服务请求驱动的语义 Web 服务组合方法①

房丙午，季红梅，汪永涛

(安徽财贸职业学院 计算机系，合肥 230601)

摘　要：以 Web 服务本体和领域本体为基础，利用语义匹配算法，提出一种以服务请求驱动的服务组合方法。递归调用此算法可以得到满足服务请求功能及性能约束的服务组合树。通过一个具体的实例，讨论服务组合树的构造过程。实例应用表明，本方法支持自动的、动态的 Web 服务组合，能够很好地满足 QOS 约束。

关键词：Web 服务组合；语义匹配；前驱服务；组合树

Service Request-Driven Semantic Web Service Composition

FANG Bing-Wu, JI Hong-Mei, WANG Yong-Tao

(Dept.of Computer, Anhui Finance and Trade Vocational College, Hefei 230601, China)

Abstract: Using the semantic matching algorithm, this paper proposes a service request-driven service composition method based on Web services ontology and domain ontology. By recursive calls this method to find the combination tree which should satisfy the service request function and performance constraints. To explain how to get the combination tree, a concrete example is used in this combination method. Example applications which support automatically and dynamic Web service composition can satisfy QOS constraints.

Key words: Web service composition; semantic matching; PreServices; combination tree

1　引言

Web 服务是一种能够被其它 Internet 上的软件组件访问的标准组件。如何使多个功能单一的 Web 服务相互协作，实现一个复杂的功能以满足用户的服务请求，是目前 Web 服务领域的研究热点。随着语义 Web 本体标记语言 OWL 的发展和成熟，使得机器可以理解 Web 服务，实现服务自动、智能、动态的发现和组合[1]。关于语义 Web 服务组合方法的研究有很多，文献[2]提出一种基于领域本体的 web 服务动态组合方法,利用领域本体生成一个优化的服务组合图并给出组合算法。文献[3]提出了有效的服务组合策略 BITS，使得 Web 服务的组合问题转化为树的表示及遍历并且 BITS 能够得到全部和最优的满足用户需求的 Web 服务组合方案。文献[4]提出一种基于语义匹配的查找服务链方法,认为服务链中邻接服务在语义上是匹配的。但是在以上基于语义的 Web 服务组合方法中也存在着一些不足：(1)只针对顺序关系的服务组合，没有考虑到并发、选择等服务组合问题。(2)在判断服务是否能被组合时，只 1 用到了输入输出等特定类型的参数进行语义判断，这在服务组合中具有一定的局限性。

本文提出了一种语义精确匹配的生成组合树服务组合方法。文章第一节给出相关概念和定义，第二节给出了服务组合实现相关算法，第三节通过具体的实例说明组合算法的应用，最后给出了总结。

2　相关概念和定义

Web 服务可以被看作程序或动作。在前者中，服务的输入输出显得重要，在后者中，前提条件和执行效果就很重要[5]。语义 Web 服务研究者利用这些信息，采用 Web 服务本体，例如 OWL-S[6]和 WSML[7]形式化的表示服务的功能。在海量的 Web 服务中存在着服务的输入输出相同，但却有着不同的前提条件和执行效果。为了更精确的发现、组合 Web 服务，就要充分利用服务的输入、输出参数以及服务的前提条件和执行

① 基金项目:安徽省高校省级优秀青年人才基金(2009SQRZ200);安徽省高校省级自然科学基金(KJ2010B010)

收稿时间:2010-12-09;收到修改稿时间:2011-01-23

效果。

2.1 Web 服务、服务请求的定义

定义 1. WebService={I,Pre,O,Ect},其中，I 为 WS 的输入参数集合，I={I_1,I_2,…,I_n}；Pre 为前提条件集合，Pre={pre_1,pre_2, … ,pre_m}；O 为输出参数集合，0={O_1,O_2, … ,O_p}；Ect 为执行效果集合，Ect={Ect_1,Ect_2,…,Ect_n}。

