基于语义相似性的Web服务匹配算法

欧伟杰[1] 曾 承[1,3] 韩 栋[1] 彭智勇[2] 刘 洋[1] 马景燕[1] 刘 波[1]
(武汉大学软件工程国家重点实验室 武汉 430072)[1] (武汉大学计算机学院 武汉 430072)[2]
(清华大学软件学院 北京 100084)[3]

摘 要 随着互联网应用的不断丰富,如何高效而准确地发现用户所需的Web服务已成为重要的挑战。传统基于关键字和语义匹配的方法存在查全率低和效率低下的问题,无法满足实际应用。提出基于语义相似性的服务匹配算法,实现了基于层次本体的概念相似性计算,并改进了原有二分图最优匹配算法的不足。根据该算法实现了一个Web服务发现原型系统。经实验证明,该方法不仅具有较高的查全率,且算法效率是满足目前服务发现需要的。

关键词 Web服务,语义相似性,二分图匹配
中图法分类号 TP311 **文献标识码** A

Web Service Matching Algorithm Based on Semantic Similarity

OU Wei-jie[1] ZENG Cheng[1,3] HAN Dong[1] PENG Zhi-yong[2] LIU Yang[1] MA Jing-yan[1] LIU Bo[1]
(State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China)[1]
(School of Computer Science, Wuhan University, Wuhan 430072, China)[2]
(School of Software, Tsinghua University, Beijing 100084, China)[3]

Abstract With the increasing growth of Web applications, how to discover the desired services for users efficiently becomes a significant challenge. A novel approach for service matching based on semantic similarity was proposed, which employs hierarchical ontology to compute the semantic similarity of concepts from two compared services. The maximum weight matching algorithm was improved according to the situation of Web service. The method was implemented in a prototype of service discovery. The experiments illustrate that our approach not only enhances the recall rate, but also meets the needs of the current service discovery.

Keywords Web service, Semantic similarity, Bipartite graph matching

1 引言

作为面向服务应用的重要组成部分,Web服务越来越多地被关注和使用[1]。企业可以利用服务的可重用性建立各种新颖的业务流程,而服务的多样性也可以满足广大一般用户的需求。Web服务提供了一系列标准的通信方式,这使跨系统、松耦合的服务组合得以实现。但随着互联网上服务数量的不断增加,高效而准确的服务发现变得难以实现。

服务发现通常被定义为一个匹配的过程,即服务的能力可满足服务请求者的需求。而服务的能力是隐含在服务描述中的,其中包括服务名、操作名以及服务的输入/输出参数等。服务能力可以表示为一个抽象的接口,这些接口通过标准化的服务描述语言(Web Services Description Language, WSDL)[2]加以描述。而服务组合也可以看作是两个不同服务间输入与输出接口的匹配。由此可见,服务匹配对于Web服务能力的扩展与应用起着十分关键的作用。

在WSDL语言的支持下,研究者们提出了不少Web服务匹配的方法,其中有基于关键字或基于向量空间模型(VSM)的语法匹配方法以及基于本体的语义匹配方法。虽然语法匹配方法可以帮助用户快速地找到那些潜在的、满足他们需求的服务,但是它们对于语义信息的丢失很大程度上降低了服务发现的查全率。另一方面,基于本体的语义匹配方法虽然可以解决这个问题,但其自身计算复杂性过高且本体的制定和整合也没有很好的统一标准,这就使得这类方法实施难度大,无法广泛使用。

为了解决这些问题,提出了基于WordNet的服务匹配方法,其中利用OSS算法计算概念之间的语义相似性。该方法不但将OSS算法应用于WordNet词汇库,通过概念间的相似性扩展了服务的语义,并以二分图模型定义服务匹配问题,在原有KM算法的基础上做出了优化。其主要贡献表现在以

到稿日期:2011-02-12 返修日期:2011-06-01 本文受国家自然科学基金面上项目(61070011),国家重点基础研究发展计划(973)(2007CB310806)资助。

欧伟杰(1981-),男,博士生,主要研究方向为服务计算,E-mail:oweijie@gmail.com;曾 承(1978-),男,博士,副教授,主要研究方向为服务计算、跨媒体;韩 栋(1988-),男,硕士生,主要研究方向为服务计算;彭智勇(1963-),男,教授,博士生导师,主要研究方向为Web数据管理、复杂数据管理、可信数据管理;刘 洋(1985-),硕士生,主要研究方向为服务计算;马景燕(1986-),硕士生,主要研究方向为服务计算;刘 波(1986-),男,硕士生,主要研究方向为服务计算。

