软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(9):2697−2715 [doi: 10.13328/j.cnki.jos.005279]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于逻辑 Petri 网的服务流程结构演化研究*

胡 强[1], 任志考[1], 赵 振[1], 杜军威[1], 杜玉越[2]

[1](青岛科技大学 信息科学技术学院,山东 青岛 266061)

[2](山东科技大学 计算机科学与工程学院,山东 青岛 266590)

通讯作者: 杜军威, E-mail: djwqd@163.com

摘 要: 流程结构演化是实现服务流程重构的一种有效手段,可以充分利用已有流程资源快速定制满足新业务需求的服务流程.然而,当前服务演化研究多关注于流程局部组成服务以及接口参数的兼容替换,对于流程结构演化所提供的操作过于简单,难以应对复杂流程演化场景.针对上述问题,提出一种基于逻辑 Petri 网的服务流程结构演化形式化描述方法.利用逻辑 Petri 网将服务流程建模为服务网,在服务网的基础上,针对不同的演化需求构建相应的结构演化运算;引入结构范式概念评价服务流程的结构健壮性,并借助逻辑 Petri 网的结构性质来分析与验证所建立的演化运算对流程结构范式的级别保持问题.给出了基于流程结构演化的服务流程定制框架,并基于所提出结构演化运算设计开发了仿真验证平台,验证了方法的有效性.

关键词: Web 服务;服务演化;结构范式;Petri 网

中图法分类号: TP311

中文引用格式: 胡强,任志考,赵振,杜军威,杜玉越.基于逻辑 Petri 网的服务流程结构演化研究.软件学报,2018,29(9): 2697−2715. http://www.jos.org.cn/1000-9825/5279.htm

英文引用格式: Hu Q, Ren ZK, Zhao Z, Du JW, Du YY. Study on structure evolution for service processes base on logic Petri net. Ruan Jian Xue Bao/Journal of Software, 2018,29(9):2697−2715 (in Chinese). http://www.jos.org.cn/1000-9825/5279.htm

Study on Structure Evolution for Service Processes Base on Logic Petri Net

HU Qiang[1], REN Zhi-Kao[1], ZHAO Zhen[1], DU Jun-Wei[1], DU Yu-Yue[2]

[1](School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao 266061, China)

[2](School of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China)

Abstract: Structure evolution is an efficient way for service processes to perform process reconstruction. It can make good use of the existing service processes to build a new value-added service process. However, the traditional research methods of service evolution focus more on compatible substitution of the partial component services or interface parameters in the service processes. Meanwhile, the operations of structure evolution are too simple in the existing theoretical methods and fail to cope with the complex evolutionary requirements. To solve these problems, a formal method is proposed to achieve structure evolution for service processes based on logic Petri net in this paper. The service process is modeled as a service net based on logic Petri net. Several structure evolution operations are defined to deal with different evolutionary requirements. Structure normal form is introduced to evaluate the soundness of service processes. Property preservation based on the soundness of service process is also investigated by using the structure analysis and validation methods of Petri net. Finally, a framework for customization of service processes based on structure evolution is proposed and

* 基金项目: 国家自然科学基金(61170078, 61273180, 61472228); 山东省优秀中青年科学家科研奖励基金(BS2015DX010, BS2015ZZ006); 山东省重点研发项目(2016GGX101031)

Foundation item: National Natural Science Foundation of China (61170078, 61273180, 61472228); the Promotive Research Fund for Young and Middle-Aged Scientisits of Shandong Province (BS2015DX010, BS2015ZZ006); the Key Research Program of Shandong Province (2016GGX101031)

收稿时间: 2016-10-09; 修改时间: 2016-11-25; 采用时间: 2017-02-15; jos 在线出版时间: 2017-03-31

CNKI 网络优先出版: 2017-03-31 21:54:59, http://kns.cnki.net/kcms/detail/11.2560.TP.20170331.2154.012.html

the simulation platform where evolution operations can be performed is also designed to illustrate the effectiveness of the proposed method.

Key words: Web service; service evolution; structure normal form; Petri net

伴随着云计算与物联网等信息技术的发展与日趋成熟,网构软件的开发和部署模式体现出了显著的服务化特征[1].网构软件在开发过程中大多采用面向服务的体系结构(service oriented architecture,简称 SOA),将现实处理中的业务操作抽象和封装为服务,通过网络发布以供用户调用[2].特别是近年来,随着“互联网+”战略的提出,基于 SOA 模式的业务系统的普及程度越来越高[3].在众多基于 SOA 架构的软件开发模式中,采取 Web 服务作为网构软件的基本构件是一种常用的方式[4,5].Web 服务技术出现了已有 20 余年,当前主流软件开发工具中均支持 Web 服务的构建,是一种非常成熟且得到广泛应用的 SOA 架构软件开发方法[6].本文中的服务是指以 Web 服务应用技术为代表的一切通过网络发布可供用户调用的应用程序模块.

作为一种可以通过网络远程调用的应用程序模块,服务通常被设计得功能粒度较小,大多采取将若干服务按照一定的流程逻辑链接成为组合服务流程的形式来应对复杂的业务需求.然而,不论单个的原子服务还是组合服务流程,均面临着复杂多变的分布式应用环境,在服务或者服务流程生命周期内可能存在个别服务失效的情况;同时,随着业务流程的执行,用户已有的服务需求也可能会产生变化.因此,为能够使得基于 SOA 架构的软件有效应对外部使用环境或者内部应用需求的变化,必须为之提供一套有效的服务演化机制.

服务演化研究在失效服务替换[7−10]、服务流程实例迁移[11,12]、服务接口演化[13,14]以及服务演化生命周期与版本管理[15]等方面已经取得丰硕的成果.近年来,服务流程演化研究逐渐成为热点,研究人员对服务流程的在线演化[16,17]、动态演化[18]以及可信演化[19,20]等方面展开了研究,取得了一系列的理论成果.在服务流程演化研究中大多包含流程的结构演化,然而,当前的流程结构演化研究主要集中在流程结构的局部替换以及简单流程模式之间的逻辑变换,所提供的演化操作较为简单且多面向流程实例层面,缺乏从抽象流程模型层面支持流程结构演化的形式化理论研究.

本文从服务流程的模型层面对流程结构演化展开研究,提供一组丰富的结构演化操作,借助逻辑 Petri 网作为形式化工具,构建服务流程的结构演化运算,为流程的结构演化提供形式化描述方法,并探讨结构演化过程中对流程结构健壮性的保持问题.借助本文研究所提出的结构演化操作,可以有效地实现服务流程的重构和业务功能的宽幅转换,在已有服务流程资源的基础上,快速实现满足新服务需求的业务流程定制.

1 相关研究

服务演化的目的是为当前响应服务(流程)寻找一个适合新服务需求或应用场景的替换服务(流程).如前文所述,目前,关于服务演化的研究工作主要集中在服务接口参数演化、流程失效服务替换、服务流程实例的迁移以及结构演化等几个方面.其中,接口演化通常用于原子服务的等价替换,在参数接口和功能层面为当前服务寻找可替换服务,所使用的方法主要通过计算参数相似度实现接口匹配,进而判定是否可以进行接口参数演化.例如,文献[14]构建了一种可以从 WSDL 文档中提取影响接口参数变化的细粒度语法要素工具,并根据所提供的参数接口相似度判定接口参数演化是否可行.

在服务流程的局部组成服务演化方面,研究者主要开展了失效服务的替换研究,提出了诸多的服务替换方法,如基于组合上下文的服务替换[7]、基于行为效果的服务替换[8]、基于社交网络的服务协同与替换[9]以及基于 WS-BPEL 与面向方面编程的服务容错策略及替换方案[10].流程的局部组成服务的演化要充分考虑替换服务所处的流程上下文环境,确保替换后的服务与流程的组成服务以及外部交互服务在参数和功能层面的兼容性.在流程实例的迁移研究中,宋敏等人[21]基于着色 Petri 网提出了面向数据流和控制流的网构软件服务模型,分析了 5 种动态演化操作可能引发的数据流错误,并建立了面向数据流的服务实例可迁移性准则.宋巍等人[11]提出了一种服务组合动态演化过程框架,在此框架下,形式化地定义了一种新的实例可迁移性标准,并给出了相应的判定算法.与已有的可迁移性标准相比,该标准在确保不会产生动态演化错误的同时,可允许更多的实例迁移.

Ryu SH 等人[22]在交互协议层面研究了 Web 服务流程结构的动态演化问题,给出了流程的迁移和演化规则.

在流程演化研究中,何俊等人对经典 π 演算进行扩展,形式化表示和描述 SaaS 服务流程并建立服务流程演化模型,对服务流程演化前后的互模拟和模型簇膨胀问题进行了研究[23].邓水光等人研究了大粒度服务组合问题,提出了一种在服务集的基础上演出 top-k 质量优化的组合服务流程的方法[24].Hamadi 利用 Petri 网对 Web 服务进行建模,提出了用于描述基本流程逻辑模式和构建服务流程的合成与分解运算,可为流程结构演化提供良好的借鉴[25].Silva 提出一种将基因编程与有向图相结合的服务组合演化技术 GraphEvol,有效简化了利用遗传算法进行流程演化时的变异和交叉操作[26].Mohammad 等人提出了基于生物遗传的 SaaS 服务演化,对于从流程继承角度研究流程定制具有重要意义[27].Ralph 等人在流程模板中引入可变点选项,根据租户请求和可变点选项演化来实现 SaaS 流程定制[28].Schumm 等人[29]提出一个粗粒度服务组合与演化框架,提供了服务发布、检索、抽取以及集成操作.从上述研究可以看出:多数服务流程演化研究是以服务组合和流程定制为最终目的,所提供的演化方法多以原子服务为演化处理对象,并且提供的演化模式较为简单,主要用于处理常见的顺序、选择、并发以及循环等最基本的流程结构演化,缺乏对复杂演化需求下服务流程演化的支持.

流程演化会引发流程结构和功能的变化,因此,演化得到的服务流程可能无法保持已有的结构健壮性.流程结构健壮性研究来源于工作流领域,Alst 在文献[30]中给出了工作流网健壮性定义,并在文献[31]将健壮性应用于普通业务流程的评价.同时,对健壮性进行了层次划分,划分为:弱良好、k 良好与最大 k 良好.近年来,有关某些特定子类工作流网的健壮性得到广泛关注,并取得一系列的研究成果[32−34].当前,也有文献研究服务与服务流程的健壮性,这类研究多数采取 Petri 网对组合服务流程建模,在工作流网健壮性的基础上,从流程的控制流和数据流两个角度提出服务流程的健壮性判定准则,然后,借助 Petri 网的结构性质分析方法,提出自己的流程健壮性判定方法[35−37].在前期工作中,我们提出了用于评价 Web 服务流程健壮性的结构范式概念[36],从结构可达性和冗余性两个角度将服务流程健壮性划分为 4 个层次.本文将在上述工作的基础上,探讨构建演化运算对流程结构健壮性的保持问题.

2 服务流程形式化描述

在对服务流程的建模和结构性质研究过程中,进程代数[38−40]、状态机[41,42]以及 Petri 网[25,33,34,36,37]是常用的形式化工具.进程代数属于语法层次的形式化描述工具,Petri 网和状态机同属于状态类描述模型,隶属于语义层次的形式化描述工具,具有直观的建模和良好的结构性质分析方法.然而在进行复杂系统建模和性质分析时,状态类形式化工具通常面临状态空间爆炸问题.为了更好地展示服务流程的结构演化,同时缓解结构分析时状态空间爆炸问题,本文采取逻辑 Petri 网作为形式化工具研究服务流程的结构演化及其健壮性保持问题.