Pre 和 Ect 的定义是为了提高服务匹配的精确度，因为具有相同输入输出的服务可能在不同的前提条件下产生不同的预期执行效果。

定义 2. 服务请求 WSR={I,Pre,O,Ect}，其各个参数的定义与 WS 相同。这里把服务请求分成两个特殊的服务进行考虑：服务 S_0={I、pre、、}，即 S_0.O=WSR.I,S_0.Ect=WSR.Pre；S_1={、、O、Ect}，即 S_1.I=WSR.O, S_1.Pre=WSR.Ect。

2.2 前驱服务

定义 3. 在领域本体中，对于参数 A，若存在服务 WS，其输出 O 中存在参数 B，满足 A 语义被定义为 B 语义的等价类，或者 A 被定义为 B 的子类，即 B 包含 A(A=>B)，则服务 WS 是 A 的前驱服务。记 Pre(A)为参数 A 的前驱服务集函数。

定义 4. 对于服务 WS_1 和服务 WS_2，若存在 WS_1.O=>WS_2.I,即 WS_1.O 语义包含 WS_2I，则称 WS_l 是 WS_2 的前驱服务。

定义 5. 服务集 S={Sm_1 … Sm_i … Sm_t},m_i ∈ {0,1,2,…,n},O(S)= $\cup_{j=m_1}^{m_t} s_j.O$。服务集 S 被定义为服务 S_i 的前驱服务集，当且仅当：（1）O(S)⊇I(S_i)；(2)∀ Sm_i∈S，I(Si)⊄(O(S)- Smi.O)。

定义 6. 服务匹配度是指一个服务的输出、执行效果与另一个服务的输入、前提条件间的相似度。设有服务 WS1、WS2,分别定义 WS1 的输出和执行效果以及 WS2 的输入和前提条件如下：O_1={O_{11}, O_{12}, … ,O_{1m}}，Ect_1={Ect_{11},Ect_{12}, … ,Ect_{1n}}，I_2={ I_{21}, I_{22},…,I_{2h}}, Pre_2={ Pre_{21},Pre_{22},…, Pre_{2k}},则服务 WS_1 和 WS_2 的匹配度记为 SIM（WS_1，WS_2）。SIM（WS_1，WS_2）=

$$\sum_{j=1}^{h} w_j \max(\overset{m}{sim}(O_{1i}, I_{2j})) + \sum_{j=1}^{k} \max(\overset{n}{sim}(Ect_{1i}, \Pr e_{2j}))$$

,其中 w_j 是表示各个输入参数重要性的权重，且 $\sum_{j=1}^{h} w_j = 1$。

3 服务组合的实现

3.1 优化搜索空间

采用模糊聚类方法，在未知聚类结果数目的前提下，对领域本体内所描述的语义 Web 服务进行模糊聚类。模糊聚类将功能相似的 Web 服务放在一个类中，将聚类后的结果 C_1，C_2，…，C_n 存储在 UDDI 注册数据库里，类作为一个服务来描述。在服务组合时首先定位到类中，然后选择和目标服务匹配度最高的服务，同时考虑匹配服务的 Qos 的约束。

优化后的搜索空间可以完成服务的注册、发现和绑定。对于新加入的服务，首先和服务类 C_1，C_2，…，C_n 逐一匹配，把它归属于相似度最高的那一类，当相似度小于某一给定的阀值时，则把新服务看成一个新类 C_{n+1}。服务绑定到某一类后，则可以利用定义 7 来进行实际服务匹配，以完成服务的发现和调用。

3.2 动态服务组合的实现

Web 服务组合是以服务请求 WSR 的输出及前提条件为驱动，以其输出及执行效果为结束条件，在服务注册库 UDDI 和领域本体作为 Web 服务基础数据的基础上，找到一系列的 Web 服务，使其满足服务请求功能及性能要求的过程。Web 服务组合不仅要满足服务请求的功能要求，而且还要满足其 Qos 约束，下面给出求解前驱服务集和语义组合树的构造算法。

