异步交互的Web服务的相容性分析

刘方方 余 洁 李成玲
(上海大学计算机工程与科学学院 上海 200072)

摘 要 Web服务合成的相容性是服务合成研究领域的重要问题。相容性分析需要考虑异步交互的服务合成环境。用形式化的分析方法为Web服务的合成建模,给出服务合成满足相容性要求的限制条件,并提供了相容性的判断方法。

关键词 Web服务,服务合成,相容性
中图法分类号 TP301 **文献标识码** A

On Compatibility of Asynchronously Communicating Web Services

LIU Fang-fang YU Jie LI Cheng-ling
(School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China)

Abstract Compatibility of Web services is an important issue in the research field of Web services. In Web environment, Web services from different organizations communicate asynchronously. In this paper, based on formal description of Web services and services composition, we analyzed the constraints of compatibility of services and developed the method to check compatibility of component services.

Keywords Web service, Services composition, Compatibility

Web服务是重要的分布式网络应用集成技术。Web服务合成是Web服务的基本需求。已提出描述Web服务合成流程的几种语言规范,如BPEL4WS[1]和WSCI[2],但它们大多数仅关注规范Web服务消息流程的描述,缺乏验证合成流程是否正确的分析机制。为保证Web服务合成的正确性,需要用形式化的分析方法分析Web服务之间的交互流程。

目前,分析合成流程的研究方法主要有两种:一种是通过验证Web服务合成形成的全局交互流程是否与预先定义的合成模型相匹配,从而检测合成流程的正确性,有代表性的是文献[3-5]中的研究工作;另一种是考虑每个组件服务之间的流程是否相容,相容性指的是每个组件服务与其他组件通过消息交互执行各自的消息流程,最终都会达到终止状态,有代表性的是文献[6-9]中的工作。

其中,文献[3-5]中用自顶向下的方法分析异步交互环境下的Web服务合成。用Mealy机为Web服务和异步通信的Web服务合成建模。每个Web服务有一个FIFO队列保存收到的消息,合成服务中交互的消息被一个虚拟的全局观察器保存。参与合成的服务必须满足保证服务正确合成的3个约束条件:无损连接、同步适应性、自治性。对于自顶向下的设计方法,当有新的服务需要加入现有的合成系统时,服务合成需要重新设计,因而在扩展性方面显得不够灵活。

对于Web服务相容性的分析,文献[6]中使用LTS描述基于WSCI的Web服务,考察两个Web服务之间不同情况的相容性,为Web服务的相容性提供三层定义。最严格的相容性定义是两个服务的发送与接收消息的操作流程如果正好相反,即一个服务能够发送的消息与另一个服务能够接收的消息集合相同,那么两个服务是相容的。但是这样的情形很少见,因此作者又将相容性定义的限制放松,认为一个服务发送的消息的集合包含在另一个服务能够接收的消息集合中,那么服务也可以相容。而限制最少的是,若两个服务交互执行过程中至少存在一条执行路径可以使两个服务均达到终止状态,那么两个服务就相容。

文献[7]中,作者利用Petri网作为描述参与合成的组件服务的行为模型,基于这个模型,提出了两个Web服务之间的相容性问题。其相容性分析包括两部分:词法相容性及语义相容性。语义相容性考虑了Web服务合成的行为是否有效,并且提出了限定条件,保证合成时两个交互的Web服务之间语义的相容性,但是这个定理没有考虑多个Web服务的合成情况。文献[8]用FSP为BPEL4WS建模考虑服务相容性,同样将相容性分为语法相容和语义相容两类。这里的语义相容指的是服务的流程相容。文献[9]中认为将Web服务用CCS进程表示,合成后的进程在执行过程中如果存在一条执行路径使每个组件服务都能终止,那么服务就是相容的。而如果合成服务不满足相容性定义,则协调Web服务合成的适配器可以自动地为服务生成符合相容性要求的规范说明。

以上研究中的相容性都是对同步交互的消息而言的,没有考虑Web服务异步交互的情形,以致使某些在同步情况下符合相容性要求的服务合成,在异步交互的实际运行中并不

到稿日期:2009-12-28 返修日期:2010-03-02 本文受国家自然科学基金项目(60803143)资助。
刘方方(1979-),女,博士,讲师,主要研究方向为分布对象计算、Web服务等,E-mail:ffliu@shu.edu.cn;余 洁(1981-),女,博士,讲师,主要研究方向为体系结构、交互计算等;李成玲(1986-),女,硕士生,主要研究方向为Web服务。

能得到正确的结果。

因此，本文通过自底向上的分析，对文献[9]中的 Web 服务定义进行扩充，将用来存放消息的消息队列与服务流程一起考虑，为异步交互的 Web 服务合成建模，分析合成的相容性，给出判定方法。