逻辑 Petri 网是在变迁中添加了逻辑表达式的一类高级 Petri 网,通过引入逻辑表达式对 Petri 网中的标识触发逻辑做出约束.相比传统 Petri 网,逻辑 Petri 网在进行系统建模和流程描述时得到的模型规模较小且结构清晰.在前期研究中已经应用逻辑 Petri 网对 Web 服务流程[43]、电子商务[44]和工作流[45]等进行了建模和性质分析.下面介绍逻辑 Petri 网与服务网的相关概念.

定义 1(逻辑表达式[36]). $P=\{p_1,p_2,\ldots,p_n\}$是一个布尔变量集合, $f=\overline{p}_{i_1}\Delta_1\overline{p}_{i_2}\ldots\Delta_{j-1}\overline{p}_{i_j}\Delta_j\ldots\Delta_{m-1}\overline{p}_{i_m}$ 称为集合 P 上的逻辑表达式,其中, $\Delta_j\in\{\vee,\wedge\}$,$\vee$ 与 $\wedge$ 分别为逻辑析取与合取运算符, $\overline{p}_{i_j}$ 称为 f 的组成元项, $\overline{p}_{i_j}=\overline{p}_{i_j}$或$\neg\overline{p}_{i_j}$, $p_{i_j}\in P$,$\neg$ 为逻辑非运算符,$1\leqslant i_j\leqslant n$.

引入符号Δ和 $\overline{P}$ 分别表示逻辑表达式 f 中的 $\Delta_1\ldots\Delta_{j-1}\Delta_j\ldots\Delta_{m-1}$ 和 $\overline{p}_{i_1}\overline{p}_{i_2}\ldots\overline{p}_{i_j}\ldots\overline{p}_{i_m}$ 的元素序列,并将 f 的所有组成元项的集合 $\overline{P}$ 标记为 Ω_f.

例如,$P=\{p_1,p_2,\ldots,p_8\}$,$f=p_1\wedge p_3\vee p_6\wedge p_8$ 为集合 P 上的逻辑表达式,则 $\Omega_f=\{p_1,p_3,p_6,p_8\}$.

定义 2(Petri 网). Petri 网是一个四元组集合 $PN=(P,T;F,M)$,其中:P 是库所集合,T 是变迁集合,且 $P\cap T=\varnothing$;$F\subseteq(P\times T)\cup(T\times P)$称为 Petri 网的流关系,映射 $M:P\to\{0,1,2,\ldots\}$是 Petri 网的标识.

若$\forall p\in{}^\bullet t:M(p)\geqslant 1$,则称 t 在 M 是下使能的,记为 $M[t\rangle$;若 $M[t\rangle M'$,则对于$\forall p\in P,M'(p)$可通过以下公式计算:

$$M'(p)=\begin{cases}M(p)-1, & p\in {}^\bullet t-t^\bullet\\ M(p)+1, & p\in t^\bullet-{}^\bullet t.\\ M(p), & \text{otherwise}\end{cases}$$

为区别 Petri 网定义中的符号 T 和 F,引入符号$_\bullet T_\bullet$和$_\bullet F_\bullet$分别表示逻辑真与假.设 $PN=(P,T,F,M)$为一个 Petri 网,其中,$P=\{p_1,p_2,\dots,p_n\}$是一个有限的库所集,f 是 P 上的一个逻辑表达式.对 $p_i\in P$,$p_i|_M$ 表示逻辑表达式 f 中逻辑变量 p_i 在标识 M 下的真值,其中,

$$p_i\,|\,M=\begin{cases}{}_\bullet T_\bullet, & \text{if } M(p_i)\geqslant 1\\ {}_\bullet F_\bullet, & \text{if } M(p_i)=0\end{cases};$$

并且,$f|_M$ 表示逻辑表达式 f 在 M 下的逻辑值,其中,$f|_M=f(p_1,p_2,\dots,p_n)|_M=f(p_1|_M,p_2|_M,\dots,p_n|_M)$.

定义 3(逻辑 Petri 网[36]). 设 $LN=(P,T,F,I,O)$,$LPN=(LN,M)$称为一个逻辑 Petri 网,其中,

(1) P 是一个有限库所集合;

(2) $T=T_D\cup T_I\cup T_O$ 是一个有限变迁集合,且 $T\cup P\neq\varnothing$,$\forall t\in T_I\cup T_O$:${}^\bullet t\cap t^\bullet=\varnothing$,其中,

- T_D 为一个传统 Petri 网变迁集合;
- T_I 为一个逻辑输入变迁集合.$\forall t\in T_I$,t 的触发受其输入库所集上的逻辑表达式 $f_I(t)$的约束,即,$f_I(t)$中的谓词变量包含 t 的所有输入库所,称 $f_I(t)$为逻辑输入表达式;
- T_O 为一个逻辑输出变迁的集合.$\forall t\in T_O$,t 触发后的结果受其输出库所集上的逻辑表达式$f_O(t)$的约束,即,$f_O(t)$中的谓词变量包含 t 的所有输出库所,称 $f_O(t)$为逻辑输出表达式;

(3) $F\subseteq(P\times T)\cup(T\times P)$是一个弧的集合;

(4) I 是一个从逻辑输入变迁到逻辑输入表达式的映射,即,$\forall t\in T_I$,$I(t)=f_I(t)$;

(5) O 是一个从逻辑输出变迁到逻辑输出表达式的映射,即,$\forall t\in T_O$,$O(t)=f_O(t)$;

(6) $M:P\to\{0,1\}$是一个标识函数;

(7) 变迁发生规则:

- 对$\forall t\in T_D$,变迁发生规则与普通 Petri 网中变迁发生规则相同;
- 对$\forall t\in T_I$,若 $f_I(t)|_M={}_\bullet T_\bullet$,则 $M[t\rangle$,并且,若 $M[t\rangle M'$,则$\forall p\in{}^\bullet t$:$M'(p)=0$;$\forall p\in t^\bullet$:$M(p)=0$,$M'(p)=1$;且$\forall p\notin{}^\bullet t\cup t^\bullet$:$M'(p)=M(p)$;
- 对$\forall t\in T_O$,若$\forall p\in{}^\bullet t$:$M(p)=1$,则 t 是使能的,并且,若 $M[t\rangle M'$,则$\forall p\in{}^\bullet t$:$M'(p)=M(p)-1$;$\forall p\notin t^\bullet\cup{}^\bullet t$:$M'(p)=M(p)$;且$\forall p\in t^\bullet$:$f_O(t)|_{M'}={}_\bullet T_\bullet$.

图 1 为一个逻辑 Petri 网,库所 p_1 与 p_3 中具备令牌.该模型中,变迁 t_1 为逻辑输入变迁,标记了表达式 $p_1\wedge(p_2\vee p_3)$,表示若 p_1 与 p_2 或 p_3 中的一个库所同时具备令牌,t_1 可触发.变迁 t_3 为逻辑输出标签,标记了表达式 $p_7\wedge\neg p_8\vee\neg p_7\wedge p_8$,表示 t_3 触发后只能使得 p_7 与 p_8 中的一个具有令牌.

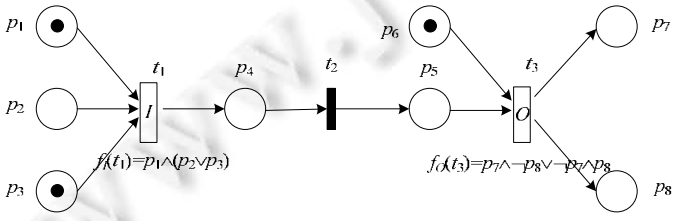

Fig.1 An example of logic Petri net

图 1 逻辑 Petri 网示例

定义 4(服务网). 服务网为一个标记逻辑 Petri 网,表示为五元组 $SN=(LPN,i,o,\varphi,L)$,其中,

(1) LPN 是业务流程模型,为一个具有唯一起始库所 i 和终止库所 o 的逻辑 Petri 网;

(2) $P=P_D\cup P_I\cup P_C$,P_D 为外部数据库所集合,用于与外部服务进行数据交互;P_I 为内部数据库所集合,用于

服务流程内部的数据交互;P_C 为控制库所集合,用于描述服务流程的转移;

(3) $T=T_s\cup T_I\cup T_O$:T_s 为服务变迁集合,是服务流程的组成 Web 服务集合;T_I 和 T_O 为分别逻辑输入和输出变迁集合,用于控制服务流程的业务逻辑;

(4) M 称为 SN 的标识,令 M_0 为初始标识,且 $M_0(i)=1,\forall p\in P-\{p\}:M_0(p)=0$;

(5) $\varphi:T\to L$ 是一个映射函数,其中,L 表示流程中的变迁名称集合.

SN 是一个服务网,$x\in T\cup P$ 为 SN 中的一个结点,则:(1) ${}^\bullet x=\{y|(y,x)\in F\}$称为 x 的前集;(2) $x^\bullet=\{y|(x,y)\in F\}$称为 x 的后集.引入获取服务网中结点前集与后集的运算算子π,即:对$\forall x\in T\cup P$,令$\pi_1(x)={}^\bullet x,\pi_2(x)=x^\bullet$,且$\pi(x)=\pi_1(p)\cup\pi_2(p)$.在前集和后集的定义基础上,给出逻辑表达式范型的定义:

定义 5(表达式范型). $F=\{f_1,f_2,\ldots,f_n\}$是一组服务流程,t_o 和 t_i 是连接 F 的分支和汇集逻辑变迁,$\forall f_j:\pi_1(f_j.i)=\{t_o\}$,$\forall f_j:\pi_2(f_j.o)=\{t_i\}(1\leqslant j\leqslant n)$.对分支和汇集变迁,定义以下 4 种形式的逻辑表达式范型.

(1) $O_\vee:O(t_o)=(f_1.i\wedge\neg f_2.i\neg\wedge\ldots\wedge\neg f_n.i)\vee(\neg f_1.i\wedge f_2.i\neg\wedge\ldots\wedge\neg f_n.i)\vee\ldots\vee(\neg f_1.i\wedge\neg f_2.i\wedge\ldots\wedge f_j.i\wedge\ldots\wedge\neg f_n.i)\vee\ldots\vee(\neg f_1.i\wedge\neg f_2.i\neg\wedge\ldots\wedge f_n.i)$;

(2) $O_\wedge:O(t_o)=f_1.i\wedge f_2.i\wedge\ldots\wedge f_n.i$;

(3) $I_\vee:I(t_i)=f_1.o\vee f_2.o\vee\ldots\vee f_n.o$;

(4) $I_\wedge:I(t_i)=f_1.o\wedge f_2.o\wedge\ldots\wedge f_n.o$.

对于给定的一组服务流程 $F=\{f_1,f_2,\ldots,f_n\}$,图 2 给出了由上述服务流程组成的服务网的 4 种基本模式,分别为顺序、选择、并行以及循环.在选择模式下,规定构成服务网的逻辑变迁所标注的逻辑表达式必须是 $O_\vee$和 $I_\vee$型;在并行模式下,则要求逻辑变迁标注的逻辑表达式必须是 $O_\wedge$和 $I_\wedge$型.

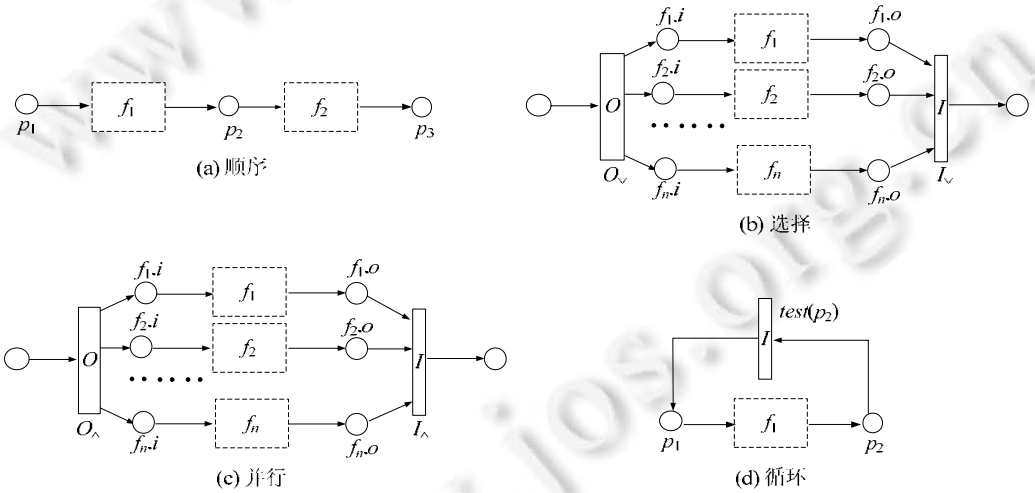

Fig.2 Four basic process patterns of service net

图 2 4 种基本的服务网流程模式

3 服务网的结构演化运算

服务流程的结构演化会引发流程组成服务的变化与业务逻辑的改变,因此,反映到流程模型层面会触发服务网的组成结点与逻辑表达式的变化.下面首先给出 3 种逻辑表达式的演化运算,然后,在此基础上构建服务网的结构演化运算.

