基于文本聚类和概念相似度的语义Web服务发现

刘一松 杨玉成
(江苏大学计算机科学与通信工程学院 镇江 212013)

摘 要 语义Web服务在进行服务发现时,需要按顺序依次匹配注册库中的服务,这将大量时间浪费在不相干的服务上,从而造成服务发现效率低下。针对该问题,提出了一种新的基于文本聚类和概念相似度的语义Web服务发现方法。该方法主要分为两个阶段,第一阶段根据服务源文件中的描述性文本信息将类别一致的服务聚类到一起,在此过程中利用了向量空间模型对文本进行表示和处理,并在前人的基础上提出了一种多重混合聚类算法MHC;第二阶段进行服务间的功能属性匹配,结合本体概念层次树中有向边的深度、强度以及概念的继承度等因素计算概念间的语义相似度。最后,实验结果表明,提出的方法在兼顾匹配准确率的基础上,大大提高了匹配效率。

关键词 语义Web服务,服务发现,文本聚类,本体,语义相似度
中图法分类号 TP393 **文献标识码** A

Semantic Web Service Discovery Based on Text Clustering and Similarity of Concepts

LIU Yi-song YANG Yu-cheng
(School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang 212013, China)

Abstract Semantic Web Services need to match services in the registry in succession in the discovery of services, which wastes a lot of time on irrelevant services, and reduces efficiency of discovery. Thus, a new discovery method of semantic Web service based on text clustering and similarity of concepts was proposed which can be divided into two phases: in the first phase, services of identical category are clustered according to descriptive texts in the service source file when texts are expressed and processed by vector space modal (VSM) and a multiple hybrid clustering algorithm MHC was proposed in the second phase, functions and properties between services are matched and semantic similarity between concepts is calculated combined with factors such as depth, strength and inheritance of directed edge in the hierarchical tree of ontology concepts. Finally, the experimental result shows that the method proposed in the article improves the matching efficiency greatly based on accurate rate.

Keywords Semantic Web services, Web service discovery, Text clustering, Ontology, Semantic similarity

Web服务是一种面向服务架构的技术,通过标准的Web协议提供服务,保证了异构平台上应用程序之间的互操作。随着Web服务的快速发展,如何准确而又高效地寻找到满足用户需求的Web服务已成为当前研究的重点之一。语义Web服务由于采用本体概念来描述服务的功能属性,因而较传统的基于UDDI服务的发现机制在匹配准确率上有很大程度的提高。文献[1]通过构建Web服务本体,提出了一种基于词汇语义的Web服务相似度度量方法,服务发现效果较基于关键词的服务发现方法有明显的改善。文献[2]对传统的OWL-S/UDDI匹配算法进行了改进,使之能够精确计算请求服务和发布服务间的匹配相似度。文献[7]提出了一种混合语义和词法的匹配方法,其实验结果证明基于语义的匹配准确率要比基于词法的匹配准确率高得多。然而它们在进行服务发现时,依然是按照顺序和注册库中的服务进行一一匹配,这就浪费了大量时间在不相干的服务上,严重地影响了服务的发现效率。

通过对注册库中的服务进行聚类处理,避免请求服务和无关服务进行匹配,可以有效地降低服务搜索空间,从而达到高效发现服务的目的。文献[4]从服务的功能相似和过程相似两个层面对服务进行聚类预处理,将能够实现相似功能的服务聚合在一起。文献[5]对WSDL中描述功能操作的信息进行处理,然后按照功能相似进行聚类得到候选服务集。文献[8]利用领域本体及其形式化概念,并基于最小生成树算法对服务进行聚类,简化了聚类计算,提高了聚类结果的准确性。但是它们都是侧重于对服务的功能接口进行聚类,而忽略了一些重要的非功能属性,比如服务的名称和服务的描述信息等。

为了充分利用OWL-S本体中的文本描述信息,本文提出了一种新的基于文本聚类和概念相似度的语义Web服务发现方法。该方法主要分为两个阶段,第一阶段根据服务的

到稿日期:2013-01-16 返修日期:2013-03-26 本文受江苏大学高级专业人才科研启动基金项目(10JDG063),江苏省社会发展计划(BS2001046),江苏省高校自然科学研究计划(03kjd520075)资助。
刘一松(1966—),男,博士,教授,主要研究领域为分布式人工智能、虚拟智能主体、语义网等;杨玉成(1987—),男,硕士生,主要研究领域为语义网、语义Web服务等,E-mail:darknoll@126.com。

