一种改进的基于 WSDL 描述的操作相似性度量方法

何玲娟 刘连臣 吴 澄
（清华大学国家 CIMS 工程研究中心 北京 100084）

摘 要 在目前通用的 Web 服务描述标准 WSDL 基础上，文中提出一种改进的操作相似性度量方法 MOSM. MOSM 在数据预处理后将 Web 服务内含的操作(operation)建模为无序标签树，并通过计算满足约束的编辑距离对其进行相似性度量. 其具体做法是抽取操作的 XML 模式的树形结构，对结构进行变形，只保留标签结点；然后计算生成的无序标签树之间满足约束的编辑距离，将操作相似性度量的问题转化为无序标签树匹配的问题. 该文的创新主要在：建模时采用满足约束的无序树模型，在编辑距离算法中引入支持不对称性的代价模型，另外为结构和文字标签匹配引入了相似系数. 文中最后给出实验了对比结果，MOSM 算法能有效提高 top-k 查准率，对找寻相似的备选操作具有重要意义.

关键词 Web 服务发现；WSDL；相似性度量；无序标签树；编辑距离；模式匹配
中图法分类号 TP311

A Modified Operation Similarity Measure Method Based on WSDL Description

HE Ling-Juan LIU Lian-Chen WU Cheng
(*National CIMS Engineering Research Center*, *Tsinghua University*, *Beijing* 100084)

Abstract Based on the current Web service description standard WSDL (Web Service Description Language), a Modified Operation Similarity Measure (MOSM) method is proposed. After the data preprocessing, an operation included in a Web service is modeled as an unordered labeled tree and get the similarities measured with other operations by calculating the constrained edit distances in MOSM. First the tree structure of the operation′s XML schema is extracted and transformed so as only the tag nodes are kept; then the constrained edit distances among the generated unordered labeled trees are calculated, which turns the operation similarity measure problem into the unordered labeled tree matching problem. The innovations of this paper mainly are: Using the constrained unordered trees instead of the ordered trees as the model, introducing the asymmetric characteristic into the cost model of the tree edit distance algorithm, and introducing the similarity weight into both the structure matching and the string label matching. In the end of the paper the experiments shows that MOSM can effectively improve the top-k precision, which makes much sense to look for the similar back up operations.

Keywords Web services; WSDL; similarity measure; unordered labeled tree; edit distance; schema matching

---

收稿日期：2008-04-08；最终修改稿收到日期：2008-06-24. 本课题得到国家科技基础条件平台“大型科学仪器设备资源的建设与整合”(2005DKA10100)资助. **何玲娟**，女，1982 年生，博士研究生，主要研究方向为服务发现与匹配. E-mail：helingjuan@gmail.com. **刘连臣**，男，1971 年生，副研究员，主要研究领域为网格应用和服务科学. **吴 澄**，男，1940 年生，教授，博士生导师，研究领域为企业 CIMS 应用信息集成和复杂系统.

1 引 言

现有的 Web 服务架构一般基于经典的三角模式，采用 WSDL[①]（Web Service Description Language）标准来对服务进行描述，并基于 XML 实现消息传递机制，具有跨平台、松耦合等特性，为信息的自动化处理提供了基础，具有广阔的发展空间. 服务发现研究如何针对服务描述文档的描述来寻找相关的 Web 服务，而服务匹配对于这些相关的服务进行更进一步的相似性的计算. 当前已有的线上 Web 服务搜索方法，如 Binding Point[②]，SalCentral[③]，基本都是利用关键字查询来查找相关的服务. 这种方案的优点是查全率较高而查准率较低，其主要不足如下：

(1) 返回服务的数量较多，却无法保证对于查询者来说最为重要的 top-k 查准率；

(2) 只提供针对 Web 服务的查询，不提供针对服务操作的查询；

(3) 缺乏这样一种机制，返回最为相似的服务操作的集合并具有相似度排名，以供查询者找寻相似的备选操作.

借鉴传统信息检索的成功经验，许多研究方案引入了向量空间模型以提高服务查询的查准率，并且利用向量夹角余弦值进行相似性度量. 由于 Web 服务发现比传统信息检索的复杂度更高，这种方案虽然提供了关键字方案缺乏的相似度量化排序的机制，但是因为忽略了 WSDL 提供的丰富的符合标准规范的结构信息，而使得 top-k 查准率不尽如人意.

本文的目的是提供一种新的 Web 服务操作查询匹配机制 MOSM（Modified Operation Similarity Measure，改进的操作相似性度量），它具有如下特点：

(1) 支持面向 Web 服务操作层次的查询；

(2) 同时利用统计语义信息和 WSDL 提供的结构信息进行相似性度量；

(3) 返回的操作序列按语义和结构的综合相似度排序；

(4) 具有较高的 top-k 查准率，特别是 k 较小的情况；

(5) 引入匹配的不对称性概念为服务增加匹配概率.

这种机制的研究动机是首先为了适应动态的服务状态变化的环境. 当查询者选择的服务在运行时不可使用时，能尽快为查询者提供功能相似的后备服务的参考；其次是为了使不完全相同但具备替代可行性的服务操作增大匹配概率，最大程度实现服务的重用和增值.