3.1 逻辑表达式的演化

服务流程在演化过程中会涉及到流程的分解、部分流程分支的替换以及流程分支的合并,对应上述 3 种情况,建立了 3 种逻辑表达式的演化操作,分别是逻辑投影、逻辑替换和逻辑合成.

定义 6(逻辑投影). $P=\{p_1,p_2,\ldots,p_n\}$为一个有限集合,f 为 P 上逻辑表达式且 $P'\subseteq P$,对$\forall p_i\in P-P'$以及 $p_i\in\Omega_f$, $f'=\Phi(f,P')$称为 f 在 P'的逻辑投影,当且仅当以下条件满足:

$$\begin{cases}\Delta_{i-1}\overline{p}_i\Delta_i\to\varnothing, & \text{if } \Delta_{i-1}=\varnothing\\ \Delta_{i-1}\overline{p}_i\Delta_i\to\Delta_i, & \text{if } \Delta_{i-1}=\wedge\\ \Delta_{i-1}\overline{p}_i\Delta_i\to\Delta_{i-1}, & \text{if } \Delta_i=\wedge\\ \Delta_{i-1}\overline{p}_i\Delta_i\to\varnothing, & \text{if } \Delta_i=\varnothing\\ \Delta_{i-1}\overline{p}_i\Delta_i\to\Delta_{i-1} \text{ or } \Delta_i, & \text{if } \Delta_{i-1}=\vee \text{ and } \Delta_i=\vee\end{cases}.$$

例如,$P=\{p_1,p_2,\ldots,p_{10}\}$,$f=p_1\vee p_3\wedge\neg p_5\vee p_6\vee p_7\wedge\neg p_8$,$\Omega_f=\{p_1,p_3,p_5,p_6,p_7,p_8\}$.若 $P'=\{p_1,p_3,p_7,p_8\}$,则 $f'=\Phi(f,P')=p_1\vee p_3\vee p_7\wedge\neg p_8$;若 $P'=\{p_1,p_5,p_6,p_7\}$,则$f'=\Phi(f,P')=p_1\vee p_5\vee p_6\vee p_7$.

定义 7(逻辑替换). $P=\{p_1,p_2,\ldots,p_n\}$为一个有限集合,f 为 P 上逻辑表达式且 $P'\subseteq\Omega_f$.逻辑表达式$f'=\omega(f,P',m)$称为 f的逻辑替换,当且仅当任意 $p\in P'$,p 被一个值 m 替换.

定义 8(逻辑合成). $P=\{p_1,p_2,\ldots,p_n\}$为一个有限集合,f,f_1与f_2均为 P 上逻辑表达式且 $\Omega_f=\Omega_{f_1}\cup\Omega_{f_2}$,则$f=\Psi(f_1,f_2,\oplus)$称为 f_1 与 f_2 的逻辑合成,当且仅当 $f=f_1\oplus f_2$,$\oplus=\vee$或$\wedge$.

例如,$P=\{p_1,p_2,\ldots,p_5\}$,$f=\neg p_1\wedge p_2\vee p_3\wedge p_4$,$f_1=p_2\vee\neg p_3$,$f_2=p_1$.令 $m=1$,$P'=\{p_2,p_3\}$,则:

$$f'=\omega(f,P',m)=\neg p_1\vee p_4,f''=\Psi(f_1,f_2,\wedge)=(p_2\vee\neg p_3)\wedge p_1=p_2\wedge p_1\vee\neg p_3\wedge p_1.$$

3.2 服务流程结构演化

服务流程的结构演化表现为其对应服务网模型的结构和流程逻辑的变化,因此,在服务网中可能会移除一些已有的变迁与库所或者增加一些新变迁与库所.为了与演化前的服务网中的变迁和库所进行区别,将所有新加入的库所和变迁在命名时均添加“′”作为上标.同时,为了更好地展示结构演化过程中服务网的变化,后文中将服务网表达为 $SN=(P,T,F,I,O,i,o,L)$形式,即,将服务网的过程模型展开表示[43].

本文建立了服务网的投影、链接、替换、集合、合成演化运算,在替换和集合演化运算中,要求参加演化的服务网在结构上具备子网约束关系,下面首先给出子网的概念:

定义 9(子网). 逻辑 Petri 网 LPN'是 LPN 的子网,当且仅当 $P'\subseteq P,T'\subseteq T,F'\subseteq F,I'(t')=\Phi(I(t),P')$且:

$$O(t')=\Phi(O(t),P').$$

对于给定服务网 SN,其过程模型为 $SN.LPN$,若 LPN'是 LPN 的子网,则将由 LPN'构成的网 SN'称为 SN 的子网,记为 $SN'\ll SN$;若 SN'中包含起始库所 i 和终止库所 o,则 SN'称为 SN 的完美子网,记为 $SN'\underline{\ll}SN$.

3.2.1 投影演化运算

投影演化用于从已有的服务流程中获取包含特定组成服务的服务流程片段,在形式化模型层角度,表现为从服务网中截取一段包含特定组成变迁的服务网片段.

在对服务网进行投影演化运算时,用户通常只能给出参与运算的服务变迁,因此在获取服务网片段时,首先应该将参加投影演化的变迁进行扩展,将与上述服务变迁相关联的逻辑变迁添加到参与投影演化的变迁集合中.对给定服务网 SN 与投影变迁集合为 T^*,其中,$T^*\subseteq S.T$,则扩展投影变迁集合定义为 $T_\chi=\{t|t\in\pi(\pi(T^*))\wedge\exists p\in\pi_1(t)\cap\pi(T)\wedge\exists p'\in\pi_2(t)\cap\pi(T)\}$.扩展投影变迁集合 T_χ获取过程如下:

(1) 在投影变迁集 T^*的基础上构建投影库所集合 P^*,P^*由投影变迁集 T^*所有变迁的前集和后集库所组成,P^*所有的库所都是构建投影服务网所必须的;

(2) 获取变迁集 T_χ,T_χ是由与 P^*中库所相关联的变迁组成;删除 T_χ中无意义逻辑变迁(只将至少同时包含一个前集或后集库所在 P^*中的逻辑变迁保留).

定义 10(投影演化). χ定义为投影演化操作算子,$SN=(P,T,F,I,O,i,o,L)$为服务网,$T^*\subseteq T$,$\chi(SN,T^*)=(P',T',F',I',O',i',o',L')$,其中,

(1) $T'=T_\chi$,$T_\chi=\{t|t\in\pi(\pi(T^*))\wedge\exists p\in\pi_1(t)\cap\pi(T)\wedge\exists p'\in\pi_2(t)\cap\pi(T)\}$;

(2) $P'=\pi(T_\chi)$;

(3) $F'=P'\times T'\cup F$;

(4) $I'(t)=\begin{cases}\Phi(I(t),\Omega_{I(t)}-\pi(T')), & \forall t\in T'-T^*\\ I(t), & \text{else}\end{cases}$,$O'(t)=\begin{cases}\Phi(O(t),\Omega-\pi(T')), & \forall t\in T'-T^*\\ O(t), & \text{else}\end{cases}$;

(5) $i'=\{p|p\in P'.P_c,\text{且}\forall t\in\pi_1(p),t\notin T'\}$;$o'=\{p|p\in P'.P_c,\text{且}\forall t\in\pi_2(p),t\notin T'\}$;

(6) $L'=\Theta(T')$.

对于投影演化,若扩展投影变迁集 T_χ中的变迁无法构成有效的连通路径,则投影出的服务网中则可能存在非连通的流程片段,因此,投影演化要求给定的投影变迁所对应的扩展投影变迁集合中的变迁在服务网中必须构成连通路径,否则投影操作是无意义的.

随着互联网经济模式的确立,众多制造企业将其制造业务功能进行封装,并借助网络通过相关服务平台进行发布供用户调用,可以有效地降低企业运营成本,提供企业经营效益.本文以制造企业的发布的模具定制服务流程为背景,说明所建立的服务流程结构演化运算的作用.服务提供商将所提供的服务流程注册在服务发布平台的流程模型库中,现有一模具制造企业 *A* 发布的业务流程 *Mould_s*,其对应的服务网 S_1 如图 3(a)所示.该流程首先接受来自模具定制用户的查询(变迁 query),再查询该企业库存后,如果存在用户所需的指定型号的模具,则触发出库子流程.该流程如下:接受模具预定(变迁 reserve),发起支付交易(变迁 defray),最后进行模具配送(变迁 delivery);如果该企业库存中不存在用户所需的模具,则触发查询失败处理操作(变迁 query_fail).

假设某模具制造企业 *B* 在查询流程模型库后,发现企业 *A* 的流程 *Mould_s* 中的查询、预定、支付以及配送子流程符合企业经营需求,但同时认为 *Mould_s* 流程中的模具查询失效处理(变迁 query_fail)不符合本企业需求,因此,企业 *B* 可以执行服务流程的投影演化运算,将子查询、预定、支付以及配送子流程截取出即可.令投影集合 T^*={query,reserve,defray,delivery},则执行$\chi(S_1,T^*)$运算后得到的服务网如图 3(b)所示.

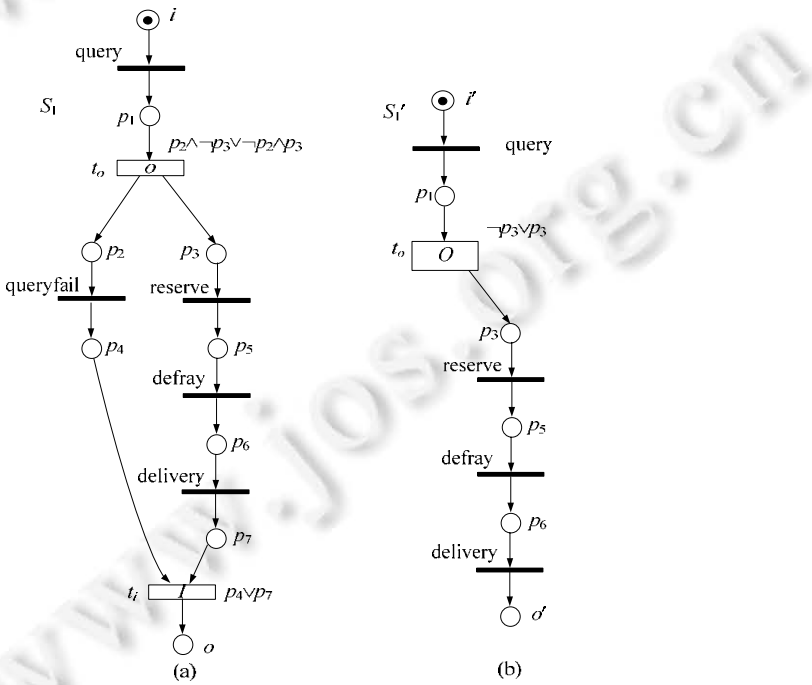

Fig.3 Evolution operations for projection

图 3 投影演化运算

3.2.2 链接演化运算

链接演化用于将一段服务流程添加到其他服务流程中,链接的模式分为选择链接和并行链接两种;在形式化模型层角度,表现为将一个服务网片段以选择或并发模式添加到另外一个服务网中.