描述性文本信息将类别一致的服务聚类到一起，在该阶段利用了向量空间模型对文本进行表示和处理，并在文献[6]的基础上提出了一种多重混合聚类算法 MHC；第二阶段进行服务间的功能属性匹配，结合本体概念层次树中有向边的深度、强度以及概念的继承度等因素来计算概念间的语义相似度。

1 基于文本聚类和概念相似度的发现框架

本文所研究的语义 Web 服务基于 OWL-S 本体语言描述，OWL-S 是由 W3C 推荐的 Web 服务描述语言，其中 ServiceProfile 用于说明服务的用途，是服务发现的主要研究对象。例 1 是公共测试数据集 OWLS-TC4[10] 的 food 领域中的某个服务源文件的 ServiceProfile 部分的信息。可以发现它除了包含输入输出等功能属性，还有 serviceName 和 textDescription 这两个非功能属性，它们一个用于标识服务名称，一个用于描述服务的用途，通过这些文本信息可以了解到该服务的主要用途和所属领域。

例 1　某 OWL-S 源文件中 ServiceProfile 示例

```
……
〈profile:Profile rdf:ID="GROCERYSTORE_BUTTERQUANTITY_PROFILE"〉
〈service:isPresentedBy rdf:resource="#GROCERYSTORE_BUTTERQUANTITY_SERVICE"/〉
〈profile:serviceName xml:lang="en"〉
GroceryStore Butter Quantity Service
〈/profile:serviceName〉
〈profile:textDescription xml:lang="en"〉
This service informs about butter items that are available in a given grocery store and its available quantity
〈/profile:textDescription〉
〈profile:hasInput rdf:resource="#_GROCERYSTORE"/〉
〈profile:hasOutput rdf:resource="#_BUTTER"/〉
〈profile:hasOutput rdf:resource="#_QUANTITY"/〉
……
```

定义 1(服务描述信息 SDID)　服务信息描述集是一个文档集合，其中每个文档和服务注册库中的一个语义 Web 服务相对应，这些文档包括两个部分：一是服务名称，即标签〈profile:serviceName〉中的内容；二是服务描述信息，即标签〈profile:textDescription〉中的内容。

如例 1 的 SDID 可以表示为：(“GroceryStore Butter Quantity Service”，“This service informs about butter items that are available in a given grocery store and its available quantity.”)。

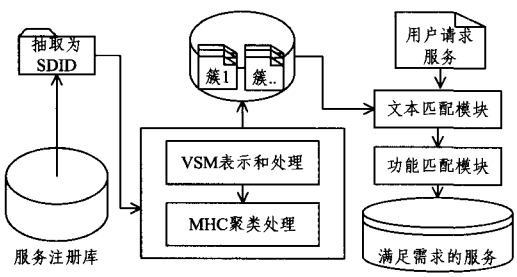

图 1　基于文本聚类和概念相似度的语义 Web 服务发现框架

本文提出的基于文本聚类和概念相似度的语义 Web 服务发现过程如图 1 所示：首先抽取服务注册库中各个服务源文件非功能属性信息组成 SDID，然后用 VSM 表示和处理这些文本信息，并使用提出的 MHC 聚类算法将类别或者所属领域一致的服务聚类到一起，这样请求服务便只需和各簇的代表服务进行匹配，将匹配结果降序排列，只选取相似度比较靠前的几个服务簇进入并进行下一步的功能匹配，最终得到满足用户需求的语义 Web 服务。

2 基于文本描述信息的语义 Web 服务聚类

2.1 SDID 的表示以及文档间相似度计算

向量空间模型是当前最常用的检索模型之一，它把文档集中的各文档表示成多维空间中的向量，这样可以很方便地计算文档间的相似度。首先对 SDID 中各文档进行分词、词根还原、去除停用词等预处理操作，得到 SDID 的初始特征项。

