

# 一种基于范例的自动 Web 服务组合方法

袁东维, 李蜀瑜

(陕西师范大学 计算机科学学院, 陕西 西安 710062)

**摘要:**在面向服务的环境下,为快速、高效地进行自动 Web 服务组合,定义了范例并集的概念,明确了当多个范例合并时才可满足服务请求的情况,简化了范例检索过程,使得在整个组合过程中只需要检索一次范例库。然后提出了一种基于范例的服务组合方法,对检索到的范例按相似度降序排列,在排名靠前的范例中选择范例实施调整,其中若有单个范例满足请求则删除该范例中的冗余服务,否则进行范例合并,最后复用调整后的范例实现服务组合。仿真实例说明了本方法的可行性和有效性。

**关键词:**范例;范例调整;Web 服务组合

**中图分类号:**TP311

**文献标识码:**A

**文章编号:**1673-629X(2010)10-0103-04

## A Case - Based Automatic Composition Method of Semantic Web Service

YUAN Dong-wei, LI Shu-yu

(College of Computer Science, Shaanxi Normal University, Xi'an 710062, China)

**Abstract:** Under the service-oriented environment, in order to carry out automatic Web service composition fast and efficiently, defined the concept of case union set, clarified the situation of satisfying request when several case were united, and made us retrieving case base only once in the entire process of composition. Then a method of case-based service composition is provided in this thesis, which lines the returned cases according to the degree of similarity in descending order, select one to carry out case adjustment with regard to those ranking-front by deleting redundant services of single case or combining them, and finally reuses the adjusted case to support Web services composition. Simulation instances have showed the feasibility and effectiveness of the proposed method.

**Key words:** case; case adjustment; Web service composition

## 0 引言

随着面向服务计算技术的广泛应用,如何高效、快速地进行自动 Web 服务组合,从而为消费者提供增值的服务是该领域中的热点问题。

面对现有的已经成功组合的大量组合服务实例,范例<sup>[1]</sup>技术被应用到服务组合中来<sup>[2~7]</sup>,它通过将以往成功的服务组合实例存储成范例,当有新的服务请求时检索范例库,对检索到的范例进行适当的调整后复用。Limthanmaphon<sup>[2]</sup>和 Lajmi<sup>[3]</sup>分别基于关键字和活动来评价请求服务与范例的相似度。成睿星<sup>[6]</sup>等对范例进行了层层抽象,从抽象到具体逐层调整。Liu<sup>[5]</sup>使用聚类存储范例,使范例的检索效率得到极大提高。

以上方法有的使用范例时没做调整,而做范例调整的都存在反复检索范例或服务现象。

为解决上述问题,文中提出一种基于范例的自动 Web 服务组合方法,通过定义范例并集概念,明确了当多个范例合并时才可满足服务请求的情况,使得在范例检索中可以一次性地检索出能够被复用的范例和范例并集,并通过选择相似度高的范例或范例并集实施范例调整,提高了范例复用的效率。

## 1 范例的表示

### 1.1 服务的表示

$S(\text{Input}, \text{Output})$  (1)

其中,  $\text{Input} = \{I_1, I_2, \dots, I_n\}$ ,  $\text{Output} = \{O_1, O_2, \dots, O_n\}$ ,  $S$  是服务名称,  $\text{Input}(\text{Output})$  是输入(输出)参数本体概念集,  $I_i(O_i)$  指具体的第  $i$  个输入(输出)参数。

同样服务请求  $R$  表示为:

收稿日期:2010-02-02;修回日期:2010-05-19

基金项目:国家自然科学基金(60671063)

作者简介:袁东维(1976-),女,陕西礼泉人,硕士研究生,研究领域为语义 Web 服务;李蜀瑜,副教授,主要研究领域为嵌入式系统、Web 服务。

$$R(\text{Input}, \text{Output}) \quad (2)$$

## 1.2 范例的表示

$$C(\text{Input}, \text{Output}, \text{graph}(C)) \quad (3)$$

其中,  $\text{graph}(C) = \{\text{WS}\{\text{start}, S_1, S_2, \dots, S_n, \text{end}\} L\{l_{ij}\}$ ,  $C$  是范例的名称, 有向无环图  $\text{graph}(C)$  是对范例具体的执行步骤的说明, 其中  $\text{WS}$  是范例  $C$  中包含服务的集合,  $\text{start}$  和  $\text{end}$  是两个虚拟服务, 规定  $\text{start}.\text{Output} = C.\text{Input}$ ,  $\text{end}.\text{Input} = C.\text{Output}$ ,  $L$  指服务关联(服务关联概念见第 2 部分)的集合。

