一种关于语义 Web 服务匹配的策略和实现*)

张 献[1] 李舟军[2] 李梦君[1]

(国防科技大学计算机学院 长沙 410073)[1] (北京航空航天大学计算机学院 北京 100083)[2]

摘 要 Web服务的迅速发展使得Web服务的匹配和查找问题成为研究热点。引用领域本体对Web服务进行语义描述,再进行语义层上的匹配,是Web服务匹配研究领域的一个重要研究方向。本文从Web服务的输入、输出参数的语义描述出发,把Web服务的匹配问题转化为领域本体库中概念间语义距离的计算,并根据服务请求者对Web服务的需求,提出了用不同的方法来计算Web服务输入、输出概念间的语义距离,其中处理Web服务输出概念时着重提高Web服务查找的查准率,而处理Web服务输入概念时则注重提高Web服务查找的查全率。实验结果显示了该方法较别的服务查找方法在查准率特别是查全率上得到了提高。

关键词 语义Web服务,本体,服务匹配,语义距离

An Approach and Implementation to Semantic Web Service Matchmaking

ZHANG Xian[1] LI Zhou-Jun[2] LI Meng-Jun[1]

(School of Computer, National University of Defense Technology, Changsha 410073)[1]

(School of Computer Science & Engineering, Beihang University, Beijing 100083)[2]

Abstract With the growing number of Web Services, the importance of matchmaking and discovery of Web Service is increasing. Using domain ontology to describe the semantic of Web Service and Matchmaking Web Service on the semantic level is a hot research spot. Based on the semantic description of Web Service's Input and Output, the problem of Matchmaking of semantic Web Service is transformed to the calculation of semantic distance between two concepts. Given Web Service requester's demand, different algorithms are applied to calculating the semantic distance of Web Service's Inputs and Outputs. Experiment results demonstrate the enhancement to accuracy and recall.

Keywords Semantic Web service, Ontology, Web service matchmaking, Semantic distance

1 引言

随着Web服务技术的迅速发展,Internet上充斥着越来越多的Web服务。Web服务提出的目标是通过Web服务间的松耦合,来支持各种领域(包括企业到消费者、企业到企业)中各种系统的无缝集成。要达到这个目标,找到满足用户需求的Web服务是关键。

Web服务请求者从众多Web服务中找到满足其需求的Web服务的过程就叫做Web服务发现。在Web服务发现过程中,需要把服务请求者的理想Web服务描述与已有的Web服务描述相比较,以判断已有的Web服务是否满足请求者的需求。这个比较过程就称为Web服务匹配。

本体是用于描述或表达某一领域知识的一组概念或术语,可用来描述特定领域的知识[1]。利用本体对特定领域的知识进行描述,然后再利用本体表达的知识对Web服务进行标注,就可把传统的基于词法的Web服务匹配提升到语义层次上来。在语义层次上定位服务请求者所要求的服务一般称为语义Web服务发现(Semantic Web Service Discovery),而在语义Web服务发现中的服务匹配也就称为语义Web服务匹配。

2 语义Web服务匹配的研究概况

语义Web服务匹配首先要对请求者的理想Web服务进行语义描述,同时要对已有的Web服务进行语义描述。现今,对Web服务应该如何进行语义描述有着多种方案,如OWL-S[2],WSDL-S[3],SWSL[4]等等。上述方案中,都把对Web服务的描述分为服务的功能性描述和非功能性描述两部分。其中对服务功能性信息(Web服务的Input,Output,Precondition,Effect)进行语义描述的基本思想都是首先用本体来表达领域内的语义信息,然后利用本体中的概念对Web服务的功能信息进行描述。而对Web服务非功能信息(如服务提供者的联系方式、服务的大概功能等),在现阶段一般都是用自然语言的方法对其进行语义描述。

从而语义Web服务匹配可分为服务功能信息的语义匹配和非功能信息的语义匹配两方面。对Web服务非功能信息的语义匹配方面,现行的处理方法大都是利用自然语言处理的成果,如WordNet[5]等,对Web服务进行描述的自然语言加以处理、比较,得到Web服务之间的相似程度(也称为匹配度)值。本文只考虑Web服务功能信息的语义匹配,特别地,只考虑Web服务功能信息中的输入(Input)和输出(Output)参数语义描述的匹配。

