软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(10):2665−2678 [doi: 10.3724/SP.J.1001.2012.04185]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

面向约束的 Web 服务发现方法研究*

柯昌博[1+], 黄志球[1], 刘林源[2], 曹子宁[1]

[1](南京航空航天大学 计算机科学与技术学院,江苏 南京 210016)

[2](南京审计学院 信息科学学院,江苏 南京 211815)

Research on Constraint-Oriented Web Service Discovery

KE Chang-Bo[1+], HUANG Zhi-Qiu[1], LIU Lin-Yuan[2], CAO Zi-Ning[1]

[1](College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

[2](School of Information Science, Nanjing Audit University, Nanjing 211815, China)

+ Corresponding author: E-mail: brobo.ke@gmail.com

Ke CB, Huang ZQ, Liu LY, Cao ZN. Research on constraint-oriented Web service discovery. *Journal of Software*, 2012,23(10):2665−2678 (in Chinese). http://www.jos.org.cn/1000-9825/4185.htm

Abstract: Web Service is a major computing resource and a main software paradigm. With rapid increase of Web Services in recent years, methods of accurately discovering the required service are becoming a research focus. By using service matching method, based on conception similarity, in this paper, the study transforms user requirements and Semantic Web Service description documents OWL-S profile into ontology trees separately. Next, by hierarchical and taxonomical method, according to the ontology trees, the study computes the conception similarity, attribute similarity, and structure similarity respectively, which effectively avoids complex reason. Thereafter, according to the relationship between conception similarity and structure similarity, the study defines the sets of constraints and restructures the requirement trees with constraints to improve the precision and recall of the service discovery. Finally, the paper presents the algorithm of Semantic Web Service discovery and conducts experiments by developing the prototype system OWLS-CSR, which proves the feasibility and affectivity of this method.

Key words: Web Service; ontology; conception similarity; attribute similarity; structure similarity

摘 要: Web Service 已经成为主要的计算资源和软件的主要存在形态.为了满足用户的各种需求,使得 Web 服务的数量快速增加,而能从大量的服务中准确地发现满足用户需求的服务,成为研究热点和难点.结合成熟的基于概念相似度的服务匹配方法,分别将用户需求和语义 Web 服务描述文档 OWL-S profile 转化为本体树,并采用分层、分类的方式分别计算对应节点的概念相似度、属性相似度和结构相似度,有效地避免了复杂的推理.根据概念相似度和结构相似度之间的关系定义一系列的约束,并利用约束对查询树进行重组,以提高服务发现的查准率和查全率.最后,给出了语义 Web 服务发现的算法,并通过开发原型系统 OWLS-CSR 进行实验,证明了该理论方法的可行性与有效性.

关键词: Web Service;本体;概念相似度;属性相似度;结构相似度

中图法分类号: TP311 **文献标识码**: A

* 基金项目: 国家自然科学基金(60873025); 江苏省自然科学基金(BK2008389)
收稿时间: 2011-08-28; 修改时间: 2011-11-02; 定稿时间: 2012-01-16

Web服务通过异构领域间的互操作,实现了领域间的通信.随着Web服务相关标准的完善和支持Web服务开发与运行平台的不断成熟,Web 服务已经成为互联网中主要的软件形态和计算资源.Web 服务是具有自组织、自适应、自管理、自描述和模块化的应用程序[1].由于具有良好的扩展性和互操作性,广泛应用于电子商务、知识集成、流程管理等领域[2].但当前的 Web 服务大多数都采用基于 XML 的 Web 描述语言(WSDL[3])来描述服务,能够保证其语法的互操作性,但缺乏对服务的语义描述,从而使得 Web 服务发现的查准率较低.而面对Web服务数量的日益增长,如何从大量的Web服务中搜索到用户或代理所要的目标服务,成为Web服务的研究热点和难点.因此,Web 服务发现吸引了国内外学者的广泛关注,成为面向服务计算(service oriented computing)领域中的关键问题[4].

语义 Web 服务(SWSs)[5]是机器可理解的、支持上下文语义推理的 Web 服务表现形式.采用本体 Web 语言(OWL-S)对服务进行描述,对 Web 服务发现提供语义支持,相对于传统上仅仅基于概念相似度的服务匹配而言,提高了发现的效率.语义 Web 服务发现的核心理论是本体和描述逻辑.然而,目前大多数还是采用概念相似度匹配[6−10],但是由于忽略了上下文语义的结构信息,其查准率与查全率不高.针对这一弱点,也有将概念相似度与描述逻辑推理器相结合进行匹配的[11−15],但定义推理规则不具有完备性,并且相对较为复杂;OWLS-MX[16]匹配器基于逻辑推理,并在概念相似度的基础上定义了过滤器,增加了匹配的查准率和查全率,但不能动态地使用本体规则,其有效性在一定程度上受到了限制.

本文采用 WSMO[17]的语义 Web 服务发现框架 WSMO-DF.在概念相似度的基础上,提出了基于分层分类的概念相似度和结构相似度,有效地避免了复杂的逻辑推理;并根据两种相似度之间的关系定义了一系列重组约束,对查询本体树进行重组,提高了 Web 服务发现的查准率和查全率;最后,开发了原型系统 OWLS-CSR,并与业界著名的服务匹配器 OWLS-MX 进行了对比实验,并进行了数据分析,证明了这一理论方法的可行性和有效性.

本文第 1 节主要讨论本文的基本理论,主要有第 1.1 节的本体、本体树及节点;第 1.2 节的相似度,包括概念相似度、属性相似度和结构相似度;第 1.3 节定义约束规则并对查询树进行重组;第 1.4 节的映射与匹配,主要给出了语义 Web 服务发现算法.第 2 节主要讨论基于 OWL-S 的 Web 服务发现流程.第 3 节介绍原型系统 OWLS-CSR,并进行数据分析.第 4 节是相关工作.第 5 节是结论及进一步的工作.

1 基本理论

Web 服务发现是将用户需求文档与服务描述文档进行匹配的过程.语义 Web 服务描述文档 OWL-S 包括 3 个组件:

- Service Profile:主要是对服务功能的描述(即服务的输入、输出、前置条件和结果),包括服务名、服务提供者等.服务搜索代理通过 service profile 实现服务匹配,寻找到满足服务请求者需求的 Web service;
- Service Model:主要描述服务的实现细节,即服务是怎么做的;
- Service Grounding:主要描述怎么访问服务,包括通信协议等.

每一个 service advertisement 都是 service profile 的一个直接实例[18,19].因此,基于 service profile 的 Web 服务发现包括 profile 实例的表示和用户需求与 service advertisement 的匹配.我们可以利用 Xpath 将 OWL-S 中的语义标注的概念取出,得到概念之间的分层关系(用描述逻辑进行描述),见表 1.然后,将用户需求文档和上述的分层关系都转化为本体树.即,将服务的匹配最终转化为本体树之间的匹配.