3.2.1 前驱服务集算法

求服务 Si 的前驱服务集 Preservices(Si)算法描述如下：

(1) $\forall I_i \in s_i$，利用定义 4 求 Pre(I_i)，Pre(I_i)为参数 I_i 的前驱服务集。

(2) 求 Pre(I_1)Pre(I_2)…Pre(I_i)…Pre(I_h)，h 为服务 Si 输入参数的个数。

(3) 对(2)的结果利用定义 5 进行处理，去除笛卡尔积中相同元素、包含其它元素的元素以及非最小服务集。

(4) 处理后的笛卡尔积中的各个元素即是 Preservices(Si)，并返回。

在优化后的搜索空间上执行 Preservices(Si)算法，得到 Si 的前驱服务集中的服务不是具体的可调用的 Web 服务，而是聚类分析后所得到的类，最后的绑定要在服务组合方法中利用定义 6 来发现和绑定某一类中最符合组合要求的服务。

3.2.2 服务组合树构造算法

定义组合树根结点的服务质量 Qos 为其孩子结点 Qos 的和。根结点 Qos 的计算要根据组合服务的关系以及服务属性的类别来计算，Qos 属性值计算分为累加级如 Price,级乘级如 available，混合级如 Time 等，相同的服务在二次调用时 Qos 不参与计算。下面给出以服务请求输出服务 S’为根结点，以服务请求输入服务 S0 各个参数为叶子结点的服务组合树构造算法：

(1) If(S’.IS0.O and S’.preS0.Ect)then (2)else(3);

(2) If (S.Qos meet S0.Qos limit) then node (S,)<-newnode(S,); S’.children=Null；Qos (S’);return(S’);else return Null；

(3) Preservices(S’);if(Preservices(S’)=Null) then return Null;

(4) Preservices(S’) element Do

① WSPreservices(S’) element，S’<-WS;Goto(1);

② (Certain Preservices(S’))=Null;Remove Preservices(S’);

③ 根据定义 6 计算服务匹配度,选择最优的孩子节点;

Qos(S’);if(S’.Qos not meet S0.Qos) Remove Preservices(S’);

(5) Choose Best Preservices(S’).Qos as it children;construct S’and its children node；Return S’;

本算法是一个递归调用算法，构造服务组合树时从叶子结点开始，而不是从根结点开始构造。这样能保证构造出的服务组合树功能及 Qos 都满足服务请求的要求。这是由于组合树中结点的 QoS 属性值是根据孩子结点的 QoS 属性值计算得到，所以组合算法应采取由下到上的构造方法。

4 实例分析

一个用户想知道将要抵达的某个城市景点的天气情况以及当天是否允许进入参观，这个服务请求-WSR 将景点名称（scenicspot）、景点所在城市（city）和要查询日期（date）作为输入参数，并提供景点的天气情况和景点所属路段作为输出参数。在这个案例中 WSR 描述如下：

WSR{I(scenicspot,city,date), Pre:know (province), unknow(district),O(weather,temper,road),Ect:admission}

领域本体内的服务描述如表 1 所示：

表 1　领域本体内的服务描述

ID	输入 I	前提条件 Pre	输出 O	执行效果 Ect
S0			scenicspot, city,date	know(province), unknow(district)
S1	city	Unknown (district)	province	know(country)
S2	scenicspot	Know (province)	district, road	admission
S3	district, province,date	Know (country)	temper, weather	know(peoplecount)
S4	postalcode, date	Know (scenicspot)	temper, weather	Know(city)
S5	postalcode, scenispot	Know (provinceun	district,city	know(country)
S6	scenicspot, city	Unknown (district)	province, country	unadmission
S’	weather, temper,road	admission		

其中 S0, S’是 WSR 转化的两个服务。经过聚类分析得到如下分类结果 S1 与 S6 属于一类，其余服务各属于一类，记 Sc={S1,S6}=S6∪S1。WSR 的服务组合就是建立一个以 S’为根，S0 输出参数为叶子结点的组合树，过程如下：由于 S’.I 和 S’.Pre 不符合 S0 要求，所以求前驱服务集 Preservices(S’)={{S2,S3},{S2,S4}}，对{S2,S4}中的 S2、S4 分别递归调用服务组合方法，对于{S2,S4}分别求 Preservinces(S2)、Preservinces(S4)都为空,则剪去前驱服务集{S2,S4}。对{S2,S3}中的 S2、S3 分别递归使用服务组合方法，求前驱服务集 Preservinces (S3)={{Sc,S2},{Sc,S5}}，所以 Preservices(S，)={S2,S3}。由于 S3.I 和 S3.Pre 还不符合 S0 的要求，对 Preservinces(S3)中的每个元素{Sc，S2},{Sc，S5}递归调用组合算法，则{Sc,S2}和{Sc,S5}的输入及前提条件都满足 S0 要求，利用定义 6 在服务类 Sc 中找出一个与 S3 匹配度最高的服务，结果为 S1。

