文章编号:1001-506X(2010)02-0415-04

基于着色 Petri 网的最小代价服务合成方法

李小艳[1], 张晓松[2], 方 敏[1]
(1. 西安电子科技大学计算机学院, 陕西 西安 710071;
2. 西安公路研究院交通工程研究所, 陕西 西安 710054)

摘 要: 如何动态地把现存的各种 Web 服务整合起来以形成新的、满足不同用户需求的服务已成为新的应用需求和研究热点.针对现有服务合成中服务选择技术的不足,提出了一种基于着色 Petri 网的最小代价服务合成方法。该方法主要思想是根据候选服务输入输出之间的逻辑关系建立有色 Petri 网,从而找出所有合成方案,然后根据原子服务的服务质量(quality of service, QOS)参数选择具有最小代价的合成方案,通过服务实例验证,基于着色 Petri 网服务合成方法能够获得最小代价的组合方式。

关键词: Web 服务; 服务合成; 有色 Petri 网; 服务质量
中图分类号: TP 393 **文献标志码**: A

Optimal Web service composition algorithm based on colored Petri nets

LI Xiao-yan[1], ZHANG Xiao-song[2], FANG Min[1]
(*1. School of Computer Science & Technology, Xidian Univ., Xi'an 710071, China;*
2. Transportation Engineering Inst., Xi'an Highway Scientific Academy, Xi'an 710054, China)

Abstract: Recently, people intend to integrate the existent Web services to form a new service that can meet the needs of different users. How to do it dynamically is a hot research area. An optimal web service composition algorithm based on colored petri-nets is proposed. The essence of the algorithm is that of ours, on the basis of the input and output logical relation of candidate Web services, the colored petri-nets is constructed in order to find all the integration methods. From those methods, the optimal one is selected by using the QoS attributes of the Proto Web service. The effectiveness of the algorithm based on colored petri nets is verified by an example.

Keywords: Web service; service composition; CP-nets; quality of service

0 引 言

Web 服务合成(Web services composition, Web-SC)就是将多个现存的 Web 服务根据需要进行组合,从而提供新的、功能更强的 Web Service。合成后的新服务被称为复合服务(composite service, CS),用于合成复合服务的子服务称之为原子服务(proto service, PS)。Web 服务合成大致可以分为两种类型:静态合成和动态合成。在设计阶段就定义了复合服务规范的合成方法是静态合成,相对地,如果在运行时刻所需服务才被选择和调用的服务合成方法属于动态合成。那么在进行 Web 服务合成时应选取哪些服务呢,一个重要因素就是考虑每一个服务的 QoS 是否满足用户的需求等。

合成 Web 服务的开发可分成几个阶段[3]:

(1) 发现可用的原子服务;

(2) 选择服务合成的方案;

(3) 使用特定的服务合成语言描述其合成;

(4) 将描述文档发布到特定执行引擎供服务使用者调用执行。

文献[2]和文献[4]基于深度优先搜索的 Web 服务合成算法来实现服务合成,但是这种合成方法只找出一种合成方案,并且不能根据 QoS 找出所有合成方案中最优方案。本文旨在研究 Web 服务合成方案的选择,利用有色 Petri 网(colored Petri nets, CP-nets)[5-6]建立服务合成的模

收稿日期:2008-08-21; **修回日期**:2009-03-27。
基金项目:陕西省科学技术研究发展计划项目(2008K04-14);西安市科技创新计划(YF07012)资助课题
作者简介:李小艳(1982-),女,硕士研究生,主要研究方向为计算机网络和 Web 服务等。E-mail:lixiaoyan417@126.com