TF-IDF 是经典的特征项权值计算公式，TF_{ij} 是词频，表明特征项 T_j 在文档 D_i 中出现的次数；IDF_j 是逆文档频率，表示特征项 T_j 在整个文档集中的分布情况，$TF_{ij} * IDF_j$ 就是特征项 T_j 在文档 D_i 中的权值。为了避免某些特征项词频很高从而造成其他特征项被淹没的现象，定义特征项的权值计算公式如下：

$$W_{ij}=\frac{(\lg TF_{ij}+1.0)\times IDF_j}{\sum_{j=1}^{i}[(\lg TF_{ij}+1.0)\times IDF_j]^2} \tag{1}$$

向量空间模型默认特征项的权值与其所处位置无关，即在一个文档中，无论特征项位于何处，计算公式都是采用式(1)。然而，在 SDID 中，特征项之间却是有位置关系的，ServiceProfile 中 serviceName 里面的词语是用来标识服务名称的，因而要比 textDescription 更加重要。本文提出一种基于位置的加权权值计算方法，首先分别计算特征项在 servieName 和 textDescription 中的权值，然后根据加权计算得到特征项的最终权值，计算公式如下：

$$W_{ij}=\alpha W_{ijN}+(1-\alpha)W_{ijD} \tag{2}$$

式中，W_{ijN} 是特征项 T_j 在 SDID 中基于 serviceName 位置的权值，即只计算 T_j 在 SDID 的 serviceName 中的权值，并不包括 textDescription 里的内容。α 是调整因子，考虑到名称标签中的词项更加重要，令 $\alpha=0.6$。

接下来进行 SDID 中各文档之间的相似度计算，使用向量的余弦夹角计算文档间相似度，计算公式如下：

$$\mathrm{Sim}(D_x,D_y)=\frac{\sum_{j=1}^{i}W_{xj}W_{yj}}{\sqrt{\sum_{j=1}^{i}(W_{xj})^2\sum_{j=1}^{i}(W_{yj})^2}} \tag{3}$$

特征值的计算完成可以得到 SDID 的数据矩阵 A，相似度的计算完成可以得到相似度矩阵 B，这些都为下一步要进行的聚类操作提供了数据基础。

2.2 SDID 中文档的聚类

凝聚层次算法和 K-Means 算法是聚类算法中的两个经典算法，凝聚层次算法的优点是生成簇的效果较好，而时间复杂度却较高。K-Means 算法的优点是收敛速度快，然而需要预先指定初始聚类中心和生成簇个数。通过将这两种算法有机结合，可以有效减少凝聚聚类算法所需的时间以及解决 K-

Means 需指定初值的问题。

文献[6]提出了一种结合这两种方法的混合聚类算法 HCAP，它将整个聚类过程分为 4 个阶段，第一阶段使用凝聚层次聚类，定义一个相似度阈值来控制生成的初始子簇，第二阶段使用 K-Means 算法对这些子簇进行调整，第三阶段将相似度阈值调高，使生成簇减少，最后再次用 K-Means 算法进行调整。HCAP 算法虽然结合了两种聚类算法的优点，但却存在如下不足：(1) HCAP 定义了质心是簇中各数据点的均值，由于簇之间是不断进行合并的，因此质心也在不断变化，数据点的向量维数若比较高会影响质心的计算，降低聚类过程的速度。(2) HCAP 未能充分利用 K-Means 算法将局部最优转换为全局最优，只是在中间和最后一步调整了，中间大部分过程仍然是单独的凝聚层次聚类。

针对以上不足，本文提出了一种改进的多重混合聚类算法 MHC，用于对 SDID 中的文档进行聚类处理。首先给出几个定义：

定义 2(簇间距离) 簇 C_x 和 C_y 之间的距离：

$$D(C_x,C_y)=\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}d(x_i,y_j)}{m\times n} \tag{4}$$

式中，x_i 是簇 C_x 中的数据点，y_j 是簇 C_y 中的数据点，m 是 C_x 中数据点个数，n 是 C_y 中数据点个数。该公式表明了两个簇的距离是两个簇中所有点两两之和的均值。使用该公式计算簇间距离可以使簇间距离定义更准确，且避免频繁使用数据矩阵来计算质心而影响计算效率。