本文研究的问题性质是 Schema 匹配问题在 Web 服务整合方向的应用，目标是使 Schema 匹配在该背景下最大限度地提高服务操作匹配的成功率. 本文的创新主要在于选择满足约束的无序树编辑距离模型对 WSDL 中的 Schema 进行建模，在编辑距离算法中引入支持不对称性的代价模型，另外为结构和文字标签匹配引入了相似系数.

本文第 2 节描述如何对基于 WSDL 描述的服务操作进行建模；第 3 节对针对这个模型改进的编辑距离算法进行了详细介绍；第 4 节给出了实验对比结果及评价；第 5 节介绍相关研究；第 6 节给出结论和对未来工作的展望.

2 基于 WSDL 描述的操作模型

一个 WSDL 文档是一个存储在各种形式的注册中心中的满足 W3C 系列规范的 XML 文档，它包含了 Web 服务的名称和命名空间等基本信息、若干个用结构化语言描述的操作、对服务和操作的自然语言描述以及调用这些操作的方法和地址等. 对于面向服务功能进行的服务查询而言，研究者关注与服务功能直接相关的部分，即以自然语言描述的服务及操作内容概要和以结构化语言描述的服务操作结构.

2.1 WSDL 结构简析

一个 Web 服务的 WSDL 描述文档的主要内容可以形成一个逻辑上的树状结构，其中根节点代表该 WSDL 文档，第 2 层的节点代表该 Web 服务包含的操作，第 3 层节点代表操作调用的消息，第 4 层节点代表消息调用的参数. 以每个操作代表的节点作为根节点的子树包含两类子节点，一类是输入消息的子节点，一类是输出消息的子节点. 第 4 层 Data type 不是叶子节点，而是基于 XML Schema[④] 构建的，具有规范良好的结构信息的模式树. 如图 1 所示.

MOSM 在数据预处理时为了充分利用文档中

① WSDL. http://www.w3.org/TR/2007/WD-wsdl20-primer-20070326/
② Binding Point. http://www.bindingpoint.com
③ SalCentral. http://www.salcentral.com
④ XML Schema. www.w3.org/XML/Schema

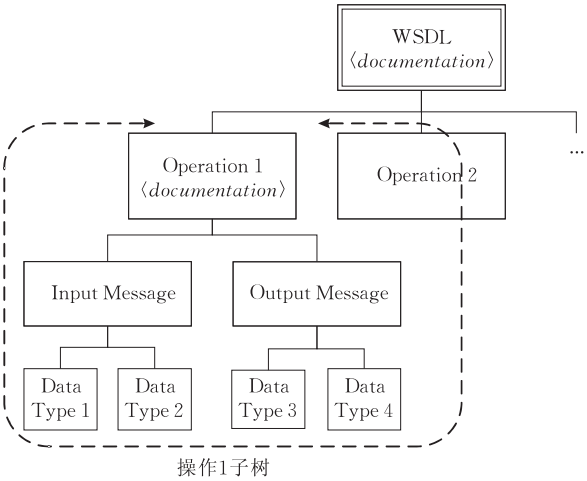

图 1　Web 服务描述文档树状结构

的自然语言信息，参照传统信息检索的向量空间模型 VSM(Vector Space Model)，为每个 WSDL 文档抽取代表该 Web 服务特征的词汇集合，并构造为一个服务向量空间中的服务向量. 主要做法如下：

在 WSDL 文档中抽取特征词汇集合：该做法根据结构文档的特点，与无结构文档稍有不同. 根据 WSDL 提供的标签含义来解析文档，抽取 Service 和 Operation 的描述信息以及上图各层节点的名称，即 Service 和 Operation 的〈*documentation*〉标签下的内容，以及服务名称、操作名称和参数名称. 将获得的信息作分词处理，并做去除停词(stop words)和取词干(stemming)等预处理，最终构成文档特征词汇集合. 然后参照传统 VSM 技术的统计词频，构造向量以及计算向量夹角余弦值，将 Web 服务按照统计语义的相似程度进行粗分类.

2.2　操作树的变形

除去使用统计语义，MOSM 最重要的特点是充分利用了 WSDL 提供的结构信息. 因为 WSDL 是基于 XML 模式描述的，并不存在数据类型实例，因此只能利用 Schema 提供的信息来进行匹配. Schema 匹配的方法有好几个种类(见第 5 节)，为了引入操作匹配的不对称性和子结构相似系数，量化操作和操作之间的相似度，本文基于测量树之间的编辑距离以比较树形结构相似性的工作，将操作的结构变形为适合使用树编辑距离经典算法的结构，再对算法进行修正和改进.

在图 1 的模型中，Data Type 子树包含两种节点：一种是对应 WSDL 中的 element 标签的节点，称之为标签节点(tag node)，它们提供了与操作相关的语义信息；另一种对应于 XML 模式内建的 3 种 model group 标签④，通常为 complexType 结点所嵌套，将其称之为约束节点(constraint node). 经典算法无法处理这种结点，因此需要通过变形既要在结构中消去它们，又要以合适的方式保留他们嵌套的标签节点的语义信息.