## 2 基本概念

文中所用到的本体概念<sup>[8,9]</sup>  $A_i$  和  $B_j$  的相似度  $\text{similarity}(A_i, B_j)$ , 本体概念集合  $CA$  和  $CB$  的相似度  $\text{similarity}(CA, CB)$  均采用文献[5]的方法。

$$\text{similarity}(A_i, B_j) = \frac{\alpha}{d + \alpha} \quad (4)$$

$$\text{similarity}(CA, CB) =$$

$$\frac{1}{n} \left[ \sum_{j=1}^n \sum_{i=1}^m \max(\text{similarity}(A_i, B_j)) \right] \quad (5)$$

其中  $\alpha$  是调整因子,  $d$  是本体概念  $A_i$  和  $B_j$  在本体概念树中的最短距离, 本体概念集合  $CA$  和  $CB$  中各包含  $m$  和  $n$  个本体概念。

范例相似度的计算如下:

$$\text{Sim}(S, S') = \frac{1}{n} (\text{similarity}(\text{Input}, \text{Input}') + \text{similarity}(\text{Output}, \text{Output}')) \quad (6)$$

为精确表示范例和进行范例检索, 文中定义以下相关概念:

**概念 1** 本体概念包含关系: 在领域本体中, 如果本体概念  $A_i$  被定义为本体概念  $B_j$  的子概念<sup>[10,11]</sup> 或  $A_i$  和  $B_j$  的相似度  $\text{similarity}(A_i, B_j) > \epsilon$ ,  $\epsilon$  是接近于 1 的阈值, 则认为本体概念  $B_j$  包含本体概念  $A_i$ , 记为  $B_j \succ A_i$ 。

**概念 2** 本体概念集合包含关系: 对于本体概念<sup>[12]</sup> 集合  $CA$  中的每个本体概念  $A_i$ , 总能在本体概念集合  $CB$  或  $\bigcup CB_i$  ( $\bigcup CB_i$  是包含  $CA$  的最小概念集合) 中找到一个本体概念  $B_j$ , 使得  $B_j \succ A_i$ , 则称  $CB$  或  $\bigcup CB_i$  包含  $CA$ , 记作  $CB \supseteq CA$  或  $\bigcup CB_i \supseteq CA$ 。

**概念 3** 服务关联: 服务集合  $\text{WS}$  中若存在服务  $S_i$  的某个输出参数  $O_a$  和服务  $S_j$  的某个输入参数  $I_b$ , 使得  $I_b \succ O_a$ , 则服务  $S_i$  的输出参数  $O_a$  满足服务  $S_j$  的输入参数  $I_b$ , 记为  $s_i^{O_a} \rightarrow I_{s_j}$ , 则称服务  $S_i$  和  $S_j$  关联, 为方便书写, 可记为  $l_{ij}$ 。

**概念 4** 范例并集: 当单个范例  $C$  不能满足  $R$  的要求时, 若存在  $R.\text{Input} \supseteq \bigcup C_i.\text{Input}$  并且  $\bigcup C_i.$

$\text{Output} \supseteq R.\text{Output}$ , 则称范例并集  $\bigcup C_i$  满足  $R$ , 并称  $\bigcup C_i$  中和  $R$  相似度最大的范例为范例并集中的主范例。

范例附加代价的计算: 对主范例  $C_j$  不能满足的  $R$  的输出本体概念  $O_k$ , 计算由  $C_j$  求得  $O_k$  所需付出的附加代价。为和后面的组合算法保持一致, 对  $C_j$  的  $\text{end}$  点赋值, 使  $C_j.\text{end}.\text{Input} = R.\text{Output}$ , 即  $\text{end}$  的  $I_k$  不能被满足, 在  $C_j$  所属的范例并集中计算由  $C_j$  求得  $I_k$  所需的附加代价  $h(I_k)$ , 其计算方法借鉴  $A^*$  算法的思想:

$$h(I_k) = \begin{cases} 0, & \text{if } \exists s_i \in C_j \text{ 或 } \exists s_i \in h(I_k), \text{ 有} \\ & s_i^{O_c} \rightarrow I_{s_j}, \text{ 且 } C_j \text{ 中不存在 } r_{ji} \\ 1 + \min h(\text{pre}(s_i)), & s_i^{O_c} \rightarrow I_{s_j}, s_i \text{ 包含在其它} \\ & \text{范例中} \end{cases} \quad (7)$$