Table 1 Domain ontology axioms

表 1 领域本体公理

$Order \subseteq Economy \cap \exists Username.\top \cap \exists Password.\top \cap \exists Number.\top \cap \exists Item.\top \cap \exists Location.\top$
$Search \subseteq Transportation \cap \exists Username.\top \cap \exists Password.\top \cap \exists Number.\top \cap \exists Location.\top$
$Economy \subseteq Profile$, $Transportation \subseteq Profile$, $Book \subseteq Item$, $Magazine \subseteq Item$, $Profile \subseteq \top$, $Hongkong:Location$, $Beijing:Location$
$Name \in Book$, $Price \in Book$, $ISBN \in Book$

例子:利用 Xpath 将 OWL-S 中的语义标注的概念取出,用描述逻辑进行描述(描述逻辑与本体树中的元素之

间具有一一对应的关系),见表 1.

表 2 利用 OWL-S 中的 Service Profile 描述了 5 个 Web 服务的 Advertisement:

- A1 是订单类,表示当用户输入自己的用户名和密码时,就会对应输出订单号和书名,并将此书送往北京;
- A2~A4 都为订单类,与 A1 具有相似的表达;
- A5 是搜索类,当用户输入用户名、密码和订单号时,就可以查询此类订单的状态.

Table 2 Instances of OWL-S Service Profile

表 2 OWL-S Service Profile 的实例

(1) A1:Order, ⟨A1,username⟩:hasInput, ⟨A1,password⟩:hasInput, ⟨A1,Number⟩:hasoutput, ⟨A1,Book⟩:hasoutput, ⟨A1,Beijing⟩:to
(2) A2:Order, ⟨A2,username⟩:hasInput, ⟨A2,password⟩:hasInput, ⟨A2,Number⟩:hasoutput, ⟨A2,Book⟩:hasoutput, ⟨A2,Hongkong⟩:to
(3) A3:Order, ⟨A3,username⟩:hasInput, ⟨A3,password⟩:hasInput, ⟨A3,Number⟩:hasoutput, ⟨A3,Magazine⟩:hasoutput, ⟨A3,Beijing⟩:to
(4) A4:Order, ⟨A4,username⟩:hasInput, ⟨A4,password⟩:hasInput, ⟨A4,Number⟩:hasoutput, ⟨A4,Book⟩:hasoutput, ⟨A4,Hongkong⟩:to
(5) A5:Search, ⟨A5,username⟩:hasInput, ⟨A5,password⟩:hasInput, ⟨A5,Number⟩:hasInput, ⟨A5,Book-state or Magazine-state⟩:hasoutput

1.1 本体、本体树及节点

定义 1(本体). 本体(ontology)可以用一个四元组来表示,即 $O=\{C,R,I,A^0\}$,其中,C(concepts)表示概念的集合,R(relations)表示概念之间关系的集合,I(instances)表示实例的集合,A^0(axioms)表示公理的集合.通常,将概念、关系、实例和公理称为本体的元素[20,21].

定义 2(本体树). 树是图的一种特例,用 T 表示,$T=(V,E,root(T))$,在更多时候省略树根 $Root(T)$,即记为 $T=(V,E)$.其中,V 表示树中点的集合,E 表示边的集合.我们将树中的每一个点的关键字记为节点的标签 $Label(v)$,对应本体中的概念(concept),因此也称为本体树(ontology tree).由表 1 的领域本体得到本体树,此服务对应的本体树中,Thing 类中包含 5 个子类,即用户名(Username)、Profile、订单号(Number)、Item 和 Location.而子类(Book)包含 3 个属性,即名称(Name)、价格(Price)和 ISBN,如图 1 所示.

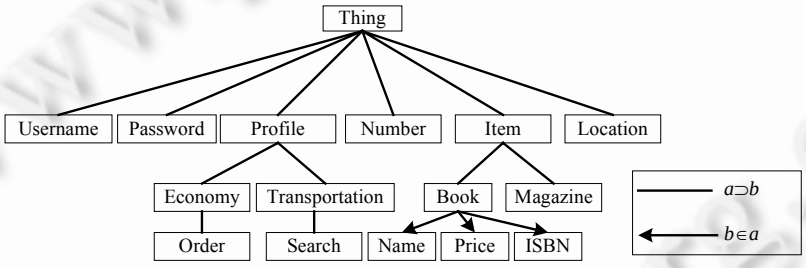


Fig.1 Corresponding ontology tree of domain ontology

图 1 领域本体对应的本体树

定义 3(节点). 节点(node)用一个五元组表示,即 $Node=\{Name,Supc,Subc,Attr,Simi\}$.其中,

- $Name$ 表示节点的名字;
- $Supc$ 表示此节点的超类集合,即 $Supc=\{Supc_1,Supc_2,Supc_3,\ldots,Supc_i\}$.例如:图 1 中的 *Thing* 为超类;
- $Subc$ 为此节点的子类;
- $Attr$ 表示为此节点的属性集合,即 $Attr=\{Attr_1,Attr_2,Attr_3,\ldots,Attr_i\}$.例如:图 1 中的 $Book_{Attr}=\{Name,Price,ISBN\}$;
- $Simi$ 为相似度的值,是 $Name,Supc,Subc,Attr$ 的函数,即 $Simi=\boldsymbol{f}(Name,Supc,Subc,Attr)$.

1.2 相似度的度量

假设查询本体树和描述本体树(下文简称为查询树和描述树)之间具有上下文层次(语义)关系的一致性,即,假若查询树中的某个节点 s_q 在描述树中所对应的层次为 i 层,则其子类节点或其属性必须在描述树中的第 $i+\alpha$

层,如图 2 所示(左为查询树,右为描述树).

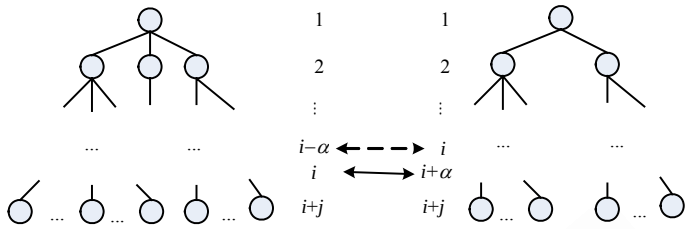


Fig.2 Hierarchy relationships of ontology trees

图 2 层次对应关系

本文中的相似度分为 3 类,即:概念相似度、属性相似度和结构相似度.

