高技术通讯 2012 年 第22 卷 第6 期:596 ~ 603

doi:10.3772/j.issn.1002-0470.2012.06.007

基于语义 Web 服务内部流程接口匹配的服务组合算法①

邱 爽②* 王亚东③*** 刘永壮**

(*哈尔滨工业大学生物医学工程研究中心 哈尔滨 150080)

(**哈尔滨工业大学计算机科学与技术学院 哈尔滨 150001)

摘 要 针对基于服务一级粗粒度的语义 Web 服务组合方法无法有效地发现满足用户需求的潜在的服务组合,导致组合成功率低的问题,提出了基于语义 Web 服务内部流程接口匹配的自动服务组合方法。该方法利用基于语义的 Web 服务描述语言(OWL-S)并在领域本体的支持下对 Web 服务语义进行描述,对服务与服务流程进行区分,将服务流程作为组合操作对象,通过计算不同的 Web 服务内部流程接口之间语义关联程度,遵从后继服务流程选择策略自动地生成能够满足用户需求的流程组合方案,并通过反向检索组合流程,消除组合方案中冗余的服务流程。通过一系列的仿真实验对该组合方法的成功率、效率等方面进行了验证,结果证明了该方法可以更加有效地根据用户请求自动生成服务组合方案。

关键词 语义 Web 服务,服务组合,Web 服务本体描述语言(OWL-S),领域本体,服务流程

0 引 言

Web 服务是一种基于网络环境实现的应用程序,它具有自适应、自描述、模块化特点及良好的互操作能力,但这些服务大都功能有限,无法满足复杂应用的需求。如何有效地组合分布于网络中的功能各异的 Web 服务,实现服务之间的无缝集成,形成功能强大的复合服务来实现用户目标,已成为 Web 服务发展的一个热点问题。

目前服务组合主要采用三类方法。一类是基于工作流的服务组合方法,主要以工作流技术为基础,基于工作流模型建立服务组合流程,如 e-Flow[1],CAFISE[2],METEOR-S[3]。该类服务组合方法针对固定的业务流程,有成熟的工具支持,易于实现,但需要大量领域专家参与,导致组合方法自动化程度不高,灵活性不足。第二类是基于人工智能理论的 Web 服务自动组合方法,如文献[4]和文献[5]将服务组合问题抽象为规划问题从而自动求解,即给定一个初始状态和目标状态,在一个服务集合中寻求一条服务的路径从初始状态到达目标状态的演变。该类方法虽然摆脱了人工参与,能够完全实现服务的自动组合,但都基于某种形式化方法或者推理系统,需要对服务进行预处理和形式化转换,因而复杂度较高,且随着规划空间的增大,复杂度会显著增高。第三类是语义驱动的 Web 服务组合方法,即将语义 Web 技术与 Web 服务相结合,通过对 Web 服务进行语义层面的描述,使得 Web 服务能够被机器或者程序所理解,从而实现自动的 Web 服务组合。文献[6]与文献[7]的方法是典型的基于语义的 Web 服务组合方法,它们利用领域本体及其推理能力生成一个优化的服务组合图,将服务组合问题转化为在该优化的服务组合图中查找一条满足要求的路径的问题。由于需要对服务注册库中的服务建立服务依赖关系,所以当服务数量很大、服务之间关系复杂时,建立服务依赖关系图及对图进行搜索操作的时间开销都会很大,大大降低了服务组合效率。文献[8]提出了一种无回溯的反向链的 Web 服务的自动组合方法,该方法可以根据不同的组合需求建立服务组合,虽然不需要事先对整个服务库中的服务建立关系依赖图,但服务组合时与文献[6]和文献[7]类似,并没有对服务与服务内部的功能单元

① 国家科技支撑计划(2008BAI64B03)资助项目。

② 男,1980 年生,博士生;研究方向:语义 Web 服务组合及其应用;E-mail:qiushuang.hit@gmail.com

③ 通讯作者,E-mail:ydwang@hit.edu.cn

(收稿日期:2011-03-02)

做区分,导致服务发现的粒度较大,不能准确地发现满足功能需求的 Web 服务。本文提出了一种基于语义 Web 服务内部流程接口匹配的自动服务组合方法。该方法将语义 Web 服务的内部流程作为功能组合的操作对象,从而更加有效地发现满足用户需求的潜在的服务组合。在组合过程中无需事先建立服务依赖关系图,可以实时地将语义 Web 服务内部流程进行组合完成不同的用户需求。由于有相应的优化机制,该组合方法可利用后继服务搜索策略既能保证满足用户需求,又可以高效地进行服务组合,同时利用反向检索冗余流程可以去除对产生用户需求没有贡献的冗余服务流程。