变形的步骤如下：

(1) 消去 Message 层，同时将 Operation 层的每个操作结点分裂为两个结点：一个操作输入结点和一个操作输出结点. 于是一个操作子树分裂为两个：一个输入子树，其子节点为该操作对应的输入参数，一个输出子树，其子节点为该操作对应的输出参数，如图 2 所示. 计算两个操作间的距离时将两棵输入子树之间的距离与对应的两棵输出子树之间的距离线形相加即可. 即

$$DistOf(Oper1, Oper2) = DistOf(Oper1input, Oper2input) + DistOf(Oper1output + Oper2output) \quad (1)$$

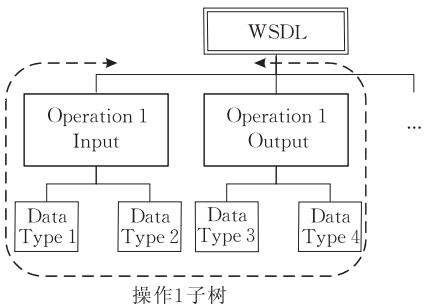

图 2　操作子树变形步骤 1

(2) 消去 Data Type 模式树的约束节点.

XML 模式将 Data Type 分为两种：简单数据类型和复杂数据类型. 简单数据类型只声明参数名和 int，string 等 xsd 内建参数类型；而复杂数据类型除了声明参数名，还可以通过使用内建的 model group 标签来嵌套其它简单或复杂数据类型. 在形成 Data Type 模式树的时候，由简单数据类型形成的子树只有一个节点，而由复杂数据类型形成的子树可能具有多层次多个节点. XML 模式内建的 model group 有〈*xsd*：*sequence*〉，〈*xsd*：*choice*〉，〈*xsd*：*all*〉，对应 3 种约束节点，这些节点将相应标签嵌套的内容作为子节点.

(1) 顺序节点. 对应于〈*xsd*：*sequence*〉标签，要求在实际数据类型中子节点按顺序出现.

(2) 选择节点. 对应于〈*xsd*：*choice*〉标签，要求在实际数据类型中仅只出现子节点之一.

(3) 多选节点. 对应于〈*xsd*：*all*〉标签或属性出现 *minOccurs*/*maxOccurs* 的标签. 对于前者来说，在实际数据类型中每个子节点可以选择的最低出现

次数 *minOccurs* 为 0，最高出现次数 *maxOccurs* 为 1；对于后者来说这两个值是可以指定的.

需要说明的是，依据 Data Type 模式树构建的应用中的数据类型实例将不再含有这些约束节点，而这些数据类型的子节点都将按照如上约束来出现，也就是说一种模式可能对应多种数据类型实例，但是它们都严格遵守模式提出的约束.

约束节点消去原则包括：

(1) 对于顺序节点，简单地消去约束节点，并将其子节点变更为其父节点的子节点.

假设模式 A 和模式 B 的顺序节点具有相同的子节点，而只是顺序受顺序约束节点约束，认为这种情况是数据类型匹配的(见 2.3 节). 因此消去约束节点即可.

(2) 对于选择节点而言，消去约束节点，将所有子节点合并为一个子节点，并将合并的标签名作为新节点的标签名，如图 3 所示.

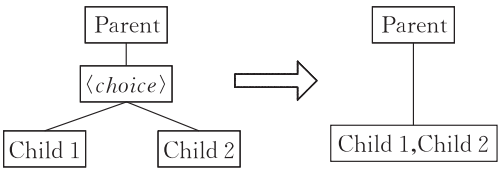

图 3 选择节点的消去

假设模式 A 和模式 B 的选择节点分别具有子节点$\{a_1, a_2, \cdots, a_n\}$和$\{b_1, b_2, \cdots, b_m\}$，未变形前模式之间的距离是 $\min \sum \{a_1 b_{j_1}, \cdots, a_n b_{j_n}, \cdots, \Lambda b_{j_m}\}$，变形后是 $\min \sum \{a_1 \cdots a_n \cdots \Lambda, b_{j_1} \cdots b_{j_n} \cdots b_{j_m}\}$，这二者是等价的.

(3) 对于多选节点，由于数据类型实例中子节点出现次数是未知的，采用的操作同顺序节点.

至此操作子树变形完成，可以对其使用经典树编辑距离算法来计算操作间相似度. 但是出于应用对象的特殊性，还需要对算法进行部分改进.

2.3 操作的无序标签树模型

下文均针对输入子树来分析和建立模型，输出子树的情况类似，不再赘述.

如图 2，此时分析的对象为以 Operation Input 作为根节点的树，其下一级为并列的 Data Type，它们是没有顺序的. 而 Data Type 模式树本身可能是有序的，这种特性体现在⟨*xsd*:*sequence*⟩约束节点上. 经过分析，在上述变形的约束节点消去原则中消去顺序节点，对寻找相近的操作是有利的，只要能在树匹配的同时提供一种机制(如设计一种 Mediator)，使两棵树的子节点对应顺序得以知晓即可. 例如同为查询天气预报的服务，服务 1 的输入有一个 Data Type，下有 countryCode，state，city 3 个子节点，而服务 2 的输入也是一个 Data Type，有 city，state，countryCode 3 个子节点，可知节点次序的迥异不会影响 Data Type 结构的类似. 同样，由于数据类型不同造成的匹配差异也交给 Mediator 处理. 将 Data Type 模式树建模为无序标签树的还有一个原因是，除去⟨*xsd*:*sequence*⟩约束节点，其它约束节点下的子节点都是无序的. 经过第 4 节对比实验的验证，选择无序树进行建模比有序树的确能有效地提高 top-*k* 查准率.