(1) 概念相似度(sim_d)

概念相似度是在分层的基础上加以定义的,根据两棵树中节点之间的关系,将其分为 3 个层次,记为 $A \xleftrightarrow{h} B$.其中,A 表示查询树中的节点;B 表示描述中的节点;h 表示两节点间的层次关系,即 $h=\{e,su,p\}$.由于两棵树有相同的根节点(Thing),故:

① 同层(exact):查询树中的 A 节点与描述树中的 B 节点具有相同的层次数,并且后代节点之间是一一对应的,即

$$A_i \xleftrightarrow{e} B_j \Leftrightarrow (j=i) \wedge (son(A_i) \mapsto son(B_j)).$$

② 上下层(subsume):查询树中 A 节点的层次数小于描述树中 B 节点的层次数,并且查询树中节点 A 的孩子与描述树中节点 B 的后代节点对应,即

$$A_i \xleftrightarrow{su} B_j \Leftrightarrow (j=i+\alpha) \wedge (descendant(A_i) \mapsto descendant(B_j)).$$

③ 下上层(plug-in):查询树中 A 节点的层次数大于描述树中 B 节点的层次数,并且查询树中节点 A 的孩子与描述树中节点 B 的后代节点对应,即

$$A_i \xleftrightarrow{p} B_j \Leftrightarrow (j=i-\alpha) \wedge (descendant(A_i) \mapsto descendant(B_j)).$$

本文是基于语义词典 Wordnet 的概念相似度的计算方法.在 Wordnet 中,每个节点 s 表示一个概念,Pantel 和 Lin 等人根据 Wordnet 定义了两个概念的相似度[22]:

$$sim_d(s_1,s_2)=\begin{cases}\dfrac{2\times \log\ p(s)}{\log\ p(s_1)+\log\ p(s_2)}, & \text{if } A_i \xleftrightarrow{h} B_j \\ 0, & \text{otherwise}\end{cases} \tag{1}$$

其中,$p(s)=count(s)/total$ 表示在 Wordnet 中概念节点 s 及其子节点所包含的单词个数在整个词典中所占的比例,$total$ 是 Wordnet 的单词总数,节点 s 是 s_1 和 s_2 的公共祖先节点.

(2) 属性相似度(sim_T)

设在两棵树 T_q,T_d 中,假如某节点为对象,两个对象分别为 O_A,O_B.而 O_A,O_B 中的属性分为两类:一类是简单属性,如整型或者是字符型,这种类型的相似度可以直接根据概念相似度(sim_d)求得;另一类为关系类,即两个属性值可以通过某种关系函数计算得到其相似度,关系函数计为 $f_r=(ID_A,ID_B,P_A,P_B)$.

设两个对象 $O_A=\langle ID_A,C_A,P_A\rangle$,$O_B=\langle ID_B,C_B,P_B\rangle$,并且有共同的属性 $P\in T$(其中,T 为共同的属性集),则属性相似度 sim_p 可以定义为

$$sim_p(O_A,O_B)=\begin{cases}sim_d, & \text{(integer, char)} \\ f_r=\Upsilon(ID_A,ID_B,P_A,P_B), & \text{(relationship)}\end{cases} \tag{2}$$

其中,f_r 是由属性之间特定的语义关系所确定的.不同的属性有着不同的语义关系,这其中有一对一的关系,也有

一对多的关系.例如:对象 O_A 是指某个人 Tom,O_B 是指 Jack,假如 Tom 的属性项包含年龄(age),而 Jack 的属性项包含生日(birthday),则这两个属性就是关系型的,即 $f_r=\Upsilon(Age,Birthday),Age=PresentYear-Birthday$;又如:$O_A$ 和 O_B 分别是指两个圆,而 O_A 的属性项包含圆的半径(r);而 O_B 的属性项包含面积(S),同理,其属性为关系型的,即 $f=\Upsilon(r,S)$, $S=\pi r$.此时,如果所得到的属性值是一致的,则属性相似度为 1,否则为 0.于是,对象节点的属性相似度 sim_T 可以定义为

$$sim_T(O_A,O_B)=\underset{\forall p\in T}{\Delta} sim_p(O_A,O_B) \tag{3}$$

(3) 结构相似度(sim_s)

设查询本体树和描述本体树分别为 T_q,T_d,由于树中有 3 种节点:类、对象和属性,则应分 3 类情况进行讨论:

1) 假若此节点为类,则可以根据超类和子类的相似度计算.而超类的相似度为

$$sim_d(supc(s_q),supc(s_d)),$$

子类的相似度为 $\sum_{subc\in d} sim_d(s_q,s_d)$.因此,节点(类)的结构相似度为

$$sim_s(s_q,s_d)=\frac{sim_d(supc(s_q),supc(s_d))+\sum_{subc\in d} sim_d(s_q,s_d)}{\min(|subc(s_q)|,|subc(s_d)|,|subc(s_q)\cap subc(s_d)|)+1} \tag{4}$$

证明:要证明公式(4)为属性节点的结构相似度是合理的,必须分两个方面加以证明:

① 语义的合理性:必须要有结构信息,即满足从超类到类的层次关系.由定义可知是满足的;

② 数值的合理性:相似度的值必须要在 0~1 之间.设

$$a=\min(|subc(s_q)|,|subc(s_d)|,|subc(s_q)\cap subc(s_d)|).$$

由于:

(a) $0\leqslant sim_d(supc(s_q),supc(s_d))\leqslant 1$;

(b) $0\leqslant \sum_{subc\in d} sim_d(s_q,s_d)\leqslant a$.

所以,$0\leqslant(a)+(b)\leqslant a+1$,故 $0\leqslant sim_s(s_q,s_d)\leqslant 1$.得证. □

2) 假若此节点为对象,如果节点 s_q 与 s_d 有相似的祖先节点,并且这两个节点属性相似.这里,两个节点的属性可以分以下 3 种情况来考虑:

① 对于两个节点 s_q,s_d,有 $|T_{s_q}|=|T_{s_d}|$,并且 $sim_T(s_q,s_d)=1$,则有 $T_{s_q}\equiv T_{s_d}$;

② 对于两个节点 s_q,s_d,有包含关系,即 $T_{s_q}\subseteq T_{s_d}\vee T_{s_q}\supseteq T_{s_d}$;

③ 对于两个节点 s_q,s_d,$\exists A: T_{s_q}\subseteq A\wedge T_{s_d}\subseteq A\wedge T_{s_q}\cap T_{s_d}\neq\varnothing$.

这 3 种关系称为属性间的相容关系.

若 $\exists S_D: S_D\in subc(s_d),\exists S_Q: S_Q\in subc(s_q)$,使得 $\exists S_D\forall supc(s_q)(sim_d(S_D,S_Q))>\alpha$,并且 $sim_T(s_q,s_d)>\beta$.其中,α和β分别为概念相似和结构相似的阈值.