下两点。

• 基于 WordNet 的 OSS 概念相似性算法，不但实现了服务语义的泛本体相似性计算，且无需建立特定领域本体，易于实现和扩展。

• 基于 KM 算法优化的服务匹配方法，借鉴二分图匹配的基本思想计算两个服务之间的相似或匹配关系，针对服务自身的特点做了优化，这对于服务发现、组合有较大帮助。

本文第 2 节介绍了如何以二分图匹配表示服务发现；第 3 节阐述方法的整体流程，并概述 OSS 算法在 WordNet 上的实现；第 4 节详细说明 KM 算法及其针对服务匹配的改进；第 5 节是所提方法的具体实验和结果分析。

2 相关研究

根据所调研的资料，目前服务发现还依赖于对服务本身的分类，而 UDDI 是采用这种方法的主流技术。虽然用户可以通过系统提供的接口利用关键字来查询他们所需的服务，但这类简单的、基于关键字的搜索仅有较低的查全率和查准率。在此基础上，人们提出了基于向量空间模型的方法，其虽然可以根据相关性排序可用的服务，但未考虑语义上有关联的服务，因为这类方法还是基于关键字的。为了解决词汇的多义性，人们利用本体技术来扩充服务本身的描述，设计出了一系列的语义服务描述语言，如 OWL-S 等；并根据这些语言设计了具体的服务匹配方法，其中的代表性工作如文献[9]。但这类方法存在一个根本性的问题，即本体自身的构建和维护需要大量人工参与，导致这类方法扩展性差，且领域性过强。各领域本体没有统一的标准，本体自身的集成和重用也是一个无法回避的问题。

通用文本集构建轻量级的语义服务匹配是不少研究者的共识[11-14]。URBE[1]提出了基于实例的服务查询，将服务发现转变为一个计算 WSDL 实例之间相似性的过程。相似性计算分为两部分：语义相似性和语法相似性。其中，语义方面与该 Web 服务所要完成的功能有关，表现为整个服务、操作以及参数的名称；语法方面与该服务的输入、输出接口之间的关系以及接口的数据类型相关。其中值得注意的是，作者讨论了两个接口之间相似性的不对称性，这种不对称主要来自两接口的概念数量不一致。作者仅提出通过概念数量比较来确定相似性最大值，没有详细分析。公式中并未给出具体解释。

浙大吴朝晖等提出的二分图匹配方法[2]关注点在于服务发现，通过服务、操作以及接口参数 3 个层次讨论需求与服务的关系。其中接口参数相似性计算中，加入了输出与输入的依赖关系，并在相似性计算中有所体现，这是其他研究所没有考虑的。但对于 KM 算法应用于不平衡二分图的情况，文中通过扩展虚拟节点得到匹配后再去除虚拟点。

文献[3]中分析了基于 WSDL 的服务相似性计算方法，指出 WordNet 不如搜索引擎所支持的概念广泛，而且不能适应新生概念。因此文中利用 Google 搜索引擎的返回结果作为两个概念之间相似性的度量，这也是引用其他研究的结果，并未展开介绍。另外，分析目前二分图算法中每个服务接口的概念都是相对孤立的，没有考虑未匹配节点的语义。文中作者提出了几种相似性计算公式，它们均考虑了各服务中的所有节点(匹配节点和未匹配节点)。

3 服务匹配的抽象定义

在如何表示一个服务的问题上，有很多不同的观点。一种观点认为服务的描述信息可以很好地反映一个服务的特征，另一种观点则认为服务接口的信息更适合描述服务的功能。本文的观点建立在后者的基础上，再赋予每个接口一个包含语义信息的概念，这就能充分体现一个服务的特征。若仅仅考虑服务接口的语义信息，则可以将一个服务抽象成一个带有方向性边的图，如图 1 所示。这里给出服务的形式化定义。