3 改进的编辑距离算法

3.1 满足约束的编辑距离映射

定义 1. 编辑操作[1]. 一棵树可以通过如下 3 种编辑操作变为另一棵树，分别是插入、删除和改写操作. 其中插入操作为指定父节点插入一个子节点，并将部分该父节点的原子节点转移为插入子节点的子节点；删除操作使某节点的子节点变为其父节点的子节点，然后删去该节点；改写操作为更改一个节点的标签.

定义 2. 满足约束的编辑距离映射[2]. 假设 $t[i]$是树 T 上的第 i 个节点，$T[i]$是以 $t[i]$为根节点的子树，$F[i]$是从 $T[i]$中删去 $t[i]$获得的无序的森林. 定义三元组(M, T_1, T_2)为从 T_1 到 T_2 的满足约束的编辑距离映射，其中 M 是有序整数对(i, j)的集合，满足如下条件：

(1) M 是一个编辑距离映射[2]；

(2) 对于 M 中的任何 3 对(i_1, j_1)，(i_2, j_2)和(i_3, j_3)，令 $t_1[I]$为 $t_1[i_1]$与 $t_1[i_2]$的最小共同祖先，$t_2[J]$为 $t_2[j_1]$与 $t_2[j_2]$的最小共同祖先，则 $t_1[I]$是 $t_1[i_3]$的真祖先(即不包括本身)当且仅当 $t_2[J]$是 $t_2[j_3]$的真祖先.

为了下文需要在此明确源(source)和目标(object)的概念. 当为某一个操作寻找语义和结构上最为相似的操作时，将前者称为源，后者称为目标.

定理 1. 源的 Data Type 子树到目标的 Data Type 子树的编辑距离映射是满足约束的编辑距离映射.

证明. 在操作输入子树匹配的场景下，源的 Data Type 子树到目标的 Data Type 子树的映射满足定义 2：考虑源的两棵独立的子树，它们只会与目

标的两棵独立的子树彼此映射，而不会进行交叉映射，指同一棵源的子树上的节点映射到两棵目标子树的节点，或来自两棵源的子树的节点映射到同一棵目标子树的节点上.

根据定理 1 和 2.3 节的模型，本文将使用无序标签树的满足约束的编辑距离映射来定义操作输入子树之间的映射，并借鉴文献[2]的多项式时间复杂度的动态规划算法对该映射的编辑距离进行计算. 以下两小节详述本文研究问题的特点，以及如何在该算法中体现.

3.2　操作匹配中的不对称代价模型

如图 4 的匹配实例. 从在线的 Web 服务描述文档中摘取两个 operation 分别叫做 GetLatestStockDailyValue 和 GetStockDailyValue，它们各含有一个同名的 Data Type 作为输入. 图 4 显示了这两个 Data Type 的描述片段. 顾名思义，GetLatestStockDailyValue 的功能是返回某只股票的最近的日值，输入的 Data Type 包含 Stock Exchange 和 Stock，输出的 Data Type 包含最近的日值；而 GetStockDailyValue 的功能是返回某只股票在某一天的日值，输入的 Data Type 包含 Stock Exchange，Stock 和 Date，输出的 Data Type 包含当日的日值. 直观地想象这两个操作的匹配结果. GetStockDailyValue 操作可以成为 GetLatestStockDailyValue 的替代操作，只需要在输入日期时输入最近的日期即可. 而

```
〈s: element name="GetLatestStockDailyValue"〉
    〈s: complexType〉
     〈s: sequence〉
      〈s: element minOccurs="1"
maxOccurs="2" name="strStockExchange"
type="s: string"/〉
      〈s: element minOccurs="1" maxOccurs="1"
       name="strStock" type="s:string"/〉
     〈/s: sequence〉
    〈/s: complexType〉
   〈/s: element〉

〈s: element name="GetStockDailyValue"〉
    〈s: complexType〉
     〈s: sequence〉
      〈s: element minOccurs="1" maxOccurs="2"
       name="strStockExchange" type="s: string"/〉
      〈s: element minOccurs="1" maxOccurs="1"
       name="strStock" type="s: string"/〉
      〈s: element minOccurs="1" maxOccurs="1"
       name="strDate" type="s: string"/〉
     〈/s: sequence〉
    〈/s: complexType〉
   〈/s: element〉
```

图 4　不对称 Data Type 实例

GetLatestStockDailyValue 却无法完全替代 GetStockDailyValue 操作，原因显而易见. 这个例子揭示了在操作匹配中存在的不对称性. 这也是前文强调要区分源和目标的主要原因.

定理 2.　源树到目标树的编辑距离≠目标树到源树的编辑距离.

但是文献[2]提供的编辑距离算法只支持对称的树匹配. 为了使之支持操作匹配的不对称性，本文对该算法进行了改进.

定义 3.　编辑距离映射的代价[1]. 令 Λ 为空节点，I 和 J 表示 T_1 和 T_2 的节点集合，集合内部的节点都未被包含在映射 M 中. 则 M 的代价定义为

$$\gamma(M)=\sum_{(i,j)\in M}\gamma(t_1[i]\rightarrow t_2[j])+\sum_{i\in I}\gamma(t_1[i]\rightarrow\Lambda)+\sum_{j\in J}\gamma(\Lambda\rightarrow t_2[j])\quad(2)$$

定理 3.　两棵树之间的编辑距离等于最小的编辑距离映射的代价.