1　Web 服务语义描述

Web 服务组合所采用的具体方法和技术依赖于 Web 服务描述所采用的模型。本文提出的是基于语义 Web 服务内部流程接口匹配的自动服务组合方法,在实现 Web 服务自动组合之前必须对 Web 服务进行语义描述。

目前在语义 Web 服务研究领域,较有影响力的描述语言有 OWL-S[9]、WSMO/WSML[10]、SWSO/SWSL[11]、WSDL-S[12] 和 USDL[13]。为了支持 Web 服务语义化描述,本文采用描述 Web 服务的 Web 本体语言(ontology Web language for services, OWL-S)的 Web 服务描述模型。OWL-S 是基于语义的 Web 服务描述语言,使用 OWL 语言定义 Web 服务本体,并将每个具体的 Web 服务作为 Web 服务本体的一个实例加以描述。在 OWL-S 中服务描述的基本信息主要有三部分:(1) 服务轮廓(service profile),提供了服务的基本描述;(2) 过程模型(process model),以流程的形式描述了服务中各子过程的逻辑执行顺序;(3) 服务基点(service grounding),描述用于访问服务的具体细节,如服务的 URI 地址、传输协议、消息格式等。

OWL-S 的服务模型(service model)描述了服务内部的细节,可以为服务匹配、发现及组合提供更为详细的信息。根据 OWL-S 描述的 Web 服务我们可知,Web 服务内部包括若干原子流程(atomic process),这些原子流程有的独立完成某一功能,有些通过由过程模型(process model)定义的控制结构组合在一起形成复合流程(composite process),本文将这样的复合流程视为 Web 服务内部能够完成某一功能的独立的功能单元,即不考虑其内部控制结构,并与其它原子流程统称为 Web 服务内部流程。

定义 1　Web 服务的内部流程 p 是一个七元组 $p = (ID_{ws}:p_{id}, Ip, Op, Pp, Ep, On)$,其中:

(1) ID_{ws} 是流程 p 所属的 Web 服务在服务注册库中的唯一标识;

(2) p_{id} 是该流程的名称,在 Web 服务内部起到唯一标识作用;

(3) $Ip = \{i_1, i_2, \cdots, i_n\}$ 是该流程的输入集合;

(4) $Op = \{o_1, o_2, \cdots, o_m\}$ 是该流程的输出集合;

(5) Pp, Ep 分别是该流程执行时的前置条件与后置效果;

(6)对于 $\forall e \in Ip \cup Op$,均与某领域本体中的概念关联;

(7) On 代表概念所属的领域本体。

Web 服务流程接口主要包含两类信息:

(1)功能描述信息 I 和 O;

(2) 功能约束条件 P (precondition) 和 E (effect)。

在基于 OWL-S 的 Web 服务功能属性匹配方面更多的研究主要侧重于基于 I 和 O 的匹配,而针对 P 和 E 的研究由于描述标准不统一而关注较少,而现有的基于功能约束条件的匹配主要采用"归纳逻辑程序设计(inductive logic programming-ILO)"领域的 θ 包含方法[14-16],该方法只是在语法层次进行匹配,同时 θ 包含是 NP 完全问题。本文的组合方法主要是基于功能描述信息 I 和 O 的匹配。

在 Web 服务内部的流程虽然包括两种类型流程,即原子流程与复合流程,但是在本文中对二者并不加以区分,复合流程对外暴露的是其接口信息而不考虑其内部控制结构,与其它原子流程一样都视为具备单一功能的服务内部流程,在此基础上我们给出 Web 服务以及服务组合请求的形式化描述。

定义 2　一个 Web 服务 S 是一个四元组 $S = (ID_{ws}, P_{ws}, QoS, On)$,其中:

(1) ID_{ws} 是该 Web 服务在服务注册库中的唯一标识;

(2) $P_{ws} = \{p_1, p_2, \cdots, p_n\}$ 是该服务内部流程集合;

(3) QoS 描述该 Web 服务的服务质量,如响应时间、可靠性、信誉度、价格等;

(4) On 代表概念所属的领域本体。

定义 3　Web 服务请求 WSR 可以形式化为一个四元组 $WSR = (IR, OR, Constrain, \delta)$,其中:

(1) WSR 是 Web 服务请求的名字;

(2) IR 是该服务请求能够提供的输入集合;

(3) OR 是该服务请求期望得到的输出集合;

(4) $Constrain$ 是用户提出的约束条件, $Constrain = (P_r, E_r, QoS_r)$;

(5) $0 \leqslant \delta \leqslant 1$ 是用户设定的服务功能满意度阈值。

2 基于流程接口匹配的自动服务组合

2.1 基本思想

Web 服务提供的功能实质上是由其包含的若干流程所提供,流程是 Web 服务的基本功能单元实体,而 Web 服务组合最终体现的也是 Web 服务流程间的组合。本文提出的基于服务流程接口匹配的组合算法(process interface-match based composition algorithm,PICA)基于这一现实,利用 OWL-S 服务模型以及领域本体对 Web 服务流程进行语义描述,将服务内部流程作为服务组合操作的对象,通过流程接口之间的语义匹配寻找一组功能关联的 Web 服务流程使其通过组合满足用户需求。

2.2 服务流程功能关联度

2.2.1 概念集合相似度计算

本文采用功能关联度来描述不同服务流程之间功能关联的强弱程度,而功能关联度是用接口参数集合的语义相似程度来刻画的。接口参数集合是领域本体中概念的集合,任意两个领域本体概念集合的语义相似度都可以利用扩展二分图的最佳匹配求解[17],即对于领域本体中两个概念集合 C_i 和 C_j,可以得到加权二分图 $G = (C_i, C_j, E)$,其中 $E = \{(x,y) \mid x \in C_i, y \in C_j, C_{sim}(x,y) > 0\}$,再利用 KM[18]算法计算可以得到 C_i 和 C_j 两个概念集合的最优匹配 M 及其权重 $W(M)$,下面给出任意两个概念集合语义相似程度 $CS_{sim}(C_i, C_j)$ 的详细定义。

定义 4 概念集合语义相似度:对于给定的两个概念集合 C_i 和 C_j,利用 KM 算法可以得到对应的一个加权二分图 $G = (C_i, C_j, E)$ 的最佳匹配 M,则 C_i 和 C_j 的语义相似程度 $CS_{sim}(C_i, C_j) = Weight(M) = \frac{1}{k}\sum_{i=1}^{k} w_i$,$k$ 为 M 的匹配数,即为边的条数,其中:

(1) w_i 是 M 中边的权值。w_i 代表两个概念集合中对应的概念 c_i 与 c_j 之间的语义相似度,可以利用公式

$$C_{sim}(c_i, c_j) = \begin{cases} 1, & (c_i = c_j) \\ e^{-\alpha l} \times \dfrac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}}, & (c_i \neq c_j) \end{cases} \quad (1)$$

得到[19],其中 l 表示两个概念之间最短路径长度,h 表示两个概念在概念树中最近的相同上层概念在树中的高度,$\alpha, \beta \geqslant 0$ 是用于调节 l 和 h 在相似度计算中的影响度。

(2) $Weight(M)$ 是二分图 $G = (C_i, C_j, E)$ 最佳匹配的权重。

通过给出计算概念集合语义相似度,我们接下来给出概念集合的等价性定义:

定义 5 概念集合的等价性:对于领域本体中两个概念集合 C_i 和 C_j,如果满足 $CS_{sim}(C_i, C_j) \geqslant \gamma$($\gamma$ 为可接受的概念集合相似度的下限),称 C_i 语义包含 C_j,记为 $C_i \supseteq C_j$;如果有 $C_i \supseteq C_j$ 且 $C_j \supseteq C_i$,则称 C_i 和 C_j 语义相等,记为 $C_i \equiv C_j$。

2.2.2 流程功能关联度

本文利用服务流程功能关联度(association degree, AD)来定量刻画服务流程之间的功能关联程度。如前文所述,流程接口之间功能关联度可以通过对应接口概念参数集合的扩展二分图最佳匹配求解,通过得到的参数集合间的最佳匹配不但可以定量计算服务流程间的功能关联度,而且可以得到接口参数间的一一映射关系,即接口参数赋值关系。