1 基本概念

WSCI 从基于协作的角度描述 Web 服务合成，它建立在 WSDL 描述的 Web 服务之上，将服务的操作用各种程序结构联系起来，形成一个 Web 服务的消息流程。进程代数可以描述基于 WSDL 和 WSCI 的 Web 服务与 Web 服务合成，并推导服务的行为特性，进行服务的相容性分析。

文献[9]中使用进程代数的一种 CCS(calculus of communicating system)描述 Web 服务，CCS 提供用来描述 Web 服务消息流程的进程表达式。为此，我们选用 CCS 为 Web 服务建模。

首先介绍 CCS[11]的一些基础概念。

一个 CCS 进程(由 P,Q,R,M 等表示)在 CCS 中定义如下：

$$P::=0\mid\alpha.Q\mid P+Q\mid P\mid\mid Q\mid P\backslash sm$$

$$\alpha::=a?(x)\mid a!(x)\mid\tau$$

式中，α 代表一个原子动作，该动作可以通过通信信道发送或接收消息(消息由 a,b 等表示)，或是执行进程内部的隐藏操作 τ。发送消息用 $a!(x)$(a 表示消息的名称，x 表示消息中的参数)表示，接收消息用 $a?(x)$表示。隐藏操作代表进程内部执行的、外界不可观察到的行为。一个终止的进程记为 0，表示进程处于终止状态；通常，一个 CCS 进程由一系列的形如 $\alpha.P$ 的子进程组成。一个进程可能执行选择操作，操作符为"+"(如 $P+Q$)；或者一个进程由一些并行运行的子进程组成，子进程间可相互通信：$P\mid\mid Q$。一个进程可能会受到一些限制，用 $P\backslash sm$ 表示，sm 为一组信道名称的集合，意思是若进程 P 的一个子进程想要通过属于 sm 的一个信道发送消息 $a!(x)$，只有当存在另一个 P 的子进程可以通过同样属于 sm 的信道接收消息 $a?(x)$时，该操作才能成功执行，这样一个进程的两个子进程之间发送和接收消息的动作，在进程外部是观察不到的，所以也称为隐藏操作，用 τ 表示。因为我们所考虑的服务合成都是封闭的，所以在用进程描述 Web 服务时，sm 通常省略。

分析 Web 服务合成的消息交互行为时，本身不涉及消息传递过程中的参数及其数据类型，因而为便于分析模型，在分析合成的流程时省去消息信道中的参数，接收与发送消息的操作分别用 $a?$，$a!$ 表示。

由 CCS 进程的定义，可以看到 CCS 能够描述进程间的相互通信行为。进程间的通信是按照一定的推导规则进行的，下面给出一些与 Web 服务合成相关的推导规则。

规则 1 $\dfrac{}{\alpha.P\xrightarrow{\alpha}P}$；规则 2 $\dfrac{P\xrightarrow{\alpha}P'}{P+Q\xrightarrow{\alpha}P'}$；

规则 3 $\dfrac{Q\xrightarrow{\alpha}Q'}{P+Q\xrightarrow{\alpha}Q'}$；规则 4 $\dfrac{P\xrightarrow{\alpha}P'}{P\mid\mid Q\xrightarrow{\alpha}P'}$；

规则 5 $\dfrac{Q\xrightarrow{\alpha}Q'}{P\mid\mid Q\xrightarrow{\alpha}Q'}$；规则 6 $\dfrac{P\xrightarrow{a!}P',Q\xrightarrow{a?}Q'}{P\mid\mid Q\xrightarrow{\tau}P'\mid\mid Q'}$；

规则 7 $\dfrac{P\xrightarrow{a?}P',Q\xrightarrow{a!}Q'}{P\mid\mid Q\xrightarrow{\tau}P'\mid\mid Q'}$。

规则 1 表明一个进程 $\alpha.P$ 通过执行 α 所代表的操作，可以转化为进程 P；规则 2 表示含有选择结构的进程可沿着某一选择分支进行转化，并且进程的一次执行只会沿一个分支进行，分支的选择取决于 α 的具体内容；规则 3 与规则 2 是对称的；规则 4 表示参与并行执行的进程可以独立执行自己的操作 α；规则 6 表示并行执行的进程间的相互通信，某个并行的子进程发送消息，另一个子进程接收消息，则并行进程执行隐藏动作 τ 进行转化；规则 5 和规则 7 是与表示规则 4 和规则 6 对称的推导规则。

2 异步交互的 Web 服务合成模型

由于 Web 服务异步交互的特性，为正确处理消息，每个服务需要为接收到的消息设置缓冲区。这样，我们有图 1 所示的 Web 服务交互模型。服务之间通过单向信道发送和接收消息，消息缓冲区由队列实现，每一个消息队列有唯一的发送者和接收者。如服务 WS_1 向服务 WS_2 发送消息的信道为 q_{12}，服务 WS_2 向服务 WS_3 发送消息的信道为 q_{23}，选择消息队列的执行语义为先进先出(first-in first-out，FIFO)。