3 Web服务功能信息语义匹配的定义和相关研究

由于本文只考虑Web服务语义描述中的输入和输出方面,故Web服务的语义描述可以抽象定义如下:

定义1 Web服务语义描述模型$W=\langle I,O\rangle$。其中:$I=$

*)本文受到国家自然科学基金项目(90104026,60473057,90604007)的资助。**张 献** 硕士研究生,主要研究方向为语义网、Web服务发现;**李舟军** 博士,教授,CCF会员,博士生导师,主要研究方向为进程代数理论、安全协议的形式化验证、语义网、Web服务与P2P计算、数据挖掘与生物信息学;**李梦君** 博士,讲师,主要研究方向为安全协议的形式化验证。

$\langle I_1, I_2, \cdots, I_n \rangle$为一个概念矢量，表示 Web 服务 W 的 N 个输入参数的语义描述，$I_1, I_2, \cdots, I_n$ 表示服务 W 各个输入参数在领域本体库中相对应的语义概念；$O=\langle O_1, O_2, \cdots, O_m \rangle$为一个概念矢量，表示 Web 服务 W 的 M 个输出参数的语义描述，$O_1, O_2, \cdots, O_m$ 表示服务 W 的各个输出参数在领域本体库中相对应的语义概念。

这样就把 Web 服务的匹配转换为请求者理想的服务语义描述模型 $W=\langle I,O \rangle$和服务库中服务语义描述模型 $W'=\langle I',O' \rangle$的匹配，进而可以转化为同一领域本体中概念矢量 I 和 I'以及 O 和 O'的匹配。

在网络环境中，服务请求者在找到满足其需求的服务以前，服务请求者和服务提供者是相互隔离的，因而他们各自对 Web 服务的语义描述是相互独立的，这样就造成了概念矢量 I 和 I'、O 和 O'在维数以及元素顺序方面都可能会有不相同的情况。要形式定义 Web 服务匹配的概念，必须定义同一领域本体库中任意两个概念矢量匹配的含义。在此之前，先定义下面这个函数：

定义 2 同一领域本体库中任意两个概念相似度的计算函数 $Sim: C_1 \times C_2 \rightarrow R^+$。其中，$C_1, C_2$ 是领域本体库中任意两个概念，$Sim(C_1, C_2)$表示衡量 C_1, C_2 相近程度的一个正实数，且 $Sim(C_1, C_2) \in [0,1]$。

$Sim(C_1, C_2)$越大，表示两个概念越相似。关于相似度函数的定义，我们将在下面做详细解释。在此基础上，就可以定义同一领域本体中任意两个概念矢量 V, V'的相似度为

定义 3 同一领域本体库中任意两个概念矢量相似度的计算公式：

$$S(V,V')=\sum_{i=1}^{n}\max(Sim(V_i,V'_j)),1\leqslant j\leqslant m$$

其中：$V=(V_1,V_2,\cdots,V_n)$，$V'=(V'_1,V'_2,\cdots,V'_m)$，$V_1,V_2,\cdots,V_n,V'_1,V'_2,\cdots,V'_m$ 均为领域本体库任意概念。

这样，通过定义 1 就把语义 Web 服务匹配抽象为 Web 服务语义描述模型的匹配，也即概念矢量间的匹配（I 和 I'，O 和 O'），然后通过定义 3 进而把概念矢量的匹配转换成了概念矢量元素的匹配，亦即领域本体库中任意两个概念的匹配。而要判断这两个概念是否匹配，一般都是先用定义 2 的相似函数计算出两个概念间相似度，然后把这个相似度与阈值进行比较，从而得到是否匹配的结果。

对用于 Web 服务匹配的领域本体库中概念间语义相似度的计算，研究者们提出了各种各样的计算方法。如在文[6]中，就根据 Web 服务的输入、输出对应概念的上下位（subClassOf）关系把这种语义相似性分为 Exact，Plug in，Subsumes，Fail 四种类型。文[7]根据 DAML-S[8] 中 Profile 对 Web 服务的描述，把服务输入、输出参数的相似度分为九个等级，其依据除了输入、输出参数对应概念的上下位（subClassOf）关系外，还考虑了关系的（Property）上下位关系（subPropertyOf），但这也使得这种匹配算法只适用于 DAML-S[8]，对其它 Web 服务语义描述标准如 WSDL-S[3] 则不能运用此算法。文[9]定义了一个值域为正实数的函数来计算同一领域本体中两个概念的相似性。文[10]第 3 章对两个概念的相似性也提出了多种计算方法。