定义 1(服务) 一个服务 S 可以通过 1 个四元组来表示：$S=\{n_s,d_s,I,O\}$，其中

(1)n_s 是该服务的名称；

(2)d_s 表示该服务的文本描述信息；

(3)$I=\{i_1,i_2,\cdots,i_n\}$是该服务的输入信息集合；$O=\{o_1,o_2,\cdots,o_n\}$为该服务的输出信息集合。

这样的抽象虽然在一定程度上简化了服务，但对服务匹配并没有影响，因为我们重点考虑的是服务的接口信息。用户的需求可以借鉴服务的定义方式，在经过需求形式化后，可以视为一个虚拟服务。满足用户需求的过程就是从服务库之中找到一个可以和虚拟服务最大匹配的潜在服务，可以将其描述为服务之间的匹配问题。

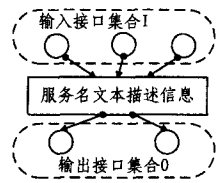

图 1 抽象服务图

每个服务抽象成的节点均有若干个入口和出口，分别对应服务的输出和输入。需要注意的是，这里讨论的服务是最小粒度的原子服务，仅包含一个操作，所以只有当这些输入同时满足时，该服务才能调用。进一步细化服务匹配的相似性计算，可以发现服务匹配其实包含每个概念节点的相似性计算。假如能够得到每个概念节点之间的相似性，则可以通过一定的算法得到服务整体之间的相似性。当一个 Web 服务抽象成图 1 的模型之后，因为服务匹配主要是接口上的匹配，所以可以将服务匹配抽象成二分图的模式，如图 2 所示。其中每条边的权重表示的是输入/输出参数之间的语义相似性。这样，服务匹配的问题就可以转换成二分图求最优匹配的问题，也可以借用二分图最佳匹配的很多经典思路。图中粗线表示的是该二分图的最佳匹配，也就是还原成服务后的服务匹配结果。由此，得到服务匹配的形式化定义。

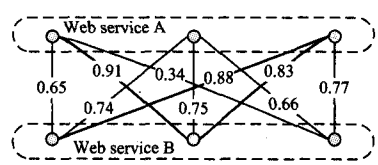

图 2 抽象成二分图的服务接口匹配

定义 2(接口匹配) 假设服务 $S_1=\{n_1,d_1,I_1,O_1\}$和另一服务 $S_2=\{n_2,d_2,I_2,O_2\}$，若 S_1 与 S_2 输入接口的相似度 $\mathrm{Sim}(I_1,I_2)\geqslant\omega$，称集合 I_1 与 I_2 之间存在接口匹配。其中 ω

是人为设定的阈值，根据实际应用环境调节接口相似性。

接口匹配不仅适用于同类接口之间（如输入与输入、输出与输出），也可以应用在输出、输入之类接口之间，这对服务组合有着重要意义。

定义 3（服务匹配） 假设存在服务 $S_1=\{n_1,d_1,I_1,O_1\}$ 和另一服务 $S_2=\{n_2,d_2,I_2,O_2\}$，当且仅当 S_1 与 S_2 的输入/输出接口均存在接口匹配时，则称服务 S_1 是服务 S_2 的一个匹配。若 S_2 代表了一个用户需求，则 S_1 是满足当前需求的可用服务，这也是服务发现的实现。

根据以上抽象定义，在 973 项目原型系统[7]中设计并实现了一个服务匹配工具。下面介绍该系统的整体结构及相关技术。

4 服务匹配系统及具体实现

4.1 整体框架

图 3 为服务匹配系统的整体框架。一方面从数据库中读取服务的接口参数信息，另一方面根据 WordNet 计算出概念之间的两两相似性，然后构建出服务对二分图来计算服务之间的最优匹配，最后得到服务相似性结果。系统中利用了 WordNet 词库和 OSS 算法计算概念相似性。