定义 11(链接演化). θ定义为投影演化操作算子,$SN_i=(P_i,T_i,F_i,LI_i,LO_i,i_i,o_i,L_i)$为两个服务网($i$=1,2),$\theta(S_1,S_2,p_m,p_n,\gamma)=(P,T,F,LI,LO,i,o,L)$,其中,

(1) $P=P_1\cup P_2\cup\{p_1',p_2'\}$;

(2) $T=T_1\cup T_2\cup\{t_o',t_i'\}$;

(3) $F=F_1\cup F_2\cup\{\pi_1(p_m)\times p_1'\times t_o'\}\cup\{t_o'\times\{p_m,i_2\}\}\cup\{\{p_n,o_2\}\times t_i'\}\cup\{t_i'\times p_2'\times\pi_2(p_n)\}$;

(4)
$$I'(t)=\begin{cases}\omega(I(t),p_n,p_2'), & \forall t\in\Omega I_{(t)}\\ I(t_i'), & t=t_i'\end{cases},\text{且}I(t_i')=\begin{cases}p_n\vee o_2, & r=\otimes\\ p_n\wedge o_2, & r=\|\end{cases}$$
$$O'(t)=\begin{cases}\omega(O(t),p_m,p_1'), & \forall t:p_m\in\Omega_{O(t)}\\ O(t_o'), & t=t_o'\end{cases},\text{且}O(t_o')=\begin{cases}p_m\wedge\neg i_2\vee p_m\wedge i_2, & r=\otimes\\ p_m\wedge i_2, & r=\|\end{cases};$$

(5) $i=i_1,o=o_1$;

(6) $L=L_1\cup L_2$.

在上述定义中,服务网 S_2 被链接到服务网 S_1 内部,定义中的 p_m 与 p_n 是指服务网 S_1 内部的两个库所,为服务网 S_2 的接入点,即,用户需要指明 S_2 在 S_1 中的具体链接位置;γ为链接模式,分为并行链接(γ=∨)和选择链(γ=∧)接两种模式.

例如,模具企业 A 在服务发布一段时间后,通过调研发现一部分用户希望先接受到模具,经过试用验证模具质量合格后再完成相应支付,企业 A 通过检索服务模型库,获得一段流程,该流程依次完成预定、配送和支付(参见图 4(a)中服务网 S).通过执行链接服务演化,将该服务网以选择模式的方式链接到 p_3 和 p_7 库所处,使得演化后的服务流程即可先支付后配送,也可以完成先配送后发货的业务功能.执行$\theta(S_1,S,p_3,p_7,\otimes)$后,得到的服务网参见图 4(b)中右侧服务网 S_2'.

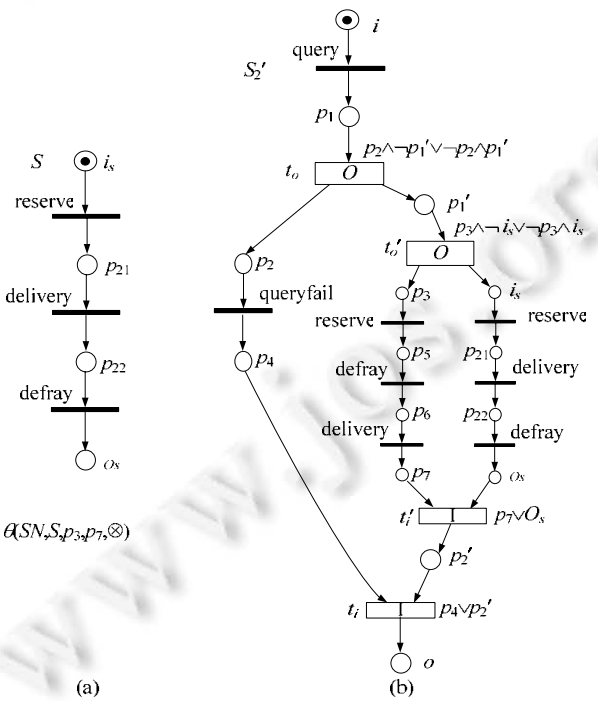

Fig.4 Evolution operations for link

图 4 链接演化运算

3.2.3 替换演化运算

替换演化是指将已有服务流程中的一个流程片段利用其他服务流程片段进行代替,从而形成具有新的业

务逻辑功能的服务流程,常用于服务流程的功能调整、流程展开与折叠.在形式化模型层角度,表现为将一个服务网中的局部流程片段用其他服务网片段进行替换.

在进行替换演化时,需要对参与替换的服务流程的输入和输出数据做约束,在满足以下条件时,服务网 SN 可以被 SN'替换:(1) SN'接受的数据是 SN 的子集;(2) SN 输出的数据是 SN'的子集.上述条件形式化表示为:

$$InData(SN'.i)\subseteq InData(SN.i)\wedge OutData(SN.o)\subseteq OutData(SN'.o).$$

定义 12(替换演化). ξ定义为替换演化操作算子,$SN_i=(P_i,T_i,F_i,I_i,O_i,i_i,o_i,L_i)$为一组服务网($i$=1,2,3),$SN_2\ll SN_1$,并且 $InData(SN_3.i)\subseteq InData(SN_2.i)\wedge OutData(SN_2.o)\subseteq OutData(SN_3.o)$.令 $SN=\xi(SN_1,SN_2,SN_3)$,则 $SN=(P,T,F,I,O,i,o,L)$,其中,

(1) $P=\{SN_1.P_1-SN_2.P_2\}\cup S_3.P_3$;

(2) $T=\{SN_1.T_1-SN_2.T_2\}\cup S_3.T_3$;

(3) $F=\{SN_1.F_1-SN_2.F_2\}\cup\{\pi_1(S_2.i)\times S_3.i\}\cup\{S_3.o\times\pi_2(S_2.o)\}$;

(4) $i=\begin{cases}S_3.i, & S_1.i=S_2.i\\ S_1.i, & S_1.i\neq S_2.i\end{cases}, o=\begin{cases}S_3.o, & S_1.o=S_2.o\\ S_1.o, & S_1.o\neq S_2.o\end{cases}$;

(5) $I'(t)=\begin{cases}\omega(I(t),S_2.o,S_3.o), & \forall t:S_2.o\in\Omega_{I(t)}\\ I(t), & \text{else}\end{cases}, O'(t)=\begin{cases}\omega(O(t),S_2.i,S_3.i), & \forall t:S_2.i\in\Omega_{I(t)}\\ O(t), & \text{else}\end{cases}$;

(6) $L=\{SN_1.L_1-SN_2.L_2\}\cup S_3.L_3$.

例如,若企业 A 现需要将查询失效处理(变迁 query_fail)变更为向其所属的加工车间配送订单,并再生产后重新模具入库上架的业务处理流程(称为入库子流程),则可将变迁 query_fail 用具备入库流程功能的服务网进行替换演化.如图 5 所示,将待演化的变迁 query_fail 定义为服务网 S_{11},服务网 S_2 用于替换 S_{11},则执行$\xi(S_1,S_{11},S_2)$后可得到满足需求的服务网 S_3'.

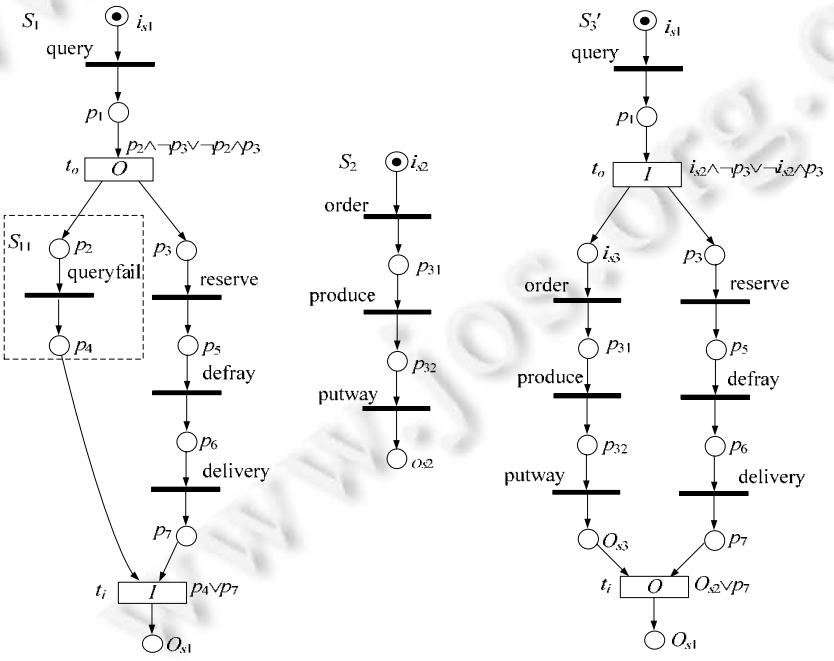

Fig.5 Evolution operations for substitution

图 5 替换演化运算

3.2.4 集合演化运算

服务流程的集合运算是指流程的交集、并集以及补集演化运算,适用于用户对寻租的服务流程进行内部组

织部门之间的功能分割与合并.例如,某用户寻租到一个包含有 5 个分支流程的服务流程供其两个生产部门使用,其中一个部门使用 3 个分支流程,另外一个部门使用剩余的分支流程.此时,用户可以先采用投影获取其中的 3 个分支流程,然后再利用补集演化运算为另外一个部门获取剩余的分支流程.下面分别给出并集演化、交集演化和补集演化的形式化定义.

定义 13(并集演化). Π定义为并集演化操作算子,$SN_i=(P_i,T_i,F_i,LI_i,LO_i,i_i,o_i,L_i)$为服务网($i$=1,2,3),并且满足$SN_1 \underline{\ll} SN_3, SN_2 \underline{\ll} SN_3$.令 $SN'=\Pi(SN_1,SN_2)$,则 $SN'=(P',T',F',LI',LO',i',o',L')$,其中,

(1) $P'=P_1\cup P_2$;

(2) $T'=T_1\cup T_2$;

(3) $F'=F_1\cup F_2$;

(4)
$$I'(t)=\begin{cases} I(t), & \forall t\in T'.T_I \wedge \Theta(t)\in L_1\cup L_2-L_1\cap L_2 \\ \Psi(I_1(t),I_2(t),\vee), & \forall t\in T'.T_I \wedge \Theta(t)\in L_1\cap L_2 \end{cases}$$
$$O'(t)=\begin{cases} O(t), & \forall t\in T'.To \wedge \Theta(t)\in L_1\cup L_2-L_1\cap L_2 \\ \Psi(O_1(t),O_2(t),\vee), & \forall t\in T'.To \wedge \Theta(t)\in L_1\cap L_2 \end{cases};$$

(5) $i'=i,o'=o$;

(6) $L'=L_1\cup L_2$.

定义 14(交集演化). Λ定义为交集演化操作算子,$SN_i=(P_i,T_i,F_i,LI_i,LO_i,i_i,o_i,L_i)$为服务网($i$=1,2,3),并且满足$SN_1 \underline{\ll} SN_3, SN_2 \underline{\ll} SN_3$.令 $SN'=\Lambda(SN_1,SN_2)$,则 $SN'=(P',T',F',LI',LO',i',o',L')$,其中,

(1) $P'=P_1\cap P_2$;

(2) $T'=T_1\cap T_2$;

(3) $F'=F_1\cap F_2$;

(4) $I'(t)=\Phi(I(t),T')$;$O'(t)=\Phi(O(t),T')$;

(5) $i'=i,o'=o$;

(6) $L'=L_1\cap L_2$.