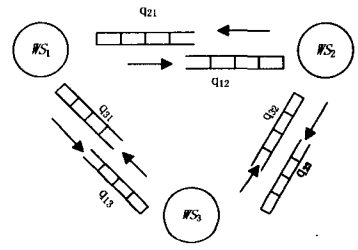

图 1 异步交互 Web 服务合成模型

在定义 Web 服务之前，先给出一些符号说明。

进程 P 经过一系列的转换后可到达 P'，即 $P=P_1\xrightarrow{\alpha_1}\cdots\xrightarrow{\alpha_{n-1}}P_n=P'$，记为 $P\rightarrow P'$。

$P\xrightarrow{a!}$或 $P\xrightarrow{a?}$分别表示进程 P 可发送或接受消息 a，或者说 P 对消息 a 有定义。

$Snd=\{a\mid\exists P_i,P\rightarrow P_i \text{ and } P_i\xrightarrow{a!}\}$表示进程 P 可发送的所有消息的集合；$Rcv=\{a\mid\exists P_i,P\rightarrow P_i \text{ and } P_i\xrightarrow{a?}\}$表示进程 P 可接收的所有消息的集合。

一个消息队列中的内容由其存储的消息序列组成。对于消息 a 和 b，$a\cdot b$ 表示 a 和 b 的连接。如果消息队列为空，则消息队列中的内容记为 ε。

参与合成的 Web 服务称为组件 Web 服务，其定义为：

定义 1(Web 服务) 一个 Web 服务是一个三元组 $WS=\langle P,Q_{in},Q_{out}\rangle$，其中

(i)P 为由 CCS 过程描述的 Web 服务消息执行流程；

(ii)Q_{in} 表示服务接收消息的队列集合；

(iii)Q_{out} 表示服务发送消息的队列集合。

Web 服务流程的执行过程通过 Web 服务的构造(Configuration)和构造间的转换来描述。Web 服务的构造由服务当前所处状态和服务消息队列中的内容组成。

定义 2(Web 服务的构造) 一个 Web 服务 WS 的构造记为 $C^k=\langle P^k,S^k_{in},S^k_{out}\rangle$，其中：

(i) 进程 P^k 由 P 转换而来，$P \rightarrow P^k$；

(ii) $S_{in}^k \subseteq Rcv^* \cup \varepsilon$，$\forall q \in Q_{in}$ 存在唯一对应的 $s \in S_{in}^k$，s 为 q 中的内容；

(iii) $S_{out}^k \subseteq Snd^* \cup \varepsilon$，$\forall q \in Q_{out}$ 存在唯一对应的 $s \in S_{out}^k$，s 为 q 中的内容。

一个服务的初始构造记为 $C^0 = \langle P^0, S_{in}^0, S_{out}^0 \rangle$，其中 $P^0 = P$，通信信道中的内容都为空，即 $\forall s \in S_{in}^0 \cup S_{out}^0$，$s = \varepsilon$。

定义 3(构造间的转换) 一个 Web 服务 WS 的构造 $C^{k+1} = \langle P^{k+1}, S_{in}^{k+1}, S_{out}^{k+1} \rangle$，由构造 C^k 转换而来，当且仅当下列条件之一满足：

(i)如果 $P^k \xrightarrow{a!}$，那么 $P^k \xrightarrow{a!} P^{k+1}$，$S_{in}^{k+1} = S_{in}^k$，$\exists s \in S_{out}^k$ 有 $s' \in S_{out}^{k+1}$ 使得 $s' = s \cdot a$，S_{out}^{k+1} 中其它元素与 S_{out}^k 相同；

(ii)如果 $P^k \xrightarrow{a?}$，那么 $P^k \xrightarrow{a?} P^{k+1}$，$S_{out}^{k+1} = S_{out}^k$，$\exists s \in S_{in}^k$ 有 $s' \in S_{in}^{k+1}$ 使得 $s = a \cdot s'$，S_{in}^{k+1} 中其它元素与 S_{in}^k 相同；

(iii) $P^k \xrightarrow{\tau} P^{k+1}$，$S_{in}^{k+1} = S_{in}^k$，$S_{out}^{k+1} = S_{out}^k$。

定义 4(服务合成) 组件服务 $WS_1, WS_2, \cdots, WS_n$ 组成的 Web 服务合成记为 $C = \{WS_i\}_{1 \leqslant i \leqslant n}$，其中 $WS_i.Q_{in} \cap WS_j.Q_{in} = \Phi, i \neq j$，且 $WS_i.Q_{out} \cap WS_j.Q_{out} = \Phi, i \neq j$。Web 服务组合 C 是封闭的，如果 $\bigcup_{i=1}^n WS_i.Q_{in} = \bigcup_{i=1}^n WS_i.Q_{out}$。