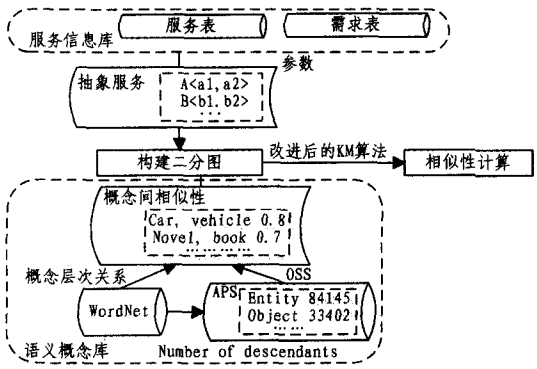

图 3 系统结构图

4.2 OSS 算法实现

WordNet[3]是由普林斯顿大学的心理学家、语言学家和计算机工程师联合设计的一种基于认知语言学的英语词典。它不仅将单词以字母顺序排列，而且按照单词的意义组成一个“单词的网络”。大部分概念相似性方法都是利用这个单词网络计算得到单词之间的语义距离。OSS 算法[4]的全称为基本本体结构的相似性，是一种在层次化本体上计算概念相似度的方法。经实验证实，OSS 算法以 WordNet 计算得到的概念相似性优于同类其他方法，这也是本文利用 OSS 方法计算服务中概念之间语义距离的原因。

在计算概念语义距离时，需要计算某节点的所有子孙数，故必须遍历该节点的子树。当该节点在 WordNet 的层次比较高时，这个算法会非常耗时，有时候会产生内存溢出。同时，概念子孙数是会被频繁计算的值，所以在实现 OSS 算法时，预先将 WordNet 中层次很高的节点的子孙数目存储到数据库中，并且在计算时，一旦需要某个节点的子孙数，就先查看对应的数据表，看数据库中是否缓存有该节点子孙数信息。若有，则直接得到；若没有，则访问 WordNet 计算，并将计算得到的子孙数更新到数据库中。这样，经过多次实验之后，计算子孙数目基本都已缓存，已经不是算法的效率瓶颈。

另一个值得注意的是，计算概念 A 和 B 的最近公共父节点的算法，称为 LCA（Least Common Ancestors）算法。该算法会逐层向上遍历 A 的父亲节点。对于每个父亲节点，遍历它的所有子树，看 B 是否在子树之中，这也是一个很耗时的操作。其解决方式是将 WordNet 导入到数据库之中，不用 WordNet 自带的接口来访问它，这样通过数据库查询操作来替换对于 WordNet 接口的频繁多次访问，从而提高实际算法的执行效率。

如图 3 所示，在计算得到概念之间的相似性（即语义距离）后，通过构建服务对二分图，可以引入二分图匹配的最优匹配算法来计算服务对之间的最优匹配。

5 最优匹配算法 KMP

5.1 最优匹配算法的缺陷

Kuhn Munkres（KM）算法[5]是计算带权二分图最优匹配的经典算法，但将服务匹配问题抽象成二分图匹配之后，发现原始算法在求最优匹配时存在一些不足，使得最后的相似性计算并不准确，甚至某些情况根本无法求得相似性结果。

5.1.1 KM 算法的条件限制

由于 KM 算法本身是计算二分图匹配的，因此算法要求二分图的两个部分的元素个数相同。在计算服务之间相似性时，如果使用 KM 算法，则必须要求服务的接口个数相同。这一点在实际服务中显然很难保证。克服这种缺陷的一个方法通常是添加虚拟节点，见文献[6]。图 4 是一个添加了虚拟节点之后的二分图匹配模型。

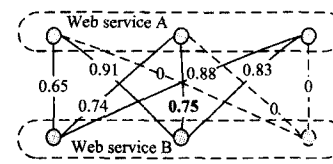

图 4 含有虚拟节点的二分图模型

由图 4 可见，虚线表示的点为虚拟节点，它和所有实际的顶点之间的权重都是 0。当平衡二分图计算出最优匹配后，再将和虚拟节点匹配上的那个边剔除掉，就得到了最优匹配。

5.1.2 对未匹配的节点的忽视

上面介绍的添加虚拟节点方法，其核心思想是构造 KM 算法的适用条件，并且丢弃那些和虚拟节点匹配上的顶点。但实际上这种做法会丢失很多语义信息，尤其是在服务匹配的语义性计算中，被丢弃的节点可能含有一些语义信息。假如用户提出一个需求，需要找到输出为“汽车，颜色”的服务。而系统现在含有两个备选服务：服务 A 的输出为“汽车”、“跑车”和“色彩”，服务 B 的输出为“汽车”、“轮船”和“色彩”，那么根据上一节的算法，服务 A 和服务 B 对于用户需求的相似性都是一样的，因为“汽车”和“汽车”、“颜色”和“色彩”相匹配，而“跑车”和“轮船”会因为未能匹配上而被丢弃。而实际上，服务 A 满足用户需求的可能性远大于服务 B。