定义 15(补集演化). υ定义为补集演化操作算子,$SN_i=(P_i,T_i,F_i,LI_i,LO_i,i_i,o_i,L_i)$为两个服务网($i$=1,2),并且满足$SN_1 \underline{\ll} SN_2$.令 $SN'=\upsilon(SN_1,SN_2)$,则 $SN'=(P',T',F',LI',LO',i',o',L')$,其中,

(1) $P'=P_2-P_1$;

(2) $T'=T_2-T_1$;

(3) $F'=P'\times T'\cap F_2$;

(4) $I'(t)=\Phi(I(t),T')$;$O'(t)=\Phi(O(t),T')$;

(5) $i'=i,o'=o$;

(6) $L'=L_2-L_1$;

集合演化运算要求参与运算的服务网具备完美子网约束关系,因此通常应用于企业服务流程的内部合并、协同以及拆解.例如,对于企业 A,在进行了如图 5 所示的替换演化运算后,新得到的服务网 S_3' 满足以下功能.

- 存在指定查询模具时,进行预定、支付和配送服务;
- 在不存在时,则触发生产和入库流程操作,该流程与最初流程 *Mould_s* 相比缺少了查询失效处理操作.

如果企业 A 希望同时提供出库流程、查询失效处理流程以及入库流程,则在企业 A 内部将最初服务流程与经过替换后得到的服务流程进行合并即可,即,执行一次并集演化运算.如图 6 所示,服务网 S_1 提供出库和查询失效处理两个子流程,服务网 S_3' 提供出库和入库两个子流程,执行并集演化 $S_4'=\Pi(S_1,S_3')$,S_4' 则同时具备了出库、查询失效处理以及入库这 3 个子流程.

同样可知,交集演化运算 $S_5'=\Lambda(S_1,S_3')$ 和补集演化运算 $S_6'=\upsilon(S_3',S_5')$ 的结果分别如图 7(a)和图 7(b)所示.

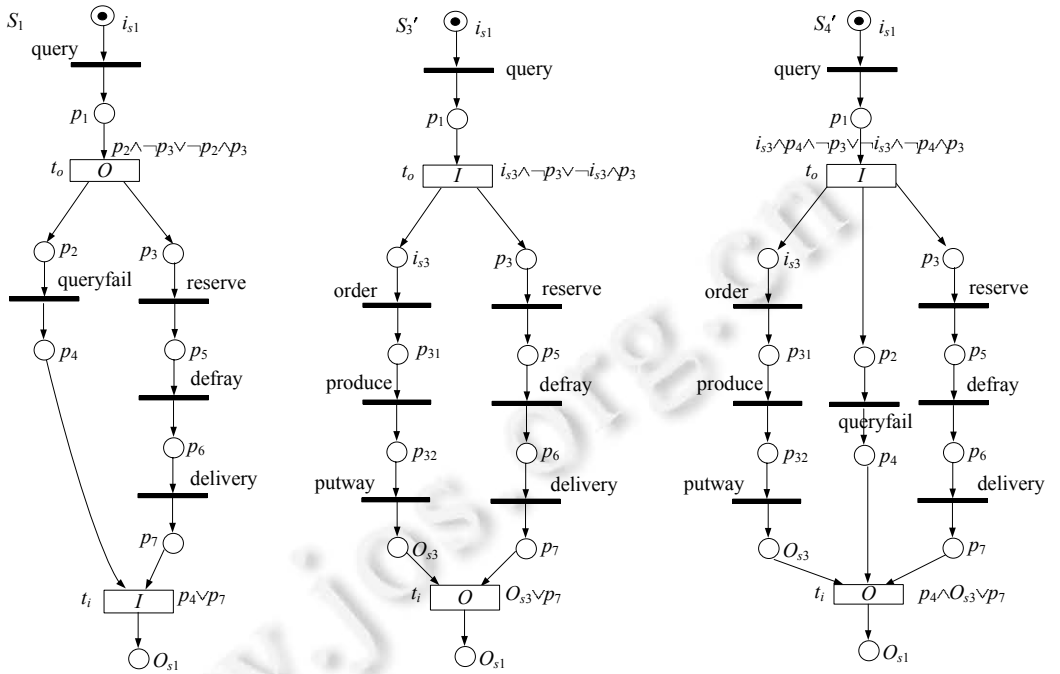

Fig.6 Evolution operations for union

图 6 并集演化运算

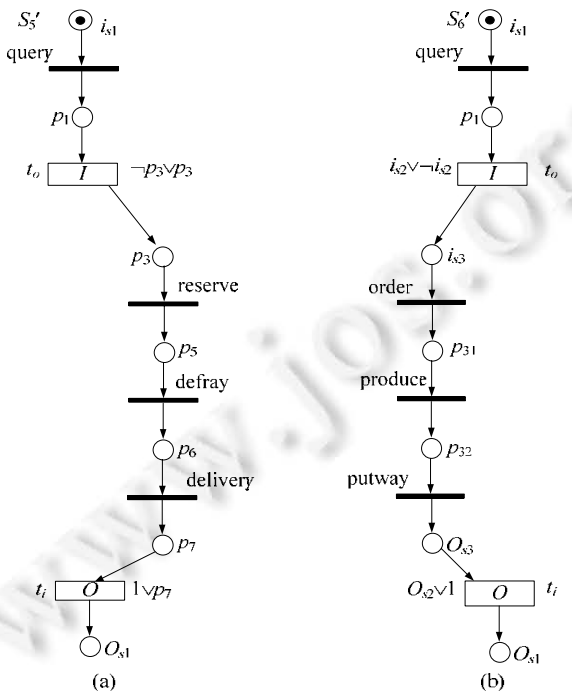

Fig.7 Evolution operations for intersection and complement

图 7 交集和补集演化运算

3.2.5 合成演化运算

合成演化用于将两个具有公共数据接口的服务流程合并成为一个交互服务流程,从而完成两个服务流程之间的数据传递,二者协同完成一个新的服务功能.

定义 16(协同数据库所). 服务网 SN_1 和 SN_2 的数据库所 p_1 和 p_2 称为协同数据库所,当且仅当库所 p_1 发送的数据可以被数据库所 p_2 接受,记为 $p_1 \Diamond p_2$,其中,发送型库所 p_1 记为 $s(p_1)$,接受型库所 p_2 记为 $r(p_2)$.

定义 17(合成演化). Σ定义为合成演化操作算子,$SN_i=(P_i,T_i,F_i,LI_i,LO_i,i_i,o_i,L_i)$为两个服务网($i$=1,2),若存在 $Pd'_1 \subseteq Pd_1$ 以及 $Pd'_2 \subseteq Pd_2$ 使得 $Pd'_1 \Diamond Pd'_2$,令 $Pd' = Pd'_1 \cap Pd'_2$,$SN'=\Sigma(SN_1,SN_2)$,则 $SN'=(P',T',F',LI',LO',i',o',L')$,其中,

(1) $P'=Pd \cup Pc \cup \{i,o\}$,$Pd=Pd_1 \cup Pd_2 - Pd'$,$Pc=Pc_1 \cup Pc_2 \cup Pd'$;

(2) $T'=T_1 \cup T_2 \cup \{t'_o,t'_i\}$;

(3) $F'=F_1 \cup F_2 \cup \{i \times t'_o \times \{i_1,i_2\}\} \cup \{\{o_1,o_2\} \times t'_i \times o\}$;

(4) $I'(t)=\begin{cases} I(t), & \forall t \in T_{I1} \cup T_{I2} \\ o_1 \wedge o_2, & t=t'_i \end{cases}$,$O'(t)=\begin{cases} O(t), & \forall t \in T_{o1} \cup T_{o2} \\ i_1 \wedge i_2, & t=t'_o \end{cases}$;

(5) $L'=L_1 \cup L_2$.

例如,在图 8 中构建了服务网 S_3 和 S_4,其中:S_3 是由服务网 S_1 的中的出库流程组成;服务网 S_4 则为一个客户代理服务程序,主要完成用户登录、预定、收货和支付功能.现利用合成演化运算,将 S_3 与 S_4 合成为一个服务网,二者具备公共数据交互接口集合$\{(p_{d11},p_{d21}),(p_{d12},p_{d22}),(p_{d13},p_{d23})\}$,三组公共交互接口分别表示为 p_{d1},p_{d2} 和 p_{d3}.根据合成演化运算的定义,执行 $S'_7=\Sigma(S_3,S_4)$后,得到的服务网 S'_7 参见图 8.在上述演化运算中,$\vartheta(S_3)$=4 且 $\vartheta(S_4)$=0,但执行合成演化运算后,得到的服务网 S'_7 只满足 $\vartheta(S'_7)=0$.引发结构范式变化的原因如下:出库流程 S_3 中要求先支付后发货;而在客户代理服务流程 S_4 中,则要求先收货后支付,二者在进行合成后出现了死锁交互等待,判定方法参见第 4.2 节定理 2.

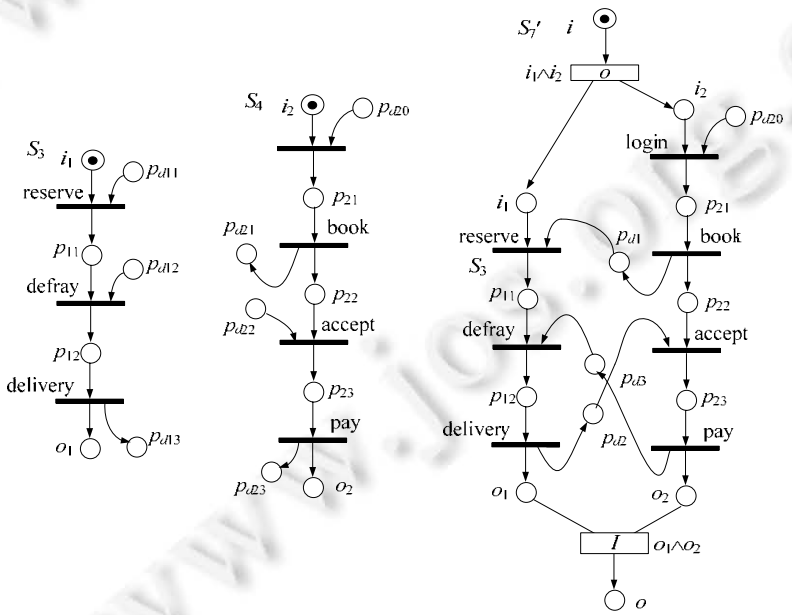

Fig.8 Evolution operations for synthesis

图 8 合成演化运算

4 服务流程的健壮性保持分析

4.1 服务网的结构范式

在文献[36]中,作者提出了用于判定组合 Web 服务流程结健壮性(即结构合理性)的结构范式概念,从流程可

达和冗余两个层面将流程结构健壮性划分为 4 个级别:第 1 范式要求服务流程中存在一条可达路径;第 2 范式要求服务流程是无阻塞的,即所有潜在的服务流程路径可达;第 3 范式要求并发或选择结构中不存在等价子服务流程;第 4 范式则要求不存在重复子流程结构.本节探讨上一节中提出的几类结构演化运算对服务流程结构范式的层次保持问题.

SN为一服务网,将删除SN中外部数据库所和同时删除内外部数控库所的得到流程网分别称为服务流程的内网与控制流网,分别用符号 SN_I 和 SN_C 表示;G_f表示 SN 的终止状态,即 G_f 为 SN 对应 $M(SN.o)=1$ 标识状态,服务网结构范式 4 个层次的形式化定义如下[36].

(1) 若 SN_I 中存在变迁序列σ,使得 $M_0[\sigma\rangle G_f$,则称 SN 满足 1NF,记为$\vartheta(SN)=1$;

(2) 若$\vartheta(SN)=1$且 SN_I 是无阻塞的,则称 SN 满足 2NF,记为$\vartheta(SN)=2$;

(3) 若$\vartheta(SN)=2$且 SN_I 中无等价子控制流网,则称 SN 满足 3NF,记为$\vartheta(SN)=3$;

(4) 若$\vartheta(SN)=3$且 SN_I 中无重复子控制流网,则称 SN 满足 4NF,记为$\vartheta(SN)=4$.