$h(I_k)$  是范例  $C_j$  中已经计算出来其它未被满足的本体概念  $I_k$  的附加代价。

范例  $C_j$  的附加代价是所有  $C_j$  不能满足的  $\text{end}.$   $I_k$  的附加代价之和, 为:

$$h(\delta) = \sum h(I_k) \quad (8)$$

## 3 基于范例的服务组合

### 3.1 范例的检索过程

文中采用聚类算法存储范例<sup>[5]</sup>, 范例的检索过程描述如下:

a) 对于服务请求  $R$ , 首先和存储目录中的索引范例(簇首)进行范例相似度计算。

b) 在相似度最高的索引所指向的那一类范例中, 计算范例  $C$  和  $R$  的输入概念集相似度, 如果  $R.\text{Input} \supseteq C.\text{Input}$  或  $R.\text{Input} \supseteq \bigcup C_i.\text{Input}$ , 则计算范例相似度, 否则检索下一个范例。

c) 在已经被计算出相似度的范例中, 如果存在  $C.\text{Output} \supseteq R.\text{Output}$  或者  $\bigcup C_i.\text{Output} \supseteq R.\text{Output}$ , 则将  $C$  或  $\bigcup C_i$  返回服务组合算法, 否则返回失败。

### 3.2 基于范例的服务组合算法

根据相似度降序排列检索返回的范例或范例并集, 对排名靠前的  $k$  个范例或范例并集做如下调整: 如果有单个范例满足  $R$ , 则该范例中可能存在冗余服务, 调用范例剪枝算法  $\text{Prune}(R, C)$  进行处理, 复用调整后的范例  $C'$ ; 若只有范例并集, 因其中任何一个范例都不能独立地满足  $R$ , 则对相似度最高的范例并集调用范例合并算法  $\text{Union}(R, \bigcup C_i)$ 。具体算法描述如下:

算法 1: CaseWSC

输入: 服务请求  $R$

输出:满足服务请求  $R$  需求的一个服务组合规划

1 检索范例库,按相似度降序排列返回的范例或范例并集;对排名靠前的  $k$  个范例或范例并集做如下调整:

2 if 在  $k$  个范例中存在  $C_i.Output \supseteq R.Output$

3 Prune( $R, C$ );

4 else if 在  $k$  个范例中存在  $\bigcup C_i.Output \supseteq R.$

Output

5 Union( $R, \bigcup C_i$ );

6 else

7 return fail;

8 }

在范例剪枝算法 Prune( $R, C$ ) 中,先删除冗余的服务关联,再删除冗余服务,返回调整后的范例  $C'$ 。算法如下所示:

算法 2: Prune( $R, C$ )

输入: 服务请求  $R$ , 和  $R$  最相似的范例

输出: 满足服务请求  $R$  需求的一个服务组合规划

$C'$

1  $C' \leftarrow C$ ;

2  $C'.start.Output = R.Input$ ;

3  $C'.end.Input = R.Output$ ;

4 从服务 end 开始向前查找

5 {if  $C'$  的服务关联集合  $L$  中存在  $l_{ij}$ ,但相应的服务之间不存在  $s_i^{O_i} \rightarrow l_{sj}$

6 {则删除  $l_{ij}$ ;

5 if 存在服务  $S_h$ ,在  $L$  中不存在形如  $l_{hj}$  的服务关联

6 { 删除服务  $S_h$ ;

7 在  $L$  中删除形如  $l_{kh}$  的服务关联;}

8 else if WS 中的服务  $S_k$  和  $S_j$  存在  $s_k^{O_k} \rightarrow l_{sj}$ ,在  $L$  中不存在形如  $l_{kj}$  的服务关联

9 在  $L$  中添加  $l_{kj}$  }

10 return  $C'$

范例合并算法 Union( $R, \bigcup C_i$ ) 首先确定主范例,然后对所有主范例不能满足的 end.  $I_k$ ,由公式(7)和(8)计算单个  $I_k$  的附加代价  $h(I_k)$  和范例的附加代价,返回  $h(\delta)$  最小的子图,将其合并到主范例中来。

算法 3: Union( $R, \bigcup C_i$ )

输入: 服务请求  $R$ , 和  $R$  最相似的范例并集  $\bigcup C_i$

输出: 满足服务请求  $R$  需求的一个服务组合规划

$\overline{C_j}$

1 在  $\bigcup C_i$  中和  $R$  相似度最大的范例  $C_j$  为主范例;

2  $\overline{C_j} = \text{Prune}(R, C_j)$ ;