此时,节点的结构相似度为(证明同上,略)

$$sim_s(s_q,s_d)=\frac{sim_d(supc(s_q),supc(s_d))+\sum_{p\in T} sim_T(s_q,s_d)}{\min(|T_{s_q}|,|T_{s_d}|,|T_{s_q}\cap T_{s_d}|)+1} \tag{5}$$

3) 假若此节点为属性,则可以直接用属性相似度与所属对象的概念相似度进行计算.故,节点的结构相似度为(证明同上,略)

$$sim_s(s_q,s_d)=sim_d(O(s_q),O(s_d)) \tag{6}$$

此节点的概念相似度为

$$sim_d(s_q,s_d)=sim_p(O_A,O_B) \tag{7}$$

因此,两棵本体树之间对应节点总的相似度为(总相似度的阈值为γ)

$$sim(s_d,s_q)=\frac{a\times sim_d(s_q,s_d)+b\times sim_s(s_q,s_d)}{a+b} \tag{8}$$

1.3 本体树的映射与需求本体树的重组

1.3.1 本体树的映射

根据两棵本体树之间节点的相似度计算,找到大于阈值的那些对应节点($sim(s_d,s_q)\geqslant\gamma$).因此,映射关系函数可以表示为

$$f(m)=\begin{cases}1, & sim(s_d,s_q)\geqslant\gamma\\ 0, & \text{otherwise}\end{cases} \tag{9}$$

由映射关系函数可以得到树之间的映射关系,如图 3 所示.

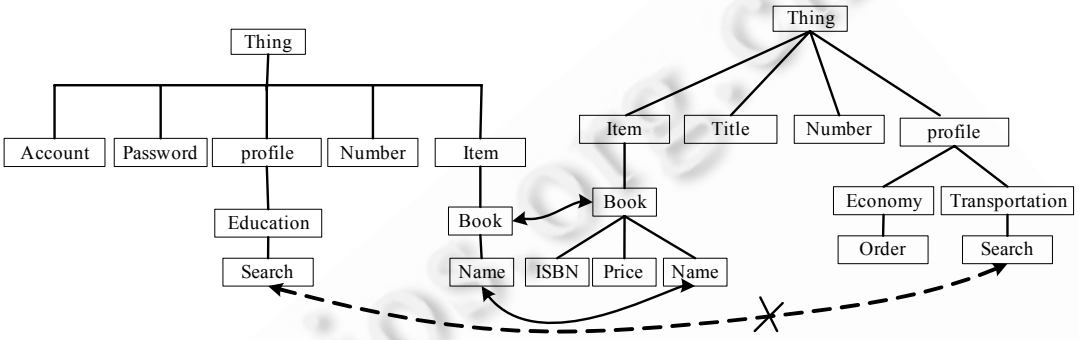


Fig.3 Mapping of ontology trees

图 3 本体树的映射

由于 Book 为对象节点,故利用公式(1)和公式(5)分别计算出该节点的概念相似度和结构相似度.假设权值 a=0.8,b=1,γ=0.5,代入公式(8)中可得 $sim(s_d,s_q)=\frac{0.8\times1+1\times1}{0.8+1}$,即 $sim(s_d,s_q)$=1,$sim(s_d,s_q)\geqslant\gamma$.此时,由映射关系函数公式(9)可知,图中的 Book 与 Book 的 $f(m)$=1,故存在映射关系.同理,Name 与 Name 也存在映射关系;由于 Search 节点为类节点,故利用公式(1)和公式(4),将计算的结果代入公式(8)中可得 $sim(s_d,s_q)<\gamma$,故不存在映射关系.

1.3.2 需求本体树的重构

为了获得更大的查全率和查准率,仅仅使用公式(7)计算总的相似度、再利用映射关系函数(公式(8))直接确定映射结果是不够的,通常需要对需求本体树进行重构.因此,根据第 1.2 节对本体树的对应节点进行度量(分层分类的方法)后,得到对应节点的概念相似度、属性相似度和结构相似度的值.然后,根据不同分类节点的概念相似度和结构相似度的取值范围确定约束条件,对需求本体树进行必要的调整,以获得更为精确、更为全面的目标服务.对需求本体树进行重构分为 3 个方面,即:替换、添加和删除.见表 3.

① 假若此节点为类节点或对象节点,结构相似度大于阈值β、但概念相似度在 0~1 之间,则将 s_q 替换为 s_d;

② 假若此节点为类节点或对象节点,结构相似度大于阈值β、但概念相似度等于 0,则将 s_d 添加为 s_q 的兄弟节点;

③ 假若此节点为类节点或对象节点,结构相似度小于阈值β,则将节点 s_q 删除;

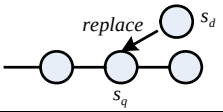
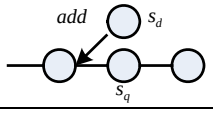
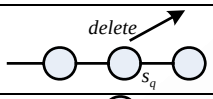
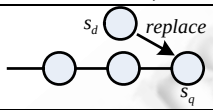
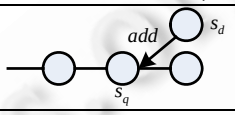
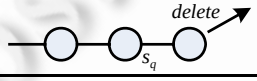
④ 假若此节点为属性节点,结构相似度大于阈值β、但概念相似度在 0~1 之间,则将 s_q 替换为 s_d;

⑤ 假若此节点为属性节点,结构相似度大于阈值β、但概念相似度等于 0,则将 s_d 添加为 s_q 的兄弟节点;

⑥ 假若此节点为属性节点,结构相似度小于阈值β,则将节点 s_q 删除.

Table 3 Restructuring of requirement ontology tree

表 3 需求本体树的重构

约束条件	实例	规则	描述
$(s_q\in C_A,s_d\in C_B)\vee(s_q\in O_A,s_d\in O_B)$, $\beta\leqslant Sim_s(s_q,s_d)\leqslant 1$, $0<Sim_d(s_q,s_d)<1$		$s_d \xrightarrow{replace} s_q$	①
$(s_q\in C_A,s_d\in C_B)\vee(s_q\in O_A,s_d\in O_B)$, $\beta\leqslant Sim_s(s_q,s_d)\leqslant 1$, $Sim_d(s_q,s_d)=0$		$s_d \xrightarrow{add} sibling(s_q)$	②
$(s_q\in C_A,s_d\in C_B)\vee(s_q\in O_A,s_d\in O_B)$, $0<Sim_s(s_q,s_d)<\beta$		$s_q \xrightarrow{delete}$	③
$s_q\in T_A,s_d\in T_B$, $\beta\leqslant Sim_s(s_q,s_d)\leqslant 1$, $0<Sim_d(s_q,s_d)<1$		$s_d \xrightarrow{replace} s_q$	④
$s_q\in T_A,s_d\in T_B$, $\beta\leqslant Sim_s(s_q,s_d)\leqslant 1$, $Sim_d(s_q,s_d)=0$		$s_d \xrightarrow{add} sibling(s_q)$	⑤
$s_q\in T_A,s_d\in T_B$, $0<Sim_s(s_q,s_d)<\beta$		$s_q \xrightarrow{delete}$	⑥

1.4 本体树的匹配算法

本体树的匹配是服务发现的关键,采用一边匹配一边精化的思想,得到最终的用户所需要的服务.