综合研究上述计算概念相似度的算法，发现存在下述一些不足：

(1)对概念相似度的区分过于简单。如文[6]只把概念相似度分为四种类型，过于简单的分类显然会不适合网络上 Web 服务数目众多的情况。

(2)过于重视领域本体库中概念间的上下位（subClassOf）关系，而忽略了其中更存在着丰富的其它二元关系。如文[6]、[7]、[9]均只利用概念间的上下位关系来计算概念间的相似度。

(3)对 Web 服务的输入、输出参数不加区分地对待，而没有意识到对服务请求者而言，服务输出具有更重要的意义，并且服务请求者对服务输入参数具有更大的控制权。上述所有算法均不加区分地用同样方法来计算输入、输出概念间相似度。文[10]虽然意识到对请求者来说输出的匹配比输入的匹配具有更重要的意义，并通过对输出的匹配结果进行加权来体现，但是本文认为，对输入、输出的区别对待，不应该仅仅体现在对输出参数匹配结果进行加权，更应体现在计算输出概念相似度时采用不同的方法上。

针对以上不足，本文对 Web 服务相似性计算的算法进行了改进。

4 改进后的 Web 服务语义匹配算法

改进后的 Web 服务语义匹配算法，采用了不同的方法来计算 Web 服务输入、输出参数的相似度。

对 Web 服务输出参数对应概念的相似度计算，本文算法仅考虑领域本体库中的上下位关系（subClassOf Property）。因为对 Web 服务请求者而言，这样所找到的 Web 服务的输出才是所需要的。据此对 Web 服务输出对应的概念进行细化（子概念）和泛化（父概念）操作，所找到的 Web 服务可能也会满足服务请求者的需要。例如在图 1 所示的本体库下，有许多 Web 服务提供各自不同的 Pizza，假设服务请求者要找的为提供 FamousPizza 的 Web 服务，根据本文算法，仅考虑本体概念间的上下位关系，返回的提供 AmericanHot（细化的概念）的 Web 服务和提供 Pizza（泛化的概念）的 Web 服务或许都能满足服务请求者的需要。但如果像文[10]中的做法那样，在计算输出参数相似度时考虑本体概念间一般的二元关系（如 madeIn 关系），则很难相信，请求者要找的是一个提供 FamousPizza 的服务，而返回他一个提供 Country 的服务会满足需要。

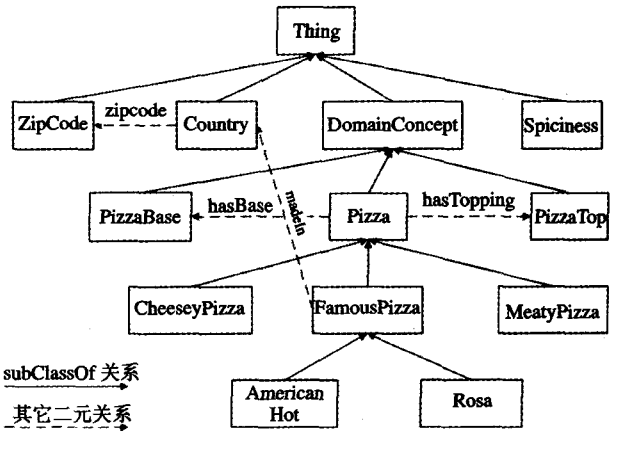

图 1 领域本体库

在信息检索领域，要计算两个概念的相似度，一般都是先计算出与之相反的两个概念间的语义距离。两个概念的语义距离越大，则其相似度越低；语义距离越小，则相似度越高。本文在计算两个 Web 服务概念矢量中元素间的相似度时，也是先计算其语义距离。对概念距离和概念相似度之间的关

系，有如下定义：

定义 4 概念间距离和概念间相似度的关系是 $Sim(C_1, C_2)=\frac{1}{e^{Dis(C_1,C_2)}}$。其中 C_1，C_2 为领域本体库中任意两个概念；$Dis(C_1,C_2)$为两个概念间的距离。