即 $Dist(T_1,T_2)=\min\gamma(M)$. 此定理也适用于满足约束的编辑距离映射.

因此为了计算体现不对称性的编辑距离，需要修改与编辑距离映射的代价相关的代价模型. 文献[1-2]的代价模型比较简易，即三种单位操作的代价都是 1.

$$\gamma(node_1\rightarrow node_2)=1,\ node_{1,2}\in\{\Lambda,T_1,T_2\}\quad(3)$$

由于插入和删除互为逆操作，式(3)中为互逆操作赋予的代价是相等的，所以这种代价模型是对称的. 下面提出一种具支持匹配不对称性的代价模型. 这种代价模型依据输入和输出分为两种情况，在每种情况下互逆操作的代价是不相等的.

支持匹配不对称性的代价模型形式化如下：

(1) 输入子树匹配的情况

$$\gamma(\Lambda\rightarrow node_2)=1,\ \gamma(node_1\rightarrow\Lambda)=0,$$
$$\gamma(node_1\rightarrow node_2)=Diff_{\mathrm{sem}}(node_1,node_2)\quad(4)$$

(2) 输出子树匹配的情况

$$\gamma(\Lambda\rightarrow node_2)=0,\ \gamma(node_1\rightarrow\Lambda)=1,$$
$$\gamma(node_1\rightarrow node_2)=Diff_{\mathrm{sem}}(node_1,node_2)\quad(5)$$

代价模型说明如下：

当匹配输入子树时，进行节点插入操作的代价为 1，而进行节点删除操作则无需代价. 这是因为假设在输出结果同等的情况下，目标操作比源操作要求的输入内容少，那无需代价就可以完成匹配.

当匹配输出子树时，进行节点删除操作的代价为 1，而进行节点插入操作则无需代价. 这是因为假设在输入参数同等的情况下，目标操作比源操作要

求的输出内容多，那无碍于使用目标操作来替代源操作并且不必付出操作代价.

另外在进行改写操作时，将代价由 1 改为两个标签名之间的语义距离. 例如 GetValueByName 与 GetValueByID 之间的改写代价应当小于 GetValueByName 与 GetZipByState. 下一节将详细介绍如何计算语义距离.

3.3 引入相似系数的语义距离计算

下面介绍如何计算两个标签名之间的语义距离 $Diff_{\mathrm{sem}}(node_1, node_2)$. 一个标签名是一个短语，或者短语的集合(2.2 节，消去约束节点后将其子节点的标签名合并为新节点的标签). 度量标签名之间的距离的步骤如下：

(1) 将标签名除去停词后分割为单词.

(2) 将单词当作一棵每个字母是一个节点的树. 其根节点是单词的首字母，树的层数等于单词的字母数，单词与单词之间的编辑距离就成为树与树编辑距离的一个特例，使用树编辑距离算法计算.

(3) 当单词之间的距离已知后，需要寻找一种单词的配对关系，使得标签与标签之间的距离最小.

将此抽象为一个最小费用最大流的问题，即在考虑节点网络上流量最大的同时，还要使得所安排流量的费用或者代价达到最小. 以计算 LatestDailyValue 与 DailyValue 的距离为例，构造图 $G=(V, E)$ 如图 5.

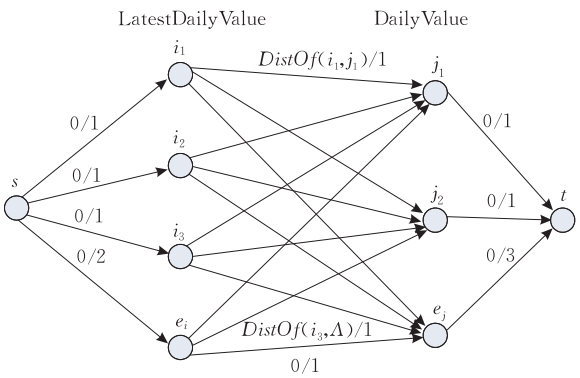

图 5 标签语义距离问题的最小费用最大流网络

顶点集合：$V=\{s, t, e_i, e_j\} \cup \{i_1, i_2, i_3\} \cup \{j_1, j_2\}$，其中 $i_1=Latest$，$i_2=Daily$，$i_3=Value$，$j_1=Daily$，$j_2=Value$. s 为网络流量出发点，t 为终点，e_i，e_j 是占位符，代表空字符串.

边集合：$[s, i_k]$，$[s, e_i]$，$[j_l, t]$，$[e_j, t]$ 的费用为 0，$[i_k, j_l]$ 的费用为 $DistOf(i_k, j_l)$，$[e_i, j_l]$ 的费用为 $DistOf(\Lambda, j_l)$，$[i_k, e_j]$ 的费用为 $DistOf(i_k, \Lambda)$，$[e_i, e_j]$ 的费用为 0. $[s, e_i]$ 的容量为 $|J|=2$，$[e_j, t]$ 的容量为 $|I|=3$，$[e_i, e_j]$ 的容量为 $\max\{|I|, |J|\}-\min\{|J|, |I|\}=1$，其它边的容量均为 1.

