第43卷 第4期 西 安 交 通 大 学 学 报 Vol. 43 №4
2009年4月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Apr. 2009

面向投影时序逻辑的 Web 服务模型检测

王小兵，段振华
（西安电子科技大学计算理论与技术研究所，710071，西安）

摘要：为了满足 Web 服务的可靠性，利用投影时序逻辑的模型检测方法来验证 Web 服务. 利用投影时序逻辑的一个可执行子集对 OWL-S 进行建模，用命题投影时序逻辑来描述期望的性质. 模型 M 和性质 P 统一以投影时序逻辑来表示，通过判定 M 蕴含 P 的有效性，即判定 M 和非 P 的合取的不可满足性，亦利用 M 和非 P 的正则形可构造合取式的正则图，判定合取的不可满足性，从而达到模型检测的目的. 通过运行实例表明，所提模型检测器可验证 Web 服务系统性质，保证 Web 服务的可靠性.

关键词：形式逻辑；投影时序逻辑；Web 服务；模型检测
中图分类号：TP393 **文献标志码**：A **文章编号**：0253-987X(2009)04-0039-05

Projection Temporal Logic Oriented Model Checking for Web Services

WANG Xiaobing, DUAN Zhenhua
(Institute of Computing Theory and Technology, Xidian University, Xi′an 710071, China)

Abstract: To satisfy the reliability of web services, a model checking approach with the projection temporal logic(PTL) is adopted to verify web services. An executable subset of PTL is used to model OWL-S, and the propositional projection temporal logic(PPTL) is selected to describe desired properties. The model M and property P are both presented in PTL, then the aim of model checking is obtained by determining the validity of M implies P, namely, the unsatisfiability of the conjunction of M and not P, and the unsatisfiability of the conjunction is checked by constructing its normal form graph with the help of the normal forms of M and not P. Executing the example, it shows that the presented model checker can be used to verify the system properties and hence to ensure the reliability of web services.

Keywords: formal logic; projection temporal logic; web service; model checking

Web 服务的验证是目前研究的热点，常用的方法是自动测试和模型检测. 模型检测是一种形式化的验证方法，能够验证系统的灵活性、安全性及公平性等性质. 模型检测中性质的描述是基于时序逻辑的，Web 服务建模方式多集中于进程代数、Petri 网、自动机等，采用时序逻辑的方法才刚刚兴起[1]，较为少见.

Web 服务的过程描述模型主要有 BPEL 和 OWL-S. OWL-S 的提出，使 Web 服务成为计算机可以理解的实体，从而能够自动完成 Web 服务的发现、调用、组合等任务. 采用时序逻辑对 OWL-S 建模，而不用其他的非逻辑建模语言[2]，能够将 Web 服务的性质和模型统一在时序逻辑的框架中，有利于进行模型检测.

本文提出了一种基于投影时序逻辑(PTL)的 Web 服务模型检测方法，使用 PTL 的可执行子集 Framed Tempura 语言为 OWL-S 建立模型，使用命题投影时序逻辑 PPTL 描述性质，将 Web 服务的模

收稿日期：2008-10-22. 作者简介：王小兵(1979－)，男，讲师. 基金项目：国家自然科学基金重点资助项目(60433010)；国家自然科学基金资助项目(60873018)；武汉大学软件工程国家重点实验室开放基金资助项目(SKLSE20080713).

型检测转化为PTL的可满足性问题加以解决.

1 投影时序逻辑

设Π是命题集合,V是变量集合,项e和公式p如下

$$e::=u\mid x\mid f(e_1,\cdots,e_m)\mid \bigcirc e\mid \Theta e$$

$$p::=\pi\mid e_1=e_2\mid W(e_1,\cdots,e_m)\mid \neg p\mid p_1\wedge p_2\mid p_1\vee p_2\mid \exists x:p\mid \bigcirc p\mid \Theta p\mid (p_1,\cdots,p_m)\text{prj } q$$

式中:逻辑操作符有=、¬、∧、∨和∃;时序操作符有下一状态○,上一状态Θ和投影prj等;u和x是静态和动态变量;π是命题;f和W为函数和谓词.