在对 Web 服务输出参数对应概念相似度的计算中，本文把 $Dis(C_1,C_2)$定义为本体库中由上下位关系所形成概念树[1)]中连接这两个概念结点的最短边数。

下面给出计算这个最短边数的一个朴素的实现算法：

1) 输入 O 对应的标志符 class1 以及 O' 对应的标志符 class2，depth=0，try_list=Φ，abandoned_list=Φ；
2) 把 Class1 加入到 try_list 的首部；
3) 如果try_list 不为空，则进行如下操作：
   取走 try_list 的第一个元素 u：
   3.1) 如果 Synset(u)包含 class2，那么 depth 为该元素的 depth，算法返回；
   3.2) 利用推理机从本体库中得与 u 具有直接上下位关系的概念集合 H(u)；
   3.3) 置 H(u)中每个元素的 depth 为 depth+1；
   3.4) 如果 H(u)包含 class2，则返回该元素的 depth；
   3.5) 对于 H(u)中的每一个元素 v；
      3.5.1) 如果 abandoned_list 和 try_list 中包含 v，那么 continue；
      3.5.2)把 v 加入到 try_list 的尾部；
   3.6) 把 u 加入到 abandoned_list。

算法中 Synset(u)表示利用推理机获得概念 u 所有的同义概念；try_list 为一列表，其中元素表示本体库中尚未遍历的概念；abandoned_list 也为一列表，其中元素表示本体库中已经遍历过的概念。该算法采用宽度优先的方式从请求者理想的服务输出概念开始遍历整个概念树，直到遍历的节点包括已有 Web 服务的输出为止。采用宽度优先的搜索方式保证了该算法找到的从概念 O 到 O' 概念的路径为概念树中这两个节点间的最短路径；采用禁忌节点表(abandoned_list)的做法可以使只对遍历过的节点扩展一次，从而可以提高算法的效率。

对 Web 服务输入参数对应概念间相似度的计算，同样是先计算出两个概念间的语义距离。在计算两个概念间的距离时，不仅仅只考虑概念间的上下位(subClassOf)关系，还考虑了本体库中概念之间存在着的其它丰富的二元关系。之所以在输入参数相似度的计算中考虑了除上下位关系外的二元关系，主要是因为 Web 服务输入参数是由请求者提供的，从而请求者对这些参数具有更大的控制权。同时，本体库中概念之间的关系是人对现实世界一些共同认识的反映(本体是人关于客观世界共享的概念化的显式的说明[13])，请求者作为现实世界中的一个实体(本文认为在现阶段 Web 服务的调用是需要人工干预的，暂时还不能实现完全的自动化)，当根据本文的算法返回一些服务时，一般情况下请求者是有能力填补发现的服务和他理想中的服务在输入参数上的一些差别的。比如在图 1 中，请求者理想中的服务是接受一个 Country 的概念作为输入，而实际只能找到一个接受 ZipCode 概念作为输入的 Web 服务。在 Web 服务的调用需要人工干预的情况下，有理由认为这个 Web 服务(接受 ZipCode 作为输入)比那些接受同一层次的其它概念(如 DomainConcept，Spiciness)作为输入的 Web 服务具有更大的相似性(之所以和同一层次的其它概念做比较，是因为如果不加区别地对待 Web 服务的输入、输出参数相似度的计算，则根据计算 Web 服务输出参数对应概念相似度的算法，那些接受 DomainConcept，Spiciness 的 Web 服务与接受 ZipCode 的 Web 服务具有相等的语义距离，从而也具有相等的相似性)。

一般来说，在领域本体库中，概念间的上下位关系比其它的二元关系在计算概念间的距离方面具有更重要的地位(这也是本文在计算 Web 服务输出参数时只考虑上下位关系的原因)。这样在计算概念间语义距离时，可以分别对上下位关系以及其它二元关系给予不同的权重。在本文中，给概念间的上下位关系赋予权重 1，而对概念间的其它关系赋予权重 1.6(因为相似度与距离成反比关系，这里的权重指的是计算距离的权重，所以上下位关系的权重小于其它二元关系)。这样，要计算两个 Web 服务输入矢量元素所对应概念间的距离，就转化为一个带权重的有向图(因为考虑了各种各样的二元关系，所以是图)中寻找最短路径的问题，对此可以运用图论中最短路径的算法(BFS 或 Dijkstra)来计算两个概念间的距离。