构造完成以后，采用弧标号法来计算网络的最小费用. 步骤如下：

1. 将所有路按费用从小到大排序. 每条弧的两端各标上数字，前端为 a=该弧容量，后端为 $b=0$.

2. 从出发点到终点选定一条费用最小的非饱和路，即这条路的每一段弧的前端数字都大于 0 且费用最小，如果找不到这样的路，说明已经求得最小费用最大流.

3. 在该选定的路上，前端数字中最小的定为 c.

4. 对选定的路上每段弧的前端数字 a 修改为 $a-c$，后端数字 b 修改为 $b+c$. 转步 2.

但仅仅使用这样的计算方法无法体现部分相似字符串对整个字符串的相似度造成的影响，即如果两个标签中存在至少一个相同的单词，那它们之间的距离应该大大降低. 例如用标号法计算标签 Value 和 Name 的距离为 3，标签 StockValue 和 StockName 的距离也为 3，而实际上后两者之间的距离应该是远远小于前两者的. 为了充分利用部分相似结构对整体相似度的影响，下面引入相似系数的概念.

定义 4. 相似系数 s. 在计算最小费用最大流网络的费用时，如果存在 n 对 $[i_k, j_l]$，使得 $DistOf(i_k, j_l)=0$，则此网络的最终费用=原费用$\times s^n$，其中 $0<s<1$.

随着相似子结构的增多，整体的相似度呈指数级下降. 例如若取相似系数为 0.1，则 StockValue 和 StockName 的距离为 0.3，大大低于 Value 和 Name 的距离之间的距离. 同样的情况也发生在树编辑距离的计算中. 具有相似子树结构的树之间的距离应当大大减小. 事实上，在语义距离和树距离计算中同时使用相似系数还有一个原因，就是使得 2.2 节中的约束节点消去原则(2)仍然是成立的.

4 实验结果与评价

为了说明本文提出的改进算法的有效性设计了两组对比实验. 由于 Web 服务发现领域尚无明确的指标体系，本文沿袭使用了信息检索领域常用的检索指标，如查全率、查准率、top-k 查准率等. 实验环境为 PIV 2.4GHz，内存 DDR333 1.5GB 的 Windows PC，实验数据集是来自 xmethods① 符合规范的 WSDL 文档，使用开源 API wsdl4j 对其进行解析，抽取需要的 XML 模式和其它数据项. 因为网上

① Xmethods. http://www.xmethods.com/

提供的在线免费 Web 服务有限，本文一共挑选了 4 个所使用的术语交集比较小的类别，其中每类各 8 个服务做实验，类别分别简称为 zip，currency，SMS 和 stock. 这 32 个服务一共包含 190 个操作，除去如 Login，Logout 等通用服务以后，保留 180 个操作进行下一步实验.

在数据预处理阶段，从每类服务中挑选一个作为中心向量，使用简单向量距离分类法将服务做粗分类. 根据事先手动标记的类别，VSM 分类得到的结果如图 6. 图中使用的指标是查全率和查准率，而且这两个指标都在 80%，87.5% 和 100% 这 3 个数值上. 由此可知用 VSM 对服务进行粗分类是十分有效的.

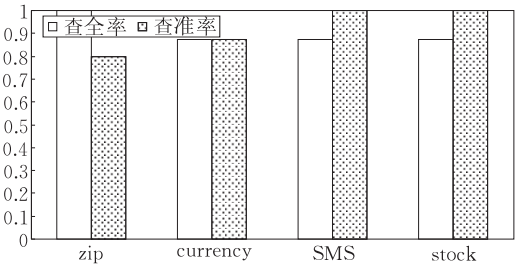

图 6　对服务使用 VSM 模型进行粗分类的结果

预处理完成后，为了比较起见，采用如下几种方法计算距离：

(1) MOSM 模型(MOSM)；

(2) 操作向量的 VSM 模型(OperVSM)；

(3) 有序树编辑距离映射模型(Ordered)；

(4) 叶子节点计算模型(Leaves)；

(5) 路径标签名计算模型(NPath)；

(6) MOSM 的有序树编辑距离映射模型(M-O)；

(7) MOSM 模型除去对不对称性的支持(Symmetry)；

(8) MOSM 模型除去对相似系数的支持(M-SW)；

以上的实验设计包含了两组对比：一组使用 Schema 混合匹配方法的各种子类型(见第 5 节表 1)进行匹配(方法 1,2,3,4,5)，这组模型是根据表 1 中的 5 个子类挑选的. 其中 MOSM 和 Ordered 模型属于 Children 类，OperVSM 属于 Name 类，Leaves 和 NPath 模型分别属于 Leaves 类和 NamePath 类，因为拟将数据类型的差异交与 Mediator 处理，所以此处不讨论 TypeName 类型；另一组是 MOSM 的各个特征做出的贡献比较(方法 1,6,7,8). 将对比结果放在同一个图 7 中，图中使用的指标是 top-k 查准率，k 分别等于 2 和 5，图中将各个算法按指标由高到低从左至右排序.