5.2 基于 KMP 算法的服务相似性

本节提出对 KM 算法思想的改进，既考虑了如何突破 KM 算法要求平衡二分图的限制，又不会完全丢弃未匹配上节点的语义信息。在推导本文改进的算法之前，将可能用到的符号在表 1 中说明。

表1 公式符号表

符号	含义
KMP	对于参数个数不相等且不相同的服务间用 KMP 算法求出的结果
K_U,T_V	需要做匹配的服务,下标为匹配的接口集合
σ	调整参数
$\cap K_U,T_V$	K_U和 T_V 之中相同参数的个数
$\cup K_U,T_V$	K_U和 T_V 之中所有不相同的参数的个数
$Sim(K_U,T_V)$	K_U和 T_V 的相似性

下面介绍 KMP 算法的主要思想。如果 K_U和 T_V 之中不存在相同的参数,参数个数不相等,假设 K_U的参数个数多于 T_V,那么可以通过添加虚拟节点的方式求得匹配结果。但结果是 K_U的某些参数势必会被丢弃。为了在结果中也体现被丢弃的参数的语义信息,取出 K_U中被丢弃参数和 T_V 中所有参数的权重最大的边,加到匹配结果中,最后将所有边权重加起来之后,除以 K_U的参数个数。

如图 5 所示,添加虚拟节点之后,只会计算 0.91 和 0.88 这两条边,而节点 C 会被丢弃,相似性为(0.91+0.88)/2=0.89。经修改后,0.75 那条边也会参与到最后结果,相似性为(0.91+0.88+0.75)/3=0.85。

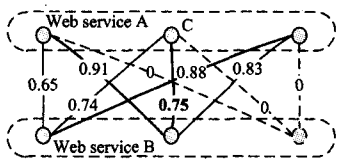

图 5 修改过的添加伪节点相似性计算方法

通过上述方法计算出的 K_U和 T_V 之中不存在相同的参数,参数个数不相等的值称为:

$$Sim(K_U,T_V)=KMP \tag{1}$$

特殊地,如 K_U和 T_V 的参数个数相同,也可以用式(1)来求出相似性,其区别是不再需要添加虚拟节点。

如果 K_U和 T_V 之中含有一些相同的参数,那么这些相同的参数肯定能匹配上,相应地,K_U和 T_V 的相似性会变高。但是仍需要量化一下相同的参数到底有多大的比重。本文在这里采用 $\cap K_U,T_V/\cup K_U,T_V$ 来量化 K_U和 T_V 之中的相同参数的比率。所以最后的相似性应该是在原始的相似性之上加上一个和 $\cap K_U,T_V/\cup K_U,T_V$ 有关的值。进一步考虑,如果去掉相同元素之后的相似性越接近 1,那么 $\cap K_U,T_V/\cup K_U,T_V$,$T_V$ 在最后的相似性中占据的权重就越小。因为相似性接近 1,表示尽管 K_U和 T_V 具有相同的元素,在去掉相同元素之后它们之间相似性仍然很大,那么相同元素的重要性就相应降低了。所以在这种 K_U和 T_V 含有相同元素的情况下,相似性计算应该如下:

$$Sim(K_U,T_V)=KMP+(1-KMP)\frac{\cap K_U,T_V}{\cup K_U,T_V} \tag{2}$$

当基于此种情况做实验时,发现还有一种特殊的情况。即 K_U和 T_V 的参数集合是真包含关系时,在去掉相同元素之后,某服务会不再含有参数,从而导致 KM 的值无法求出。为了处理这种情况,参照式(2)的推导过程,用一个调整参数来代替 KM 值:

$$Sim(K_U,T_V)=\sigma+(1-\sigma)\frac{\cap K_U,T_V}{\cup K_U,T_V} \tag{3}$$

经多次实验测试发现,σ 取 0.8 时,计算出的相似性结果更加符合经验值。最后将式(1)、式(2)和式(3)合并,得到式(4)。

$$Sim(K_U,T_V)=\frac{\cup K_U,T_V(KMP+\sigma)(\cup K_U,T_V-\cap K_U,T_V)}{\cup K_U,T_V} \tag{4}$$

