具有QoS约束的语义Web服务发现的研究*)

李春梅[1,2] 蒋运承[2,3]

(云南楚雄师范学院计算机科学系 楚雄 675000)[1]

(广西师范大学计算机科学与信息工程学院 桂林 541004)[2] (中山大学计算机科学系 广州 510275)[3]

摘 要 传统的Web服务主要是基于UDDI的技术规范,只提供了按照简单分类和关键字的服务发现方法,造成查准率与查全率低,影响服务复用和服务组合的相容性。带有语义的Web服务通常也只关注服务功能性的匹配,因缺乏服务质量描述和灵活、有效的服务匹配方法,而难以保证服务的全面匹配和快速定位。本文分析了现有的一些相关研究,在Web服务模型的基础上,结合"逐渐逼近"的思想,提出一个三层次的匹配筛选算法,并对基本描述、IOPE、服务质量各层的匹配算法进行了深入讨论,用相似函数来度量服务提供方与服务请求方的Web服务相似程度,为Web服务发现提供了一种有效的方法。

关键词 服务发现,服务匹配,服务质量,相似函数,相离度

Study on Semantic Web Services Discovery with QoS Constraint

LI Chun-Mei[1,2] JIANG Yun-Cheng[2,3]

(Department of Computer Science, Chuxiong Normal University, Chuxiong 675000)[1]

(College of Computer Science and Information Engineering, Guangxi Normal University, Guilin 541004)[2]

(Department of Computer Sciences, Sun Yat-sen University, Guangzhou 510275)[3]

Abstract Traditional Web services only provide simple classification and keywords-based methods based on UDDI technical specification mainly, lead to low precision and recall, compatibility is influenced of service reuse and composition. Web services with semantics also always set focus on function matching, thorough matching and quick location are guaranteed difficulty without quality of service and vivid, efficiency matching method. In this paper, existing related researches are analyzed, on Web service model, combine the thought of"approach gradually", a kind of matching and filter algorithm with three level is presented, each of algorithm, namely basic description, IOPE and QoS matching is thoroughly discussed, similarity functions is introduced to measure Web service similarity degree of provider and requester, this approach will be significant to Web services discovery.

Keywords Services discovery, Services matching, Quality of service, Similarity functions, Deviation degree

1 引言

在动态、多样的Internet服务市场中,发现合适的服务是实现服务共享、复用的重要前提,因此,Web服务发现作为面向服务体系结构的一个重要组成部分,备受工业界和学术界关注[1,2]。服务发现的主要过程是进行服务匹配,传统的Web服务匹配,是通过对UDDI上的服务注册信息进行关键词精确匹配实现的,这种方法存在以下缺点:(1)服务查全率和查准率低,匹配结果中存在大量垃圾信息,增加了人工寻找及筛选的工作量;(2)服务描述基本上没有语义,所以无法识别服务之间所具有的联系。

针对传统Web服务发现技术中出现的问题,目前国内外关于发现的匹配算法讨论得还是比较多的,如:K Sycara[3]、K Arisha[4]、GJ Wickler[5]分别提出不同的服务匹配算法,文[6]和[7]中提出的匹配算法,基本上都是从功能性和接口上进行的匹配,只将请求者的输出与现存服务的所有输出相匹配,以及将服务的输入与请求者提供的输入相匹配。这种匹配方法的效率是较低的,尤其当本体库的规模越庞大,概念之间的传递关系越复杂,进行大规模搜索时消耗时间也越多。然而在通常情况下,服务请求者的需求也是多层次的,比如对于一个网上酒店预定服务来说,有的客户方只要求满足基本的入住条件,而有的客户不仅有基本条件方面的要求,而且想得到高质量的网络服务,因此他对服务质量(QoS)方面(如时间、费用、可靠性等)也有要求,而有的客户在面对满足前面条件的一大批服务来说,当他在决定选择哪一家酒店时,还会考察酒店的信誉度是好还是不好?自己对酒店提供的服务条件(包括硬件设施与软件设施)是否达到预期的满意,因此,在进行服务匹配与服务定位时,要充分考虑不同用户对服务的不同需求,以体现个性化的网络服务。

2 Web服务描述模型

在提出匹配算法前,先给出Web服务描述模型,我们可定义一个通用的Web服务描述模型为:

$WS=\{S,F,NF,C\}$

其中S是基本描述,是服务的公共属性,包括服务分类、服务ID、服务名称、服务提供者ID、服务提供者名称、文本描述、联系方式、版本等,基本描述是概要性的描述。

F是服务功能描述,包括输入与输出,前提与结果等,是

*)基金项目:国家自然科学基金项目(60573010)和云南楚雄师院基金项目(05-YJQN02)。李春梅 讲师;蒋运承 副教授,硕士生导师,博士,博士后。

服务请求者判断 Web 服务能否满足其功能需求的主要依据。

NF 是非功能的属性描述，比如代价(Cost)、响应时间(Time)、服务优先级(Priorities)等非功能属性，这些属性反映了一个 Web 服务的性能，我们把这些属性称为 QoS 属性。