计算两个 Web 服务输入矢量元素所对应概念间的距离的算法如下：

1) 输入 I 对应的标志符 class1 以及 I' 对应的标志符 class2，try_list=Φ，abandoned_list=Φ；
2) 把 class1 的权重初始化为 0，并加入到 try_list 中；
3) 如果try_list 不为空，则进行如下操作：
   取走 try_list 中具有最小权重的元素 u：
   3.1) 如果 Synset(u)包含 class2，则返回 u 具有的权重，算法结束；
   3.2) 利用推理机从本体库中得到与 u 具有直接上下位关系的概念集合 H(u)；
   3.3) 如果 H(u)中包含 class2，则返回(u 的权重+1)，算法结束；
   3.4) 对 H(u)∩try_list 中的每一个元素 v，如果(u 的权重+1)<v 的权重，则把 v 的权重修改为(u 的权重+1)；对 H(u)－try_list－abandoned_list 中的每个元素 w，设定 w 的权重为(u 的权重+1)，并把它加入到 try_list 中；
   3.5) 利用推理机得到以 u 为定义域的所有直接二元关系的值域集合 B(u)；
   3.6) 如果 B(u)中包含 class2，则返回(u 的权重+1.6)，算法结束；
   3.7) 对 B(u)∩try_list 中的每一个元素 v，如果(u 的权重+1.6)<v 的权重，则把 v 的权重修改为(u 的权重+1.6)；对 B(u)－try_list－abandoned_list 中的每个元素 w，设定 w 的权重为(u 的权重+1.6)，并把它加入到 try_list 中；

1)严格来说，由本体库中概念以及概念间上下位关系(subClassOf)形成的图形不是严格的树，而是近似于树的一个图，因为本体中允许一个概念有多个父概念(上位概念)。但这对本文的算法没有影响，因为该算法基于宽度优先的搜索可以适用于图这种情形。

3.8) 把 u 加入到 abandoned_list 中。

在这个算法中，Synset(u)，try_list，abandoned_list 的含义同上；步骤 3.3 表示赋予上下位关系的权重为 1；步骤 3.6 表示赋予其它二元关系的权重为 1.6。

经上述两个算法就得到两个 Web 服务关于输入和输出参数的距离，进而得到了这两个 Web 服务关于输入和输出参数的相似度(定义 4)，而最后要得到的是这两个 Web 服务的相似度。对此，一般把两个 Web 服务 $W=\langle I,O\rangle$ 和 $W'=\langle I',O'\rangle$ 相似度定义为 $\alpha\times S(O,O')+(1-\alpha)\times S(I,I')$，其中 α 是一个大于 0 的常量，一般建议把其设为大于 0.5，这样就可以加大输出参数的相似性在整个 Web 服务相似性计算中的权重。

5 实现及实验数据分析

由于目前没有与语义 Web 服务相关的标准平台和标准测试数据集，而且带语义标注的 Web 服务在网络上很难收集，本文采用计算机随机产生和人工标注已有 Web 服务相结合的方法产生测试集。特别地，从简单性方面考虑，对测试集中每个 Web 服务的语义描述模型 $W=\langle I,O\rangle$，其概念矢量 I 和概念矢量 O 都为一元矢量的形式。

在本文实验中，随机生成了 970 个 Web 服务的语义描述，人工标注了 30 个 Web 服务(共 1000 个已发布的 Web 服务语义描述)，人工产生了 5 个 Web 服务请求的语义描述。对每个 Web 服务请求的语义描述，实验用 3 种不同的方法[1)]计算它与每个已发布 Web 服务语义描述之间的语义距离，最后把每种方法返回的语义距离值进行排序。

表 1 3 种匹配算法查找结果相似权重比较

用户请求 Web 服务的语义描述	基于关键词匹配的算法(一)的查找结果	只考虑上下位关系处理 Web 服务输入输出算法(二)的查找结果	本文改进后的算法(三)的查找结果
⟨Hotel, MeetingRoom⟩	⟨Hotel, MeetingRoom⟩ 0.0	⟨Hotel, MeetingRoom⟩ 0.0 ⟨Company, MeetingRoom⟩ 0.39 ⟨Location, MeetingRoom⟩ 2.4	⟨Hotel, MeetingRoom⟩ 0.0 ⟨Company, MeetingRoom⟩ 0.39 ⟨Location, MeetingRoom⟩ 0.6