定义 6 服务流程功能关联度(AD):对于给定一个服务流程 $p_i = (I_i, O_i)$ 与服务流程集合 $P_{i-prior} = \{p_1, p_2, \cdots, p_n\}$,$n \geqslant 1$,$p_i$ 与 $P_{i-prior}$ 功能关联度 $AD(P_{i-prior}, p_i) = CS_{sim}(OP_{i-prior}, I_i)$,其中 $OP_{i-prior} = O_1 \cup O_2 \cup \cdots \cup O_n$。

通过服务流程功能关联度的计算以及概念集合等价性的定义我们可知,当服务流程之间的 $AD(P_{i-prior}, p_i)$ 满足 $AD(P_{i-prior}, p_i) \geqslant \gamma$ 时(γ 此时为设定的服务流程之间最小关联度),表明 $P_{i-prior}$ 的输出参数集合与 p_i 的输入参数集合语义关联,即 $P_{i-prior}$ 与 p_i 功能关联,$P_{i-prior}$ 可以为 p_i 提供所需要的输入参数,这时我们将 $P_{i-prior}$ 称为 p_i 的前驱服务集合,p_i 为 $P_{i-prior}$ 后继服务。

2.3 后继服务流程选择策略

2.3.1 组合功能满意度

为了保证组合流程在功能上满足用户的需求,本文给出组合服务流程功能满意度(function satisfaction,FS)的概念,利用 FS 来选择满足组合需求的后继服务流程。

定义 7 组合功能满意度(FS):对于给定服务

流程 p_i 及其前驱服务流程集合 $P_{i-\text{prior}} = \{p_1, p_2, \cdots, p_n\}, n \geqslant 1$ 是 p_i 的前驱服务流程个数，$FS(p_i) = AD(P_{i-\text{prior}}, p_i) \times \min FS(P_{i-\text{prior}})$。

由 FS 定义可知，当 $n = 1$，即 p_i 只有一个前驱服务流程 p_j 时，$FS(p_i) = AD(p_j, p_i) \times FS_j$；当 $n > 1$ 时，即 p_i 存在多个并行前驱服务流程时，$FS(p_i) = AD(P_{i-\text{prior}}, p_i) \times \min FS(P_{i-\text{prior}}) = CS_{\text{sim}}(OP_{i-\text{prior}}, I_i) \times \min FS(P_{i-\text{prior}})$，由其并行前驱服务流程中 FS 值最小的服务流程决定。

2.3.2　后继服务流程选择策略

为了保证服务组合方案在功能上能够满足需求，同时防止死锁及避免将那些对产生用户需求无用的冗余的服务流程被加入到组合方案中，针对服务组合请求 $WSR = (IR, OR, \delta)$，对于任意服务流程 p_i 及其前驱服务流程集合 $P_{i-\text{prior}} = \{p_1, p_2, \cdots, p_n\}, n \geqslant 1$，同时满足以下四个条件时，可以将 p_i 加入服务流程组合方案中：

(1) $\{p_i\} \notin P_{i-\text{prior}}$，避免死锁的出现；

(2) $AD(P_{i-\text{prior}}, p_i) \geqslant \delta$，保证前驱后继服务功能关联；

(3) $FS(p_i) \geqslant \delta$，保证组合流程功能满足用户需求；

(4) $CS_{\text{sim}}(OP_{i-\text{prior}}, O_i) < \delta$，保证 p_i 的输出不被其前驱服务的输出集合语义包含，提高后继服务搜索效率。