定义 3(聚类阈值) 聚类阈值：

$$F=\alpha D_{\min}+(1-\alpha)D_{\max} \tag{5}$$

式中，$D_{\min}$ 是 SDID 中所有文档中相似度最小的值，$D_{\max}$ 是 SDID 中所有文档中相似度最大的值，α 是调节因子，根据取值的大小可以对阈值进行伸缩处理。

本文所提出的多重混合聚类算法 MHC 如下：

输入：SDID 的数据矩阵 A 和相似度矩阵 B

输出：SDID 的聚类簇集合$\{C_1,C_2,C_3,\cdots,C_m\}$

步骤 1 将 SDID 中每一个文档都看作一个簇，则 SDID=$\{C_1,C_2,\cdots,C_n\}$，其中 n 为 SDID 中文档个数。将相似度矩阵 B 的上三角数据按照升序排序并存放在一个数组 array[]中。从 array[0]开始扫描整个数组，根据当前相似度值找到对应的两个数据点 X_i 和 X_j。若 array[i]≤F，检查对应的 X_i 和 X_j 是否属于同一个簇，若是则不做处理，否则将两个簇合并。依次进行下去直至数组 array[]中的所有值均大于聚类阈值 F。

步骤 2 对步骤 1 中所得到的聚类，分别计算各个聚类的质心；对于 SDID 中各个数据点，分别计算它们到这些聚类质心的距离，将这些点重新分配到距离自己最近的簇中。再次计算各聚类质心以及其他点到各质心的距离，重新分配这些数据点。重复以上步骤直至各簇的质心不再发生变化为止。

步骤 3 对步骤 2 所生成的各簇，用各自质心代替，重新生成数据矩阵 A′和相似度矩阵 B′。定义伸缩因子 S 来控制聚类阈值，首先可以取 S=1.5，此时令聚类阈值 F′=SF。和步骤 1 类似，将 B′的上三角数据按照升序排列存放在数组 array2[]中，从 array2[0]开始扫描，若 array[i]≤SF，则将相关的两个簇合并直至数组 array2[]中，所有数据均大于聚类阈值 SF。

步骤 4 类似于步骤 2，分别计算各聚类的质心，根据 K-Means 原理调整各聚类中的数据点，直至不再变化为止。

步骤 5 类似于步骤 3，调整 S 的值，使其逐渐增大，以得到更少的聚类数，本文令 S+=0.5。

步骤 6 重复步骤 4 和步骤 5，直至得到最后满足用户需求的聚类集合$\{C_1,C_2,\cdots,C_m\}$。

3 语义 Web 服务的功能接口匹配

在 OWL-S 文件中，语义 Web 服务的功能属性主要由 ServiceProfile 中的 IOPE 来定义。本文采用输入输出这两个属性来表征一个语义 Web 服务的功能，则服务 W_i 和服务 W_j 之间的功能相似度可以用式(6)来计算：

$$\mathrm{Sim}(W_i,W_j)=\alpha\mathrm{Sim}_{out}(W_i,W_j)+(1-\alpha)\mathrm{Sim}_{in}(W_i,W_j) \tag{6}$$

式中，Sim_{out} 和 Sim_{in} 分别是两个服务的输出概念的相似度和输入概念的相似度，α 是调整因子，考虑到输入和输出属性对于服务的功能来说是同等重要的，因此取 $\alpha=0.5$。

由于服务的输入输出都是采用本体概念来描述的，因此输入或者输出方面的相似度计算实际上就是本体概念间的语义相似度匹配。假设服务 W_i 和 W_j 的输出接口均用一个本体概念 C_x 和 C_y 来描述，则它们的输出相似度可以表示为：

$$\mathrm{Sim}_{out}(W_i,Wj)=\frac{1}{1+Dist(C_x,C_y)} \tag{7}$$

式中，$Dist(C_x,C_y)$是概念 C_x 和 C_y 在本体概念层次树中的语义距离。关于本体概念的相似度计算问题，文献[9]提出应根据概念的包含关系将概念匹配分为精确匹配、插入匹配、可替代匹配和失败匹配这 4 个等级，然而这种方法只是定性说明了服务间的匹配程度，未能精确计算匹配结果。为了具体得到语义相似度的值，本文充分考虑了本体概念树中有向边的权值，而这些权值与有向边的深度、强度和概念的继承度有关，下面给出相关定义。