型找出所有合成方案，并根据服务质量 QoS 属性选出最优合成方案。这种合成方法能够清晰地表达各个子服务之间的逻辑关系，并从所有合成方案中选择出最优的方案。

1 基本概念和符号

1.1 Web 服务的 QoS 属性

Web 服务合成时可能存在多种组合方案，最后应该取哪个组合方案呢，一个重要因素就是考虑每一种组合方案的服务质量 QoS，从中选择出服务质量最好的组合方案。

在服务语义描述中[7]，给出 5 种通用的 QoS 度量的概念，这些度量有响应时间、声誉、成功率、可靠性和价格。

(1) 响应时间：服务请求发出和该服务执行完毕之间的时间延迟。

(2) 声誉：表示用户对服务的信用度评价。该标准主要依赖用户对服务的评价。

(3) 服务代价：服务请求者调用服务所需支付的代价。

(4) 成功率：在某段时间内服务可用的概率。

(5) 可靠性：请求在期望时间内被响应的概率。

Web 服务的 QoS 属性由以上 5 个方面来定性评价，根据质量模型[7]有

$$Q(P)=(Q_{du}(p),Q_{rep}(p),Q_{price}(p),Q_{suc}(p),Q_{re}(p))$$

1.2 有色 Petri 网的简单介绍

有色网 CP-nets 是一个八元组 $CPN=(\Sigma, P,T,F,C,G,E,I)$[5]，其中，

Σ 是类型的非空有限集，也称颜色集；

P 是有限库所集；T 是有限变迁集；F 是有限弧集且 $F\subseteq P\times T\cup T\times P$；

C 是颜色函数：$P\rightarrow\Sigma$；

G 为警戒函数

$$T\rightarrow \text{Boolexpression};\ \forall t\in T$$

$$Type(G(t))=Boolean\wedge type(\text{var}(G(t)))\subseteq\Sigma$$

E 为弧函数

$$F\rightarrow \text{Boolexpression};\ \forall f\in F$$

$$Type(E(f))=C(P)_{MS}\wedge type(\text{var}(E(f)))\subseteq\Sigma$$

式中 p 为 f 所连接的库所。

I 为初始化函数，满足

$$\forall p\in P: Type(I(p))=C(p)_{ms}$$

模型中还需要定义如下函数：

B：变迁的 body 函数。它的作用是，对于那些能够通过变迁 t 的托肯，执行该函数，对输出托肯的出现形式进行相应的处理。

2 有色 Petri 网的最小代价合成算法

算法的基本思想如下：假定单个原子服务不能满足用户提出的 Web 服务要求，那么就需要利用现有的原子服务进行合成来满足要求，即希望合成后的 Web 服务能够根据特定输入得到特定输出。算法根据用户当前能够访问的候选 Web 服务建立可访问的 Web 服务集合 S，然后根据合成服务的输入输出要求以及可选的 Web 服务之间的输入输出关系建立 CP-nets，找出所有合成方案，然后利用每个服务的代价，从所有合成方案中找出最小代价的方案。

2.1 Web 服务的代价

在 Web 环境中可能有多个 Web 服务都提供相同的功能，但是其非功能属性(如响应时间、声誉、成功率、可靠性和价格)可能不相同。上节介绍的 Web 服务 QoS 可衡量标准包括 5 个方面，本文根据 Web 服务的代价选择最优方案。

Web 服务的代价 $Q_{price}(S)$ 是指服务请求者调用该 Web 服务所应当支付的费用，即 QoS 中所说的价格。一个合成的 Web 服务如果需要调用 n 个 Web 服务：$S_1,\cdots,S_n$，那么它总的代价

是：$Q_{price}(S)=Q_{price}(S_1)+Q_{price}(S_2)+\cdots+Q_{price}(S_n)$

2.2 Web 服务的有色网模型

将单个的 Web 服务表示为：$S(I,O,P)$，其中 S 表示原子服务，I 表示输入参数的集合，O 表示输出参数的集合，P 表示调用该服务的代价。

Web 服务 S 是一个 CP-nets $S=(\Sigma,P,T,F,C,G,E,I,In,Out)$，其中

库所集 $P=\{In,Out\}$；变迁集 $T=\{\text{service}\}$；

In 和 Out 分别是服务 S 的输入、输出集，即变迁 service 的输入和输出库所，$In,Out\in P$ 且 $\neg\exists t\in T:\langle t,In\rangle\in F$，即库所 $\neg\exists t\in T:\langle Out,t\rangle\in F$；

In 在 S 中没有前集元素，库所 Out 没有后继元素；

Guard 函数依据具体服务的语义来确定，它决定服务调用能否执行，即变迁能否发生。

假设某一原子服务为：$S(A^iB^i,D^o,P)$ 则它的有色网模型为

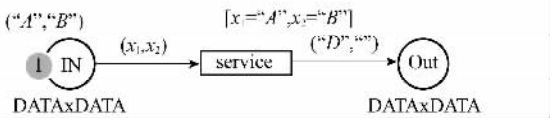

图 1 原子服务的 CP-nets 模型

服务合成是由原子服务，按一定规则组合而成，其 CPN 模型是由多个原子服务的 CP-nets 模型通过控制变迁、库所按流关系组合而成。现对合成服务中的变迁进行分类，用于表示被调用服务的变迁称为服务变迁，如图 1 中的 service。用于把服务组织成合成服务的变迁称为控制变迁。在图 2 中名称含有字母 t 的变迁表示控制变迁，含有字母 s 的变迁表示服务变迁。