本文中所考虑的 Web 服务合成都是封闭的，即一个组件服务的发送消息队列对应另一个组件服务的接收消息队列，服务发送的消息直接放到接收服务的消息队列中。记 q_{ij} 为服务 WS_i 向服务 WS_j 发送消息的队列，它同时也是服务 WS_j 接收由服务 WS_i 发送来的消息的队列，s_{ij} 为消息队列的内容。合成服务中所有消息的集合记为 $A = \bigcup_{i=1}^n Rcv_i \cup \varepsilon = \bigcup_{i=1}^n Snd_i \cup \varepsilon$。

3 Web 服务合成的相容性

Web 服务合成的运行过程是每个组件服务在其不同的构造间转换的过程。对于 Web 服务合成 $C = \{WS_i\}_{1 \leqslant i \leqslant n}$ 的一次运行过程是一个有限或无限的序列：

$$\rho = \{\rho_k\}_{k \geqslant 0} = \{C_1^k.P^k, C_2^k.P^k, \cdots, C_n^k.P^k, \langle s_{ij}^k \rangle_{i \neq j, i,j \in [1,n]}\}$$

式中，ρ_{k+1} 由 ρ_k 转换而来。服务合成是相容的，即指参与合成的组件服务都能够到达各自的终止状态。用 ter 表示组件服务的终止状态的集合，服务合成的相容性定义如下。

定义 5(合成的相容性) Web 服务合成 $C = \{WS_i\}_{1 \leqslant i \leqslant n}$ 中的组件服务 $WS_1, WS_2, \cdots, WS_n$ 是相容的，如果合成的所有运行序列 ρ 都会终止。处于终止状态的合成的构造为

$$\rho_{end} = \{C_1^e.P^e, C_2^e.P^e, \cdots, C_n^e.P^e, \langle s_{ij}^e \rangle_{i \neq j, i,j \in [1,n]}\}$$

式中，$C_i^e.P^e \in ter, 1 \leqslant i \leqslant n$，且 $s_{ij}^e = \varepsilon, i \neq j, i,j \in [1,n]$。

对于服务合成是否相容的判定，需要考虑通信信道的完整性问题。如果服务合成的通信信道是正确的(perfect)，并且每个组件服务的消息队列是无界的(unbounded)，那么 Web 服务合成的运行就相当于一个图灵机[10]，其相容性问题是不可判定的(undecidable)。为了判断服务合成是否相容，我们考虑采用具有有损信道的 Web 服务合成系统[10,12,13]来分析服务合成的行为特性。在下一节中将介绍具有有损信道的 Web 服务合成系统，由其构造转换所构建的可达树是可终止的，从而使得合成服务相容与否可判定。且具有正确信道的服务的构造序列，是包含在具有有损信道的合成服务的运行序列中的，从而可根据具有有损信道的合成服务是否相容来判断正常的(即具有正确的通信信道的)合成服务是否相容。

"有损信道(non-perfect or lossy channel)"指的是通信信道存在消息丢失的情况。假设一个 Web 服务 WS，其所具有的通信信道是有损的，可以通过对原有的服务中描述服务流程的进程 P 进行扩充，得到具有有损信道的 Web 服务的进程 P_α。方法如下：记 Re $a(P)$ 为进程 P 的可达集，Re $a(P) = \{M | P \rightarrow M\}$。由于一个 Web 服务 WS 接收和发送的消息个数是有限的，进程 P 的可达集 Re $a(P)$ 中的元素个数也是有限的。对可达集中的任意进程 $M \in$ Re $a(P)$，如果 M 对服务 WS 可接收的消息 a，$a \in Rcv$ 没有定义，那么添加 $M \xrightarrow{a?} M$ 至进程 P；如果 M 对消息 a 有定义，$M \xrightarrow{a}$ 保持不变，同样添加 $M \xrightarrow{a?} M$ 至进程 P。此处 M 也可为处于终止状态的进程。依上述过程得到的进程 P_α 称为定义完全的(completely specified)进程。

假定在服务合成系统运行过程中，每一步只有一个组件服务执行转换。描述具有有损信道的 Web 服务合成运行的定义如下。

定义 6(具有有损信道的服务合成) 具有无界消息队列和有损信道的 Web 服务合成 $C = \{WS_i\}_{1 \leqslant i \leqslant n}$，将进程 $WS_i.P$，$1 \leqslant i \leqslant n$ 转换为定义完全的进程 $WS_i.P_\alpha$。这样得到的合成服务 C' 的一次运行过程是一个有限或无限的序列：

$$\rho = \{\rho_k\}_{k \geqslant 0} = \{C_1^k.P_\alpha^k, C_2^k.P_\alpha^k, \cdots, C_n^k.P_\alpha^k, \langle s_{ij}^k \rangle_{i \neq j, i,j \in [1,n]}\}$$