C 是一个 Web 服务的信誉值，该值反映了一个 Web 服务信誉的好坏，服务请求方可根据该信誉值作为判断服务提供方是否值得与其进行交易的依据。

3 语义 Web 服务匹配策略分析与匹配器设计

通常情况下，普通用户由于缺乏一定的发散性思维训练，因此难以通过关键词来准确的表达其查询需求，然而如果能提供大量较相关的词汇，让用户选择的话，就要容易得多，因此，在进行查询时，用户可以先通过一些不太准确的查询条件进行搜索，而在所得的查询结果中，往往能够发现一些更准确的描述词汇，然后可以根据需要更改查询条件重新检索，以得到更准确的查询结果。这种方式需要经过多次查询就能"逐渐逼近"用户所需的信息。

结合"逐渐逼近"所述的思想与 Web 服务模型 $WS=\{S, F, NF, C\}$ 结构，在 Web 服务中，服务匹配也可以是个逐步精化过程：经由基本描述匹配筛选的候选服务再参与功能匹配，合格的服务进一步参与具有 QoS 非功能属性的匹配，这就可以满足服务请求者不同层次的需要。因此，对语义 Web 服务的匹配，我们采取的是三级细粒度匹配策略，这三级匹配的作用和地位并不完全一样。第一级属于过滤性质的匹配，其作用主要是为第二级的功能匹配过滤大量的无关的垃圾服务。因为发布在注册中心的 Web 服务的数量是海量的，其中绝大部分服务是与请求者所请求的服务相差很远的，对这些服务就没有必要进行复杂的功能性匹配。第二级是在第一层匹配的基础上进一步进行功能上的匹配筛选，再次缩小了匹配的服务数。第三级根据用户的特殊需要进行服务质量上的匹配，因为不同的请求者对 Web 服务的 QoS 要求是不同的，是动态变化的，QoS 所反映的是服务的"性格"特征，这些虽然和服务的执行过程无关，但是会直接影响请求者使用服务的效果，因此在功能语义匹配的基础上，很有必要对服务提供者的服务 QoS 和服务请求者的服务 QoS 进行匹配，以满足用户更高级的需求。其匹配过滤器设计如下：

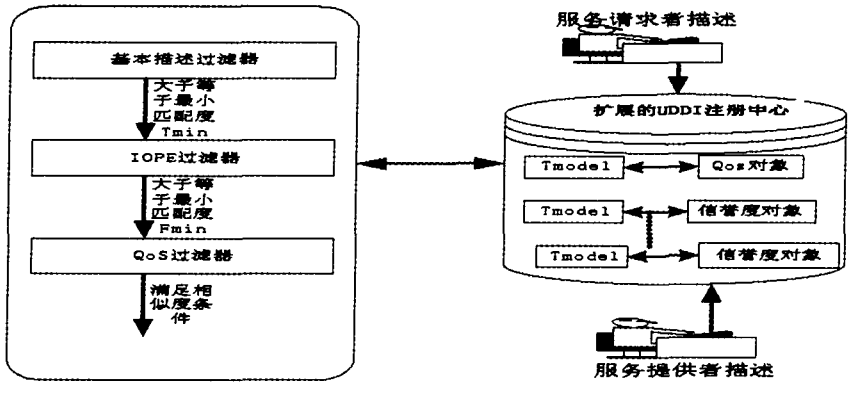

图 1 匹配过滤器

匹配引擎主要包含三个过滤器，在每一级分别设置一个满足匹配的阈值。

基本描述匹配过滤器：其主要功能是将服务请求者的请求描述与发布在扩展的 UDDI 注册中心提供的所有服务描述逐一进行比较，如果第一级上不能满足规定的匹配度就说明了服务不是所请求的服务，因此该服务的匹配就可以到此为止，不再进行后面的匹配。如果提供服务的综合相似度大于等于规定的最小匹配度 T_{min}，则将该服务添加到第一级的匹配服务记录集中，以备进行第二级上的功能匹配。

IOPE 匹配过滤器：其主要功能是对基本描述匹配成功的服务，从第一级的匹配服务记录集中对提供服务与请求服务的输入参数、输出参数、前置与结果分别进行匹配，然后进行综合相似度的计算。如果目标服务在第二级上的综合相似度不能满足规定的最小匹配度 F_{min}，就说明了服务不是所请求的服务，因此该服务的匹配就也可以到此为止不再进行后面的匹配，把满足条件的服务添加到数据库的第二级的匹配服务记录集中，以备进行第三级上的 QoS(服务质量)匹配。

QoS 匹配器：其主要功能是对在 IOPE 上已成功匹配的服务进行请求服务与提供服务上的 QoS 匹配，如果目标服务在第三级上满足规定的相似度条件，则把该服务添加到数据库的第三级的匹配服务记录集中。QoS 匹配器同时将根据匹配度排列所有通过该级匹配的服务，并可把服务提供给请求者。

4 基于语义的服务匹配算法

发现服务的关键，是服务匹配。服务匹配是将请求者提供的所需服务的描述与服务提供者已发布的服务描述进行匹配，判断后者是否满足匹配。从理论上来讲，当一个发布的服务描述与一个请求的服务描述一模一样时，这是最完美的服务匹配。然而，因为服务提供者与服务请求者双方事先不可能就服务的要求达成一致的协定，显然，这种定义过于严格，势必会导致服务匹配的失败。这就需要匹配算法具有一定的灵活的匹配能力，好的匹配算法对服务的匹配应有一定的松驰度，使得服务请求者的要求与服务提供者的描述按一定的相似度进行匹配。这种相似可以是基于语法的，也可以是基于语义的，以保证一定的查准率，也要保证提高查全率。根据前面所做分析，从三个层次来对整个服务进行匹配筛选，各层采用不同的匹配算法来实现。