算法的第 1 行和第 2 行是输入和输出:输入为 OWL-S 服务描述文档的 Advertisements(记为 S_A)和用户的需求文档.此处的需求主要是指用户的功能需求,即用户的输入和所期望的输出,记为 Q;输出为服务集.

- 首先,将所有的 S_A (service advertisements)和用户需求 Q 都分别转化了查询本体树和描述本体树;
- 然后,将两棵本体树的节点进行分类分层匹配,并获得每个节点的概念相似度和结构相似度,根据相应的约束条件分别对查询本体树中的节点进行替换、添加和删除操作,以得到最为精化的服务集;
- 最后,将服务集放到集合 *MathchesSet* 中,使用二分法对相似度进行降序排序,并将排序结果放入数组 $S[n]$中,供用户选取.

具体的匹配算法如下:

算法 1.

1. Input: The set of OWL-S Service Advertisement (S_A); A user Requirement (Q);
2. Output: Service collection ($S[n]$).
3. $MathchesSet=\varnothing$;
4. $S[n]=[\varnothing,\varnothing,\ldots,\varnothing]$;
5. $Q\rightarrow T_q$;
6. **for all** $A_i\in S_A$ **do**
7. $A_i\rightarrow T_d$;
8. **for** $i\leftarrow 1,n$ **do**
9. **for all** $s_q\in T_q,s_d\in T_d$ **do**
10. $Sim_d=Sim_d(s_q,s_d)$;
11. $Sim_s=Sim_d(s_q,s_d)$;

12. **while** ($\beta \leqslant Sim_s \leqslant 1$) **then**
13. **if** ($0<Sim<1$) **then**
14. $s_d \xrightarrow{replace} s_q$;
15. **if else** ($Sim_d=0$) **then**
16. $s_d \xrightarrow{add} sibling(s_q)$;
17. **else**
18. **continue**;
19. **end if**
20. **end if**
21. **end if**
22. **end while**
23. **if** $0<Sim_s<\beta$ **then**
24. $s_q \xrightarrow{delete}$;
25. **end if**
26. $sim(s_d, s_q) = \dfrac{a.sim_d + b.sim_s}{a+b}$;
27. $sim = \dfrac{\sum_{s_q \in T_q, s_d \in T_d} sim(s_d, s_q)}{|T_q \cap T_d|}$;
28. **end for**
29. $MathchesSet=\langle A_i, sim\rangle$;
30. **end for**
31. **end for**
32. **for** $i \to 1, n$ **do**
33. $S[i]=binarySort_{sim}(S[i])$
34. **end for**
35. **return** S

2 面向约束的服务发现过程

面向约束的语义 Web 服务发现的过程如下所示(如图 4 所示):

(1) 本体匹配层(核心层).面向约束的语义 Web 服务发现的本质为本体的匹配.将服务提供者的 OWL-S 文档中的 Service Advertisement 进行解析,得到服务的描述本体树与查询本体树进行匹配.首先,分层分类地计算节点的概念相似度、属性相似度和结构相似度,根据概念相似度与结构相似度之间定义一系列的约束关系(在第 1.3.2 节中有详细介绍).利用这些约束关系对查询本体树中的节点进行重构和迭代匹配,以获得更为精确的服务集;

(2) 语义 Web 服务发现层(应用层).对服务请求者的需求文档进行解析,得到查询本体树,与描述本体树匹配.查询本体树与描述本体树之间可能是 1:1 的关系,即,服务请求者所需要的服务刚好有一个完整的服务与之相匹配,无需进行服务组合.也有可能是 1:*n* 的关系,即服务请求者所需要的服务必须经过服务组合才能得到;将获得服务集,按相似度的大小降序排序放入数组中,用户根据自己的偏好选择服务的 Advertisement,然后由机器自动地从服务提供者那里获取服务.

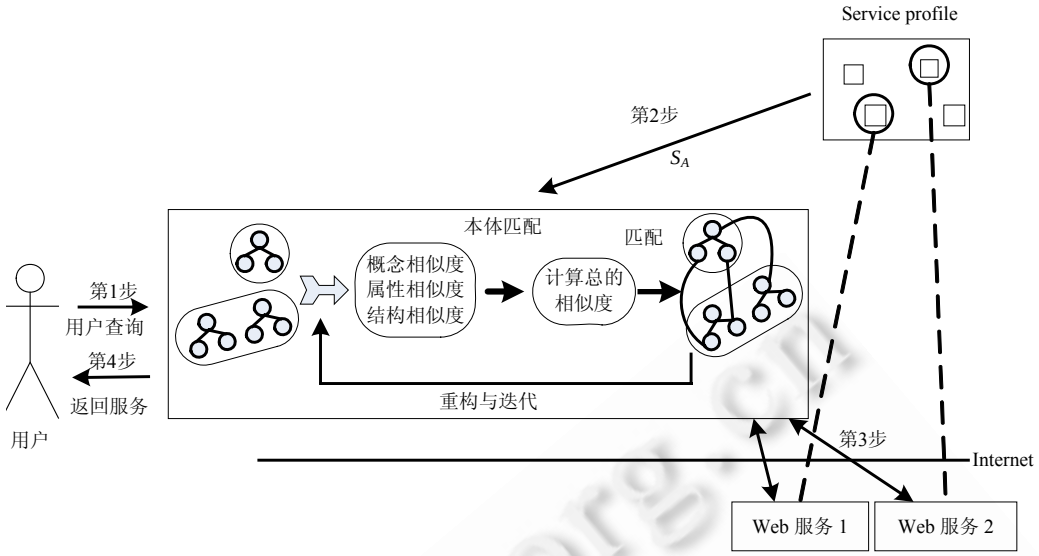


Fig.4 Constraint-Oriented service discovery process

图 4 面向约束的服务发现过程

3 实验与数据分析

本文开发了原型系统 OWLS-CSR,使用了 OWLS-TC version 4.0 中的 1 083 个和手工修改的 29 个 OWL-S advertisements 与 42 个用户查询需求,并与著名的服务匹配器 OWLS-MX 进行比较.利用查准率和查全率对数据进行分析得出,在相关的约束条件下,查准率和查全率高于 OWLS-MX.在整个实验过程中,我们使用的是 OWLS-MX 中的 M4 的配置,而对 OWLS-CSR 中参数的配置分别为 a=0.8,b=1,α=0.5,β=0.5,γ=0.5.