3 while 在  $\overline{C_j}$ .end. Input 中存在  $I_k$ ,但 WS 中的服务不存在  $s_k^{O_k} \rightarrow l_{sj}$  end

4 将  $I_k$  存入本体概念集合 NewOutput 中;

5 计算 NewOutput 中每个  $I_k$  的附加代价  $h(I_k)$  及范例附加代价  $h(\delta)$ ,返回  $h(\delta)$  最小的子图  $C_{add}$ ;

6 对  $\overline{C_j}$ . WS 中的服务从 end 开始往前查找直到 start,若服务  $S_j$  有不能满足的本体概念  $I_k$

7 {if 在  $\overline{C_j}$ . WS 中不存在服务  $S_i$ ,使得  $s_k^{O_k} \rightarrow l_{sj}$ ;

8 {在  $C_{add}$  中查找服务  $S_h$ ,其中  $s_k^{O_k} \rightarrow l_{sj}$ ;

9 将  $S_h$  加入到  $\overline{C_j}$ . WS 中的  $S_j$  服务前;

10 在服务关联集  $\overline{C_j}.L$  中添加  $l_{hj}$ };

11 else if 在  $\overline{C_j}$ . WS 中存在服务  $S_i$ ,使得  $s_i^{O_i} \rightarrow l_{sj}$ ,但  $\overline{C_j}.L$  中不存在  $l_{ij}$ ;

12 在  $\overline{C_j}.L$  中添加  $l_{ij}$ };

13 return ( $\overline{C_j}$ );

#### 4 仿真实例

仿真实例如下:

假设服务请求为  $R(\text{Input}, \text{Output})$ ,  $\text{Input} = \{a, b, c\}$ ,  $\text{Output} = \{e, f, m\}$ 。实验范例库中存储有 300 个范例,检索范例库,满足  $R$  要求的范例按相似度大小排序为  $C_4 \cup C_7$  和  $C_2 \cup C_7$ 。它们中没有单个范例满足  $R$ ,取和  $R$  相似度最大范例并集  $C_4 \cup C_7$  做调整。其中范例  $C_4$  和  $C_7$  如图 1 和图 2 所示(因篇幅所限对  $C_2 \cup C_7$  不做介绍)。

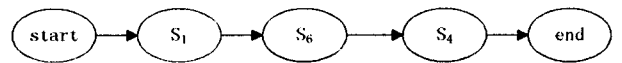


图 1 范例  $C_4$

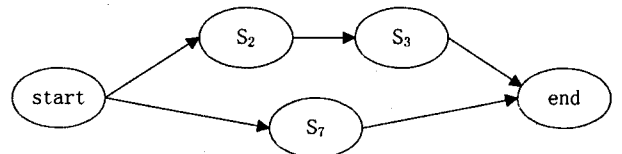


图 2 范例  $C_7$

$C_4 = (\text{Input}_4, \text{Output}_4, \text{graph}(C_4))$ ,  $\text{Input}_4 = \{a, b\}$ ,  $\text{Output}_4 = \{e, f, n\}$ ,  $\text{graph}(C_4) = \{\text{WS}\{\text{start}, S_1, S_6, S_4, \text{end}\} L_4\{l_{sta1}, l_{16}, l_{64}, l_{4end}\}\}$ .

$C_7 = (\text{Input}_7, \text{Output}_7, \text{graph}(C_7))$ ,  $\text{Input}_7 = \{a, c\}$ ,  $\text{Output}_7 = \{d, g, k, m\}$ ,  $\text{graph}(C_7) = \{\text{WS}\{\text{start}, S_2, S_3, S_7, \text{end}\} L_7\{l_{sta2}, l_{sta7}, l_{23}, l_{3end}, l_{7end}\}\}$ .

当前范例并集中所用到的服务如表 1 所示。

对范例并集  $C_4 \cup C_7$  调用范例合并算法。又因  $C_4$  和  $R$  的相似度较高,将其作为主范例,对  $C_4$  调用剪枝

算法,用  $R$  的 Output 替换掉  $Output_4$  后,服务关联  $l_{4end}$  不成立,将其删除,再删除冗余服务  $S_4$  及其相关的服务关联  $l_{64}$ ,因  $S_6$  和 end 存在关联关系,最后在  $L_4$  中添加  $l_{6end}$ (执行结果如图 3 所示)。对主范例不能满足的个体概念  $m$ ,将它存入 NewOutput 中,计算  $\overline{C}_4$  满足  $m$  的附加代价  $h(m) = 1$ ,因为 NewOutput 只有这一个个体概念,所以范例  $\overline{C}_4$  的附加代价  $h(\delta) = h(m)$ ,  $C_{add}$  中只包含一个服务  $S_7$ ,将它添加到主范例中,复用调整好的范例  $\overline{C}_4$ (见图 4)。