4.1 第一级基本描述的相似匹配

在基本描述的这些信息中，不是每一个信息都是很重要的，我们可挑选最重要的三个信息来计算匹配度，即服务分类(service category)、服务名称(service name)和文本描述(text description)。

由于服务分类、服务名称与文本描述是字符串，因此它们的比较实质上是文本相似性比较，我们采用向量空间模型[8,9]算法。文本的内容通常用它所包含的语言特征来表

示，在通常的文本表示模型中通常是使用关键词作为特征项。文本 D 可以用项集表示为 $D\{t_1,t_2,t_3,t_4,\cdots,t_n\}$，其中 n 是用于表示文本的特征项的个数。对于 n 个特征项的文本 $D\{t_1,t_2,t_3,t_4,\cdots,t_k,t_n\}$，项 t_k 通常被赋予一定的权重 W_k(Term Weight)，用于表示该特征项对于文本的重要程度。我们把 $t_1,t_2\cdots,t_n$ 视为坐标系，那么 w_1，w_2，$\cdots$，w_n 就是相对应的坐标值，$D\{w_1,w_2;\cdots,w_n\}$就可以被看作为 n 维空间中的一个向量，称 $D\{w_1,w_2;\cdots,w_n\}$为文档 D 的向量表示。重要程度越高，那么相应的权值就越大。权值大小可以是专家根据领域知识或者经验人为的确定，但是这种方法的随意性强，效率低，很难应用到大规模的真实文本处理中，所以采用的是基于统计的方法，用 TF-IDF 方法，进行文本特征提取，也就是通过计算已选出来的每个词在文本和训练文本中出现的词频、词对共现的频率来计算词的权重，并计算其对文本的分辨能力，权重的计算公式如下：

$$wt_i(d)=|tf_i(d).\log|\frac{N}{nt_i}+0.5\|/\sqrt{\sum_{j=1}^{n}(tf_i(d))^2.\log^2|\frac{N}{nt_i}+0.5|} \quad (1)$$

其中，$wt_i(d)$为词在文本 D 中的权重，而 $tf_i(d)$为词在文本 D 中的词频，N 为训练文本的总数，nt_i 为训练文本集中出现 t_i 的文本数，分母为归一化因子。当文本被表示为向量空间模型时，我们可以借助向量之间的某种距离来表示文本之间的相似程度 $Sim(D1,D2)$，通常用两个文本 $D1$ 和 $D2$ 之间的向量的夹角余弦来表示：

$$Sim(D1,D2)=\frac{\sum_{i=1}^{n}W_{1i}\times W_{2i}}{\sqrt{\sum_{i=1}^{n}W_{1i}^2}\times\sqrt{\sum_{i=1}^{n}W_{2i}^2}} \quad (2)$$

我们根据上式可算出上述服务提供者与服务请求者在服务分类(service category)、服务名称(service name)和文本描述(text description)三个方面的相似度，然后综合这三种相似度，得到第一级的基本描述相似度如下：

$$b_{sim}(Ps,Rs)=w_1*sc_{sim}(Ps,Rs)+w_2*sn_{sim}(Ps,Rs)+w_3td_{sim}(Ps,Rs) \quad (3)$$

$sc_{sim}(Ps,Rs)$是服务分类的相似度，$sn_{sim}(Ps,Rs)$是服务名称的相似度，$td_{sim}(Ps,Rs)$是文本描述的相度。

当提供服务与请求服务的基本描述的综合相似度大于或等于一个最小相似度的阈值 T_{min}时，$b_{sim}(p_s,r_s)$满足第一级匹配，即可将该提供服务添加到第一级的匹配服务记录集 Record1 中，从而可进入第二级的服务功能匹配。

4.2 第二级服务功能的匹配(IOPE 的匹配)

通过与本体中的类相关联，实现服务功能的语义描述，是语义匹配实现的基础。需要把服务请求者功能描述与第一级上筛选出的匹配服务记录集 Record1 中的提供服务的功能描述进行匹配。为了能更方便地介绍 IOPE 的匹配算法，首先对一些符号的简记作出说明：Pins 表示服务提供者输入参数的集合，Rins 表示服务请求者输入参数的集合，Pous 表示服务提供者输出参数的集合，Rous 表示服务请求者输出参数的集合，Pprs 表示服务提供者的前提条件参数的集合，Rprs 表示服务请求者前提条件参数的集合，Pes 表示服务提供者结果影响参数的集合，Res 表示服务请求者结果影响参数的集合。这里需注意的是这四个参数都是本体中的概念而不是原子数据类型。二维函数 $F_{sim}(Ps,Rs)$计算服务的功能性特征(IOPE)的匹配度，其参数是提供服务(Ps)和请求服务(Rs)，结果是 0 到 1 之间的实数值。其功能相似度公式如下：

$$F_{sim}(Ps,Rs)=u_1\,sim(Ps.\,Pins,Rs.\,Rins)+u_2\,sim(Ps.\,Pous,Rs.\,Rous)+u_3\,sim(Ps.\,Pprs,Rs.\,Rprs)+u_4\,sim(Ps.\,Pes,Rs.\,Res) \quad (4)$$

u_1、u_2、u_3、u_4 分别是输入集、输出集、前提条件集、结果影响集的权值，$\sum u_i=1$，且 $0\leqslant u_i\leqslant 1$。可由用户决定，如果用户没有对权值的要求，则可采用默认的平均权值，即这四个方面是同等重要。