定义1 状态$s=(I_v,I_p)$,$I_v:V\to D$,$I_p:\Pi\to B$. D是变量的值域,包含nil为未定义值,B是布尔集合.

定义2 区间$\sigma=<S_0,S_1,\cdots>$,长度记为$|\sigma|$,当σ无限时,$|\sigma|$为ω,否则为其状态数减1,操作符·用来将两个区间连接成一个大区间.

定义3 解释$I=(\sigma,i,k,j)$,其中i、k为整数,j为整数或ω,并且$0\leqslant i\leqslant j\leqslant|\sigma|$,其中$\leqslant$定义为$\leqslant-\{\omega,\omega\}$.

对项的解释定义为:$I[u]=s_k[u]=I_v^k[u]=I_v^i[u]$若$u$是静态变量;$I[x]=s_k[x]=I_v^k[x]$若$x$是动态变量;$I[f(e_1,\cdots,e_m)]=f(I[e_1],\cdots,I[e_m])$(若$I[e_h]\neq$nil,对$1\leqslant h\leqslant m$),否则=nil;$I[\bigcirc e]=(\sigma,i,k+1,j)[e]$(若$k<j$),否则=nil. $I[\Theta e]=(\sigma,i,k-1,j)[e]$(若$i<k$),否则=nil.

为了定义投影操作的语义,需要一个辅助操作符↓. 设$\sigma=<s_0,s_1,\cdots>$是一个区间,$r_0,\cdots,r_h$($h\geqslant 0$)是整数且有$0\leqslant r_0\leqslant\cdots\leqslant r_h\leq|\sigma|$,则$\sigma$在$r_0,\cdots,r_h$上的投影$\sigma\downarrow<r_0,\cdots,r_h>=<s_{t_0},\cdots,s_{t_l}>$,其中$t_0,\cdots,t_l$是通过删除$r_0,\cdots,r_h$中的冗余元素得到的. 例如,$<s_0,s_1,s_2,s_3>\downarrow<0,0,2,2,3>=<s_0,s_2,s_3>$.

定义4 若$I|=p$,则称解释I满足公式p,即p在I的区间σ的子区间$\sigma_{(i\cdots j)}=<s_i,\cdots,s_j>$当前状态$S_k$是满足的.

解释与公式的满足关系|=定义为:$I|=\pi$当且仅当$s_k[\pi]=$true;$I|=e_1=e_2$当且仅当$I[e_1]=I[e_2]$;$I|=W(e_1,\cdots,e_m)$当且仅当W是除等词以外的原始谓词,并且对$1\leqslant h\leqslant m$,有$I[e_h]\neq$nil和$W(I[e_1],\cdots,I[e_m])=$true;$I|=\neg p$当且仅当$I|\neq p$;$I|=p_1\wedge p_2$当且仅当$I|=p_1$并且$I|=p_2$;$I|=p_1\vee p_2$当且仅当$I|=p_1$或者$I|=p_2$;$I|=\exists x:p$当且仅当存在σ',满足$|\sigma'|=|\sigma|$,$(\sigma',i,k,j)|=p$,σ和σ'除了对x的解释可不同外,对其他变量的解释都相同;$I|=\bigcirc p$当且仅当$k<j$并且$(\sigma,i,k+1,j)|=p$;$I|=\Theta p$当且仅当$i<k$并且$(\sigma,i,k-1,j)|=p$;$I|=(p_1,\cdots,p_m)$ prj q,若存在整数$k=r_0\leqslant\cdots\leqslant r_m\leq j$,使$(\sigma,i,r_0,r_1)|=p_1$,$(\sigma,r_{l-1},r_{l-1},r_l)|=p_l$(对$1<l\leqslant m$),并且对下面的某种情况有$(\sigma',0,0,|\sigma'|)|=q$.