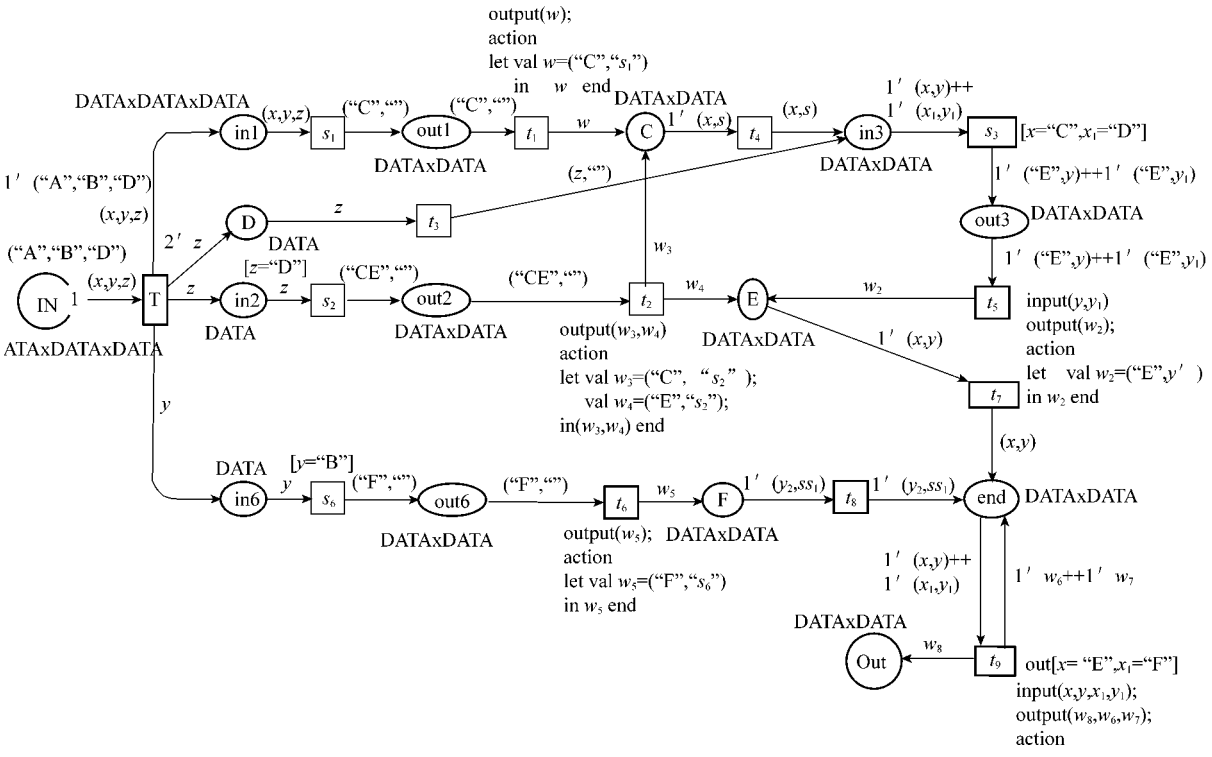

图 2　合成方案 CP-nets

2.3　合成服务的构造算法

最小代价合成服务的算法分两步。

第一步　构造合成方案的 CP-nets,找出所有合成方案。

第二步　利用原子服务的代价,计算合成方案的代价,从而选出最小的方案。

CP-nets 模型的构造与执行:

(1) 根据用户当前能够访问的候选 Web 服务建立可访问的 Web 服务集合 S ,S 中的元素表示为 $S_i:In_i \rightarrow Out_i$,其中 In_i 表示服务 S_i 的输入参数集合;Out_i 表示服务 S_i 的输出参数集合。

(2) 对 S 中的每个元素 S_i,都可以生成如图 1 中的 CP-nets模型,并严格定义输入输出库所中托肯的出现形式。需要合成的服务表示为:WS(In,Out)其中 In 表示服务 WS 的输入参数集合;Out 表示服务 WS 的输出参数的集合;CS 为当前满足输入条件的服务集合,初始值为空;CS_{out} 表示 CS 中服务的输出参数集合其初始值为空。