下面对相似函数 sim(Ps. Pins,Rs. Rins)进行说明。从本体概念类之间的各种关系来考虑，当提供服务的一个输入参数和请求服务的一个输入参数在同一个本体中定义时，有以下四种可能出现的相似值：

(1)精确匹配(exact)：该匹配要求请求服务的 $Rs.\,ins_i$ 与提供服务的 $Ps.\,ins_i$ 在语义上是等价的，即二概念是同一个概念($Ps.\,ins_i=Rs.\,ins_i$)；

(2) 包含匹配(subsumer)：即提供服务的 $Ps.\,ins_i$ 是请求服务的 $Rs.\,ins_i$ 的父概念(subclass of,($Rs.\,ins_i$, $Ps.\,ins_i$))；

(3) 插入匹配(plugin)：概念 $Rs.\,in_i$ 是概念 $Ps.\,ins_i$ 的父概念(subclass of ($Ps.\,ins_i$,$Rs.\ ins_i$))；

(4) 相交匹配(overlap)：二概念在语义上没有直接的关联，$Ps.\,ins_i\neq Rs.\,ins_i$ 即请求服务与提供服务二者有共同的交集，交集不为空。

对于以上的四种情况，则有如下输入相似度的计算。

$$Sin(Ps.\,Pins,Rs.\,Rins)K=\begin{cases}1 & ①\\ 1 & ②\\ \dfrac{|Rs.\,Rins|}{|Ps.\,Pins|} & ③\\ \sqrt{\dfrac{|Ps.\,Pins|\cap|Rs.\,Rins|}{|Ps.\,Pins|\cup|Rs.\,Rins|}*\dfrac{|Ps.\,Pins|\cap|Rs.\,Rins|}{|Rs.\,Rins|}} & ④\end{cases} \quad (5)$$

对于第四种情况，服务提供者与服务请求者的相应概念之间没有直接的联系。在这种情况下，可根据 Tversky[10,11] 的基本特征的相似性模型来评估两个概念的相似性，模型要求将两个概念的特征分为两种：它们的共同特征和它们的不同特征。共同的特征能够增强两个概念的相似性，而不同的特征则会减弱相似性，但是共同特征对相似度的增强影响要大于不同特征减弱相似度的影响。所以，在评价相似度的时候，会给予概念的共同特征以更大的信任度。本文在 Tversky 模型的基础上提出如下计算公式来度量提供服务与请求服务的相似度：其中，$\frac{|Ps.\,Pins|\cap|Rs.\,Rins|}{|Ps.\,Pins|\cup|Rs.\,Rins|}$ 表示提供服务与请求服务的共同的输入参数的个数与两者总的输入参数个数之比，$\frac{|Ps.\,Pins|\cap|Rs.\,Rins|}{|Rs.\,Rins|}$则表示匹配的输入参数在请求输入参数中所占的比率。

输出参数相似度 $sim(Ps.\,Pous,Rs.\,Rous)$、前提条件相似度 $sim(Ps.\,Pprs,Rs.\,Rprs)$、结果 $sim(Ps.\,Pes,Rs.\,Res)$相似度的计算方法与输入参数的相似度计算类似。当各项计算出相似度时，按公式(4)计算综合相似度 $F_{sim}(Ps,Rs)$。

当一个提供服务与请求服务的 IOPE 描述的综合相似度大于或等于一个最小相似度的阈值 F_{min}时满足第二级匹配，即可将该提供服务添加到第二级的匹配服务记录集中，从而可进入第三级的匹配。

4.3 第三级非功能属性 QoS 的匹配

4.3.1 QoS的模型

目前在Web服务发现的研究中，为了能够体现出服务请求者对服务性能的不同要求，一些学者对服务的QoS方面做了量化研究[12~16]，文[17]中著名学者Jorge Cardoso指导性地给出了服务质量模型中应包括的因素，具体包括费用(cost)、时间(time)、可靠性(reliability)和可信性(fidelity)。文[18]在AgFlow系统中选择以下5个通用的服务属性来评价QoS服务：服务运行成本(execution price)、运行时间(execution duration)、信任度(reputation)、成功率(successful executionrate)和可用性(availability)，文[19]提出了把服务属性分成多类，并扩大到加密、安全等属性方面，量化结果为这几个属性的取值加权后的总和。这样的量化存在着两方面的不足：

(1)这些量化的属性与服务领域无关。实际上一个Web服务是与具体的领域相关的，服务的领域属性包括与服务的业务内容、服务上下文以及服务供应商等相关的信息，它们是服务使用者在量化Web服务质量以及选取Web服务时考虑的重要因素。以酒店预订服务为例，用户往往选择入住率较高的酒店提供的订房服务，即使该酒店的酒店预订服务的响应时间相对较长，酒店的入住率表明了酒店受欢迎的程度，它是酒店预订服务的领域属性。