式中,当 K_U和 T_V 之间存在真包含关系时,σ 取 0.8,否则为 0。当两个服务之间没有相同参数时,$\cap K_U,T_V$和 σ 均为 0,式(4)将变为式 1;当两个服务含有相同参数又不真包含时,σ 为 0,式(4)将变为式(2);当两个服务真包含时,KM 将为 0,式(4)将变成式(3)。

式(4)将原始的 KMP 算法作为最大的因素计算,同时加进了对各种应用情况的处理。另外,σ 作为一个控制因子来调整这个算法。

5.3 服务匹配流程

在推导出了改进后的 KMP 公式来计算 Web 服务的相似性之后,本节将给出整个匹配算法的流程。

Algorithm Web 服务语义匹配算法

输入:Web 服务 A (input_a,output_a),Web 服务 B (input_b,output_b)

输出:Sim(A,B) //A 和 B 的语义相似性

1. 从数据库之中取得 Web 服务,得到参数信息,设 A 为 K_U,设 B 为 T_V;
2. 设 output_a 为 U,设 input_b 为 V;
3. for each u in U{
4. for each v in V{
5. 通过 OSS 算法 得到 sim[u][v];
6. }
7. }
8. 构建二分图 G(X,E,Y),set U to X,set V to Y,set sim[u][v] to E。
9. 计算 $Sim(K_U,T_V)$ //式 (4)

这个流程中最耗时的就是第 5 步,计划后期通过数据库缓存相似性来减少重复计算的开销。在构建了二分图之后,最后一步计算相似性是利用 KMP 算法实现的,原始 KM 算法本身的时间复杂度为 $O(n^3)$。这里的 n 是由服务中操作和接口的参数数量所决定。但由于服务中参数的数量为常数,因此服务匹配算法整体的时间复杂度是常数,即算法运行时间为固定常数,总体匹配时间仅与服务数量呈线性关系。

6 Web 服务匹配实验

6.1 实验环境及数据

选择与经典服务匹配方法 OWLS-MX[9] 做比较,以验证新方法的查全率。由于 OWLS-MX 的实验测试集合中仅包含 1003 个服务,因此收集了 Seekda 网站[8] 提供的真实 Web 服务描述文件来扩充测试服务库。本实验使用的服务测试库和文献[7]中一致。实验用的计算机操作系统为 Windows XP SP3,内存为 2G,CPU 为 3.0 GHz Pentium IV。测试代码为 Java 编写,Eclipse3.3 编译通过。

6.2 实验与结果分析

6.2.1 算法查全率

本试验将测试服务集改为 OWLS-MX 的测试集,采用相似性为 0.8 的阈值来在 OWLS-MX 的测试服务库中检索用户需要的服务,记录返回服务的数量。同时采用同样的需求

(下转第 129 页)

Software Technology,2009,51(2):306-324
[12] 倪友聪,应时,文静,等.一种面向方面软件体系结构中的编织机制研究[J].计算机研究与发展,2010,47(4):695-706
[13] 倪友聪,应时,张琳琳,等.基于时序逻辑的面向方面体系结构描述语言[J].计算机科学,2010,37(1):146-152
[14] Ni Youcong,Ying Shi,et al. Modeling Aspect-Oriented Software Architecture[A]//Proceedings of 2009 International Conference on Industrial and Information Systems (IIS2009),2009[C]. Haikou,China:IEEE,2009:108-113
[15] Wen Jing, Ying Shi, Zhang Lin-lin, et al. AC2-ADL: Architectural Description of Aspect-oriented Systems[J]. International Journal of Software Engineering and its Applications,2009,3(1):1-10
[16] 李未.数理逻辑基本原理与形式演算[M].北京:科学出版社,2007

(上接第 95 页)

使用 OWLS-MX 来查询服务,记录下 OWLS-MX 返回的服务数量。做了 3 组不同数据的测试,详见表 2,分别根据 3 个不同的用户需求比较两种算法的返回结果。最后结果如图 6 所示。

表 2 对比实验数据

	服务集合规模	用户需求
test1	300	car,price
test2	150	novel
test3	200	scholarship,organization