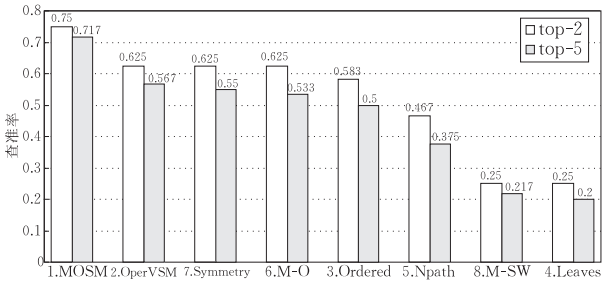

图 7　对比实验的 top-k 查准率柱状图

其中方法 2 是抽取每个操作的操作名、操作说明以及操作调用的 Data Type 名来构成操作向量，然后对其使用 VSM 模型的做法；方法 3 使用有序树对操作树进行建模；方法 4 抽取模式树的所有叶子节点进行匹配；方法 5 用路径名来表示叶子节点后进行匹配. 在本文第 2 节和第 3 节中提到 MOSM 具有如下 3 个特征：(1) 使用满足约束的无序树模型；(2) 支持不对称性的代价模型；(3) 为树结构匹配和标签匹配引入相似系数. 方法 6,7,8 分别在使用 MOSM 模型的基础上，单项地除去这些特征，以体现各个特征为提高 top-k 查准率做出的贡献.

第 1 组.　MOSM 和各种模式匹配方法的比较.

如图 7 方法 1,2,3,4,5 的 top-k 查准率可以得出：

(1) 采用无序树建模的 MOSM 模型，比有序树 Ordered 模型将 top-2 查准率提高了 28.6%，将 top-5 查准率提高了 43%. 这印证了 2.3 节中无序树建模合理性的假设.

(2) MOSM 充分利用了 XML Schema 的结构信息，与之相对，OperVSM 模型因为利用了操作的描述信息，在语义信息方面使用更充分而忽略了结构信息. MOSM 模型比 OperVSM 模型，将 top-2 查准率提高了 20%，将 top-5 查准率提高了 26.5%. 这说明了充分利用结构信息的重要性.

(3) 因为引入了路径信息，NPath 模型比 Leaves 模型的 top-2 和 top-5 查准率分别高了 86.8% 和 87.5%. 这与 WSDL 的应用背景有关，操作名和 Data Type 名处于操作树的根部和内部，引入路径信息不会忽略这些重要信息. 但 NPath 模型在计算层次比较深的节点时计算效率很低，而且由于许多节点部分路径重复带来的重复计算量也很大.

第 2 组.　MOSM 各个特征的贡献比较.

方法 1,6,7,8 进行比较.

(1) MOSM 模型比 Symmetry 模型的 top-2 和 top-5 查准率分别提高了 20%和 30.4%,这是由于不对称性代价模型增加了操作匹配的概率.针对的是两个操作可能不完全一样,但是其中之一可以替代另一个的情况.

(2) MOSM 模型比 M-O 模型的 top-2 和 top-5 查准率分别提高了 20.5%和 34.5%,这是由于满足约束的无序树模型增加了操作匹配的概率,针对的是两个 Data Type 可能参数顺序有异,但是调整后可以互相匹配的情况.

(3) 与其它模型的 top-k 查准率相比,MOSM 对相似系数的引入对结果的改善最明显,而无序树模型的引入和不对称代价模型的引入则较低.这可能与选取相似候选服务的标准中语义所占的权重较大有关.

值得一提的是,实验中进行数据预处理的做法对 top-k 查准率的影响很小,即如果不进行服务的粗分类而直接使用以上各种算法,在运行效率大幅下降的同时,结果并不会有明显的提高.

5 相关研究

基于 Schema 匹配的问题研究由来已久.文献[3]从统一的角度综述了基于模式和本体的匹配方法,将已有的各种 Schema 匹配方法从各个角度进行了分类,如模糊与精确匹配技术;元素和结构层次上的句法与语义匹配等,并将基于字符串匹配的技术简要分为前缀匹配、后缀匹配、编辑距离计算和 N-gram 计算. MOSM 采用的是基于编辑距离计算的方法,而本文的工作之一则是在此基础上引入相似系数的概念,能有效利用部分结构相似性更好地衡量语义距离.

文献[4]将 Schema 匹配简要分为简单匹配、混合匹配和基于重用的匹配,其中混合匹配又分为 Name, NamePath, TypeName, Children 和 Leaves 5 个子类,分别通过标签名、标签路径名、标签名加数据类型、所有子节点以及叶子节点的匹配来完成 Schema 匹配.标签名法将简单匹配进行简单组合叠加(如线性叠加),标签路径名法使用自根节点开始的节点序列给原始标签加注,有利于甄别标签的上下文语境并可进行不同层次的匹配,标签名加数据类型法将数据类型的差异加入了计算,所有子节点法自底向上使用递推的方法计算包含所有子节点在内的相似性,而叶子节点法则只计算叶子节点,不考虑其它节点的影响.

文献[4-5]是研究通用Schema匹配的,文献[6-8]则是 Schema 匹配在 Web 服务匹配和 Web 服务相似性度量中的应用.表 1 按文献[4]的分类法将它们归纳入混合匹配的 5 个子类.其中文献[6]假设出现在同一个标签的单词具有高度相关的语义,通过叠加标签名之间的语义距离来计算 Schema 之间的距离.文献[7]根据子结构之间的 model type 来调整子结构在匹配中所占的权重系数,是同时利用了数据类型和标签名的匹配方法.文献[8]使用有序标签树来为服务操作建模,是使用所有子节点来进行匹配的方法.MOSM 也属于所有子节点法.这种方法的优点是利用的结构信息比较完备,缺点是计算量相对较大.但是由于 Web 服务的相似性度量可以离线完成,而在运行时返回结果集即可,因此是可行的.