上述定义详见文献[36],限于篇幅不再赘述.其中,$\vartheta(SN)$表示服务网 SN 的结构范式,在 SN 内部不存在从 $SN.o$ 至 $SN.i$ 的可达路径时,此时,SN 的范式约定为 0,表示为$\vartheta(SN)=0$.引入符号$\perp$表示流程结构范式层次不确定,即:当 SN 的范式层次值不确定时,表示为$\vartheta(SN)=\perp$.

4.2 结构范式的保性分析

服务流程的演化引发流程结构的转换,而流程结构的转换可能引发流程结构健壮性的变化.因此,研究建立的演化运算对流程结构范式的影响对提前预知和判定演化后的流程结构健壮性具有重要意义,本节分析和建立文中 7 类演化运算对不同层次流程结构范式保持问题.

定理 1. χ,θ,ξ,Π演化运算可以保持服务网的 1NF.

证明:

(1) $\vartheta(SN)=1,T^*\subseteq T$,现证明$\vartheta(\chi(SN,T^*))=1$.

由于χ演化运算规定 T_χ中必须存在有效连通路径,假设该路径对应的变迁序列为σ,令 $SN'=\chi(S,T^*)$,则 $SN'.M_0[\sigma\rangle SN'.G_f$,所以$\vartheta(\chi(SN,T^*))=1$.

(2) $\vartheta(SN_1)=1$且$\vartheta(SN_2)=1$,现证明$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=1$.

由于$\vartheta(SN_1)=1$,所以$\exists\sigma_1$使得 $SN_1.M_0[\sigma_1\rangle SN_1.G_f$;同理,$\exists\sigma_2$使得 $SN_2.M_0[\sigma_2\rangle SN_2.G_f$.

令$\sigma=\sigma_1,SN'=\theta(SN_1,SN_2,p_m,p_n,\gamma)$,则 $SN'.M_0[\sigma=SN_1.M_0[\sigma\rangle SN_1.G_f=SN'.G_f$,所以$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=1$.

(3) $\vartheta(SN_i)=1,i=\{1,2,3\},SN_2\text{«}SN_1$,且 $InData(SN_3.i)\subseteq InData(SN_2.i)\wedge OutData(SN_2.o)\subseteq OutData(SN_3.o)$,现证明$\vartheta(\xi(SN_1,SN_2,SN_3))=1$.

由于$\vartheta(SN_i)=1$,所以$\exists\sigma_i$使得 $SN_i.M_0[\sigma_i\rangle SN_i.G_f,i=\{1,2,3\}$,令 $SN'=\xi(SN_1,SN_2,SN_3)$.

若$\sigma_2\subseteq\sigma_1$,令$\sigma_3$替换$\sigma_2$得到变迁序列$\sigma'$,则 $SN'.M_0[\sigma'\rangle SN'.G_f$;若$\sigma_2\not\subset\sigma_1$,则 $SN'.M_0[\sigma_1\rangle SN'.G_f$,所以$\vartheta(\xi(SN_1,SN_2,SN_3))=1$.

(4) $\vartheta(SN_1)=1$且$\vartheta(SN_2)=1$,现证明$\vartheta(\Pi(SN_1,SN_2))=1$.

由于$\vartheta(SN_1)=1$,所以$\exists\sigma_1$使得 $SN_1.M_0[\sigma_1\rangle SN_1.G_f$;同理,$\exists\sigma_2$使得 $SN_2.M_0[\sigma_2\rangle SN_2.G_f$.

令$\sigma=\sigma_1/\sigma_2,(\Pi(SN_1,SN_2)).M_0[\sigma\rangle(\Pi(SN_1,SN_2)).G_f$,所以$\vartheta(\Pi(SN_1,SN_2))=1$. □

定义 18(路径). C 称为从节点 n_1 到节点 n_k 的一条路径,当且仅当在服务网 SN 中存在节点序列$\langle n_1,n_2,\ldots,n_k\rangle$,使得$(n_i,n_{i+1})\in F,1\leqslant i\leqslant k$.利用$\&(C)$表示路径 C 的字母表,即$\&(C)=\{n_1,n_2,\ldots,n_k\}$;符号$\prec$表示两个字符存在后继关系,在 C 中,n_j 称为 n_i 的后继,记作 $n_i\prec_C n_j$.

定理 2. 假设一个服务网中存在n条从起始库所i到终止库所o的路径,任取其中的m条路径,记为$C_1,C_2,\ldots,C_m$,若两条路径C_i与C_{i+1}之间存在内部库所p,使得$s(p)\in C_i\wedge r(p)\in C_{i+1}$,将这些库所命名为集合$P_I^i$ $(1\leqslant i<m)$,在集合P_I^1到P_I^m中任取一个库所,分别记为$p_1,p_2,\ldots,p_m$,若存在$j(1\leqslant j\leqslant m)$使得$C_j:s(p_j)\prec_{Cj}r(p_{(m+j-1)\%(m+1)})$,则$C_1,C_2,\ldots,C_m$中无环.

证明:参见文献[36]. □

定理 3. Λ,Σ,υ演化运算无法保持服务网的 1NF.

证明:

(1) 若 $\vartheta(SN_1)=1$ 且 $\vartheta(SN_2)=1$,现证明 $\vartheta(\Lambda(SN_1,SN_2))=\perp$.

因为 $\vartheta(SN_1)=1$,所以$\exists\sigma_1$ 使得 $SN_1.M_0[\sigma_1\rangle SN_1.G_f$;同理,$\exists\sigma_2$ 使得 $SN_2.M_0[\sigma_2\rangle SN_2.G_f$.

令 $SN'=\Lambda(SN_1,SN_2)$,若 $SN'.M_0[\sigma_1\rangle SN'.G_f$或 $SN'.M_0[\sigma_2\rangle SN'.G_f$,$\vartheta(SN')=1$;否则,$\vartheta(SN')\neq 1$.所以,$\vartheta(SN')$的值无法确定,即 $\vartheta(\Lambda(SN_1,SN_2))=\perp$.

(2) 若 $\vartheta(SN_1)=1$ 且 $\vartheta(SN_2)=1$,现证明 $\vartheta(\Sigma(SN_1,SN_2))=\perp$.

因为 $\vartheta(SN_1)=1$ 且 $\vartheta(SN_2)=1$,假设 SN_1 和 SN_2 中均只存在一条可达路径,分别为 C_1 与 C_2,且 C_1 与 C_2 中存在对偶数据库所 p_1 和 p_2.由定理 2 可知:若 $r(p_1)\prec_{C1}SN(p_2)\wedge r(p_2)\prec_{C2}SN(p_1)$,则 C_1 与 C_2 存在环,路径 C_1 与 C_2 在 $\Sigma(SN_1,SN_2)$ 不可达,$\vartheta(\Sigma(SN_1,SN_2))=0$;反之,若不存在环,则 $\vartheta(\Sigma(SN_1,SN_2))=1$.因此,$\Sigma(SN_1,SN_2)=\perp$.

(3) 若 $SN_i\ll SN_3$ 且 $\vartheta(SN_i)=1(i=1,2)$,令 $SN'=\upsilon(SN_1,SN_2)$,现证明 $\vartheta(SN')=\perp$.

因为 $\vartheta(SN_1)=1$,所以$\exists\sigma_1$ 使得 $SN_1.M_0[\sigma_1\rangle SN_1.G_f$;同理,$\exists\sigma_2$ 使得 $SN_2.M_0[\sigma_2\rangle SN_2.G_f$.假设 SN_i 均只存在唯一从 $SN_i.i$ 到 $SN_i.o$ 的可达路径,若$\sigma_1=\sigma_2$,则 $\upsilon(SN_1,SN_2)$不存在从 $\upsilon(SN_1,SN_2).i$ 到 $\upsilon(SN_1,SN_2).o$ 的可达路径,$\vartheta(\upsilon(SN_1,SN_2))=0$.若 $\sigma_1\neq\sigma_2$ 且 $\sigma_2\in\upsilon(SN_1,SN_2)$,则 $\upsilon(SN_1,SN_2)$存在从 $\upsilon(SN_1,SN_2).i$ 到 $\upsilon(SN_1,SN_2).o$ 的可达路径,$\vartheta(\upsilon(SN_1,SN_2))=1$,因此 $\vartheta(\upsilon(SN_1,SN_2))=\perp$. □

定理 4. $\chi,\theta,\xi,\Pi,\Lambda,\upsilon$演化运算可以保持服务网的 2NF.

证明:

(1) $\vartheta(SN)=2,T^*\subseteq T$,现证明 $\vartheta(\chi(SN,T^*))=2$.

$\vartheta(SN)=2,T^*\subseteq T$,令 $SN'=\vartheta(\chi(SN,T^*))$,假设 $\vartheta(SN')\neq 2$,则在 SN'存在不可达路径;根据χ的定义,$SN'\ll SN$.因此 SN 中存在不可达路径,即 $\vartheta(SN)\neq 2$,这与 $\vartheta(SN)=2$ 相矛盾,所以假设不成立,$\vartheta(SN')=2$,即 $\vartheta(\chi(SN,T^*))=2$.

同理可以证明,Π,Λ,υ演化运算可以保持服务网的 2NF.

(2) $\vartheta(SN_1)=2$ 且 $\vartheta(SN_2)=2$,现证明 $\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=2$.

令 $SN=SN_1||SN_2,SN.M_0[t_i'\rangle M_1$,由于 $I(t_i')=SN_1.i\wedge SN_2.i$,所以 $M_1(SN_1.i)=1\wedge M_1(SN_2.i)=1$.

对$\forall M\in R((SN_1||SN_2).M_0)$,则 $M=M_1$ 或 $M=M'\wedge M'(SN_1.p_x)=1\wedge M'(SN_2.p_y)=1$.

由于 $\vartheta(SN_1)=2$,不论 $M=M_1$ 还是 $M=M'$,存在σ_1 使得 $M[\sigma_1\rangle M''\wedge M''(SN_1.o)=1$;又 $\vartheta(SN_2)=2$,对 M'',存在σ_2 使得 $M''[\sigma_2\rangle M'''\wedge M'''(SN_2.o)=1$;

在 SN 中,$O(t_o')=SN_1.o\wedge SN_2.o$,所以 $M'''[t_o'\rangle M_f\wedge M_f(SN.o)=1$;

令 $\sigma=\sigma_1\circ\sigma_2\circ t_o'$,对$\forall M\in R((SN_1||SN_2).M_0),M[\sigma\rangle M_f(SN.o)=SN.G_f$,所以 $\vartheta(SN_1||SN_2)=2$.

同理可证明 $\vartheta(SN_1\otimes SN_2)=2$.

将 SN_1 中以 p_m 和 p_n 为起始和终止库所的子服务网定义为 SN',则 $\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=\Pi(SN_1,(SN'\gamma SN_2))$,$\gamma=\{||,\otimes\}$;而$\Pi,||$以及$\otimes$均保持 2NF,所以 $\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=2$.

(3) $\vartheta(SN_i)=2,i=\{1,2,3\},SN_2\ll SN_1$,且 $InData(SN_3.i)\subseteq InData(SN_2.i)\wedge OutData(SN_2.o)\subseteq OutData(SN_3.o)$,现证明 $\vartheta(\xi(SN_1,SN_2,SN_3))=2$.