ρ_{k+1} 由 ρ_k 转换而来，当且仅当下列条件之一满足：

(i)如果服务 WS_i 发送消息 a 到服务 WS_j，$c_i^k \cdot p_\alpha^k \xrightarrow{a!} c_i^{k+1} \cdot p_\alpha^{k+1}$，$s_{ij}^{k+1} = s_{ij}^k \cdot a$，除此之外 ρ_{k+1} 与 ρ_k 中的其它元素均相同；

(ii)如果服务 WS_i 接收 WS_j 发送的消息 a，$c_i^k \cdot p^k \xrightarrow{a?} c_i^{k+1} \cdot p^{k+1}$，$s_{ji}^k = a \cdot s_{ji}^{k+1}$，除此之外 ρ_{k+1} 与 ρ_k 中的其它元素均相同；

(iii)如果服务 WS_i 执行隐藏操作，$c_i^k \cdot p_\alpha^k \xrightarrow{\tau} c_i^{k+1} \cdot p_\alpha^{k+1}$，除此之外 ρ_{k+1} 与 ρ_k 中的其它元素均相同；

(iv)如果某个消息队列内容为 $s_{ij}^k = a \cdot x$，其中 $x \in A^*$，有 $s_{ij}^{k+1} = x$，消息 a 丢失，除此之外 ρ_{k+1} 与 ρ_k 中的其它元素均相同。

合成服务的初始构造为 $\rho_0 = \{C_1^0.P^0, C_2^0.P^0, \cdots, C_n^0.P^0, \langle s_{ij}^0 \rangle_{i \neq j, i,j \in [1,n]}\}$，其中 $s_{ij}^0 = \varepsilon, i \neq j, 1 \leqslant i, j \leqslant n$。若 ρ_j 由 ρ_i 经一系列的转换而来，记 $\rho_i \Rightarrow \rho_j, j > i$。

以上是对异步交互的 Web 服务合成的建模。

有损信道的 Web 服务合成的相容性是可以判定的。相容性的判定基于如下描述的一种关系：代表通信信道内容的两个字符串 $s, w \in A^*$，$s \leq w$，如果 s 是 w 的(非连续的)子串，s, w 有如下形式：

$s = u_1 \cdot u_2 \cdot \cdots \cdot u_n$，$w = v_1 \cdot u_1 \cdot v_2 \cdot u_2 \cdot \cdots \cdot v_n \cdot u_n \cdot v_{n+1}$，$u_i \in A, v_i \in A^*$。服务合成的构造 $\rho_k \leq \rho_l$，如果 $C_i^k.P^k = C_i^l.P^l, i \in [1,n]$ 且 $s_{ij}^k \leq s_{ij}^l, i \neq j, i,j \in [1,n]$。

具有有损信道的服务合成有如下性质。

性质 1 Web 服务合成 $C = \{WS_i\}_{1 \leqslant i \leqslant n}$，将进程 $WS_i.P$，$1 \leqslant i \leqslant n$ 转换为定义完全的进程 $WS_i.P_\alpha$，得到服务合成 C'，C' 的运行序列 $\rho = \{\rho_k\}_{k \geqslant 0} = \{C_1^k.P_\alpha^k, C_2^k.P_\alpha^k, \cdots, C_n^k.P_\alpha^k,$

$\langle s_{ij}^k\rangle_{i\neq j,i,j\in[1,n]}\}$中的任意两个构造 ρ_k,ρ_l，如果满足关系 $\rho_k\leq\rho_l$，那么对于任意的消息序列 $\forall x\in A^*$ 使得 $\rho_k\Rightarrow\rho_m$，则 $\exists x'\in A^*$，使得 $\rho_l\Rightarrow\rho_n$，且 $\rho_m\leq\rho_n, x\leq x'$。

证明：我们先看只有一个消息操作的情形。记 $\rho_k=\{C_1^k.P_{cs}^k,\cdots,C_i^k.P_{cs}^k,\cdots,C_n.P_{cs}^i,\cdots,s_{ij}^k,\cdots\}$，因为 $\rho_k\leq\rho_l$，所以对于 $\rho_l=\{C_1^l.P_{cs}^l,\cdots,C_i^l.P_{cs}^l,\cdots,C_n^l.P_{cs}^l,\cdots,s_{ij}^l,\cdots\}$，有 $C_i^k.P_{cs}^k=C_i^l.P_{cs}^l, i\in[1,n], s_{ij}^k\leq s_{ij}^l$。若 ρ_k 有转换操作 $a\in A$，由过程 $C_i^k.P_{cs}^k$ 执行，得到构造 ρ_m。考虑两种情况：