2.4　服务流程组合算法

通过前文给出的定义以及后继服务流程选择策略，我们给出基于服务流程接口匹配的组合算法(PICA)，如表 1 所示。

表 1　PICA 形式化描述

基于服务流程接口匹配的组合算法 PICA
输入：用户需求：$WSR = (IR, OR, \delta)$
　　服务注册库：$Repository = \{p_1, p_2, p_3, \cdots, p_n\}$
　　Limit of steps：$StepLimit$
输出：服务流程组合图 G
步骤：
1. 初始化：创建服务流程 $p_{start} = (I_{start}, O_{start})$ 和 $p_{final} = (I_{final}, O_{final})$，服务流程组合图 $G = (V, E)$，
2. $I_{start} = O_{start} = IR, I_{final} = O_{final} = OR, V = \{p_{start}\}, E = \varnothing, Output(G) = \bigcup_{i=1}^{n} Op_i, \forall p_i \in V, 1 \leqslant i \leqslant n$,
3. $FS(p_{start}) = 1$，全局变量 $CurrentParametersMap = \varnothing, CurrentParametersList = Output(G) = IR$.
4. **While**(! Goal($CurrentParametersList$)&&$step$! = $StepLimit$)
5. 　　$step$ ++
6. 　　**for** every $p_j \in Repository \wedge !isVisited(p_j)$
7. 　　**do**
8. 　　get an optimal matching set M according to $Output(G)$ and $I(p_j)$;
9. 　　Compute the similarity according to M: $AD(P_{j-prior}, p_j) = CS_{sim}(Output(G), I(p_j)) = Weight(M)$;
10. 　　**if** $AD(P_{j-prior}, p_j) \geqslant \delta$
11. 　　**then** get all the possible prior processes from G according to M and $CurrentParametersMap$
12. 　　compute the value of
13. 　　$FS(p_j) = AD(P_{j-prior}, p_j) \times \min FS(P_{j-prior}) = CS_{sim}(Output(G), I(p_j)) \times \min FS(P_{j-prior})$;
14. 　　**if** $FS(p_j) \geqslant \delta$
15. 　　**then** compute the value of $CS_{sim}(OP_{j-prior}, O_j) = CS_{sim}(Output(G), O_j)$
16. 　　**if** $CS_{sim}(OP_{j-prior}, O_j) < \delta$
17. 　　**then** add $O(p_j)$ to the $CurrentParametersList$ and set $Visited(p_j) = true$;
18. 　　add $O(p_j)$ and their associated process to the $CurrentParametersMap$;
19. 　　add p_j to G according to their prior processes.
20. Search the prior processes to remove the redundant process nodes along the inverse direction from the goal
21. node p_{final} to the virtual process node p_{start} to get a new graph G, **return** G

PICA 通过输入用户的请求信息，返回满足用户需求的服务流程组合图。算法分为两个阶段，第一阶段是前向逐层搜索组合流程阶段。算法描述第 6 至 19 行表示依据后继服务选择策略向前搜索组合流程并构造服务流程组合 DAG 图。第二阶段是后向去除冗余服务流程节点阶段，如算法描述中 20、21 行所示，以虚拟服务流程节点 p_{final} 为起始节点，虚拟服务流程节点 p_{start} 为终点，反向遍历前一阶段

生成的服务组合 DAG 图并输出更新后的 DAG 图，其目的就是可以根据前一阶段建立的每个服务流程与其前驱服务流程的关系反方向对 DAG 图遍历从而自动地将与产生用户需求信息没有直接作用的冗余的服务流程节点去除，图的遍历搜索已有很多成熟的方法，本文不再详述。算法描述第 4 行为算法执行中止条件，如表 2 所示，当产生用户需求信息并且组合服务流程的功能满足用户设定的阈值时返回流程组合方案。

表 2 服务流程组合终止条件

Goal
Input: *CurrentParametersList*
Output: true or false to judge if the search procedure reach the goal node p_{final}
1. Get an optimal matching set M according to *CurrentParametersList* and OR;
2. Compute the similarity according to M: $AD = CS_{sim}(CurrentParametersList, OR) = Weight(M)$;
3. **if** $AD \geqslant \delta$
4. **then** get all the possible prior processes from G according to M and
5. *CurrentParametersMap*, compute the value of FS;
6. **if** $FS \geqslant \delta$
7. **then** add the virtual process p_{final} to G
8. return true
9. **else** return false

通过推导不难得出 PICA 的时间复杂度为多项式级的时间复杂度 $O(m^3nk^5)$，其中 m 为单个服务流程接口参数个数的最大值，n 是服务流程个数，k 是服务流程组合过程中人为设定的迭代计算次数，即算法描述中的 *StepLimit*。在实际情况当中绝大多数 Web 服务内部流程接口的参数个数很少，m 的变化对于整个算法执行时间的影响有限，k 是人为设定的一个常量，所以算法的时间复杂度更大程度地取决于服务注册库中服务流程的规模。

2.5 示例

例 1 根据用户提供的组合需求信息 $WSR = (IR, OR, \delta)$，$\delta = 0.7$，在服务流程注册库中寻找一组服务流程通过组合完成任务，以虚服务流程 $p_{start} = (I_{start}, O_{start})$ 为搜索起点，$p_{final} = (I_{final}, O_{final})$ 为搜索终点，服务流程组合方案生成过程如图 1 所示。