(3) 利用原子服务输入输出关系建立 CP-nets 合成模型。

(4) 首先建立一个起始库所和控制变迁,其标记分别为 IN 和 T。并加入一条由 IN 指向 T 的弧。IN 中的托肯为合成服务的输入参数,控制变迁 T 用来控制输出库所中托肯的出现形式。

(5) $\forall S_i \in (S-CS)$且 $S_i:In_i \rightarrow Out_i$,如果 $\forall x \in In_i$ 使得 $x \in$ IN,则令:

$CS=CS+\{S_i\}$,$CS_{out}=CS_{out} \cup Out_i$ 并且加入一条由控制变迁指向 S_i 的输入库所的有向弧,使得输入库所中托肯的出现形式为 S_i 的输入参数。如果 $\exists x \in In_i$ 使得 $x \in$ IN,则建立新库所,其标记为 x,并加入一条控制变迁指向 x 的有向弧,库所中的托肯的出现形式为(x,‘’)。如果 $CS=\{\}$,则转(8)。

(6) 对于 $\forall y \in CS_{out}$,建立库所,其标记为 y,同时 CS 中每个元素 S_i 的输出库所分别指向新建立的控制变迁 t_i,即 $s_i->t_i$ 的形式。t_i 指向标记为 S_i 输出参数的库所。控制变迁 t_i 中的 body 函数规定该输出库所中托肯的出现形式为(“输出参数”,“调用的服务”)。

(7) 此时 S 中所有满足 $In_i \subset$ IN 的服务被选入 *CP-nets*。且 $S=S-CS$,$CS=\{\}$,$CS_{out}=\{\}$。在当前 *CP-nets*模型中,如果对于任意一个以输出弧为零的库所,其标记$\notin Out$ 且 $S=\varnothing$,则转(8),否则如果 $S \neq \varnothing$,则对于每个输出弧为零的库所分别作为起始变迁 In,并建立控制变迁,重复(4)和(5)。

(8) 在当前 CP-nets 中,对每个输出弧为零的库所建立一个控制变迁,并指向一个新的、标识为 end 的库所。end 库所中托肯的出现形式为:(“某个输出参数”,“路径”)表示以某个输出参数为标识的库所的路径。最后,建立一个控制变迁 T_{out} 和输出库所 Out,end 库所指向 T_{out},T_{out} 指向 Out 和 end。其中 T_{out} 用来计算所有组合方案,Out 库所中托肯

的出现形式为:(“合成服务的输出参数”,“服务集合”)或者空。为空表示组合失败,算法结束。

求解最小代价的方案

假定由 CP-nets 的运算结果可知该服务合成方案分别为:CP_1,CP_2,CP_2,…,CP_n。同时我们假定 CP_1 所对应的原子服务分别为 s_{11},s_{12},…,s_{1r};CP_2 所对应的原子服务分别为 s_{21},s_{22},…,s_{2s};那么寻找最小代价的合成方案可以按两步求解:

(1) 求出 CP_1,CP_2,…,CP_n 各个合成方案的总代价,分别用 C_1,C_2,…,C_n 来表示

$$C_1=Q_{price}(s_{11})+Q_{price}(s_{12})+\cdots+Q_{price}(s_{1r})=\sum_{i=1}^{r}Q_{price}(s_{1i})$$

$$C_2=Q_{price}(s_{21})+Q_{price}(s_{22})+\cdots+Q_{price}(s_{2s})=\sum_{i=1}^{s}Q_{price}(s_{2i})$$

$$C_n=Q_{price}(s_{n1})+Q_{price}(s_{n2})+\cdots+Q_{price}(s_{nt})=\sum_{i=1}^{t}Q_{price}(s_{ni})_i$$

(2) 从 C_1,C_2,…,C_n 中选择最小的 Ci 所对应的合成方案 CP_i 就是所需的合成方案。即 $Ci=\min(C_1,C_2,\cdots,Cn)$ 对应的合成方案 CP_i。

3 实例分析

假定有以下 6 个 Web service,每个 Web service 都接收一定的输入,并产生一定的输出,每个 Web service 具有一定的代价。例如 6 个 Web service 如表 1 所示。