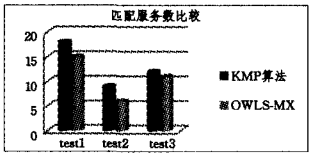

图 6 算法结果对比图

由图可以看出,改进后的 KMP 算法可以比 OWLS-MX 发现更多的潜在可匹配的服务。如在测试 2 中,用户要求输入满足概念 "novel"、返回服务中有一个输入为"publication"的服务,而 OWLS-MX 则没有返回该服务。在测试 3 中,用户检索包含"scholarship,organization"参数的接口,返回了包含"award,government"的服务,语义是强相关的,而 OWLS-MX 也没有检索到。根据以上结果可以证明,与基于本体的语义匹配相比,本文方法可以更好地发现服务之间的语义联系。

6.2.2 算法耗时分析

为了了解算法的执行效率,将特定服务与不同规模服务集进行匹配计算。图 7 为实验的耗时情况。

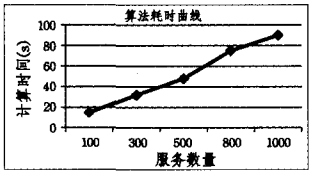

图 7 耗时曲线图

由图 7 可以看出,当服务集的大小为 300 时,计算整个服务库之间服务两两匹配需要将近 20s,而当总服务数量达到 1000 以后,计算的时间仅为 90s。计算得到的总耗时与匹配次数呈线性关系,与第 4 节中得到的计算复杂度相符。平均每次匹配操作所需时间为 100ms 左右,其中包含了服务中概念间相似性计算的时间。因此可以证明,对于管理服务数量在万级的系统(如 Seekda)来说,该方法也是完全适用的,并在原型系统[7]中得到了验证。

结束语 服务匹配是 Web 服务体系的重要支撑技术。如何提高服务匹配的准确性以及效率,是本文研究的重点。利用基于 WordNet 的语义相似性计算以及基于二分图的 KM 匹配算法,提出了的服务匹配 KMP 算法。经实验证明,该方法查全率优于传统语义方法,且算法效率满足实际应用。下一步将考虑通过数据库缓存语义相似性,进一步提高匹配系统的整体执行效率,这对大规模的服务发现是具有实际意义的。

参 考 文 献

[1] 岳昆,王晓玲,周傲英,等. Web 服务核心支撑技术:研究综述[J].软件学报,2004,15 (3):428-442
[2] Christensen E, Curbera F, Meredith G, et al. Web services description language (WSDL) 1. 1[OL]. http://www. w3. org/TR/wsdl, World Wide Web Consortium, W3C Note, March 2001
[3] Miller G A, Beckwith R, Fellbaum C D, et al. WordNet: An online lexical database [J]. International Journal of Lexicography. 1990(3):235-244
[4] Schickel-Zuber V, Faltings B. OSS: A Semantic Similarity Function Based on Hierarchical Ontologies[C]// Proceedings of IJCAI. 2007:551-556
[5] 邓水光,尹建伟,李莹,等.基于二分图匹配的语义 Web 服务发现方法[J].计算机学报,2008,31(8):1364-1375
[6] Zeng Cheng, Guo Xiao, Ou Wei-jie, et al. Cloud Computing Service Composition and Search Based on Semantic[C]// Proceedings of the 1st International Conference on Cloud Computing (CloudCom '09). 2009:290-300
[7] Seekda[OL]. http://www. seekda. com/
[8] Klusch M, Fries B, Sycara K. Automated Semantic Web Service Discovery with OWLS-MX[C]// Proceedings of 5th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 06). 2006:915-922
[9] Wu J, Wu Z H. Similarity-based Web service matchmaking[C]// International Conference on Services Computing. Orlando, FL, USA: IEEE Computer Society, 2005 (1):287-294
[10] Zhuang Z, Mitra P, Jaiswal A. Corpus-based Web Services Matchmaking[C]// AAAI. Conference. 2005:46-52
[11] Chen Lei, Yang Geng, Wang Dong-rui. WordNet-powered Web Services Discovery Using Kernel-based Similarity Matching Mechanism[C]// The Fifth IEEE International Symposium on Service Oriented System Engineering. 2010:64-68
[12] Plebani P, Pernici B. URBE: Web Service Retrieval Based on Similarity Evaluation [J]. IEEE Transactions on Knowledge and Data Engineering, 2009, 16:1926-1942
[13] Liu Fang-fang, Shi Yu-liang, Yu Jie, et al. Measuring similarity of Web services based on WSDL[C]// IEEE 8th International Conference on Web Services. 2010:155-162