对参数的取值作以下说明:

① 关于 a 与 b 的取值:本文的结构相似度是建立在概念相似度之上的,也只有结构相似度能体现概念之间的语义关系.假若某个概念的结构相似度大于阈值,说明此概念一定满足上下文语义关系,进而可以推断出该节点一定是满足用户需求的节点.故结构相似度的权值一定大于概念相似度的权值,且 b=1.即,在公式 $sim(s_d,s_q)=\frac{a.sim_d(s_q,s_d)+b.sim_s(s_q,s_d)}{a+b}$ 中,$b\geqslant a$ 且 b=1.这一点是可以确定的,所以在实验时取 a=0.8,b=1.大多数学者对权值的确定采用智能算法进行迭代得到,或者根据前人的研究结果给一个经验值,由于受篇幅所限及为突出研究重点,本文对 a 取一个小于 b 的经验值,即 a=0.8,没有对 a 的取值进行计算说明和实验.

② 对于α,β和γ的取值:在整个约束条件中,结构相似度的阈值起决定作用,即:β的取值对匹配结果起关键作用;没有与α相关的约束关系,故与α的取值没有关系,即α可以取[0,1]的任何值.而γ的取值是由α和β决定的,假设取α=0.5,此时γ与β满足线性关系,即 $\gamma=\frac{0.8\times0.5+1\times\beta}{1+0.8}$,$\gamma$是随着$\beta$的增大而增大的.关于$\gamma$与$\beta$的取值将在第 3.2 节中给出详细说明.

3.1 实验环境与评估方法

(1) 实验环境

为了检验本文提出的面向约束的 Web 服务发现方法的可行性和有效性,我们对方法的应用作了应用分析.实验分析的环境为一台笔记本电脑,其中,CPU 为 Intel 双核 T6 系列,主频为 2.20GHz,内存为 2GB.

(2) 评估方法

实验结果采用查准率(precision)和查全率(recall)对实验数据进行评估,定义如下:

- 查准率(Precision): $p=\frac{C}{C+D}$,指的是匹配结果中正确的匹配数与匹配总数的比值;
- 查全率(Recall): $p=\frac{C}{C+E}$,指的是匹配结果中的正确匹配数与全部正确匹配数的比值,

其中,C 表示识别到的正确匹配结果,D 表示识别到的错误匹配结果,E 表示未识别到的正确匹配结果.

3.2 实验数据分析

实验 1. β和γ的取值.

该实验是在同层过滤器 Exact 的条件下进行的,在其他过滤器条件下的曲线趋势类似,因此,以 Exact 过滤器为例,目的是为了得到β和γ的最佳取值,减少因取值大小对实验结果产生不利影响.我们使用 1 112 个 OWL-S advertisements 和 18 个用户查询需求在原型系统 OWLS-CSR 进行取值实验.由于β和γ之间存在线性关系,故对β的初始值设为 0,以后每次增加 0.1,直到 0.9.由此可得到β的取值与服务发现的查准率与查全率之间的关系,如图 5 所示.

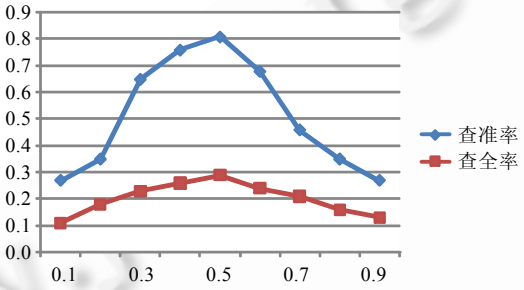


Fig.5 Effect of β value on precision and recall

图 5 β的取值对查准率和查全率的影响

在图 5 中,横坐标表示β的取值.由图 5 可知:在β的取值从 0.1~0.5 的过程中,服务发现的查准率和查全率是逐步升高的,达到 0.5 时,查准率和查全率达到最大值;在β的取值从 0.5 逐步递增到 0.9 的过程中,服务发现的查准率和查全率是依次下降的.这是由于,当β小于 0.5 时,虽然删除的节点变少,但需要插入和替换的节点增多,β的取值越小,插入和替换的节点数越多,这样,越不满足用户的需求;当β大于 0.5 时,虽然插入和替换的节点越来越少,但需要删除的节点数越来越多,此时,也越不能满足用户的需求.所以,β的取值为 0.5 是最为理想的.由此,我们由β与γ的线性关系式 $\gamma=\frac{0.8\times0.5+1\times\beta}{1+0.8}$ 可以得出,γ的取值也为 0.5.

实验 2. 当属性节点存在关系型时对服务发现性能和效率的影响.

从 1 112 个 OWL-S advertisements 中选取属性中具有关系型的 200 个 advertisements,把采用关系函数库进行服务匹配和不采用关系函数库进行服务匹配加以对比,目的是发现采用关系函数库时对实验产生的影响.通过对比得到,采用关系函数的服务匹配能够提高服务发现的查准率,但却牺牲了时间代价,如图 6 所示.

当采用关系函数时,反应时间为 53ms,查准率为 85%.这是由于,虽然减少了节点的重构和匹配迭代所需要的代价,但调用函数库中的函数需要的资源和时间相对较大;当不采用相关的关系函数时,反应时间为 46ms,查准率为 71%.这是由于,调用函数库中的函数所需要的资源和时间代价比节点的重构和匹配迭代所需求的代价要高.但是,采用关系函数的查准率明显高于不采用关系函数的查准率.

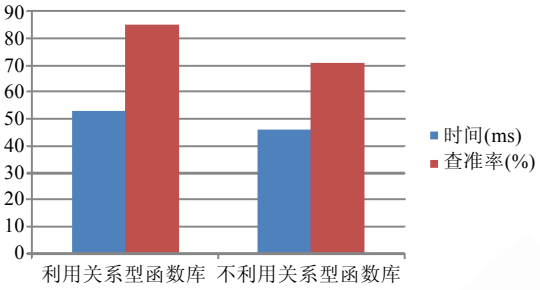


Fig.6 Effect using relation function repository on time cost and precision

图 6 利用关系型函数库对服务发现的时间代价和查准率的影响

实验 3. 服务发现的有效性实验(与 OWLS-M4 的对比).

本文分别在 Exact,Plugin 和 Subsume 这 3 个层次上与 OWLS-M4 进行了比较,而我们忽略了 OWLS-M4 中的 Sibling 和 Subsumed-by 两种过滤器.查准率如图 7 所示,当仅使用概念相似度时,查准率相比 OWLS-M4 要低,并且是所有匹配方法中最低的.使用概念相似度和结构相似度相结合后,查准率有所提高,并高于 OWLS-M4.当我们使用约束条件下的重组后,查准率有较大改善.