(2)这些用来量化的非功能属性所使用的度量单位不同，如时间使用秒，费用使用元，而对于服务可靠性、真实性没有单位，用属性的取值加权后的总和来进行量化是不能真实地反映出具体的Web服务的性能的。

基于上面所述的不足，在本文中关于QoS匹配的研究，提出一种设计模式，对任意一个Web服务 i 有如下QoS模式：

$$QoS(i)=\langle \text{tyattribute}, \text{lyattribute}\rangle$$

其中tyattribute是通用属性，适合于任何一个Web服务，是一个可扩展的属性，如时间(Time)、费用(Cost)、可靠性(Reliability)、真实度(Fidelity)等。而lyattribute是领域内专门的属性，如位置，入住率等。领域属性也是一个可扩展的属性，由领域专家根据具体的服务类别设定出相关的属性，具体过程在本文中不做详细讨论。

4.3.2 QoS的匹配算法

下面讨论提供服务与请求服务之间QoS匹配算法，由于实际生活中不同的用户对这些属性有不同表达方式，如有的人喜欢用一个具体的数值表示属性，如time=40秒，而在更多的实际情况下，请求方对服务的属性是不能用一个明确的数值来表达的，例如对于服务时间，请求方有可能要求为在[50,90]秒之间的范围内能承受。在本文中针对两种不同的表达方式，提出不同的量化方法，其框图如图2所示。对于数值属性的量化，规定一个最小相似度，对于区间属性的量化，规定一个最大相离度，无论用何种方法计算出服务提供者与服务请求者的匹配，都添加进第三级的匹配表，以进行下面的服务选取。

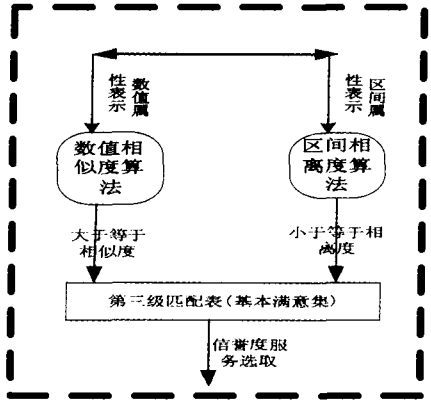

图2 QoS匹配过滤器

1)数值属性的QoS匹配。函数QoS(Ps, Rs)计算提供服务和请求服务的QoS相似度。通过计算Ps和Rs的QoS中各维相似度的几何距离来计算。函数返回一个0到1之间的实数，返回值越接近1，Ps和Rs就越相似。

$$QoS(Ps,Rs)=\sqrt[3]{Qosd(Ps,Rs,\text{属性 1})*Qosd(Ps,Rs,\text{属性 2})*\cdots Qosd(Ps,Rs,\text{属性 } n)} \tag{6}$$

$$Qosd(Ps,Rs,\text{属性 } n)=\sqrt[3]{d_{\min}(Ps,Rs,\text{属性 } n)*d_{\text{avg}}(Ps,Rs,\text{属性 } n)*d_{\max}(Ps,Rs,\text{属性 } n)} \tag{7}$$

有三种距离函数：$d_{\min}(Ps,Rs,\text{属性 } n)$，$d_{\text{avg}}(PS, RS,\text{属性 } n)$和$d_{\max}(Ps, Rs,\text{属性 } n)$.

$d_{\min}(Ps, Rs,\text{属性 } n)$可定义如下：

$$d_{\min}(Ps,Rs,\text{属性 } n)=1-\frac{|\min(Ps.\text{属性 } n)-\min(Rs.\text{属性 } n)|}{\min(Rs.\text{属性 } n)} \tag{8}$$

另外两个函数的定义类似，用max, avg代替min即可。函数min、max、avg分别返回参数中指定的QoS的那一维中的最小值、最大值和平均值。下表举例说明如何计算QoSd(Ps, Rs,时间)。

表1 提供服务Ps与请求服务Rs的QoS描述

QoS属性值	提供者Ps对服务的描述			请求者Rs对服务的描述		
	min	avg	max	min	avg	max
Time	50	60	90	40	70	80
Cost	60	70	80	50	70	80
Reliability	5	8	10	6	8	10
Fidelity	6	8	10	7	8	10

以时间属性为例计算提供服务与请求服务的相似度，代入下面公式计算：

$$d_{\min}(Ps,Rs,\text{时间})=1-\frac{|\min(Ps.\text{时间})-\min(Rs.\text{时间})|}{\min(Rs.\text{时间})} \tag{9}$$

则 $d_{\min}(\text{Ps, Rs, Time})=1-|50-40|/40=0.750$

$d_{\text{avg}}(\text{Ps, Ps, Time})=1-|60-70|/70=0.857$

$d_{\max}(\text{Ps, Rs, Time})=1-|90-80|/80=0.872$

$$Qosd(Ps,Rs,Time)=\sqrt[3]{d_{\min}(Ps,Rs,Time)*d_{avg}(Ps,Rs,Time)*d_{\max}(Ps,Rs,Time)}$$

$$=\sqrt[3]{0.750*0.857*0.872}=0.830 \tag{10}$$

同理，$d_{\min}(\text{Ps, Rs, Cost})=0.80$，$d_{avg}(\text{Ps, Rs, Cost})=1$，$d_{\max}(\text{Ps, Rs, Cost})=1$