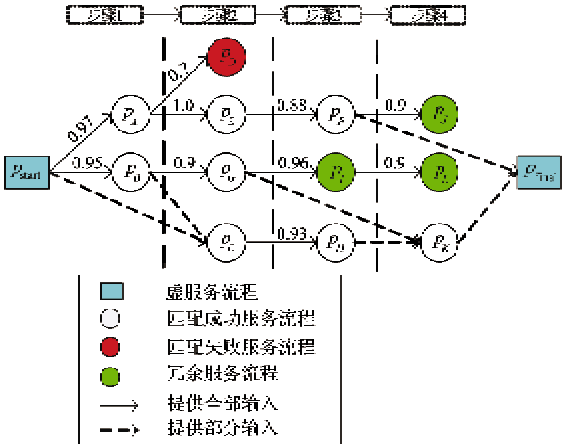

图 1 前向搜索组合服务流程

通过前向搜索过程以及后向除去冗余流程过程，最终得到的服务流程组合方案如图 2 所示。

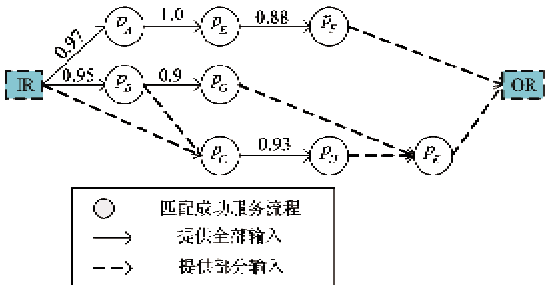

图 2 最终生成的服务流程组合方案

3 实验结果及分析

3.1 实验环境

本文中的所有实验运行在 Dell Optiplex 760 上，软硬件环境为：Intel Core Duo CPU 2.66GHz，4G RAM，Windows 7 Operating System。

3.2 实验数据集

鉴于目前没有标准的测试数据集用于基于语义的 Web 服务自动组合研究，本文基于马里兰大学开发的 OWL-S API，开发了一个服务数据集生成工具，随机生成一定数量的 Web 服务作为测试数据集。服务本身不执行有意义的操作，但接口描述符合 OWL-S 规范，并且每个服务内均包含多个服务流程。从测试角度看，这些随机生成的服务与真实的 Web 服务并没有区别。

实验数据集设置:

(1)本文不考虑跨本体的概念相似度计算,利用英国的 myGrid 项目所开发的 myGrid Ontology[20],该本体主要描述了生物信息学领域内所有的 Web 服务当中所涉及到的或可能涉及到的概念,该本体可描述绝大多数的生物信息领域当中真实的 Web 服务。我们从中抽取一个由 200 个概念组成的本体片段为模拟实验所用,本体的解析工作主要利用惠普实验室所开发的 Jena 2.6.2。

(2) 用户的服务请求 WSR 利用 OWL-S 当中的 Service Profile 描述,同样利用基于 OWL-S 开发的工具随机生成,由于暂不考虑 P、E 及 QoS,所以令 $WSR = (IR, OR, Constrain, \delta)$ 中 $Constrain = (P, E, QoS) = \varnothing$。本文中每组测试数据均采用 100 个服务请求,$IR$ 和 OR 分别都对应领域本体中的概念上,集合参数个数为 1 ~ 4。

(3) 设定每个服务流程 $p_i = (I_i, O_i)$ 的输入、输出端口信息随机映射到领域本体的概念上,输入和输出概念集合参数个数为 1 ~ 4,随机生成 5 组测试数据集,它们包含的服务流程总数分别是 200,400,600,800,1000。

(4) 对于每组服务流程测试数据集自动生成相应的 Web 服务数据集,对应的 5 组 Web 服务数据集中 Web 服务数分别是 60,109,172,231,282 如表 3 所示,设定每个 Web 服务内部包含 2 ~ 5 个流程。

表 3　测试数据集规模

测试库	数据集 1	数据集 2	数据集 3	数据集 4	数据集 5
服务流程库	200	400	600	800	1000
Web 服务库	60	109	172	231	282

3.3　实验结果及分析

实验 1　服务组合成功率测试

对于随机产生的 100 个服务组合需求,分别在 5 组测试数据集上进行服务组合,基于语义的服务组合方法中设定功能满意度阈值 $\delta = 0.7$,基于关键字的服务组合令 $\delta = 1.0$ 即可,测试结果如图 3 所示。