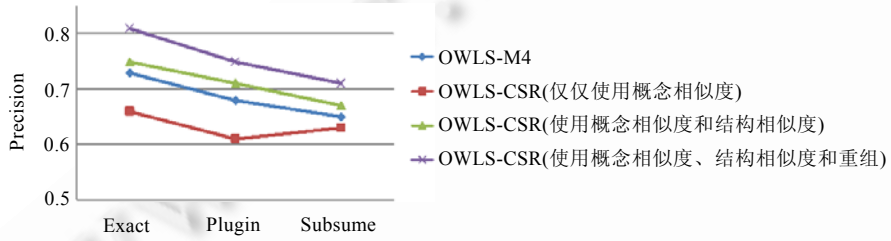


Fig.7 Comparison of precision

图 7 查准率的对比

查全率如图 8 所示,与查准率相似,仅仅使用概念相似度的匹配方法,查全率在所有的匹配方法中最低,也比 OWLS-M4 要低.但当使用结构相似度和带约束条件的重组后,匹配的查全率有了一定的提高,并高于 OWLS-M4 的查全率.

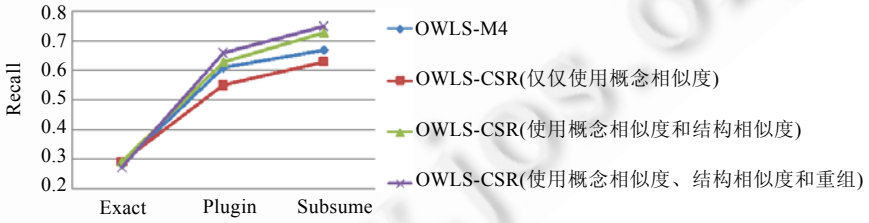


Fig.8 Comparison of recall

图 8 查全率对比

4 相关工作

(1) 基于概念相似度的 Web 服务发现方法

美国卡内基-梅隆大学智能软件实验室首先提出了基于 DAML-S 的 Web 服务发现方法[6],该方法利用本体

描述语言 OWL 将用户请求的输入/输出所对应的概念与服务提供者的输入/输出所对应的概念进行匹配,并实现了服务匹配器[7,8].文献[9]将 OWL-S 的 Service Profile 转化成 UDDI 的数据模型,并通过概念间的本体推理将服务中包含的本体建立索引.当服务发现时,利用服务的本体索引通过 UDDI 的搜索引擎实现基于语义的服务匹配.文献[10]通过在 OWL-S 的基础上扩展 QoS 信息进行服务匹配.

(2) 基于描述逻辑的 Web 服务发现方法

文献[11]利用描述逻辑对 Web Service 进行描述,并为服务设计了描述模型,最终将服务匹配转化为概念匹配.文献[12]将描述逻辑与服务推理结合起来,利用描述逻辑具有清晰模型-理论语义的特点和描述逻辑在概念分层中的优点,通过描述逻辑中对概念包含关系的有效推理和判断能力,提出了基于描述逻辑的服务匹配算法.该算法利用描述逻辑推理机可以为多系统建立服务分层图,然后在该服务分层图的基础上实施服务匹配.文献[13]将服务的自动发现问题转化成最佳覆盖问题,并利用描述逻辑作为形式化工具对此问题进行描述,然后基于超图(HyPergraPh)算法得出最佳覆盖,进而实现服务的自动发现.文献[14]利用 Pi 演算描述服务的动态属性,而采用描述逻辑刻画服务的静态属性,提出了基于 Pi 演算和描述逻辑的服务匹配算法,统一处理匹配过程中用户和服务在语义、时序结构和安全等方面的约束.文献[15]使用分布式描述逻辑描述相互关联的异构分布式本体,以适应语义 Web 服务的动态性和处理来自多信息源的异构分布式本体的进化和更新,提出了优先分布式知识库(PDK)的概念,并在 PDK 的基础上给出了 Web 服务的语义查询,从而实现了 Web 服务动态发现.

(3) 其他的 Web 服务发现方法

以美国乔治亚大学大规模分布式信息系统实验室(LSDIS)与 IBM 联合提出的 WSDL-S 为基础(WSDL-S 对 WSDL 作了语义扩展),文献[23]以 WSDL-S 为服务描述语言,从领域有关本体和领域无关本体两个角度进行服务匹配,最后将两方面的匹配结果相结合,得出最终的服务匹配结果.文献[24]利用流程本体刻画服务语义,流程本体包含属性、值、任务、资源和异常等基本元语.基于流程本体,提出了流程查询语言 PQL(process query language),从而将服务匹配问题转化为语言的模式匹配,进而采用基于模式匹配的算法实现服务匹配.文献[25]基于案例推理的方法实现服务的匹配和发现.文献[26]在 Web 服务描述中引入语义、功能和行为约束以及服务质量(QoS),对 WSDL 进行扩展,提出轻量级的 Web 服务匹配方法.文献[27]提出一种基于上下文感知的 Web 服务选择方法.该方法利用本体描述与服务提供者进行交互的上下文,根据上下文的语义关系选择用户所需的服务.文献[28]提出了一种经验的方法,该方法通过搜索引擎利用网页的数量和文本片断来估计两个单词的语义相似度.

5 结论及进一步工作

本文将用户需求和 Web Service Profile 分别转化为本体树,因此把服务匹配问题转化为本体匹配问题.采用分层分类的方式分别计算出本体树中各个节点的概念相似度和结构相似度.根据概念相似度和结构相似度之间的关系,定义了一系列的重组约束,对查询本体树进行重组,以提高服务发现的查准率与查全率.最后,开发了原型系统 OWLS-CSR,并与 OWLS-M4 进行了对比实验,证明了这一方法的可行性与有效性.下一步工作将关注服务组合中由于服务提供者违反隐私策略,需要重新绑定而发现新的服务的问题.

References:

[1] Berners-Lee T, Fischetti M, Dertouzos ML. Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web. Harper, 1999.

[2] Yue K, Wang XL, Zhou AY. Underlying techniques for Web services: A survey. Journal of Software, 2004,15(3):428−441 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/428.htm

[3] WSDL 1.1. 2001. http://www.w3.org/TR/wsdl

[4] Papazoglou M, Georgakopoulos D. Service oriented computing. Communications of the ACM, 2003,46(10):25−28.