(1)$r_m<j$并且$\sigma'=\sigma\downarrow(r_0,\cdots,r_m)\cdot\sigma_{(r_m+1\cdots j)}$.

(2)$r_m=j$并且$\sigma'=\sigma\downarrow(r_0,\cdots,r_n)(0\leqslant h\leqslant m)$.

投影操作允许用不同的时间粒度刻画计算进程,并且是PTL的核心时序操作符,由它可定义顺序和循环等语句. 为了解释$(p_1,\cdots,p_m)$pri q,需要两个不同时间粒度的状态序列,一个是执行$p_1,\cdots,p_m$的局部序列,另一个是并行的执行q的全局序列. 直观地,q与$p_1,\cdots,p_m$在一个区间上并行执行,q的区间状态仅仅是$p_1,\cdots,p_m$各自区间的初始状态和终止状态. 投影操作允许$p_1,\cdots,p_m$和q各自独立,都有权利来定义本身的执行区间. 进程独立运行,通信依赖于共享变量,且只发生在全局序列的状态上.

常用的公式定义如下,其中区间时序逻辑[3]的核心操作符";"能够被prj所表示. empty≡¬○true,more≡○true, skip≡○empty, len(n)≡$\bigcirc^n$empty,$p_1;\cdots;p_m\equiv(p_1,\cdots,p_m)$ prj empty,$\Diamond p\equiv$true;p,$\Box p\equiv\neg\Diamond\neg p$,$p^*\equiv$empty$\vee(p;p^*)\vee p\wedge\Box$more.

2 Web服务建模与性质描述

2.1 建模语言

Framed Tempura[4]是PTL的一个可执行子集,具有完全的形式化语义,可用作OWL-S的建模语言. 时序逻辑程序在一个状态序列上执行,变量的值不是自动遗传的. 为了使变量具有惰性,可以显式或隐式的重复赋值[5],但这样程序就会变得比较复杂. 定义谓词af(x)=p_x,当变量x被赋值时为真,否则为假.

定义5 状态框架操作符lbf(x)=¬af(x)→∃b:(Θx=b∧x=b),其中b为静态变量,它表示x是状态框架的,即若x在当前状态未赋值,则它继承上一状态的值.

定义6 区间框架操作符frame(x)=□(more→○lbf(x)),它表示x是区间框架的,即x在区间所有状态上都是框架的,frame($x_1,\cdots,x_n$)用于声明多个区间框架变量.

Framed Tempura中的表达式对应于PTL的

项，语句对应于 PTL 的公式，执行程序即是为 PTL 公式寻找一个区间，使公式在此区间的相应状态上能被满足. 在如下基本语句中，x 是变量，e 是表达式，b 是布尔表达式，s(可带下标)是语句.

(1)空语句：empty.

(2)基本赋值语句：$x=e$，$x:=e\equiv \text{skip}\wedge \bigcirc x=e$.

(3)区间框架语句：frame(x).

(4)存在语句：$\exists x:s$.

(5)next 语句：$\bigcirc s$.

(6)always 语句：$\Box s$.

(7)投影语句：$(s_1,\cdots,s_m)\text{prj } s$.

(8)顺序语句：$s_1;s_2\equiv(s_1,s_2)$ prj empty.

(9)选择语句：$s_1\vee s_2$.

(10)无序语句：$s_1\otimes s_2\equiv(s_1;s_2)\vee(s_2;s_1)$.

(11)合取语句：$s_1\wedge s_2$.

(12)并行语句：

$$s_1\parallel s_2\equiv s_1\wedge(s_2;\text{true})\vee s_2\wedge(s_1;\text{true}).$$

(13)条件语句：

if b then s_1 else $s_2\equiv(b\rightarrow s_1)\wedge(\neg b\rightarrow s_2)$.

(14)while 语句：

$$\text{while } b \text{ do } s\equiv(b\wedge s)^*\wedge\Box(\text{empty}\rightarrow\neg b).$$

(15)until 语句：

repeat s until $b\equiv s$; while $\neg b$ do s.