(i)a 为发送消息到进程 j，$\rho_m=\{C_1^k.P_{cs}^k,\cdots,C_i^m.P_{cs}^m,\cdots,C_n^k.P_{cs}^k,\cdots,s_{ij}^k\cdot a,\cdots\}$，那么同样存在构造 ρ_n 由 ρ_l 转换，$\rho_n=\{C_1^l.P_{cs}^l,\cdots,C_i^n.P_{cs}^n,\cdots,C_n^l.P_{cs}^l,\cdots,s_{ij}^l\cdot a,\cdots\}$，根据 $C_i^k.P_{cs}^k=C_i^l.P_{cs}^l, c_i^k\cdot p_{cs}^k\xrightarrow{a!}c_i^m\cdot p_{cs}^m$，可以推出，所以 $\rho_m\leq\rho_n$。

(ii)a 为接收进程 j 发送的消息，记 $s_{ji}^k=a\cdot s_{ji}^m$，$\rho_m=\{C_1^k.P_{cs}^k,\cdots,C_i^m.P_{cs}^m,\cdots,C_n^k.P_{cs}^k,\cdots,s_{ji}^m,\cdots\}$。记 $s_{ji}^l=v\cdot a\cdot s_{ji}^n$，其中 $v\in A^*$，v 中不包含消息 a（因为 $s_{ij}^k\leq s_{ij}^l$，所以此结论成立）。由于进程都是定义完全，那么存在构造 ρ_r 由 ρ_l 经过转换 v 得到，其中进程 $C_i^k.P_{cs}^i$ 消耗序列 v 但不进行任何其他转变，所以 $\rho_r=\{C_1^l.P_{cs}^l,\cdots,C_i^l.P_{cs}^l,\cdots,C_n^l.P_{cs}^l,\cdots,a\cdot s_{ji}^n,\cdots\}$。根据 $C_i^k.P_{cs}^k=C_i^l.P_{cs}^l, c_i^k\cdot p_{cs}^k\xrightarrow{a?}c_i^m\cdot p_{cs}^m$，构造 ρ_r 同样有转换 $c_i^l\cdot p_{cs}^l\xrightarrow{a?}c_i^n\cdot p_{cs}^n$，所以存在 $\rho_n=\{C_1^l.P_{cs}^l,\cdots,C_i^n.P_{cs}^n,\cdots,C_n^l.P_{cs}^l,\cdots,s_{ij}^n,\cdots\}$，$\rho_m\leq\rho_n$，而且 $a\leq v\cdot a$。

操作序列由一组接收或发送消息的操作组成时，如果 $x=a_1\cdot\cdots\cdot a_n$，那么根据上述过程有 $x'=v_1\cdot a_1\cdot\cdots\cdot v_n\cdot a_n\cdot v_{n+1}$，所以 $\rho_m\leq\rho_n, x\leq x'$。

此性质决定了有损信道的相容性是可判定的。下节给出具体的证明。

4 服务合成相容性的判定

判断服务的相容性，需要判断服务的所有运算序列是否都到达终止状态。通过建立可达树的方法，分析服务合成的所有运算序列，判断服务间相容与否。

具有正确信道的服务合成，其可达树的建立过程可能面临不可终止的状况。而为具有有损信道的合成服务建立可达树的过程，则在可控的时间复杂度内一定可以完成。因此通过为具有有损信道的合成服务建立可达树的算法，能够找出服务所有的运行序列。如果可达树是有限的，那么合成服务的运行序列就是有限的。如果可达树的每个分支都可到达终止状态，那么服务合成是相容的。

服务合成可达树的建立步骤如下：

初始时：合成服务 C' 的可达树 RT 为空，合成服务的初始构造为 ρ_0，终止状态的构造为 ρ_{end}；

1. 可达树的根节点标记为合成服务的初始构造 ρ_0。

2. 可达树的一个节点 nd 标记为 ρ_k。如果 ρ_k 满足要求 $C_i^k.P^k\in ter, 1\leqslant i\leqslant n$，节点所在的可达树分支建立终止；如果构造 ρ_k 不执行任何转换，节点 nd 没有后继，nd 所在的可达树分支建立终止。不然，以广度优先方式建立可达树；如果存在构造 ρ_l 由 ρ_k 转换，节点 nd 有后继节点 nd'，标记为 ρ_l。

3. 重复步骤 2 至出现以下情况之一时终止：

(i)可达树中的所有分支建立终止时的节点，其相应的构造 ρ_k 满足要求 $C_i^k.P^k\in ter, 1\leqslant i\leqslant n$，可达树建立终止；

(ii)可达树中的所有分支都是有限的，可达树建立终止；

(iii)如果一个节点 nd' 是节点 nd 的后代，则分别标记为 ρ_l 和 ρ_k，如果 $\rho_k\leq\rho_l$，则可达树是无限的，可达树建立终止。

根据可达树的构造可以看出，可达树的每一个分支代表合成服务的一个运行序列。在第一种情况下，合成服务的运行过程都可到达终止状态，参与服务合成的组件服务是相容的。

下面给出满足具有有损信道的 Web 服务，其可达树建立是可终止的依据。

性质 2 具有有损信道的合成服务的可达树是无限的当且仅当存在构造 ρ_k 与 ρ_l，$\rho_k\Rightarrow\rho_l$ 且 $\rho_k\leq\rho_l$。