因为 $\vartheta(SN_1)=2,\forall M\in R(SN_1.M_0),\exists\sigma_1\in T^*:M[\sigma_1\rangle SN_1.G_f$.假定$\sigma_2\subset\sigma_1,\sigma_2\in SN_2.T^*$,又 $\vartheta(SN_3)=2$,所以$\exists\sigma_3\in SN_3.T^*:SN_3.M_0[\sigma\rangle SN_3.G_f$.由 ξ的定义可知:令 σ为利用 σ_3 替代 σ_1 中的 σ_2 的变迁序列,则$\forall M\in R(\xi(SN_1,SN_2,SN_3).M_0),\exists\sigma\in T^*:M[\sigma\rangle\xi(SN_1,SN_2,SN_3).G_f$,因此 $\vartheta(\xi(SN_1,SN_2,SN_3))=2$. □

定理 5. $\vartheta(SN_1)=2$ 且 $\vartheta(SN_2)=2$,则 $\vartheta(\Sigma(SN_1,SN_2))=\perp$.

证明:因为 $\vartheta(SN_i)=2$,所以 SN_i 中任意从 $SN_i.i$ 到 $SN_i.o$ 的路径是可达的($i=1,2$).假设 SN_1 和 SN_2 中的路径数目之和为 n,由定理 2 可知,在这 n 条路径中任取 m 条路径命名为 $C_1,C_2,\ldots,C_m$,若满足以下条件时,$C_1,C_2,\ldots,C_m$ 中无环:若 C_i 与 C_{i+1} 之间存在内部库所 p,使得 $SN(p)\in C_i\wedge r(p)\in C_{i+1}$,将这些库所命名为集合 P_I^i ($1\leqslant i<m$),在集合 P_I^1 到 P_I^m 中任取一个库所,分别记为 $p_1,p_2,\ldots,p_m$,则存在 $j(1\leqslant j\leqslant m)$使得 $C_j.SN(p_j)\prec_{Cj}r(p_{(m+j-1)\%(m+1)})$.

$\Sigma(SN_1,SN_2)$中任意路径之间无环,则$\Sigma(SN_1,SN_2)$所有路径的可达性可以保持,此时$\vartheta(\Sigma(SN_1,SN_2))=2$,若存在一对路径之间有环,则$\vartheta(\Sigma(SN_1,SN_2))\neq 2$,因此$\vartheta(\Sigma(SN_1,SN_2))=\perp$. □

定理 6. θ,ξ,Π,Σ演化运算无法保持服务网的 3NF 和 4NF.

证明:

(1) $\vartheta(SN_1)=3$ 且 $\vartheta(SN_2)=3$,现证明$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=\perp$;

将 SN_1 中以 p_m 和 p_n 为起始和终止库所的子服务网定义为 SN',若 SN' 与 SN_2 为等价子服务网,则$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))$存在等价子流程网,$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=2$;否则,$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=3$.所以,$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=\perp$.

(2) $\vartheta(SN_1)=4$ 且 $\vartheta(SN_2)=4$,现证明$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=\perp$;

将 SN_1 中以 p_m 和 p_n 为起始和终止库所的子服务网定义为 SN',若 SN' 与 SN_2 为重复子服务网,则$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))$存在重复子流程网,$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=3$;否则,$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=4$.所以,$\vartheta(\theta(SN_1,SN_2,p_m,p_n,\gamma))=\perp$.

(3) 同理可证,ξ,Π,Σ演化运算无法保持服务网的 3NF 和 4NF. □

定理 7. χ,Λ,υ演化运算可以保持服务网的 3NF 和 4NF.

证明:对 3NF,χ,Λ,υ这 3 种演化运算的结果均为原有服务网的子网,因此该子网均可满足 3NF;否则,原有服务网不满足 3NF,与题设矛盾.所以,χ,Λ,υ演化运算可以服务网的 3NF.

同理可证明,χ,Λ,υ演化运算可以保持服务网的 4NF. □

从上述分析可知:文中建立的 7 类结构演化运算对服务网的结构范式稳定性影响如表 1 所示,其中,√表示对应演化运算执行后可以保持相应结构范式级别,×表示演化运算执行后的结构范式级别无法保持或者不确定.

Table 1 Preservation of structure normal form and evolution operations

表 1 演化运算与服务网结构范式保持关系

	χ	θ	ξ	Π	Λ	υ	Σ
1NF	√	√	√	√	×	×	×
2NF	√	√	√	√	√	√	×
3NF	√	×	×	×	√	√	×
4NF	√	×	×	×	√	√	×

5 基于流程结构演化的服务流程定制框架

图 9 是本文提出的一种基于流程结构演化的服务流程定制框架,该框架划分为 3 个层次,分别是应用接口层、演化定制层和资源存储层.演化定制层位于该定制框架的核心层次,由定制引擎和流程模型库两部分组成.

资源储存层用于存放已有的服务资源,用户在应用接口层根据框架所提供的服务需求规约模板所规定的需求描述格式提交服务流程定制需求,该需求将被提交至演化定制层的定制引擎.演化定制层中的定制引擎主要实现以下 4 个方面的功能.

(1) 需求解析:接受来至于应用接口层的用户定制需求,并对其进行分解,将服务定制需求拆分为一组粒度合理的子服务需求;

(2) 流程匹配:实现对需求解析中分解得到的子服务需求的响应服务流程的匹配和推荐,按照指定的算法查找模型库,并根据流程相似度推荐一组可以响应子服务需求的服务流程;

(3) 结构演化:该模块依据本文第 3 节建立的结构演化运算实现子服务流程的截取与装配,将流程匹配模块中查找到的子服务流程组合为一个可满足用户定制需求的服务流程;

(4) 寻租绑定:该模块主要完成定制的服务流程模型与具体服务流程实例之间的映射与绑定.

根据前面定义的各类结构演化运算的运算规则,利用 VC++6.0 开发了如图 10 所示的仿真验证平台.通过该平台可以建立服务网,采用链表作为数据结构对流程进行存储,仿真平台能够对简单流程逻辑的服务流程演化运算进行仿真,并可以判定流程的结构范式.

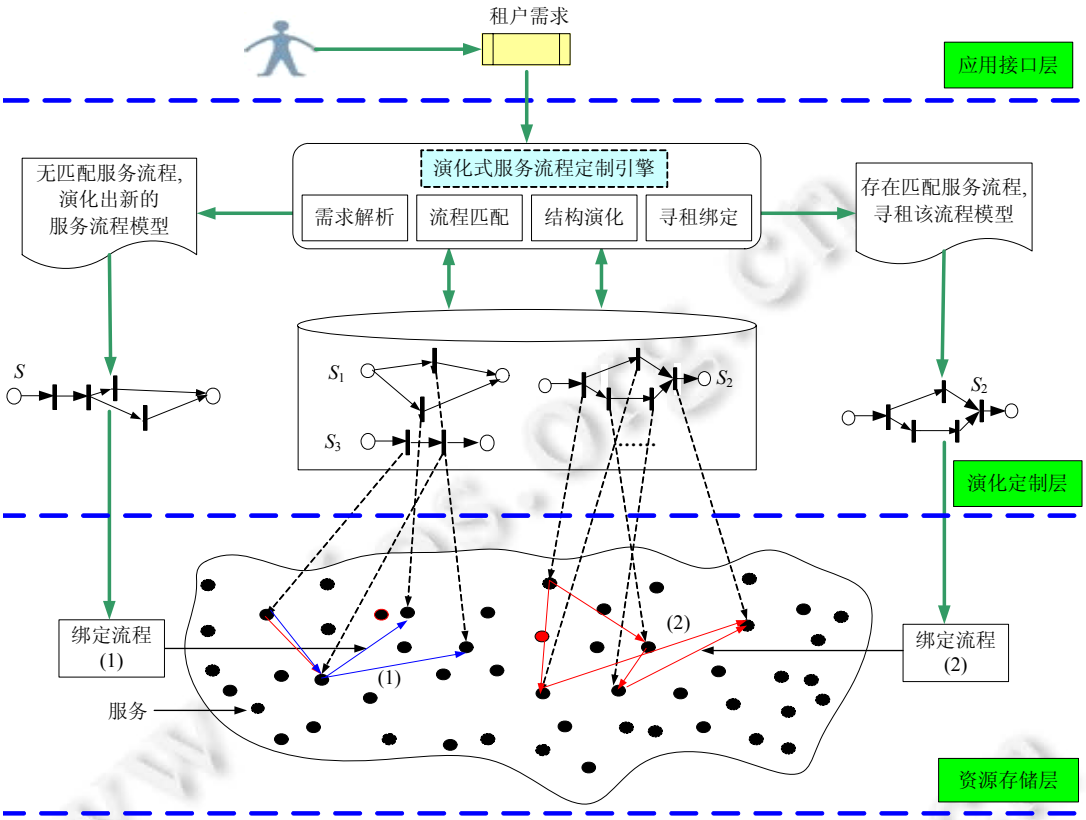

Fig.9 A framework for customization of service processes based on structure evolution

图 9 基于流程结构演化的服务流程定制框架

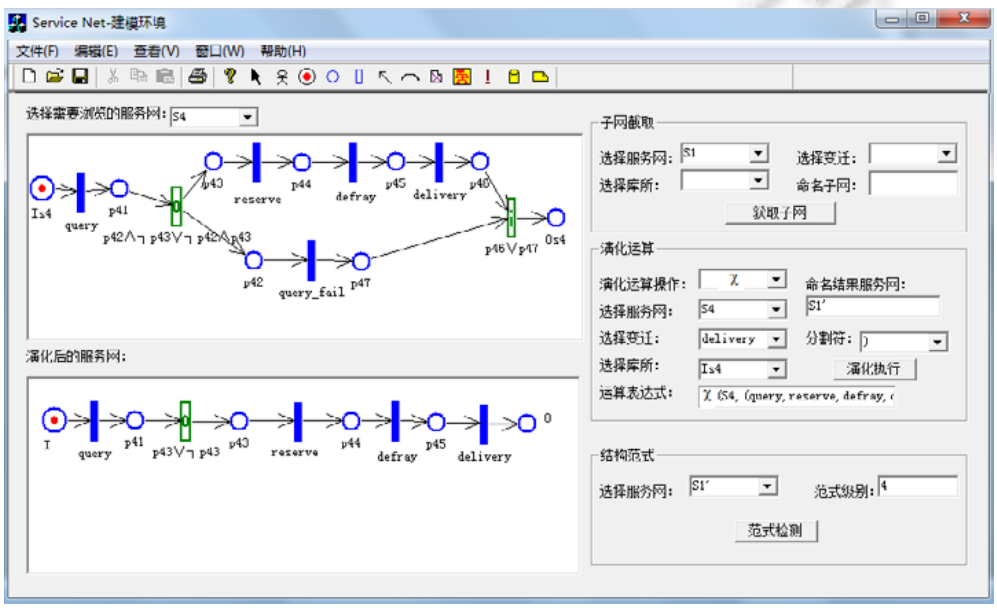

Fig.10 Simulation platform

图 10 仿真验证平台

6 结束语

服务流程的结构演化是通过对已有服务流程的分解、截取、链接以及合成等操作组建新服务流程的一种有效方法.相比现有以原子服务作为基本服务组合对象的服务组合方法,通过流程结构演化的方式构建组合服务流程,可以重用已有服务流程资源中的大粒度流程片段,从而有效地降低了服务组合的难度.

文中采用逻辑 Petri 网作为形式化建模工具,通过分析几类不同的演化需求场景,构建了基于逻辑 Petri 网的服务流程结构演化运算,提供了一种在流程模型层面描述流程结构演化的形式化方法,并在前期工作的基础上,探讨了所建立的结构演化运算对流程结构范式的保持问题.本文后续工作主要是进一步归纳和提炼演化场景,完善结构演化运算的功能,并研究所建立结构演化运算的完备性问题,同时,完善仿真验证平台的设计与开发.