(16)等待语句：

await(b)$\equiv$frame($x_1,\cdots,x_h$)$\wedge\Box$(empty$\leftrightarrow b$)，其中 $x_1,\cdots,x_h$ 是 b 中的变量.

表达式和程序的解释与 PTL 中项和公式的解释一样. 操作符的优先级从高到低依次为$\neg$，$\exists$，$\bigcirc$，Θ，$\Box$，$=$，$:=$，$\vee$，$\parallel$，$\vee$，$\otimes$，$\parallel$，$\rightarrow$，$\leftrightarrow$，prj，；. empty 表示程序在当前状态终止. 执行基本赋值语句首先要解释 e，然后将 x 解释为 e 值，使得公式 $x=e$ 为真. frame(x)表示 x 是区间框架的. 存在语句中的变量 x 只在 s 中有效，即是一个局部变量. next 语句表示语句在下一状态执行，而 always 语句表示语句在每个状态都要执行. 基于 prj 的投影语句是可执行的，常见的顺序、条件、while 和 until 语句都被定义成一个可执行的时序逻辑公式. 选择语句用于构造非确定性程序，无序语句说明进程可以任意顺序执行. 合取与并行语句比较类似，都可用于描述并发程序，但后者允许各进程定义自己的区间. 一个进程可以包含等待语句 await(b)，表达式 b 中包含若干变量，当其他进程改变这些变量使得表达式为真时，该进程才跳过等待语句继续执行，因此等待语句可以实现进程间的同步.

OWL-S 使用 IOPE 来刻画 Web 服务的功能，即 Web 服务的输入参数、输出参数、正确执行的前提条件以及执行后产生的结果. OWL-S 把 Web 服务模型区分为原子进程、简单进程和组合进程，分别对应于 Web 服务、抽象服务和组合服务. 原子进程可以通过 Web 服务消息进行直接的交互，可一步完成. 简单进程不可直接调用，用作原子进程的视图，也可作为组合进程的简化表示. 组合进程可以通过一组控制结构在原子进程或组合进程的基础上复合而成.

输入参数的约束和前提条件可视为原子进程执行前的条件，输出参数的约束和执行结果相当于原子进程执行后的条件[6]. 因此，原子进程的 IOPE 能够以 Framed Tempura 程序的形式清晰地进行表达[1]. 原子进程的描述为 $I\wedge C\rightarrow O\wedge E$，其中：$I$ 是所有输入参数描述公式的合取，若原子进程没有输入则为 true；C 是原子进程执行的所有前提条件公式的合取，若没有条件则为 true；O 是所有输出参数描述公式的合取，若原子进程没有输出则为 true；E 是原子进程执行后结果条件的合取，若没有条件则为 true. 建立在 IOPE 基础上的 Web 服务语义存在一定的局限性，具有相同输入、输出的两个服务很有可能代表完全不同的任务，难以准确地反映 Web 服务的能力[7]. 因此，如果使用 Framed Tempura 对进程的内部行为进行建模，就能够验证 Web 服务内部的性质，从而更精确地判断 Web 服务是否符合系统的需求.

下面，给出 OWL-S 中的各种控制结构及其对应的建模语句.

(1)Sequence 结构：子进程按照顺序依次执行，由顺序语句建模.

(2)Split 结构：子进程并发执行，由并行语句建模. 与文献[1]中采用交错操作符表达的交叠式语义不同，这里的并发具有真并发语义.

(3)Split+Join 结构：子进程并发执行，等所有子进程执行后该结构才完成. 该结构由并行和等待语句建模，前者实现进程的并发，后者实现进程的同步.

(4)Any-Order 结构：子进程按照任意顺序执行，但不能同时发生. 两个进程的 Any-Order 结构可直接由无序语句建模，类似地，3 个及以上进程的 Any-Order 结构由顺序和选择语句建模.