但是模型[6-8]都没有考虑操作匹配存在的不对称性.文献[9]的主要贡献是提出了操作匹配存在的不对称性,但是它主要是针对使用概念构建的基于本体语义的操作,通过计算不同操作共享的概念范围来进行匹配.文献[10]同样也关注了这种语义概念层次的匹配不对称性,使用语义覆盖率的概念来进行阐述.然而它们都没有关注结构层次提供的信息,也没有将这种特性引入到 Schema 结构匹配中.本文的主要工作即设计一种合适的模型和算法,将这种不对称特性引入到结构层次的 Schema 匹配中来合理增加匹配概率,从而增加服务重用和增值的能力.

表 1 Schema 混合匹配方法的 5 个子类及例子

匹配类型	匹配项	Schema 信息	例子
Hybrid	Name	Element names	Woogle[6]
	NamePath	Names+Paths	COMA[4]
	TypeName	Data Types+Names	FIM[7]
	Children	Child elements	文献[8],MOSM
	Leaves	Leaf elements	Cupid[5]

6 结 论

近年来 Web 服务和服务计算的概念逐步受到重视,与 Web 服务描述、发现、组合等方面相关的研究也非常的活跃,很多模型层次的研究有了长足的进展.本文旨在介绍一个工程应用中的具体方法,为工程方法和核心技术的深入研究提供参考.本文的主要内容是为基于 WSDL 描述的服务操作提供一

种基于语义和结构相结合的相似性度量方法，与其它在此背景下基于 Schema 匹配的模型相比均能获得更好的查询指标.

在未来的工作中，将会把较为成熟的语义技术应用到该度量方法中. 例如针对具体应用，使用领域本体来对 WSDL 的标签进行语义标注，进一步精确化标签的语义相似度；提供 Web 服务匹配与异构数据查询之间的互相转化，以及形成可以根据历史数据反馈进行参数自调整的服务匹配机制等. 相关的研究成果将会被应用在国家大型科学仪器设备资源的建设与整合中，具有实际的工程意义.

参　考　文　献

[1] Zhang Kai-Zhong, Shasha D. Simple fast algorithms for the editing distance between trees and related problems. SIAM Journal of Computing, 1989, 18(6): 1245-1262

[2] Zhang Kai-Zhong. A constrained edit distance between unordered labeled trees. Algorithmica, 1996, 15(3): 205-222

[3] Shvaiko P, Euzenat J. A survey of schema-based matching approaches. Journal on Data Semantics IV, 2005, LNCS 3730: 146-171

[4] Do H H, Rahm E. COMA—A system for flexible combination of schema matching approaches//Proceedings of the 27th VLDB Conference. Roma, Italy, 2001: 610-621

[5] Madhavan J, Bernstein P, Rahm E. Generic schema matching with cupid//Proceedings of the 27th VLDB Conference. Roma, Italy, 2001: 49-58

[6] Xin Dong et al. Similarity search for Web services//Proceedings of the 30th VLDB Conference. Toronto, Canada, 2004: 372-383

[7] Wang Yi-Qiao, Stroulia E. Flexible interface matching for Web-service discovery//Proceedings of the 4th International Conference on Web Information Systems Engineering. Roma, Italy, 2003: 147- 156

[8] Hao Yanan, Zhang Yan-Chun. Web services discovery based on schema matching//Proceedings of the 30th Australasian conference on Computer Science 2007. Ballarat, Australia, 2007: 107-113

[9] Cardoso J. Discovering semantic web services with and without a common ontology commitment//Proceedings of the IEEE Services Computing Workshops 2006. Chicago, USA, 2006: 183-190

[10] Gunay A, Yolum P. Structural and semantic similarity metrics for Web service matchmaking. E-Commerce and Web Technologies, 2007, LNCS 4655: 129-138

HE Ling-Juan, born in 1982, Ph. D. candidate. Her research interests include service discovery and matching.

LIU Lian-Chen, born in 1971, Ph. D. , associate professor. His research interests include grid application and service computing.

WU Cheng, born in 1940, professor, Ph. D. supervisor. His research interests include CIMS and complex systems.

Background

Web services technique plays an important role in e-industry. Since Web services are provided by different organizations, it is necessary to understand the different service compatibilities in order to select the appropriate function or compose multiple functions for the related service request. A common technique for assessing service function compatibility is the structural matching of related operations. To utilize the underlying structural information in the description documents, the authors propose the approach MOSM based on structural schema matching with the asymmetric characteristic consideration.

In 2002, a report from China Ministry of Education indicated that most of the research equipments were only being used 7 hours per week, meaning that research equipments were heavily underutilized. The 211 project committee did an extensive study on the equipment shortage problem, suggested the government to focus on improving the accessibilities of the underutilized equipments. As the popular solution of resource integration, Web service related issues have been studying in the team. Besides Web service discovery, the team has also made much progress in Web service composition and online negotiation. The authors' current work is focusing on software implementation to integrate the service provided by different parties.

This work has been supported fully under PRC government grant (2005DKA10100).