证明：充分性证明。

如果可达树是无限的，根据文献[14]的 Koenig's lemma，一个无限的树，如果其每个节点的度数（即有定义的转换的个数，对 Web 服务而言，是构造转换的个数，因为消息是有限的，所以转换的个数有限）是有限的，那么该可达树必定存在无限分支。

可达树的每个节点与合成服务的一个构造对应，所以无限分支对应一个无限的构造序列。而构造中的进程的可达集是有限的，消息集合 A 中的消息个数也是有限的，记 $N=\{\langle Rea(C_i.P)\rangle_{1\leqslant i\leqslant n},A\}$，集合中的元素个数是有限的。根据 Higman 引理[15]，一个有限的集合 M，不存在 M^* 的无限序列 $w_1,w_2,\cdots,w_n,\cdots$，对于任意的 $i<j, w_i\not\leq w_j$。因此对由有限集合 N 中的元素组成的可达树的一个无限分支中必定存在标记为 ρ_l 和 ρ_k 的节点，$\rho_k\Rightarrow\rho_l$ 且 $\rho_k\leq\rho_l$。

必要性证明。

如果有构造 $\rho_k\Rightarrow\rho_l$，那么可达树中必然存在这样的分支：节点 nd 和 nd' 分别标记为 ρ_k 和 ρ_l，节点 nd' 由 nd 可达。又因为 $\rho_k\leq\rho_l$，根据性质 1，存在构造 ρ_m,ρ_n，使 $\rho_k\Rightarrow\rho_m,\rho_m\Rightarrow\rho_l,\rho_l\Rightarrow\rho_n$，且 $\rho_m\leq\rho_n$。因此，根据 $\rho_m\Rightarrow\rho_n,\rho_m\leq\rho_n$，又可以继续推出新的属于同一分支上的构造满足关系 $\leq$，所以分支是无限的。

根据文献[12]，可达树的构造是可以终止的。首先，判断构造 ρ_k 和 ρ_l 之间是否存在关系 $\rho_k\leq\rho_l$ 的复杂度则是取决于字符串的长度，即队列中的内容的长度。其次，对于有限集合 $N=\{\langle Re\,a(C_i.P)\rangle_{1\leqslant i\leqslant n},A\}$ 上的无限构造序列 $C_1,C_2,\cdots,C_n,\cdots$，根据性质 2，存在 $i_1<i_2<\cdots<i_n<\cdots$ 使 $C_{i_1}\leq C_{i_2}\leq\cdots\leq C_{i_n}\leq\cdots$。也就是说如果可达树中有无限的分支，在无限的分支一定上存在节点标记为 ρ_k 和 ρ_l，满足 $\rho_k\leq\rho_l$，从而使有无限分支的可达树的建立可终止。

根据以上性质，我们有如下结论。

定理 1 合成服务 C' 的可达树中每个分支都达到终止状态，那么合成服务 C 相容。

证明：合成服务 C' 的可达树中每个分支都达到终止状态，意味着 Web 服务合成 C 中组件服务的进程转化为定义完全的进程后，合成 C' 中的组件服务，在异步通信的情况下，所有的运行序列都可终止。而原有合成 C 中的任意运行序列都包含在 C' 的运行序列中，所以 C 中的所有运行序列也都可达到终止状态。根据定义 5，C 中的组件服务是相容的。

例 1 如图 2(a)所示，两个用进程 P,Q 表示的 Web 服务，它们的定义完全后的进程 P_{cs},Q_{cs} 如图 2(b)所示。合成的可达树如图 2(c)所示。每个分支都能到达终止状态，所以两个 Web 服务是相容的。

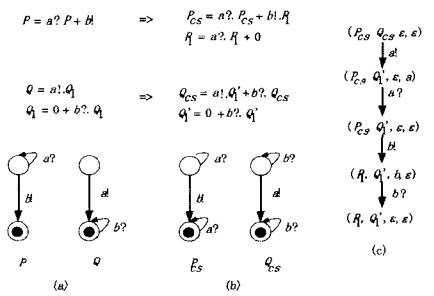

图 2　Web 服务合成相容示例

定理 1 给出的是服务合成相容的充分条件，并不是必要条件。如果 C' 不满足定理 1 的条件，并不能说明服务合成 C 一定不满足相容性要求。我们分析可达树不满足定理 1 时的情况，此时可达树或是无限的，或是有限但分支并不以终止状态结束。如果 C' 的可达树是有限的但并不是每个可达树的分支都以终止状态结束，那么合成服务 C 的可达树的所有分支也都是有限的，即合成服务的运行序列都是有限的。此时，我们可以建立合成 C 的可达树，分析每个分支是否都可以达到终止状态，因为每个分支都是有限的，可达树的建立是会终止的。如果 C' 的可达树是无限的，此时合成服务 C 的可达树是否是有限的不可确定，其相容性问题也不可确定。我们以两个例子说明上述两种情况。