(5)Choice 结构：任意选择一个子进程执行，由选择语句建模.

(6)If-Then-Else 结构:根据条件的真假决定选择一个子进程执行,由条件语句建模.

(7)Repeat-While 结构:先判断条件,为假则退出,为真则执行子进程,再循环. 该结构由 while 语句建模.

(8)Repeat-Until 结构:先执行子进程,再判断条件,为真则退出,为假则循环. 该结构由 until 语句建模.

2.2 性质描述语言

Web 服务的性质需要一种性质描述语言来表达,这可以使用命题投影时序逻辑 PPTL. PPTL 是 PTL 的命题形式,不包含变量、谓词和量词等一阶成分,其语法定义为

$$p ::= \pi \mid \neg p \mid p_1 \wedge p_2 \mid p_1 \vee p_2 \mid \bigcirc p \mid \Theta p \mid (p_1, \cdots, p_m) \text{ prj } q$$

PPTL 的语义与 PTL 的类似,每个状态对 Π 中的命题进行解释,但不包括对变量的解释. 特别地,PPTL 存在一个可判定的算法[8],为 Web 服务的模型检测提供了理论基础.

SPIN 的性质规范语言是命题线性时序逻辑 PLTL,其表达能力有限,不能表达所有的正则表达式,使得一些性质,如"命题 p 必须在每个偶数状态成立"的验证无法自动完成[8]. 反之,PPTL 则可以表达所有的正则表达式,用它作为 Web 服务的性质描述语言,与 PLTL 相比较[2],能描述更多的性质,从而确保 Web 服务满足系统的需求.

3 模型检测算法

基于 PTL 的 Web 服务模型检测的基本思想如下:先用 Framed Tempura 对 OWL-S 建模,得到程序 M 用于模拟 Web 服务,再用 PPTL 描述需要验证的性质 P,最后判定 $M \rightarrow P$ 是否有效,这可以转化为 $M \wedge \neg P$ 的可满足性问题. 如果 $M \wedge \neg P$ 不可满足,则 $M \rightarrow P \equiv \neg(M \wedge \neg P)$ 始终为真,模型检测成功;如果 $M \wedge \neg P$ 可满足,则 M 就违反了性质 P.

Web 服务的模型和性质都统一表达在 PTL 的框架内,因此公式的形式很复杂,研究它们的性质较为困难. 为了简化研究,可将 Framed Tempura 程序和 PPTL 公式转化为等价的标准形式,称为正则形,定义分别如下.

定义 7 Framed Tempura 程序的正则形为

$$\bigvee_{i=1}^{k} (q_{ei} \wedge \text{empty}) \vee \bigvee_{j=1}^{h} (q_{cj} \wedge \bigcirc q_{fj})$$

式中:$k, h \geqslant 0 (k+h \geqslant 1)$;对 $1 \leqslant j \leqslant h$,$q_{fj}$ 是一般程序(不要求是正则形);q_{ei} 和 q_{cj} 为 true,或形为 $(x_1 = e_1) \wedge \cdots \wedge (x_m = e_m) \wedge \dot{p}_{x1} \wedge \cdots \wedge \dot{p}_{xm}$,$e_1, \cdots, e_m$ 属于 D,$\dot{p}_x$ 代表 p_x 或 $\neg p_x$.

定义 8 PPTL 公式的正则形为

$$\bigvee_{i=1}^{k} (q_{ei} \wedge \text{empty}) \vee \bigvee_{j=1}^{h} (q_{cj} \wedge \bigcirc q_{fj})$$

式中:$k, h \geqslant 0 (k+h \geqslant 1)$;对 $1 \leqslant j \leqslant h$,$q_{fj}$ 是一般 PPTL 公式(不要求是正则形);q_{ei} 和 q_{cj} 为 true,或形为 $\dot{\pi}_1 \wedge \cdots \wedge \dot{\pi}_m$,$\dot{\pi}$ 代表 π 或 $\neg\pi$. 特别地,如果 $\bigvee_{j=1}^{h} q_{cj} \equiv \text{true}$,且 $\bigvee_{i \neq j} (q_{ci} \wedge q_{cj}) \equiv \text{false}$,则该正则形称为完全正则形.