$Qosd(Ps,Rs,Cost)=0.928$

$d_{\min}(\text{Ps, Rs, Reliability})=0.833$，$d_{avg}(\text{Ps, Rs, Reliability})=1$，$d_{\max}(\text{Ps,Rs, Reliability})=1$

$Qosd(Ps,Rs,Reliability)=0.941$

$d_{\min}(\text{Ps, Rs, Fidelity})=0.857$，$d_{avg}(\text{Ps, Rs, Fidelity})=1$，$d_{\max}(\text{Ps, Rs, Fidelity})=1$

$Qosd(Ps,Rs,Fidelity)=0.950$

应用公式(6),则可算出提供服务与请求服务的质量相似度为:

$QoS(Ps,Rs)=0.883$

为了匹配出符合 QoS 条件的服务,规定一个相似度的阈值 Q_{smin},若相似度$\geq Q_{smin}$,说明提供服务与请求服务相似。该提供服务可添加进第三级的匹配表中。

2) 区间属性的 QoS 匹配。为了能衡量两个区间数相似的程度,我们可用属性的区间数之间的相离度[20]来刻画两个区间是否相似。可定义相离度如下:

定义 1 设两个区间数 $a=[a^l,a^u]$,$b=[b^l,b^u]$,令区间 a 与 b 的相离度为:

$$D(a,b)=\sqrt{(a^l-b^l)^2+(a^u-b^u)^2} \tag{11}$$

相离度 $D(a,b)$ 越大,说明区间 a,b 之间不相似的程度也越大,特别地,当 $D(a,b)=0$ 时,有区间 $a=b$。

我们可扩展到多个区间,$a_i=[a_i{}^l,a_i{}^u]$,$b_i=[b_i{}^l,b_i{}^u]$,则多个区间的综合相离度可定义为如下公式:

$$ZD(a_i,b_i)=\sum\beta_i * \sqrt{(a_i{}^l-b_i{}^l)^2+(a_i{}^u+b_i{}^u)^2} \tag{12}$$

其中,β_i 是第 i 个属性的权重,$0<\beta_i<1$,$\sum\beta_i=0$。

经过前面对请求服务的基本描述匹配和功能匹配的两次筛选匹配,QoS 匹配器中可对 QoS 进行匹配,对以上的属性进行分类,按效益型和成本型来分类。效益型属性是指属性值越大越好的属性;成本型属性是指属性值越小越好的属性。对于一个 Web 服务来说,用户总是希望时间越短越好,花费越少越好,而可靠性和真实性越大越好。因此可把时间和花费归为成本型类,把可靠性和真实性归为效益型。

根据区间数的运算法则,对一个矩阵 $A=(a_{ij})_{m\times n}$ 中的具体数值有如下量化

$$r_{ij}^l=a_{ij}^l/\sqrt{\sum_{i=1}^{n}(a_{ij}^u)^2} \quad i\in I_1,j\in N \quad \text{效益型} \tag{13}$$

$$r_{ij}^l=a_{ij}^u/\sqrt{\sum_{i=1}^{n}(a_{ij}^l)^2}$$

$$r_{ij}^l=(1/a_{ij}^u)/\|(1/a_{ij}^l)^2\|,i\in I_2,j\in N \quad \text{成本型} \tag{14}$$

$$r_{ij}^u=(1/a_{ij}{}^l)/\|(1/a_{ij}{}^u)^2\|,i\in I_2,j\in N$$

下面以网上酒店订房为例说明其匹配过程,其中除了酒店本体外,还可能用到凭证、钱、日历本体,设存在酒店订房服务,其提供服务的简单抽象描述为:

〈! --酒店房间预定 Web 服务描述-- -〉
〈owl:servicename〉Hotel room book service〈/owl: servicename〉
〈owl:textdescription〉酒店房间预定服务提供方便的网上房间预定服务.〈/owl: textdescription〉
〈owl:type〉calendardate={day of week,day of year,year,month,day}〈/owl:type〉
Mony:real;icket(票据):string
Input:yourname:name;
Input:yoursex:sex
Input:booktime ValidateCalenderdate:calendardate
Input:bookfangxing:fangxing
Input:personnumber:number
Input:dingjing:mony
Output:changemony:Money;Ticket:ticket
Precondition:Money
Effect:Ticket
Qos model:
Name:Time=[80,100];unite:秒
Name:Cost=[60,90];unite:元
Name:Reliability=[5, 10]
Name:Fidelity=[5, 10]

假如还有两个提供服务的抽象描述,为简单起见,只列出它们的 Time 、Cost、Reliability、Fidelity 的 QoS 值,与第一个提供服务的描述一起,构成一个区间表 A 如下:

表 2 三个提供服务的服务描述

QoS 属性值	提供服务描述		
	提供者 1	提供者 2	提供者 3
Time	[80,100]	[70,90]	[60,100]
Cost	[60,90]	[50,80]	[60,80]
Reliability	[5,10]	[6,10]	[7,10]
Fidelity	[5,10]	[7,10]	[6,10]

下面是请求服务的抽象描述,为简单化,只简写出其 QoS 描述:

〈! --酒店房间预定 web 服务描述-- -〉
〈owl:servicename〉Hotel room book service〈/owl: servicename〉
〈owl:textdescription〉酒店网上房间预定服务.〈/owl: textdescription〉
〈owl:type〉date={day of week,day of year,year,month,day}〈/owl:type〉
Mony:real; Ticket(票据):string
…
QoS model:
Name:Time=[50 90];unite:秒
Name:Cost=[50 80];unite:元
Name:Reliability=[6,10]
Name:Fidelity=[7,10]

请求者 QoS 的 Time、Cost、Reliability、Fidelity 可用区间差值法来计算。

设有如下设区间数 $a=[a^l,a^u]$,可定义其规范化为:

$$r^l=(a^u-a^l)/a^u,\ r^u=(a^u-a^l)/a^l \tag{15}$$

根据上式可得:$Time^l=(90-50)/90=0.44$;$Time^u=(90-50)/50=0.80$

同理可得 $Cost^l=0.38$ $Cost^u=0.6$

$Reliability^l=0.40$ $Reliability^u=0.67$

$Fidelity^l=0.30$ $Fidelity^u=0.43$

$$A=\begin{bmatrix}[80,100] & [70,90] & [60,100]\\ [60,90] & [50,80] & [60,80]\\ [5,10] & [6,10] & [7,10]\\ [5,10] & [7,10] & [6,10]\end{bmatrix}$$

对提供服务的矩阵 A 用公式(13)、(14)规范化后,可得规范矩阵 R

$$R=\begin{bmatrix}[0.40,0.70] & [0.40,0.80] & [0.40,0.93]\\ [0.36,0.80] & [0.41,0.96] & [0.41,0.80]\\ [0.29,0.95] & [0.35,0.95] & [0.40,0.95]\\ [0.29,0.95] & [0.40,0.95] & [0.35,0.95]\end{bmatrix}$$

把请求服务经过规范化后的 QoS 各项均与三个提供服务各条的的规范矩阵 R 进行计算,可得到如下综合相离度:为简单方便,我们这里对 Time、Cost、Reliability、Fidelity 各属性区间的权重各取值为 0.25,则按照公式(12),得:

$$d_1=0.25*(\sqrt{(0.44-0.40)^2+(0,80-0.70)^2}+\sqrt{(0.38-0.36)^2+(0.60-0.80)^2}+\sqrt{(0.40-0.29)^2+(0.67-0.95)^2}+\sqrt{(0.30-0.29)^2+(0.43-0.95)^2})=0.283$$

同理可得:$d_2=0.304$ $d_3=0.285$

以上 d_1、d_2、d_3 分别是请求服务与提供服务者 1、提供服务者 2、提供服务者 3 之间的相离度。

为了匹配出符合 QoS 条件的服务,规定一个相离度的阈值 Q_{dmin},若相离度$\leq Q_{dmin}$,说明提供服务与请求服务相似。该提供服务可添加进第三级的匹配表中。

5 模拟实验

我们进行了一个模拟实验来验证算法的查全率与查准

率，由于条件的限制，实验以酒店服务预定为对象，让用户查找要求的服务，本实验假设基于本体的推理均已实现，用SQL数据库代替注册中心，注册中心上已有酒店服务的一些相应记录，用Pb编写匹配器。查全率与查准率对比图如图4和图5。

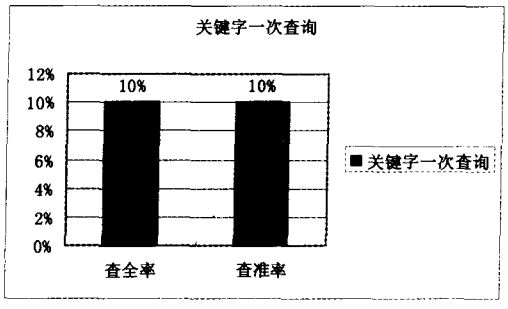

图4 关键字一次查询的查准率与查全率

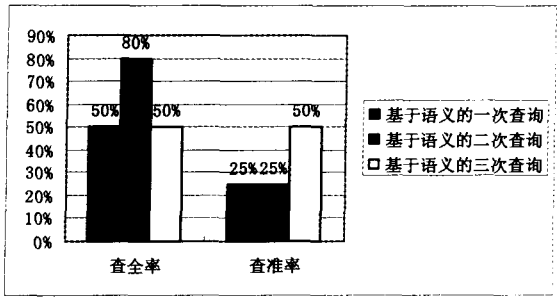

图5 基于语义的多次查询的查准率与查全率

根据查询条件逐步过滤，每一个条件就是在原来的基础上过滤一次。图4与图5对比了关键字一次查询与基于语义的多次查询的查全率与查准率，从图中可以看出，基于语义的多次查询的查准率与查全率均比基于关键字的一次性查询要高，且经过了多次查询的过滤后，逐渐逼近了服务请求者的目标。