由{S1,S2}和{S1,S5}分别计算 S3 的 Qos 属性值，选择满足 WSR 的 Qos 约束条件的前驱服务集，若都满足，则按 Qos 最优选择，这里假定{S1,S2}最优。再由 S3 和 S2 的 Qos 计算最终组合树根结点 S’的 Qos,并与 WSR.Qos 比较，满足则服务组合成功，且返回的是具有最优服务质量的组合服务。本次服务请求 WSR 返回的组合树如图 1 所示。

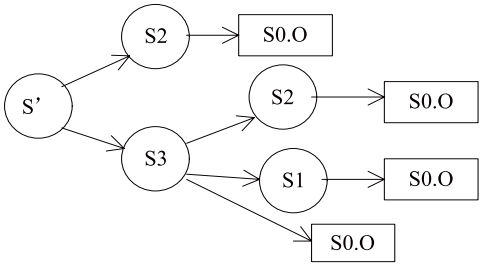

图 1 WSR 返回的服务组合树

5 结语

本文给出了 Web 服务描述方法，在优化搜索空间的前提下，利用 Web 服务语义匹配算法，提出了一种由服务请求生成满足服务质量约束的组合树的服务组合方法。通过服务的输入输出、前提条件执行效果的相互匹配，来更有效地绑定服务。它能够有效的解决顺序的、并发的、选择的服务组合等问题。

参考文献

1 Martin D, Burstein M, et al. OWL-S1.0. http:www.daml.org/services/owl-s/

2 Li M,Wang DZ, Du XY, et al. Dynamic Composition of Web Services Based on Domain Ontology. Chinese journal of Computers, 2005,28(4):644−650.

3 Zhou AY, Huang S, Wang XL. BITS:A binary tree based Web service composition system. International Journal of Web Services Research, 2007,4(1):40−58.

4 Fu YN, Liu L, Jin CZ. Service chain-based Approach for web service composition. Journal on Communications, 2007,28 (7): l92−97.

5 Marta S, Debb R, Sander VS. An experience report on using DAML-S. Proc. of 12th International World Wide Web. Hungary, 2003.

6 W3C(World Wide Web Consortium).Web ontology language (0WL-S).http:// www.w3.org / 2004 / OWL/#specs.

7 Laurenh, Romand, Kelleru. Web services modeling ontology-standard(WSMO.standard).http://wsmo.org/2004/d2/vv0.2/.

8 Shen GY, Li JH. Web Service Automatic Composition Algorithm Based on Semantics.Computer Engineering, 2009,35(16):262−263.

(上接第 198 页)

参考文献

1 EMC Corporation. EMC in Major Storage Performance Break through; First with Enterprise-Ready Solid State Flash Drive Technology.2008.http://www.emc.com/about/news/press/us/2008/011408-1.htm.

2 MySQL Performance. Should I buy a Fast SSD or more memory.2010.http://www.mysqlperformanceblog.com/2010/04/08/fast-ssd-or-more-memory.

3 Lee SW, Moon B, Park C, Kim JM, Kim SW. A case for flash memory ssd in enterprise database applications. Vancouver, Canada: Proc. of the 2008 ACM SIGMOD International Conference on Management of Data. 2008. 1075-1086.

4 Oi H. A Case Study:Performance Evaluation of a DRAM-Based Solid State Disk. Frontier of Computer Science and Technology (FCST), 2007 Japan-China Joint Workshop on; Wuhan, China. Beijing: FSCT, 2007. 57-60.

5 Chen SM. Flash Logging: exploiting flash devices for synchronous logging performance. Rhode Island, USA: Proc. of the 35th SIGMOD International Conference on Management of Data, 2009. 73-86.