完全正则形用于求解一个 PPTL 公式的否定形式[8]. 若公式 q 的完全正则形如上所示,那么 $\neg q$ 为 $\bigvee_{i=1}^{k} (\neg q_{ei} \wedge \text{empty}) \vee \bigvee_{j=1}^{h} (q_{cj} \wedge \bigcirc \neg q_{fj})$. 根据 Web 服务模型检测的基本思想,性质 P 需要转化为 $\neg P$,因此完全正则形具有重要的作用.

已经证明存在算法,可将任意 Framed Tempura 程序转化为正则形[4],将任意 PPTL 公式转化为完全正则形[8],在此不再赘述. 据正则形可以构造出 Framed Tempura 程序或 PPTL 公式的模型,称为正则图,其定义如下.

定义 9 p 是一个 Framed Tempura 程序或 PPTL 公式,则其正则图是有向图 $G(p) = (C_L(p), E_L(p))$. $C_L(p)$ 是结点集合,每个结点是一个 Framed Tempura 程序或 PPTL 公式. $E_L(p)$ 是有向边集合,包含三元组 (q, p_e, r),表示从结点 q 到结点 r 有一条标记为 p_e 的边,其中 p_e 为 Framed Tempura 程序或 PPTL 公式. 正则图 $G(p)$ 定义如下:

(1)p 在 $C_L(p)$ 中;

(2)对于 $C_L(p)$-{empty, false} 中任意 q,若其正则形为 $\bigvee_{i=1}^{k} (q_{ei} \wedge \text{empty}) \vee \bigvee_{j=1}^{h} (q_{cj} \wedge \bigcirc q_{fj})$,则 empty 在 $C_L(p)$ 中,对于所有 i 有边 $(q, q_{ei}, \text{empty})$ 在 $E_L(p)$ 中,对于所有 j 有 q_{fj} 在 $C_L(p)$ 中,且有边 (q, q_{cj}, q_{fj}) 在 $E_L(p)$ 中;

(3)有限次应用(1)和(2)生成的才是正则图.

在构造正则图的过程中,从根结点代表的公式出发,不断地求解正则形,生成新的边和结点,如果边代表的公式是不可满足的,则该路径无法继续延展到 empty 结点,它就不是程序的可能执行路径,需要将该路径上的结点删除掉. 如果一个程序不能执行,即它代表的 PTL 公式不可满足,不存在从根结点到 empty 结点的路径,除根结点外的所有结点

都将删除,那么正则图最终只包含根结点. 如果程序能执行,那么在正则图中,从根结点出发,一定可以到达 empty 结点.

给定 Web 服务模型 M 和待验证的性质 P,模型检测被转化为判定公式 $M\wedge\neg P$ 的不可满足性,这需要构造它的正则图.

算法 1:构造 $M\wedge\neg P$ 的正则图.

输入:Framed Tempura 程序 M 和 PPTL 公式 p.

输出:正则图 $G=\{C_L,E_L\}$.

(1)置结点集 C_L 为$\{M\wedge\neg P\}$,且该根结点加未处理标记,置边集 E_L 为空.

(2)若 C_L 中结点处理完毕,转向(4);否则任取一未处理结点 $r\wedge\neg\varphi$ 做已处理标记,计算 r 的正则形为 $\bigvee_{i=1}^{k}(r_{ei}\wedge\text{empty})\vee\bigvee_{j=1}^{h}(r_{cj}\wedge\text{O}r_{fj})$,计算$\neg\varphi$ 的正则形为 $\bigvee_{m=1}^{s}(\neg\varphi_{em}\wedge\text{empty})\vee\bigvee_{n=1}^{t}(\varphi_{cn}\wedge\text{O}\neg\varphi_{fn})$.