结论 服务发现的主要过程是进行服务匹配，匹配算法的目标是实现提供服务与请求服务之间的语义匹配，即通过分析提供服务与请求服务两组语义描述信息，判断它们之间的相符和相似程度，以便达到为请求者找到尽可能满足要求的服务。通常情况下，服务请求者的需求是多层次的，服务发布方与服务请求方事先没有约定如何定义某个服务，再加上它们有着不同的目标，因此一个非常苛刻的"相符"标准在匹配上不可能体现匹配的共性。本文提出的多层过滤匹配机制与文[3～7]所提方法相比有以下优越之处：采用分层过滤方法可以缩小匹配范围，避免在相似度较低的匹配上浪费时间，从而提高了匹配效率；为了适应语义匹配的一般性，我们需要匹配引擎进行较为灵活的匹配，即通过判断服务与请求之间的"相似程度"来进行匹配，这样就增大了找到满足请求的服务的可能性；匹配引擎主要包含三个过滤器，基本描述匹配过滤器、IOPE匹配过滤器、QoS匹配过滤器。第一级匹配明显为后续匹配节约了大量的存储空间，同时排除了大量的干扰数据，也为后续匹配的准确性提供了有力的支持。第二级在第一层匹配的基础上应用本体进行进一步筛选，再次缩小了匹配的服务数，同时体现了用户对功能性的要求。第三级根据用户的特殊需要进行服务质量上的动态匹配，以满足用户对服务质量上的更高级的需求。全面考虑了不同用户对服务质量的不同表达方式，对于数值型服务质量用几何距离来计算提供服务与请求服务的相似度，对区间型服务质量用相离度来计算提供服务与请求服务的相似度。这种实现方法使代理的功能非常容易扩展，任何一个匹配算法都可以方便地为其他算法取代，方法是为新的匹配算法开发一个新的过滤器。所以，算法的改进不会对整个系统结构造成影响，新的算法可以方便地引入系统。

参考文献

1 Paolucci M, Kawamura T, Payne T R, Sycara K. Semantic matching of Web services capabilities [C]. In: Proceedings of the 1st International Semantic Web Conference(ISWC), Sardinia, Italia, 2002. 34～43

2 岳昆，王晓玲，周傲英. Web服务核心支撑技术研究综述[J]. 软件学报，2004,15(3):428～442

3 Sycara K, Widoff S, Larks K M. dynamic match-making among heterogeneous software agents in cyberspace [J]. Autonomous Agents and MultiAgent Systems, 2002,5(2):173～203

4 Arisha K, Kraus S, Fetal O. *The interactive Maryland platform* for agents collaborating together [J]. IEEE Intelligent Systems, 1999 ,14 (2) :64～72

5 Wickler G J. Using expressive and flexible action representations to reason about capabilities or intelligent agent cooperation [D]. University of Edinburgh , Edinburgh , U K, 1999

6 Martin D, et al. OWL-S: Semantic Markup for Web Services [EB/OL]:[Technical report]. Daml consortium. http://www.daml. org/services/owl-s/1. 0/owl-s. pdf, February 2004

7 Paolucci M, Kawamura T. matching of Web services capabilities [C]. In: Web Conference (ISWC), volume 2342, Springer-Verlag, 2002. 333～347

8 陆玉昌，鲁羽凡. 向量空间法中单词权重函数的分析和构造[J]. 计算机研究与发展,2002,39(10):5～7

9 游荣彦，邓志才. 向量空间模型中特征词的区分度的定盆研究[J]. 中文信息学报，2002,16(3)

10 Tversky A. Features of similarity [J]. Psychological Review, 1997, 84(4): 327～352

11 Rvdriguez A, Egenhofer M. Determining Semantic Similarity Among Entity Classes from Different Ontologies [J]. IEEE Transactions on Knowledge and Data Engineering, 2002

12 Maximilien E M, Singh M P. A Frame work and Ontology for Dynamic Web Services Selection [J]. IEEE Internet Computing, 2004, 9(10):84～93

13 Ran S P. A model for Web services discovery with QoS [J]. ACM SIGCOM Exchanges, 2004(1):1～10

14 Chen H G, Yu T, Lin K I. QCW& An implementation of QoS-capatily multimedia Web services[C]. In:Proceedings of the 5 International Symposium on Multimedia Software Engineering [A]. Taichung the Institute of Electrical and Electronics Engineers, 2003. 38～45

15 杨胜文，史美林. 一种支持约束的服务发现模型[J]. 计算机学报,2005,28(4):589～594

16 蒋运承，史忠植. QoS驱动的主体服务匹配[J]. 小型微型计算机系统，2005，26(4):687～692

17 Cardoso J, Bussler C. Semantic Web Services and Processes: Semantic Composition and Quality of Service [C]. On the Move to Meaningfull Internet Computer, 2002

18 Zeng I, BenatSallam B. QoS-aware middle are for Web service composition [J]. IEEE Transactions on Software Engineering, 2004, 30(5):311～327

19 Medjahed B. Semantic Web enabled composition of Web services [D]. Virginia Polytechnic Institute and State University, Virginia, USA, 2004

20 徐泽水，孙在东. 一类不确定型多属性决策问题的排序方法[J]. 管理科学学报，2002,5(3) :35～39