本文实验在 Windows XP 环境下开发。采用 Eclipse3.0 作为开发工具，Protégé3.1.1 作为领域本体建模工具，JDK 版本为 5.0。在获取概念上下位集合时，采用 Racer-1-7-23 作为推理机，同时利用了 Protégé-owl 提供的 jar 包对本体库中的概念(Class)、关系(Property)进行操纵和遍历[2)]，硬件环境为奔四 2.8G CPU，1G 的 DDR 内存。

对同一查询目标，表 1 给出了用 3 种算法计算特定服务(人工判定是满足用户需求的)与该查询目标之间语义距离。

表 1 中，Web 服务⟨Hotel, MeetingRoom⟩表示用户想查询的是某宾馆提供会议室的情况，⟨Location, MeetingRoom⟩为列出城市某一区域拥有大型会议室的情况的一个服务，⟨Company, MeetingRoom⟩为列出一个机构(Hotel 是机构中的一种，即 Hotel subClassOf Company)拥有大型会议室情况的 Web 服务。由表中结果可以看出，算法二和算法三查找结果栏中前两项 Web 服务与用户请求的 Web 服务的语义距离相同，这是因为 Company 和 Hotel 之间具有直接上下位关系，语义距离最近。而对给出的第三个 Web 服务，由于 Location 概念和 Hotel 概念之间几乎不存在上下位语义关系，因此算法二给出的语义距离较大，而由于每个 Hotel 都有一个 locatedIn 二元属性(表示该 Hotel 位于什么地方，locatedIn 属性的值域为 Location)，并且 Hotel 又是 Web 服务的输入概念，所以算法三给出的语义距离很小。

实验用测试集对 5 个 Web 服务请求进行了查询，图 2 给出了算法二和算法三在这 5 个查找请求下平均查准率和平均查全率之间的关系。

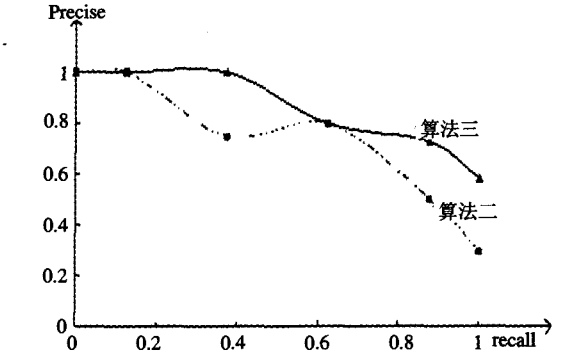

图 2 算法二和算法三的 recall-precise 曲线

由图 2 中曲线可以看出，在相同准确率下，算法三的查全率明显高于算法二，这主要是因为算法三在计算 Web 服务输入概念相似度时，考虑了领域本体库中存在着的丰富的除上下位关系外的其它二元关系，而算法二在 Web 服务匹配时由于只考虑了本体库中的上下位关系，这就使那些实际满足用户请求的、但在输入概念上与用户请求不具有上下位关系的 Web 服务在返回结果中排位过低，从而降低了算法二的查准率和查全率。

在程序执行效率方面，算法二和算法三效率差不多，在 1000 个测试集上的执行时间均为 6min 左右，这也反映了语义 Web 服务匹配具有相当高的计算复杂度。

结束语 本文首先通过一系列定义描述了如何将带语义标注的 Web 服务匹配问题化归为领域本体库中两个概念间相似度的计算，然后把概念间相似度的计算转化为概念间语义距离的计算。对概念间距离的计算，本文给出了两种算法，并分别用它们来计算 Web 服务输入和输出概念矢量的相似度。本文对算法进行了实现，并把用本文改进后的算法进行 Web 服务匹配的结果与其它算法的结果进行了比较。实验结果指出，本文改进后的算法较别的算法具有更好的查准率和查全率。

当然，本文的工作在许多方面还需进一步深化，比如如何优化领域本体库的结构来提高计算概念相似度算法的效率；如何判断找到的服务是满足用户所要求的，能被用户调用的等等。特别是在如何与别的算法进行比较方面，还需要用大规模的实验数据进行测试。