(3)上述两个正则形的合取式记为 Q,根据 Q 的 3 种不同形式执行如下不同的操作,完成后转向(2).

①Q 如 $\bigvee_{i=1}^{k}\bigvee_{m=1}^{s}(r_{ei}\wedge\neg\varphi_{em}\wedge\text{empty})$,则将结点 empty 加入 C_L,对所有 i 和 m,将边$(r\wedge\neg\varphi,r_{ei}\wedge\neg\varphi_{em},\text{empty})$加入 E_L;②Q 如 $\bigvee_{j=1}^{h}\bigvee_{n=1}^{t}(r_{cj}\wedge\varphi_{cn}\wedge\text{O}(r_{fj}\wedge\neg\varphi_{fn}))$,对所有 j 和 n,若结点 $r_{fj}\wedge\neg\varphi_{fn}$ 不在 C_L 中,则加入该结点并做未处理标记,将边$(r\wedge\neg\varphi,r_{cj}\wedge\varphi_{cn},r_{fj}\wedge\neg\varphi_{fn})$加入 E_L;③Q 如 $\bigvee_{i=1}^{k}\bigvee_{m=1}^{s}(r_{ei}\wedge\neg\varphi_{em}\wedge\text{empty})\vee\bigvee_{j=1}^{h}\bigvee_{n=1}^{t}(r_{cj}\wedge\varphi_{cn}\wedge\text{O}(r_{fj}\wedge\neg\varphi_{fn}))$,则执行①和②的操作.

(4)从根结点出发,将无法到达 empty 结点的路径上的结点删除.

如果 $M\wedge\neg P$ 的正则图只有一个根结点,没有任何路径存在,则 $M\wedge\neg P$ 不可满足,$M\rightarrow P$ 始终为真,结论为 Web 服务满足待验证的性质;如果 $M\wedge\neg P$ 的正则图含有路径,则 $M\wedge\neg P$ 可满足,结论为 Web 服务不满足待验证的性质.

4　实　例

目前,我们已实现了一个基于 PTL 的模型检测器原型. 下面通过一个购书比价的实例来说明 Web 服务模型检测的基本过程.

Web 服务 FindCheaperBookService 是组合进程,由 3 个原子进程 FindBookInfo、FindAmazonPrice、FindHudongPrice 和 1 个组合进程 ComparePrices 复合而成. 首先执行 FindBookInfo,根据书名查询返回书号,然后并发执行 FindAmazonPrice 和 FindHudong Price,根据书号在亚马逊和互动出版网查询并返回图书价格,最后执行 ComparePrices 比较图书价格,挑选价格较便宜的书店. 组合进程 ComparePrices 由 2 个原子进程 ProduceAmazonPrice 和 ProduceHudong Price 复合而成,如果亚马逊的图书价格便宜,则执行前者显示亚马逊的名称和价格,否则执行后者显示互动出版网的名称和价格.