表 1 6 个 Web service

Web services	代价
$S_1(A^i,B^i,C^o)$	5
$S_2(D^i,C^o,E^o)$	15
$S_3(D^i,C^i,E^o)$	7
$S_4(E^i,A^o)$	2
$S_5(B^i,G^i,D^o)$	2
$S_6(B^i,F^o)$	4

其中 $S_1(A^i,B^i,C^o)$ 表示该 Web Service 要求输入两个参数 A 和 B,输出一个 C,它的代价为 5 元/次;而 $S_2(D^i,E^o,C^o)$ 则表示该 Web service 输入一个参数 D,返回 C 和 E 两个输出,它的代价为 15 元/次。利用以上几个 Web Service 可以进行各种查询,有时为了完成某个查询,需要对现有的 Web Service 进行合成,并根据 QoS 值选择一种最优的组合方案。

假设用户所要求的 Web 服务是:输入三个参数 A、B、D 则能够输出 E、F。该服务可以表示为:$S(A^i,B^i,D^i,E^o,F^o)$。表 1 中任何一个服务都不能满足这一要求,但是由这些单个服务组合而成的合成服务可以满足用户要求。

根据第三节中所介绍的算法,利用 CPN 仿真工具构造 CP-nets,执行该 CP-nets 可以得到所有合成方案。可以得到所有合成方案。

由 CP-nets 的运算结果可知该服务合成方案有三种记为:CP_1,CP_2,CP_3;CP_1 所对应的原子服务为 s1,s6,s3;CP_2 所对应的原子服务为 s2,s3,s6;CP_3 所对应的原子服务为 s2,s6;那么寻找最小代价的合成方案可以按两步求解:

(1) 分别求出 CP_1,CP_2,CP_3 各个合成方案的总代价,分别用 C_1,C_2,C_3 来表示:

$C_1=Q_{price}(s1)+Q_{price}(s3)+Q_{price}(s6)=5+7+4=16$

$C_2=Q_{price}(s2)+Q_{price}(s6)=15+4=19$

$C_3=Q_{price}(s2)+Q_{price}(s3)+Q_{price}(s6)=15+7+4=26$

(2) 从 C_1,C_2,C_3 中选择最小的 C_1 所对应的合成方案 CP_1:s1s2s6 就是所需代价最小的合成方案。

4 结束语

本文是从语义的角度理解每个 Web 服务的输入输出,由此来构造 CP-nets 找出所有的合成方案,并根据每个服务的 QoS 属性即服务价格选择一种代价最小的合成方案。由于时间有限没有综合考虑 QoS 所有属性,单从服务价格这一方面来选择最优的合成方案。下一步要做的工作是综合考虑 QoS 的所有属性选择出最优的合成方案并使用特定的服务合成语言,例如 BPEL4WS[10] 描述其合成,并将描述文档发布给 BPEL4WS 执行引擎供服务使用者调用执行。

参考文献:

[1] D Booth, Hugo Haas, Francis McCabe. Web Services Architecture[EB/OL] http://www.w3.org/TR/ws2arch,2004-02.

[2]Hashemian S V, Mavaddat F. A Graph-based approach to web services composition[C]// *Proc. of the Symp on Applications and the Internet*, 2005:183-189.

[3] 顾宁,刘家茂,柴晓路. WebServices 原理与研发实践[M]. 北京:机械工业出版社,2006: 177-214.

[4] 刘峰,谭庆平,杨艳萍. 基于深度优先搜索的 Web 服务合成算法[J]. 计算机工程与科学,2006.

[5] Jensen K. Coloured Petri Nets: basic concepts, analysis methods and practical[J]. *Basic Concepts, Monographs in Theoretical Computer Science*,1997.

[6] Y Kochut K J. A CP-nets-based design and verification framework for web services composition[C]// *Proc. of IEEE International Conference on Web Services*, 2004:756-760.

[7] 高晓燕,余镇危,史银龙. 基于 QoS 的 P2P 网络服务组合的算法[J]. 计算机工程与设计,2007.

[8]高勇,刘瑜,谢昆青,等. 一个基于 Petri 网的 Web 服务组合模型.[J]. 计算机工程,2006;32(6).

[9]CPN Group. CPN Tools[EB/OL]. http://wiki.daimi.au.dk/cpntools/cpntools.wiki.20061108.

[10] Alves A. Web services business process execution language version 2.0. In http://www.oasisopen.org/committees/documents.php, 2006.