) 不确定性模式匹配与传统模式匹配的准确率比较 (集成查询接口-Web 数据库查询接口)

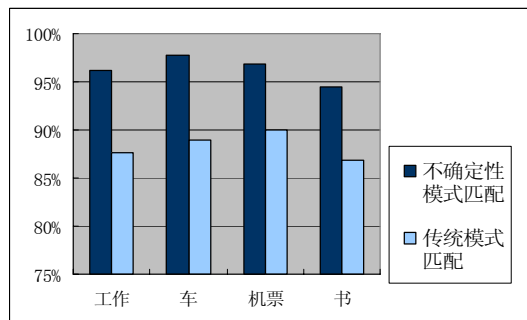


图 5(e) 不确定模式匹配与传统模式匹配的召回率比较 (查询关键字-集成查询接口)

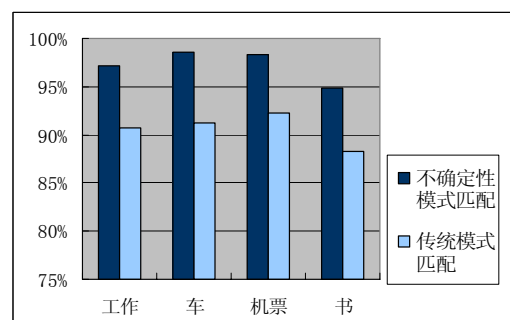


图 5(f) 不确定性模式匹配与传统模式匹配的召回率比较 (集成查询接口-Web 数据库查询接口)

我们在匹配的过程中还运用了优化方法：数据类型驱动的优化方法和数字类型匹配的剪枝方法，这明显的降低了模式匹配的时间复杂度，如图 6。

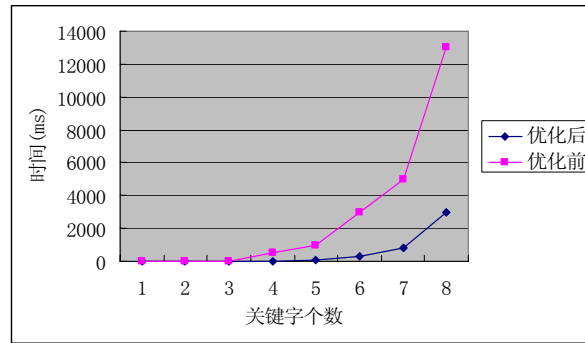


图 6 优化前后模式匹配的时间复杂度

6 结论

本文针对在 Deep Web 集成服务中模式匹配的不确定性，提出了基于相似度计算的不确定性模式匹配方法。本文首先剖析了模式匹配的不确定性，然后针对目前数字类型数据的相似度计算方法匮乏的现状，提出了一组数字类型数据的相似度计算方法；并给出了进行数字类型的模式匹配的有效的剪枝方法以及数据类型驱动的模式匹配优化方法；最后在此基础上提出了不确定性模式匹配方法。实验结果表明，本文提出的方法是非常有效的。

参考文献：

- [1] Raghavan S, Garcia-Molina H. Crawling the Hidden Web. // Proceedings of the 27th International Conference on Very Large Data Bases, Roma, 2001: 129-138
- [2] He B, Chang K C-C. Making holistic schema matching robust: an ensemble approach. // Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, 2005: 429-438.
- [3] Wang J, Wen J, Lochovsky F H, Ma W. Instance-based Schema Matching for Web Databases by Domain-specific Query Probing. // Proceedings of the 13th International Conference on Very Large Data Bases, Toronto, 2004: 408-419.
- [4] He B, Chang K C-C. Statistical Schema Matching across Web Query Interfaces. // Proceedings of the 22th ACM SIGMOD International Conference on Management of Data, San Diego, 2003: 217-228.
- [5] Rahm E, Bernstein P A. A survey of approaches to automatic schema matching. VLDB Journal, 2001, 10(4): 334-350.
- [6] Li W., Clifton C. SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. Data and Knowledge Engineering, 2000, 33(1): 49-84.
- [7] Madhavan J, Bernstein P A, Rahm E. Generic Schema Matching with Cupid. // Proceedings of the 27th International Conference on Very Large Data Bases, Roma, 2001: 49-58.
- [8] Palopoli L, Sacca D, Ursino D. An automatic technique for detecting type conflicts in database schemas. // Proceedings of the 7th ACM CIKM International Conference on Information and Knowledge Management, Bethesda, 1998: 306 -313.
- [9] Do H, Rahm E. COMA - A system for exible combination of schema matching approaches. // Proceedings of the 28th International Conference on Very Large Data Bases, Hong Kong, 2002: 610-621.
- [10] Magnani M, Montesi D. Uncertainty in data integration: current approaches and open problems. MUD Workshop of VLDB Conference, 2007.
- [11] Halevy A Y. Data Integration: A Status Report. // Proceedings of the 10th Conference on Database Systems for Business, Technology and the Web, 2003: 24-29.

- [12] Doan A, Halevy A Y. Semantic Integration Research in the Database Community: A Brief Survey. *AI Magazine*, 2005, 26(1): 83-94.
- [13] Halevy A Y, Rajaraman A, Ordille J J. Data Integration: The Teenage Years. // *Proceedings of the 32nd International Conference on Very Large Data Bases*, Seoul, 2006: 9-16.
- [14] Elmagarmid A K, Ipeirotis P G, Verykios V S. Duplicate Record Detection: A Survey. *IEEE Transactions on Knowledge and Data Engineering*. 2007, 19(1): 1-16.
- [15] He H, Meng W, Yu C T, Wu Z. WISE-Integrator: An Automatic Integrator of Web Search Interfaces for E-Commerce. // *Proceedings of the 29th International Conference on Very Large Data Bases*, Berlin, 2003: 357-368.



JIANG Fang-Jiao, born in 1971, Ph.D. candidate. Her research interests include Web data management and integration.

Meng Xiao-Feng, born in 1964, professor and Ph.D. supervisor. His research interests include Web data management, native XML database, and mobile data management.

Jia Lin-Lin, born in 1984, M.S. Her research interests include Web data management and integration.

Background:

This research was partially supported by the grants from the Natural Science Foundation of China (No: 60573091); China 863 High-Tech Program (No: 2007AA01Z155); China National Basic Research and Development Program's Semantic Grid Project (No. 2003CB317000); Program for New Century Excellent Talents in University (NCET).

With an increasing number of Web databases, it is more and more difficult for users to get their desired information among these Web data sources manually. The purpose of those projects is to provide users an automatic approach to achieve and integrate the information in Deep Web. In recent years, more and more researchers have focused on some issues in it. In the past years, the authors have researched and developed a lot of techniques in the area of Deep Web integration service, and these works mainly focus on schema matching, query translation and database selection. This paper focuses on the uncertain schema matching method.