定义 4(有向边深度) 有向边 E_{ij} 的深度为 $Dep(E_{ij})=\min(Dep(C_i,C_j))$。其中，$E_{ij}$ 是连接概念 C_i 和 C_j 的有向边，$Dep(C)$为概念 C 的深度，定义根节点深度为 1，$Dep(C)=Dep(Parent(C))+1$。随着有向边的不断加深，概念的分类也就越细致，它们之间的相似度也就越高。

定义 5(有向边强度) 有向边 E_{ij} 的强度为 $Str(E_{ij})=|IC(i)-IC(j)|$。其中，$E_{ij}$ 是连接概念 C_i 和 C_j 的有向边，$IC(C)$为概念 C 的信息量。采用文献[3]的定义，$IC(C)=-\lg_2 P(C)$，$P(C)=P(P)/n$，$P(R)=1$，其中概念 C 为概念 P 的父概念，n 为概念 P 直接子概念的数量，概念 R 为根结点。当一个概念的子结点越密集时，分类也就越细致，因此该概念与子结点的有向边强度就越大。

定义 6(概念继承度) 本体概念层次树中的概念大部分都只有一个直接父类，但也可能存在多继承的情况，若一个概念有 N 个直接父类，则该概念的继承度 $Inh(C)=N$。

如图 2 中的 Jazz 概念就有两个直接父类 Classic 和 Blues，则 Inh(Jazz)=2。若一个概念继承度大于 1，则它与其他概念之间的最短路径就可能有多条，需要分别计算各个路径的语义距离，然后取平均值得到最终的 $Dist(C_1,C_2)$。

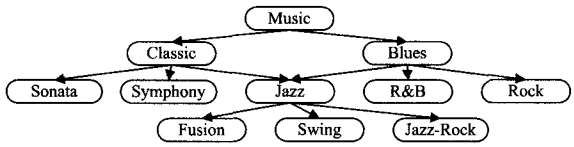

图 2　表示音乐分类的本体片段

综合以上几个因素，则相邻两个概念 C_i 和 C_j 间的有向边的权值 WE_{ij} 可以定义为：

$$WE_{ij}=\frac{1}{Dep(E_{ij})}\times\frac{1}{Str(E_{ij})+1} \tag{8}$$

概念 C_x 和 C_y 之间的语义距离 $Dist(C_x,C_y)$ 的计算过程如下：

(1)若 C_x 和 C_y 的继承度都为 1，则这两个概念间的最短路径有且仅有一条，则

$$Dist(C_x,C_y)=\sum_{i=1}^{m}WE_i \tag{9}$$

式中，m 为连接两概念间最短路径上的边数和，WE_i 为连接两概念的最短路径上的第 i 条边的权值。

(2)若 C_x 和 C_y 的继承度不全为 1，则这两个概念间的最短路径可能不止一条，应该分别计算各条路径的语义距离，最后取均值。如图 2 所示，概念 R&B 和 Jazz-Rock 间的最短路径有两条：{R&B->Blues->Rock->Jazz-Rock}和{R&B->Blues->Jazz->Jazz-Rock}。假设 C_x 和 C_y 之间的最短路径有 n 条，则有：

$$Dist(C_x,C_y)=\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m}WE_i \tag{10}$$

4　实验结果和性能分析

由于目前没有标准的语义 Web 服务测试平台，因此本文利用 JSP 技术设计并实现了一个语义 Web 服务原型发现系统。该系统所使用的计算机 CPU 为 Inter Core2 2.0GHZ，内存为 1GB，操作系统为 Windows XP。该系统的集成开发环境为 MyEclipse 8.5，并使用 Tomcat 作为 Web 容器，Racer 作为逻辑推理机，OWLS-TC4 作为测试数据集。下面从服务的匹配准确率和匹配效率两方面进行测试并和文献[3]进行比较，从而验证提出方法的可行性和有效性。