References:

[1] Puthal D, Sahoo BPS, Mishra S, Swain S. Cloud computing features, issues, and challenges: A big picture. In: Proc. of the Int'l Conf. on Computational Intelligence and Networks. Odisha: IEEE, 2015. 116−123. [doi: 10.1109/CINE.2015.31]

[2] Wortmann F, Flüchter K. Internet of things. Business & Information Systems Engineering, 2015,57(3):221−224. [doi: 10.1007/s12599-015-0383-3]

[3] Borowik G, Woźniak M, Fornaia A, *et al*. A software architecture assisting workflow executions on cloud resources. Int'l Journal of Electronics and Telecommunications, 2015,61(1):17−23. [doi: 10.1515/eletel-2015-0002]

[4] Patti E, Syrri ALA, Jahn M, Mancarella P, Acquaviva A, Macii E. Distributed software infrastructure for general purpose services in smart grid. IEEE Trans. on Smart Grid, 2016,7(2):1156−1163.

[5] Bertolino A, Blake MB, Mehra P, Mehra P, Mei H, Xie T. Software engineering for Internet computing: Internetware and beyond. IEEE Software, 2015,32(1):35−37.

[6] Lemos AL, Daniel F, Benatallah B. Web service composition: A survey of techniques and tools. ACM Computing Surveys, 2016, 48(3):33−73.

[7] Wang HY, Li SR. Service substitution method based on composition context. Journal on Communications, 2014,35(9):57−66 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-436x.2014.09.006]

[8] Liu Y, Zhang YC, Zhang B, Zhang MW, Zhu ZL. Analysis of service replaceability on behavior effect. Journal of Computer Research and Development, 2015,47(8):1442−1449 (in Chinese with English abstract).

[9] Yahyaoui H, Maamar Z, Lim E, Thiran P. Towards a community-based, social network-driven framework for Web services management. Future Generation Computer Systems, 2013,29(6):1363−1377. [doi: 10.1016/j.future.2013.02.003]

[10] Saboohi H, Amini A, Abolhassani H. Failure recovery of composite semantic web services using subgraph replacement. In: Proc. of the 5th Int'l Conf. on Secure Software Integration and Reliability Improvement. Jeju Island, 2011. 182−188.

[11] Song W, Ma XX, Lv J. Instance migration in dynamic evolution of Web service compositions. Chinese Journal of Computers, 2009, 32(9):1816−1831 (in Chinese with English abstract). [doi: 10.3724/sp.j.1016.2009.01816]

[12] Goswami A, Patel RP. Service migration in cluster based cloud computing environment. In: Proc. of the Int'l Conf. on Information Processing. Maharashtra: IEEE, 2015. 468−471.

[13] Fokaefs M, Mikhaiel R, Tsantalis N, Stroulia E. An empirical study on Web service evolution. In: Proc. of the IEEE 18th Int'l Conf. on Web Services. Washington: IEEE, 2011. 49−56.

[14] Romano D, Pinzger M. Analyzing the evolution of Web services using fine-grained changes. In: Proc. of the IEEE 19th Int'l Conf. on Web Services. Hawaii: IEEE, 2012. 392−399.

[15] Andrikopoulos V, Benbernou S, Papazoglou MP. On the evolution of services. IEEE Trans. on Software Engineering, 2012,38(3): 609−628.

[16] Wang HM, Shi PC, Ding B, Yin G, Shi DX. Online evolution of software services. Chinese Journal of Computer, 2011,34(2): 318−328 (in Chinese with English abstract). [doi: 10.3724/sp.j.1016.2011.00318]

[17] Seinturier L, Merle P, Rouvoy R, Romero D, Schiavoni V, Stefani JB. A component-based middleware platform for reconfigurable service-oriented architectures. Software: Practice and Experience, 2012,42(5):559−583.

[18] Wei L, Li YL, Zhao QY, Shu HP. Dynamic changing model of workflow process based on adaptive component. Computer Integrated Manufacturing Systems, 2010,16(12):2603−2610 (in Chinese with English abstract).

[19] Zeng J, Sun HL, Liu XD, Deng T, Huai JP. Dynamic evolution mechanism for trustworthy software based on service compositon. Ruan Jian Xue Bao/Journal of Software, 2010,21(2):261−276 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3735.htm [doi: 10.3724/SP.J.1001.2010.03735]

[20] Wang S, Huang L, Hsu CH, Yang FC. Collaboration reputation for trustworthy Web service selection in social networks. Journal of Computer and System Sciences, 2016,82(1):130−143.

[21] Song M, Wei ZX, Yin GS. Evolution analysis of data flow oriented internetware service. Ruan Jian Xue Bao/Journal of Software, 2013,24(12):2797−2813 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4396.htm [doi: 10.3724/SP.J.1001.2013.04396]

[22] Ryu SH, Casati F, Skogsrud H, Benatallah B, Paul RS. Supporting the dynamic evolution of Web service protocols in service-oriented architectures. ACM Trans. on the Web, 2008,2(2):1−43.

[23] He J. Study on the evolution for SaaS service driven by requirement [MS. Thesis]. Kunming: Yunnan University, 2013. 78−89 (in Chinese).

[24] Deng SG, Huang LT, Tan W, Wu ZH. Top-*k* automatic service composition: A parallel method for large-scale service sets. IEEE Trans. on Automation Science and Engineering, 2014,11(3):891−905.

[25] Hamadi R, Benatallath B. A Petri net-based model for Web service composition. In: Proc. of the 14th Australasian Database Conf. Adelaide, 2003. 191−200.

[26] da Silva AS, Ma H, Zhang M. GraphEvol: A graph evolution technique for Web service composition. In: Proc. of the Int'l Conf. on Database and Expert Systems Applications. Springer Int'l Publishing, 2015. 134−142.

[27] Mohammad AF, Dargham J, Mcheick HT, Noor A. Software evolution as SaaS: Evolution of intelligent design in cloud. Procedia Computer Science, 2013,19:486−493.

[28] Ralph M. Using variability descriptors to describe customizable SaaS application templates. In: Proc. of the Institute of Architecture of Application Systems. 2008. 1−27.

[29] Schumm D, Dentsas D, Hahn M, Karastoyanova D, Leymann F, Sonntag M. Web service composition reuse through shared process fragment libraries. In: Proc. of the Web Engineering. Berlin, Heidelberg: Springer-Verlag, 2012. 498−501.

[30] Van der Aalst WMP. Verification of workflow nets. In: Proc. of the Int'l Conf. on Application and Theory of Petri Nets. Berlin, Heidelberg: Springer-Verlag, 1997. 407−426.

[31] van der Aalst WMP, van Hee KM, ter Hofstede AHM, Sidorova N, Verbeek HMW, Voorhoeve M, Wynn MT. Soundness of workflow nets: Classification, decidability, and analysis. Formal Aspects of Computing, 2011,23(3):333−363. [doi: 10.1007/s00165-010-0161-4]

[32] Liu G, Jiang C. Co-NP-Hardness of the soundness problem for asymmetric-choice workflow nets. IEEE Trans. on Systems, Man, and Cybernetics: Systems, 2015,45(8):1201−1204.

[33] Clempner J. Verifying soundness of business processes: A decision process Petri nets approach. Expert Systems with Applications, 2014,41(11):5030−5040.

[34] Boucheneb H, Barkaoui K. Partial order reduction for checking soundness of time workflow nets. Information Sciences, 2014,282:261−276.

[35] Sheng QZ, Maamar Z, Yao L, *et al*. Behavior modeling and automated verification of Web services. Information Sciences, 2014,258:416−433.

[36] Hu Q, Du JW, Du YY. The structure normal form of Web service processes and its testing algorithm. Chinese Journal of Computers, 2015,38(1):178−190 (in Chinese with English abstract). [doi: 10.3724/sp.j.1016.2015.00178]

[37] Quintanilla FG, Cardin O, L'Anton A, et Szabo G, Bourne S. A Petri net-based methodology to increase flexibility in service-oriented holonic manufacturing systems. Computers in Industry, 2016,76:53−68.

[38] Mateescu R, Poizat P, Salaün G. Adaptation of service protocols using process algebra and on-the-fly reduction techniques. IEEE Trans. on Software Engineering, 2012,38(4):755−777. [doi: 10.1109/TSE.2011.62]

[39] Li YX, Yao XF, Xu C, Zhang J, Li B. Cloud manufacturing service composition modeling and Qos evaluation based on process calculus. Computer Integrated Manufacturing Systems, 2014,20(3):689−700 (in Chinese with English abstract).

[40] Wu X, Zhu H. Formalization and analysis of the REST architecture from the process algebra perspective. Future Generation Computer Systems, 2016,56:153−168.

[41] Lei LH, Duan ZH. An extended deterministic fine automata based method for the verification of composite Web service. Ruan Jian Xue Bao/Journal of Software, 2007,18(12):2980−2990 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/18/2980.htm [doi: 10.1360/jos182980]

[42] Belkhir W, Chevalier Y, Rusinowitch M. Parametrized automata simulation and application to service composition. Journal of Symbolic Computation, 2015,69:40−60. [doi: 10.1016/j.jsc.2014.09.029]

[43] Hu Q, Du YY, Yu SX. Service net algebra based on logic Petri nets. Information Sciences, 2014,268:271−289.

[44] Du YY, Qi L, Zhou MC. Analysis and application of logical Petri nets to e-commerce systems. IEEE Trans. on Systems, Man, and Cybernetics: Systems, 2014,44(4):468−481.

[45] Liu W, Du YY, Zhou MC, Yan C. Transformation of logical workflow nets. IEEE Trans. on Systems, Man, and Cybernetics: Systems, 2014,44(10):1401−1412.

附中文参考文献:

[7] 王海艳,李思瑞.基于组合上下文的服务替换方法.通信学报,2014,35(9):57−66. [doi: 10.3969/j.issn.1000-436x.2014.09.006]

[8] 刘莹,张一川,张斌,等.基于行为效果的服务可替换性分析.计算机研究与发展,2015,47(8):1442−1449.

[11] 宋巍,马晓星,吕建.Web 服务组合动态演化的实例可迁移性.计算机学报,2009,32(9):1816−1831. [doi: 10.3724/sp.j.1016.2009.01816]

[16] 王怀民,史佩昌,丁博,等.软件服务的在线演化.计算机学报,2011,34(2):318−328. [doi: 10.3724/sp.j.1016.2011.00318]

[18] 魏乐,李亚玲,赵秋云,等.基于自适应构件的工作流流程动态变更模型.计算机集成制造系统,2010,16(12):2603−2610.

[19] 曾晋,孙海龙,刘旭东,邓婷,怀进鹏.基于服务组合的可信软件动态演化机制.软件学报,2010,21(2):261−276. http://www.jos.org.cn/1000-9825/3735.htm [doi: 10.3724/SP.J.1001.2010.03735]

[21] 宋敏,韦正现,印桂生.面向数据流的网构软件服务动态演化分析.软件学报,2013,24(12):2797−2813. http://www.jos.org.cn/1000-9825/4396.htm [doi: 10.3724/SP.J.1001.2013.04396]

[23] 何俊.需求驱动的 Saas 服务演化研究[硕士学位论文].昆明:云南大学,2013.78−89.

[36] 胡强,杜军威,杜玉越.Web 服务流程的结构范式及其判定算法.计算机学报,2015,38(1):178−190. [doi: 10.3724/sp.j.1016.2015.00178]

[39] 李永湘,姚锡凡,徐川,等.基于扩展进程代数的云制造服务组合建模与 QoS 评价.计算机集成制造系统,2014,20(3):689−700.

[41] 雷丽晖,段振华.一种基于扩展有限自动机验证组合 Web 服务的方法.软件学报,2007,18(12):2980−2990. http://www.jos.org.cn/1000-9825/18/2980.htm [doi: 10.1360/jos182980]

胡强(1980－),男,山东邹城人,博士,讲师,CCF 专业会员,主要研究领域为 Petri 网理论,服务计算,软件形式化分析方法.

任志考(1969－),男,副教授,主要研究领域为软件形式化建模,流程优化.

赵振(1982－),男,博士,讲师,主要研究领域为智能生产系统,语义网.

杜军威(1974－),男,博士,教授,CCF 专业会员,主要研究领域为软件测试,形式化验证.

杜玉越(1960－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为 Petri 网理论,形式化方法,人工智能.