[5] Burstein M, Bussler C, Zaremba M, Finin T, Huhns MN, Paolucci M, Sheth AP, Williams S. A semantic Web services architecture. IEEE Internet Computing, 2005,9(5):72−81. [doi: 10.1109/MIC.2005.96]

[6] Paolueei M, Vannura KT, Payne TR, Syeara K. Semantic matching of Web services capabilities. In: Proc. of the Int'l Semantic Web Conf. (ISWC). 2002. http://www.springerlink.com/content/9lwn9fkclngm6trl/ [doi: 10.1007/3-540-48005-6_26]

[7] Kawamura T, De Blasio JA, Hasegawa T, Paolueei M, Sycara K. Preliminary report of public experiment of semantie service matchmaker with UDDI business registry. In: Proc. of the Int'l Conf. on Service Oriented Computing. 2003. http://www.springerlink.com/content/rb5pqd6fgj14p1wc/ [doi: 10.1007/b94513]

[8] Srinivasan N, Paolueei M, Sycara K. Adding OWL-s to UDDI, implementation and through put. In: Proc. of the 1st Int'l Workshop on Semantic Web Services and Web Process Composition. 2004. 6−9. http://en.scientificcommons.org/43326907

[9] Luo J, Montrose B, Kim A, Khashnobish A. Adding OWL-S support to the existing UDDI infrastructure. In: Proc. of the IEEE Int'l Conf. on Web Service (ICWS). 2006. http://dl.acm.org/citation.cfm?id=1173175 [doi: 10.1109/ICWS.2006.26]

[10] Shen GH, Huang ZQ, Zhang YP, Zhu XD, Yang J. A semantic model for matchmaking of Web services based on description logics. Fundamenta Informaticae, 2009,175(96):211−226.

[11] Zhou C, Chia LT, Lee BS. Service discovery and measurement based on DAML-QoS ontology. In: Proc. of the Int'l World Wide Web Conf. 2005. 1070−1071. [doi: 10.1145/1062745.1062873]

[12] Shi ZZ, Jiang YC, Zhang HJ, Dong MK. Agent service matchmaking based on description logic. Chinese Journal of Computers, 2004,27(5):625−635 (in Chinese with English abstract).

[13] Benatallah B, Haeid MS, Leger A, Rey C. On automating Web services discovery. Int'l Journal on Very Large Data Bases, 2005, 14(l):84−96. [doi: 10.1007/s00778-003-0117-x]

[14] Agarwal S, StUder R. Automatic matchmaking of Web services. In: Proc. of the15th Int'l Conf. on World Wide Web. 2006. 1057−1058. http://dl.acm.org/citation.cfm?id=1136013 [doi: 10.1109/ICWS.2006.35]

[15] Ma YL, Jin BH, Feng YL. Dynamic discovery for semantic Web services based on evolving distributed ontologies. Chinese Journal of Computers, 2005,28(4):603−614 (in Chinese with English abstract).

[16] Klusch M, Fries B, Sycara K. OWLS-MX: A Hybrid Semantic Web Service Matchmaker for OWL-S Services. Journal of Web Semantics, 2009,7(2):121−133.

[17] Fensel D, Lausen H, Polleres A, Bruijn JD, Stollberg M, Roman D, Domingue J. Enabling Semantic Web Services: The Web Service Modeling Ontology. Springer-Verlag, 2006.

[18] OWL-S 1.1 release: Examples. 2004. http://www.daml.org/services/owl-s/1.1/examples.html

[19] Meditskos G, Bassiliades N. Structural and role-oriented Web service discovery with taxonomies in OWL-S. IEEE Trans. on Knowledge and Data Engineering, 2010,22(2):178−290.

[20] Gruber T. A translation approach to portable ontology specifications. Knowledge Acquisition, 1993,5(2):199−220. [doi: 10.1006/knac.1993.1008]

[21] Li JZ, Tang J. RiMOM: A dynamic multistrategy ontology alignment framework. IEEE Trans. on Knowledge and Data Engineering, 2009,21(8):1218−1232. [doi: 10.1109/TKDE.2008.202]

[22] Pantel P, Lin D. Discovering word senses from text. In: Proc. of the 2002 ACM SIGKDD Conf. on Knowledge Discovery and Data Mining. Edmonton, 2002. 613−6l9. [doi: 10.1145/775047.775138]

[23] Syeda-Mahmood T, Shah G, Akkiraju R, Ivan AA. Searching service repositories by combining semantic and ontological matching. In: Proc. of the IEEE Int'l Conf. on Web Service. 2005. http://dl.acm.org/citation.cfm?id=1092135 [doi: 10.1109/ICWS.2005.102]

[24] Klein M, Bemstein A. Toward high-precision service retrieval. IEEE Internet Computing, 2004,8(11):30−36. [doi: 10.1109/MIC.2004.1260701]

[25] Osman T, Thakker D, AI-Dabass D. Semantic-Driven matchmaking of Web services using case-based reasoning. In: Proc. of the IEEE Int'l Conf. on Web Service (ICWS). 2006. http://dl.acm.org/citation.cfm?id=1173147 [doi: 10.1109/ICWS.2006.118]

[26] Hu JQ, Zou P, Wang HM, Zhou B. Research on Web service description language QWSDL and service matching model. Chinese Journal of Computers, 2005,28(4):505−513. (in Chinese with English abstract).

[27] Sensoy M, Yolum P. Ontology-Based service representation and selection. IEEE Trans. on Knowledge and Data Engineering, 2007, 19(8):1102−1115. [doi: 10.1109/TKDE.2007.1045]

[28] Bollegala D, Matsuo Y, Ishizuka M. A Web search engine-based approach to measure semantic similarity between words. IEEE Trans. on Knowledge and Data Engineering, 2011,23(7):977−990. [doi: 10.1109/TKDE.2010.172]

附中文参考文献:

[2] 岳昆,王晓玲,周傲英.Web 服务核心支撑技术:研究综述.软件学报,2004,15(3):428−441. http://www.jos.org.cn/1000-9825/15/428.htm

[12] 史忠植,蒋运承,张海俊,董明楷.基于描述逻辑的主体服务匹配.计算机学报,2004,27(5):625−635.

[15] 马应龙,金蓓弘,冯玉琳.基于进化分布式本体的语义 Web 服务动态发现.计算机学报,2005,28(4):603−614.

[26] 胡建强,邹鹏,王怀民,周斌.Web 服务描述语言 QWSDL 和服务匹配模型研究.计算机学报,2005,28(4):505−513.

柯昌博(1984－),男,陕西安康人,博士生,主要研究领域为语义 Web 服务,信息系统的安全与隐私,基于本体的软件工程.

刘林源(1981－),男,博士,讲师,主要研究领域为信息系统安全,软件工程.

黄志球(1965－),男,博士,教授,博士生导师,CFF 高级会员,主要研究领域为软件工程,形式化方法,数据仓库.

曹子宁(1972－),男,博士,教授,博士生导师,主要研究领域为形式化方法.