实验中比较了基于语义的服务组合方法与基于关键字(key words)匹配的服务组合方法的服务组合成功率,结果表明,基于语义的Web服务组合无论是以 Web 服务为组合操作对象还是以 Web 服务内部流程作为组合操作对象,其成功率都要超过基于关键字的服务组合方法,这说明了进行基于语义的 Web 服务自动自合的必要性。

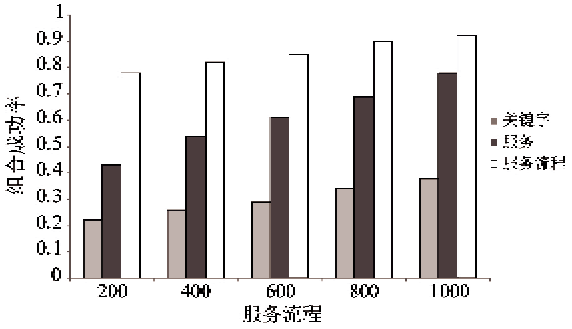

图 3　服务组合成功率测试

实验中还比较了分别以 Web 服务及 Web 服务内部流程为组合操作对象时的服务组合成功率,实验结果证明,分别在利用 OWL-S 的 Service Profile 描述的 Web 服务和 OWL-S 的 Service Model 描述的 Web 服务流程两种不同粒度上进行服务组合的成功率有所不同,以细粒度服务流程作为服务组合操作对象能够发现潜在的服务流程组合满足用户的需求,其成功率高于以 Web 服务作为操作对象的服务组合方法;同时,实验结果还表明随着服务注册库中的服务数量增加,满足组合要求的服务数量也随之增加,服务组合的成功率就会越高。

实验 2　服务组合满意度阈值 δ 对服务组合算法的影响

对服务流程总数为 800 的数据集进行测试,令服务组合满意度阈值 δ 分别为 0.4,0.5,0.6,0.7,0.8,0.9,1.0。对应每个阈值,随机生成 100 个服务组合请求,使用本文服务组合算法进行服务组合,得到相应的服务流程组合成功率和时间效率如表 4 所示。

表 4　服务组合满意度阈值对服务组合算法的影响

δ	服务组合成功率(%)	服务组合时间(s)
0.4	92.1	8.5
0.5	91.8	9.4
0.6	91.6	10.4
0.7	90.0	11.6
0.8	78.4	12.2
0.9	60.8	13.1
1.0	38.2	13.9

实验结果表明,用户设定的服务组合满意度阈值 δ 对组合算法运行时间没有太大的影响,但是对于组合成功率影响较大。δ 取值越大,说明用户对于组合服务的功能满意程度越高,满足要求的组合服务就会越少,组合成功率也就越低。

4 结 论

本文提出了基于语义 Web 服务内部流程接口匹配的自动服务组合方法,其主要贡献在于:(1)与基于 OWL-S Service Profile 一级粗粒度的 Web 服务接口匹配的组合方法不同,它是建立在 Service Model 服务流程一级细粒度的服务组合,将服务内部具有独立功能的单元服务流程作为服务组合的操作对象,从而更加有效地发现满足用户需求的潜在的服务组合,大大提高了服务组合的成功率;(2)其核心是寻找一组满足用户需求的功能上语义关联的 Web 服务流程,利用二分图最佳匹配解决服务流程间接口语义匹配的问题,通过这种方法不但可以定量地刻画服务流程间功能的关联程度,而且可以利用最佳匹配的参数之间单一映射关系确定服务流程接口参数之间的赋值关系;(3)利用后继服务选择策略来约束后继服务的选择,既可以提高后继服务搜索效率,又能够保证最终的组合方案在功能上满足用户的要求,并且在前向搜索的基础上增加了反向去除冗余服务流程的过程,使得最终生成的服务流程组合方案中不会包含对产生用户需求无用的服务流程;(4)最终生成的服务流程组合方案支持多元输入与多元输出,并且涵盖了顺序执行与并发执行的组合情况,这与以往大部分服务组合方案仅限于顺序执行不同,更加符合实际服务组合的要求。

下一步的工作是考虑跨本体的概念相似度计算从而使得服务功能语义关联度的刻画更加精确;在现有算法基础上考虑 QoS 因素,使得最终的组合方案在功能上能够完成任务的同时在性能上也能满足用户需求。以上几点国内外已有相关的研究成果可以作为下一步研究工作的参考,最终在此研究基础上,实现一个 Web 服务组合原型系统。