Web 服务 FindCheaperBookService 的 PPTL 性质描述和 Framed Tempura 程序模型如图 1 所

```
</ //使用PPTL描述性质
   define f: findB=1; define p: findA=1; define q: findH=1;
   define r: amazonPrice<hudongPrice;
   define s: Price=amazonPrice;
   always (empty -> (f and p and q and r -> s))
/>
//对进程FindCheaperBookService建模
frame (amazonStore,hudongStore,bookInfo,bookIndex,
amazonPrice,hudongPrice,findB,findA,findH,Price)
and findB=0 and findA=0 and findH=0 and amazonPrice=0
and hudongPrice=0 and Price=0
 and (                    //初始化书籍信息、书店信息
bookInfo = [["Chinese", "01"], ["English", "02"],
["Math", "03"], ["French", "04"]] and
amazonStore = [["01", 20], ["02", 24], ["03", 15]] and
hudongStore = [["02", 25], ["03", 16], ["04", 19]] and empty;
//进程FindBookInfo: 通过书名寻找书号
i = 0 and while (i < 4 and findB = 0){
   if (bookInfo[i, 0] = "English") then
   { findB := 1 and i := 0 and bookIndex := bookInfo[i, 1]}
    else{findB := 0 and i := i + 1}};
if(findB = 1) then{( //进程FindAmazonPrice
   i = 0 and while(i < 3 and findA = 0){
      if(amazonStore[i, 0] = bookIndex)then{ findA := 1 and
         i := 0 and amazonPrice := amazonStore[i, 1]}
      else{findA := 0 and i := i + 1}}
|| //进程FindHudongPrice
j = 0 and while (j < 3 and findH = 0){
   if(hudongStore[j, 0] = bookIndex) then{findH := 1 and
      j := 0 and hudongPrice := hudongStore[j, 1]}
   else{findH := 0 and j := j + 1}});
//进程ComparePrices
if (amazonPrice < hudongPrice) then{
   //进程ProduceAmazonPrice
   output ("AmazonStore") and Price = amazonPrice and skip;
   output ("Price : ", amazonPrice) and empty}
else{ //进程ProduceHudongPrice
   output ("HudongStore") and Price = hudongPrice and skip;
   output ("Price : ",hudongPrice) and empty}}
else{ output ("No book!") and empty } ).
```

图 1　程序模型

示.性质中的命题 f 为真表示书籍“*English*”存在,p 为真表示亚马逊有该书,q 为真表示互动出版网有该书,r 为真表示亚马逊该书价格低于互动出版网的价格,s 为真表示最终价格为亚马逊该书的价格. *Web* 服务期望的一条性质为 *always*(*empty*->(f *and* p *and* q *and* r->s)),即当 *Web* 服务终止时,如果书籍“*English*”存在,亚马逊和互动出版网都有该书,并且前者的售价较低,那么最终输出价格是亚马逊的价格.

模型检测运行该实例后,得到的正则图只包含一个根结点,所以 *Web* 服务满足待验证的性质.

5 结 论

本文提出了一种基于投影时序逻辑的 Web 服务模型检测方法.该方法将 Web 服务的模型和性质采用投影时序逻辑进行形式化,模型检测被转化为投影时序逻辑的可满足性问题加以解决.通过模型检测,可验证 Web 服务的性质,提高了 Web 服务的可靠性.

参考文献:

[1] 雷丽晖,段振华.基于扩展投影时序逻辑的组合 Web 服务描述与验证[J].西安交通大学学报,2007 41(10):1155-1159.
LEI Lihui, DUAN Zhenhua. Specification and verification of composite web services based on extended projection temporal logic [J]. Journal of Xi′an Jiaotong University, 2007, 41(10): 1155-1150.

[2] ANKOLEKAR A, PAOLUCCI M, SYCARA K. Towards a formal verification of OWL-S process models [C]// Proceedings of LISWC 2005. Berlin, Germany: Springer-Verlag, 2005: 37-51.

[3] MOSZKOWSKI B C. Executing temporal logic programs[M]. Cambridge, UK: Cambridge University Press, 1986.

[4] DUAN Zhenhua, YANG Xiaoxiao, KOUTRIY M. Framed temporal logic programming [J]. Science of Computer Programming, 2008, 70(1): 31-61.

[5] 唐稚松.时序逻辑程序设计与软件工程[M].北京:科学出版社,1999.

[6] NARAYANAN S, MCSHEILA A. Simulation, verification and automated composition of web services [C]// Proceedings of the 11th International Conference on World Wide Web. New York, NY, USA: ACM, 2002: 77-88.

[7] 侯丽珊,金芝,吴步丹.需求驱动的 Web 服务建模及其验证:一个基于本体的方法[J].中国科学:E 辑,2006 36(10): 1189-1219.

[8] DUAN Zhenhua, TIAN Cong, ZHANG Li. A decision procedure for propositional projection temporal logic with infinite models[J]. Acta Informatica, 2008, 45(1): 43-78.

(编辑 杜秀杰 苗凌)