表 1 当前范例并集中所用到的服务列表

Services	Input	Output
$S_1$	a, b	e
$S_2$	a, c	d
$S_3$	d	g
$S_4$	a, f	n
$S_5$	a, e	h
$S_6$	e	f
$S_7$	a, c	m, k

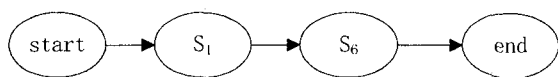


图 3 执行剪枝算法后的  $C'_4$

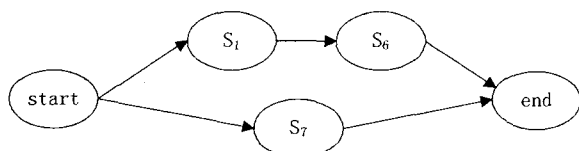


图 4 执行范例合并后的  $\overline{C}_4$

## 5 结束语

文中提出一个基于范例的自动 Web 服务组合方法,通过定义范例并集,明确界定了当多个范例合并才可满足服务请求  $R$  的情形,简化了范例检索,整个组合过程只需对范例库进行一次检索。今后将研究在组合算法加入 QoS 指标来衡量范例,以使组合结果更加

符合用户要求。

## 参考文献:

- [1] 高俊杰,邓贵仕.基于本体的范例推理系统研究综述[J].计算机应用研究,2009,26(2):406-409.
- [2] Limthanmaphon B, Zhang Y. Web Service Composition with Case - Based Reasoning[C]//Proceedings of the Fourteenth Australasian Database Conference. Adelaide: [s. n.], 2003: 201-208.
- [3] Lajmi S, Ghedira C, Ghedira K. WeSCo- CBR: How to Compose Web Services Via Case - Based Reasoning[C]//Proceeding of the IEEE International Conference on e - Business Engineering (ICEBE 2006). [s. l.]: [s. n.], 2006: 618-622.
- [4] Thakker D, Osman T, Al - Dabass D. Semantic - Driven Matchmaking and Composition of Web Services Using Case - Based Reasoning[C]//Proceedings of the Fifth European Conference on Web Services. Washington: [s. n.], 2007: 67-76.
- [5] Liu Yong zhuang, Qiu Shuang, Tao Hai jun. A Case - Based Reasoning Approach to Support Web Services Composition [C]//Proceedings of the Eighth International Conference on Machine Learning and Cybernetics. Baoding: [s. n.], 2009: 1471-1476.
- [6] 成睿星,杨放春,苏 森.基于多重服务范例适应性调整的服务组合[J].软件学报,2008,19(11):3011-3022.
- [7] 胡海涛,李 刚,韩燕波.一种面向业务用户的大粒度服务组合办法[J].计算机学报,2005,28(4):694-703.
- [8] 李选如,何洁月.语义集成:本体映射方法研究[J].计算机技术与发展,2007,17(2):121-124.
- [9] 于 琦,周 勇.一种基于本体的异构数据源模式集成[J].计算机技术与发展,2008,18(2):34-37.
- [10] 李永超,罗钧旻.语义 Web 中的本体推理研究[J].计算机技术与发展,2007,17(1):101-103.
- [11] 郭学俊,杜 春,秦雪杰.基于本体的 Web 服务发现模型研究[J].计算机技术与发展,2006,16(6):207-209.
- [12] 鲍 文,李冠宇.本体存储技术研究[J].计算机技术与发展,2008,18(1):146-150.

# 全国嵌入式系统学术年会(2010 ES CHINA·北京)

第九届全国嵌入式系统学术年会(2010 ES China)定于 2010 年 11 月 5 日至 7 日在北京召开。会议将邀请两院院士、国内、外著名专家和知名厂商就嵌入式系统设计、软硬件开发创新及应用作主题报告及应用技术交流。

主办单位:北京大学、中国计算机学会嵌入式系统专业委员会。

征文要求:论文用 Word 排版。以电子邮件方式投稿到 wwzhang@pku.edu.cn。电子邮件主题为“年会投稿”。

联系方式:张 维 北京大学信息科学技术学院(100871) 电话:010-62763331 Email: wwzhang@pku.edu.cn