参考文献

[1] Casati F, Ilnicki S, Jin L, et al. Adaptive and dynamic service composition in eFlow. In: Proceedings of the 12th International Conference on Advanced Information Systems Engineering, Berlin, Germany, 2000. 13-31

[2] Han Y B, Zhao Z F, Li G, et al. CAFISE : an approach to enabling adaptive configuration of service grid applications. *Journal of Computer Science and Technology*, 2003, 18(4): 485-494

[3] Sivashanmugam K, Verma K, Sheth A, et al. Adding semantics to web services standards. In: Proceedings of the 1st IEEE International Conference on Web Services, Las Vegas, USA, 2003. 395-401

[4] Sohrabi S, McIlraith S A. Optimizing web service composition while enforcing regulations. In: Proceedings of the 8th International Semantic Web Conference, Berlin, Germany, 2009. 601-617

[5] 王杰生, 李舟军, 李梦君. 语义 Web 服务的自动化组合方法:研究综述. 计算机科学, 2007, 34(6): 19-23

[6] 李曼, 王大治, 杜小勇等. 基于领域本体的 Web 服务动态组合. 计算机学报, 2005, 28(4): 644-650

[7] Zhang R Y, Arpinar B I, Aleman-Meza B. Automatic composition of semantic Web services. In: Proceedings of the International Conference on Web Services, ICWS 03, Las Vegas, USA, 2003. 38-41

[8] 刘家茂, 顾宁, 施伯乐. 基于 Mediator 的 Web Services 无回溯反向链动态合成. 计算机研究与发展, 2005, 42(7): 1153-1158

[9] Martin D, Burstein M, Hobbs J, et al. OWL-S: Semantic Markup for Web Services. http://www.w3.org/Submission/OWL-S/: W3C, 2004

[10] McIlraith S, Son T C. Adapting Golog for composition of semantic web services. In: Proceedings of the 8th International Conference on Knowledge Representation and Reasoning, Toulouse, France, 2002. 482-493

[11] Sirin E, Parsia B, Wu D, et al. HTN planning for web service composition using SHOP2. *Journal of Web Semantics*, 2004, 1(4):377-396

[12] Klein M, Bernstein A. Toward high-precision service retrieval. *IEEE Internet Computing*, 2004, 8(11): 30-36

[13] Bansal A, Kona S, Simon L, et al. A universal service-semantics description language. In: Proceedings of the 3rd IEEE European Conference on Web Services, Växj, Sweden, 2005. 214-225

[14] Kawamura T, Blasio J D, Hasegawa T, et al. Preliminary report of public experiment of semantic service matchmaker with UDDI business registry. In: Proceedings of the International Conference on Service Oriented Computing, Berlin, Germany, 2003. 208-225

[15] Zaremski A M, Wing J M. Specification matching of software components. *ACM Transactions on Software Engineering and Methodology*, 1997, 6(4): 333-369

[16] Sycara K, Widoff S, Klusch M J. Larks: Dynamic

matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agent and Multi-agent Systems*, 2002, 5(2): 173-203

[17] 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. 计算机学报, 2008, 31(8): 595-602

[18] DAVES A M. The optimal assignment problem. http://www. math. uwo. ca/~ mdawes/courses/344/kuhn-munkres. pdf: Department of Mathematics University of Western Ontario, 2004

[19] Li Y, Bandar Z A, McLean D. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Trans on Knowledge and Data Engineering*, 2003,15(4):871-882

[20] Wolstencroft K, Alper P, Hull D. The myGrid ontology: bioinformatics service discovery. *International Journal of Bioinformatics Research and Applications*, 2007, 3(3): 303-332

Semantic Web service composition based on the interface matching of service processes

Qiu Shuang*, Wang Yadong*,**, Liu Yongzhuang**

(*Center for Biomedical Informatics, Harbin Institute of Technology, Harbin 150080)

(**School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001)

Abstract

This study proposed an automated semantic Web service composition method based on the interface matching of service processes. The proposed method distinguishes service processes with services in order to obtain the potential service composition. By computing the relevancy of different process interfaces and following the selection strategy of succeed service processes, the method can automatically generate user-oriented combined processes using the OWL-S to describe the semantic of Web service according to ontology. It can eliminate the redundant processes as well by examining the combined processes reversely, resulting in the final combined services. The efficiency and the accuracy of the method were tested with a series of simulated experiments, and the experimental result showed its feasibility.

Key words: semantic Web service, service composition, ontology Web language for services (OWL-S), ontology, service process