例 2　如图 3 所示，两个用进程 P,Q 表示的 Web 服务，定义完全后的进程 P_{cs}，Q_{cs} 的合成的可达树是有限的，但并不是每一个分支都到达终止状态（见图 3(a)）。为两个服务的合成本身建立可达树（见图 3(b)），分支达到终止状态，两个 Web 服务相容。从此例中也可看出，合成服务的所有运行序列都包含在定义完全的合成服务的运行序列中。

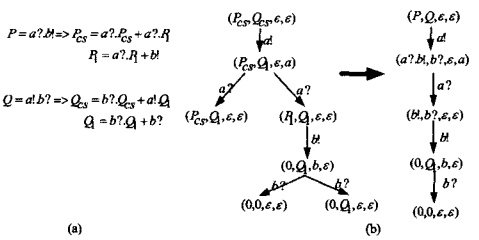

图 3　Web 服务合成相容示例

例 3　如图 4 所示，两个用进程 P,Q 表示的 Web 服务，定义完全后的进程 P_{cs}，Q_{cs} 的合成的可达树的分支中，有构造满足关系 $\preceq$，如 $(P_1',Q_{cs},\varepsilon,\varepsilon)\preceq(P_1',Q_{cs},a,\varepsilon)$，可达树是无限的，定义完全的合成服务不满足相容性要求。进程 P,Q 表示合成服务的相容性不可确定。

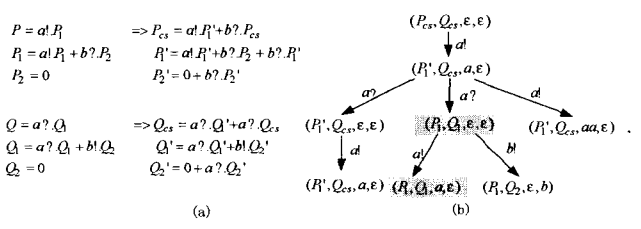

图 4　Web 服务合成相容性示例

结束语　本文介绍一种异步交互的 Web 服务合成的相容性分析方法。给出的判断算法是 Web 服务相容的充分条件，但不是必要条件。下一步的工作是分析自顶向下的 Web 服务合成分析方法所得到的结论与本文中的方法之间的关系，以进一步研究服务合成的相容性问题。

参 考 文 献

[1] Business Process Execution Language for Web Services (BPEL), Version 1.1[EB/OL]. http://www.ibm.com/developerworks/library/ws-bpel, May 2003

[2] W3C. Web Service Choreography Interface (WSCI) [EB/OL]. World W3C (2002). http://www.w3.org/TR/WSCI

[3] Bultan T, et al. Conversation Specification: A New Approach to Design and Analysis of E-Service Composition[C]// Proc. Int'l World Wide Web Conf. (WWW 2003). ACM Press, 2003: 403-410

[4] Fu X, Bultan T, Su J. A Top-Down Approach to Modeling Global Behaviors of Web services[C]// Workshop on Requirements Engineering and Open Systems (REOS). September 2003

[5] Fu X, Bultan T, Su J. Conversation Protocols: A Formalism for Specification and Verification of Reactive Electronic Services. [C] // Proceedings of the 8th International Conference on Implementation and Application of Automata (CIAA). Santa Barbara, USA, July 2003: 188-200

[6] Bordeaux L, Salun G, Berardi D, et al. When are Two Web Services Compatible? [C]// TES 2004. Springer, 2005: 15-28

[7] Martens A. On Compatibility of Web Services[J]. Petri Net Newsletter, 2003, 65: 12-20

[8] Foster H, Uchitel S, Magee J, et al. Compatibility Verification for Web Service Choreography // Proc. of the 3rd IEEE Int'l Conf. on Web Services (ICWS 2004). San Diego: IEEE Press, 2004: 738-741

[9] Brogi A, Canal C, Pimentel E, et al. Formalizing Web services Choreographies [J]. Electronic Notes in Computer Science, 2004, 105: 73-94

[10] Deutsch A, Sui L, Vianu V, et al. Verification of Communicating Data-Driven Web Services[C]// Proceedings of PODS'06. Chicago, Illinois, USA, 2006: 1-10

[11] Milner R. Communication and Concurrency[M]. Prentice Hall, 1989

[12] Finkel A. Decidability of the termination problem for completely specified protocols[J]. Distributed Computing, 1994, 7: 129-135

[13] Abdulla P, Jonsson B. Verifying Programs with Unreliable Channels[C]// Proceedings of the 8th IEEE Symposium on Logic in Computer Science. 1993

[14] Lothaire M. Combinatorics on Words [M]. Addison_Wesley, 1983

[15] Teller P. A Modern Formal Logic Primer (1st edition) [M]. Prentice Hall College Div, 1989