1)这 3 种方法分别为关键词搜索算法、用本文计算 Web 服务输出概念的方法计算 Web 服务输入输出概念相似度的算法(不加区别地对待输入输出概念)以及本文描述的改进后的算法。其中方法二与方法三的区别主要在于方法三用不同的方法计算 Web 服务输入输出概念的相似度，亦即方法三在计算 Web 服务输入概念相似度时，加入了对本体中非上下位二元关系的考虑。

2)本实验所有源代码及领域本体库可在 http://blogger.org.cn/blog/more.asp?name=01musician&id=14840 下载。

参 考 文 献

1 William S, Austin T. Ontologies. IEEE Intelligent Systems, 1999. 18～19
2 Martin D, Burstein M, Hobbs, J et al. OWL-S: Semantic Markup for Web Service. W3C Member Submission 22, November 2004
3 Akkiraju R, Farrell J, Miller J, et al. Web Service Semantics - WSDL-S. W3C Member Submission 7, November 2005
4 Battle S, Bernstein A, Boley H, et al. Semantic Web Servcies Language(SWSL). W3C Member Submission 9, September 2005
5 Miller G A, Beckwith R, Fellbaum C, et al. Introduction to WordNet: An On-line Lexical Database. International Journal of Lexicography, 1993
6 Paolucci M, Kawamura T, Payne T R, Mtching of Web Services Capabilities. In: Proc. of the First Intl Semantic Web Conference, June 2002
7 Tang S. Matching of Web Service Specifications Using DAML-S Descriptions: [Master dissertation]. Technische UniversitÄat Berlin, 2004, 3
8 Ankolekar A, Burstein M, Hobbs J R, et al. DAML-S : Web Service Description for the Semantic Web. The Semantic Web-ISWC, Springer 2002
9 Bramantoro A, Krishnaswamy S, Indrawan M. A Semantic Distance Measure for Matching Web Services. In: WWW2005, Web Service Semantics Workshop, 2005
10 张钋. 基于语义的网络服务匹配机制的研究与实现:[硕士论文]. 北京:清华大学计算机科学与技术系,2005,12
11 Sycara K, Widoff S, Klusch M, et al. Dynamic matchmaking among heterogeneous software agents in cyberspace. Autonomous Agents and Multi-Agent Systems, 2002, 5:173～203
12 吴健,吴朝晖,李莹,等. 基于本体论和词汇语义相似度的 Web 服务发现. 计算机学报,2005(4):595～602
13 Borst WN. Construction of Engineer Ontologies: [Ph D dissertation]. University of Twenty, Enschede, 1997

(上接第 74 页)

在图 5 中,每一个状态转换由检测率和成功率生成一个转移概率 P_{ij},其中 i 表示原子攻击 A_i,j 表示转移到状态 S_i^j。由于具体的攻击步骤只有 3 种结果,因此 3 种结果的状态转移概率之和为 1,如图 5 中有 $P_{10}+P_{11}+P_{12}=1$。

4.3 攻击效果预测算法

为了便于描述,取 $f(s_i^j)$为攻击效果评价值。没有后续攻击步骤的状态都是可能的最终状态。可以得出攻击效果为

$$P(f(s_5^3))=P_{10}P_{20}P_{30},P_{43},P(f(s_5^0))=P_{10}P_{21}P_{40},P(f(s_3^2))=P_{10}P_{22},\cdots \quad (3)$$

并且将效果评价值相同的状态合并,计算数学期望 E:

$$E=f(s_5^3)P(f(s_5^3))+f(s_5^0)P(f(s_5^0))+f(s_3^2)P(f(s_3^2))+\cdots \quad (4)$$

从而得出最终预测值。

4.4 决策意见生成算法

决策意见生成算法主要通过比较停止攻击所具有的攻击效果与继续攻击的最大数学期望来实现。其中停止攻击所具有的攻击效果容易计算,下面主要讨论继续攻击情况下攻击效果的最大数学期望的计算方法。用最大数学期望作为对比标准是合理的,如图 6 所示。