实验 1(匹配准确率)　从测试数据集 OWLS-TC4 的 Communication、Economy、Education 这 3 个领域分别选取若干服务在服务注册库中发布，对于某个相同的请求服务，本文所提方法和文献[3]所提方法的匹配结果如表 1 所列。从中可以发现，按照本文的方法，由于在进行功能匹配之前需要进行服务领域聚类，因此使得功能匹配的时候匹配空间要比顺序匹配的要小，这就造成了匹配准确率不如文献[3]的情况，但是仍然可以保证很高程度的精度。

表 1　服务匹配准确率的比较

领域	服务数量	服务匹配数量/个		
		文献[3]	本文	人为评价
Communication	59	47	43	51
Economy	395	322	307	335
Education	286	241	226	267

实验 2(匹配效率)　从测试数据集 OWLS-TC4 的各个不同领域中选取若干个服务在服务注册库中发布，对于若干个相同的请求服务，本文所提方法和文献[3]所提方法的匹配时间如表 2 所列。由此可以发现，当服务注册库中充满了来自各个不同领域的服务时，文献[3]的匹配效率会变得很低，而本文所提出的匹配方法由于提前对服务库中的服务进行了聚类预处理，因此大大提高了匹配效率，减少了匹配所需时间，这也是本文所提出的匹配方法的关键所在。

表 2　服务匹配效率的比较

服务个数	服务匹配时间/s	
	文献[3]	本文
150	114	39
450	360	143
900	812	386

结束语　本文提出了一种基于文本聚类和概念相似度的语义 Web 服务发现方法。首先，利用文本聚类技术对服务注册库中的语义 Web 服务进行预处理，将类别或领域一致的服务聚集到一起。然后在第二阶段，使用基于本体概念层次树的概念相似度匹配方法对服务的输入输出功能接口进行精确匹配，最终得到满足用户需求的服务。由于本文的预处理阶段对服务进行了聚类，因此减少了后者功能匹配的搜索空间，在兼顾服务匹配准确率的基础上很大程度地提高了服务发现效率。仿真实验的验证表明了所提方法的有效性和可行性。

本文对语义 Web 服务的发现做了初步的研究和探索，还有很多问题有待考虑和研究。下一步工作的重点是将服务质量 QoS 考虑到服务匹配的过程中，这样才能够使服务发现效果达到最佳。

参 考 文 献

[1] 吴建，吴朝辉，李莹，等. 基于本体论和词汇语义相似度的 Web 服务发现[J]. 计算机学报，2005，28(4)：595-602

[2] 彭辉，史忠植，邱莉榕，等. 基于本体概念相似度的语义 Web 服务匹配算法[J]. 计算机工程，2008，34(15)：51-53

[3] 杨永齐，符云清，余伟. 基于多阶段匹配的语义 Web 服务发现框架[J]. 计算机科学，2010，37(9)：164-167

[4] 孙萍，蒋昌俊. 利用服务聚类优化面向过程模型的语义 Web 服务发现[J]. 计算机学报，2008，31(8)：1340-1353

[5] Khalid E，Ahmed E，Patrick M. Clustering WSDL Documents to Bootstrap the Discovery of Web Services[C]// IEEE International Conference on Web Services. 2010：147-154

[6] 曾志雄. 一种有效的基于划分和层次的混合聚类算法[J]. 计算机应用，2007，27(7)：1692-1698

[7] Klusch M，Fries B，Sycara K. OWLS-MX：A Hybrid Semantic Web Service Matchmaker for OWL-S Services[J]. Web Semantics：Science，Services and Agents on the WorldWideWeb，2009，7(2)：121-133

[8] 徐小良，陈金奎，吴优. 基于聚类优化的 Web 服务发现方法[J]. 计算机工程，2011，37(9)：68-70

[9] Paolucci M，Kawmura T，Payne T，et al. Semantic matching of Web services capabilities[J]. Lecture Notes in Computer Science，2002，2342：333-347

[10] Klusch M，Khalid M，Kapahnke P，et al. OWLS-TC4：OWL-S service retrieval test collection version4[EB/OL]. http://projecs. semwebcentral. org/frs/download. php/487/OWLS-TC4_PDDL. zip，2010-09-21