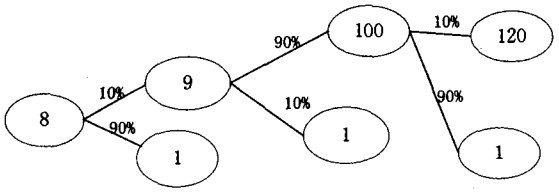

图 6 最大数学期望示例

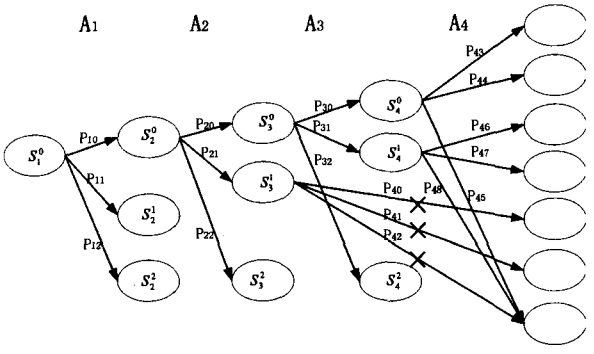

图 7 最大数学期望攻击图

与图 3 不同的是在攻击状态 100 后多了一个攻击步骤。当攻击状态为 8 时继续攻击最终的数学期望为

$$120*10\%*90\%*10\%+1*90\%*90\%*10\%+1*10\%*10\%+1*90\%=2.071$$

但是在状态 8 时我们的决策不变,仍是继续攻击。因为在状态 100 时,继续攻击的数学期望会很小,我们会在此处决策为停止攻击,在这样的攻击方案中最大数学期望仍为 9.91,所以在状态 8 时我们选择继续攻击。

用 $E(s_i^j)$表示 S_i^j 时继续攻击的最大数学期望。首先在从生成的攻击图中得出最大数学期望攻击图,如图 7 所示。决策意见生成算法如下:

step1:生成从当前状态开始的攻击图;

step2:计算最后一步攻击前所有状态的数学期望。如果 $E(s_i^j)<S_i^j$,则删除 S_i^j 的后续攻击步骤,生成一个新的攻击图;

step3:在此基础上计算倒数第二步攻击前的所有状态数学期望,同 step2 处理,直到当前需要决策的状态所在的攻击步骤为止。

step4:计算当前状态下继续攻击的数学期望。如果 $E(s_i^j)<S_i^j$,则决策为停止攻击,否则为继续攻击。

结束语 本文分析了网络攻击效果在线评估系统的目的,提出了直观有效的攻击效果评估准则和评估模型,给出了网络攻击效果在线评估的相关算法的实现方案,在实际应用中取得了良好的效果。由于网络攻击的具体情况十分复杂,如窃取数据信息量的初始值难以确定和归一化,针对软件的攻击时效性如何表现在指标初始值上等情况,使得指标权重的确定变得比较困难,同时攻击图生成算法中考虑到的隐蔽性和成功率在实际中难以精确得到,使得状态转移概率不精确,可能会影响最终的评估结果。上述这些难点问题都是以后研究中需要改进的地方。

参 考 文 献

1 Lala C, Panda B. Evaluating Damage from Cyber Attacks: A Model and Analysis[J]. IEEE Transactions on Systems, Man and Cybernetics-Part, A: Systems and Humans, JULY, 2001, 31(4)
2 Liu Peng, Zang Wanyu, Yu Meng. Incentive-based Modeling and Inference of Attacker Intent, Objectives, and Strategies [J]. ACM Transactions on Information and System Security, 2005, 8(1): 78～118
3 Menasch'ea D S, Figueiredob D R, de Souza e Silvaa E. An evolutionary game-theoretic approach to congestion control[J]. Performance Evaluation, 2005, 62: 295～312
4 Gregg M, Kim D. Inside Network Security Assessment: Guarding your IT Infrastructure[M]. Sams, 2005
5 冯登国,张阳,张玉清. 信息安全风险评估综述[J]. 通信学报,2004,25(7):10～18
6 林闯,汪洋,李泉林. 网络安全的随机模型方法与评价技术[J]. 计算机学报, 2005,28(12):1943～1956
7 卢显,等. 协同式网络对抗[M]. 北京:国防工业出版社,2003
8 刘进,王永杰,张义荣,等. 层次分析法在网络攻击效果评估中的应用[J]. 计算机应用研究,2005,22(3):113～115