一种基于 Petri 网化简的工作流过程语义验证方法*

周建涛[1,2+], 史美林[1], 叶新铭[2]

[1](清华大学 计算机科学与技术系,北京 100084)
[2](内蒙古大学 计算机学院,内蒙古 呼和浩特 010021)

A Method for Semantic Verification of Workflow Processes Based on Petri Net Reduction Technique

ZHOU Jian-Tao[1,2+], SHI Mei-Lin[1], YE Xin-Ming[2]

[1](Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)
[2](College of Computer Science, Inner Mongolia University, Huhhot 010021, China)
+ Corresponding author: Phn: +86-10-62785609, E-mail: zjtao@csnet4.cs.tsinghua.edu.cn, http://www.cs.tsinghua.edu.cn
Received 2004-03-29; Accepted 2004-12-08

Zhou JT, Shi ML, Ye XM. A method for semantic verification of workflow processes based on Petri net reduction technique. *Journal of Software*, 2005,16(7):1242−1251. DOI: 10.1360/jos161242

Abstract: Verification is meaningful for ensuring the correctness of workflow process definition. This paper focuses on semantic verification method, solving the problem of conflict checking under the combination of control flow, data flow and resource dimensions of workflow processes. Firstly, a formal model for process description—3DWFN is defined responding to the requirement of semantic verification, and then reduction rules accomplishing semantic verification are stated in details based on 3DWFN nets. Finally, their advantages on semantic verification layer are compared with the existing reduction rules in the literatures.
Key words: workflow; process; semantic verification; Petri net; reduction

摘 要: 过程验证是保证工作流过程定义正确性的重要手段.针对过程定义中控制流、数据流和资源三维元信息相结合的冲突检测问题,研究了过程的语义验证方法.首先根据语义验证的需求,定义 3DWFN 网作为过程描述的形式化模型,然后基于该模型阐述了完成语义验证的化简规则.并通过与其他相关化简规则的比较,说明了这些规则在语义验证层面的优势.
关键词: 工作流;过程;语义验证;Petri 网;化简
中图法分类号: TP311 **文献标识码**: A

工作流在办公自动化、集成制造、电子商务等领域发挥了重大作用,并向更复杂和更灵活的方向发展[1],

* Supported by the National Natural Science Foundation of China under Grant No.60073011 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2001AA113150 (国家高技术研究发展计划(863))

作者简介: **周建涛**(1974－),女,河北丰宁人,博士,讲师,主要研究领域为计算机网络,计算机支持的协同工作,形式描述技术;**史美林**(1938－),男,教授,博士生导师,主要研究领域为计算机网络,计算机支持的协同工作,分布式系统;**叶新铭**(1943－),男,教授,博士生导师,CCF 高级会员,主要研究领域为计算机网络,分布式系统,协议测试和形式描述技术.

若依靠用户的反复使用来发现过程定义中的错误,则修复代价将成倍增长,因此,系统中缺少过程正确性验证成为日益突出的问题.与工作流管理联盟定义参考框架[2]时相比,在被称为“后工作流研究”的业务过程管理[3]中,与验证相关的内容得到了丰富和发展,可见,过程验证正在成为新时期工作流系统研究的热点.

目前,已有的一些过程验证方法侧重控制流冲突的检测,控制流与数据流、资源流相结合的更复杂的语义验证问题还没有得到很好解决.本文将以正确实现过程定义的目标为出发点,突破结构验证的局限,探讨基于 Petri 网化简技术的语义验证方法.第 1 节简单介绍过程验证的现状和基于 Petri 网的化简技术.第 2 节定义基于 Petri 网的、面向语义验证的过程模型,然后详述语义验证化简规则.第 3 节分析规则集合的特性,以实例说明其有效性.第 4 节与相关研究进行比较;最后总结全文的工作.

1 工作流过程验证和 Petri 网化简技术

为了确保过程定义在引擎的控制下顺利执行,需要验证其正确性.基本要求是语法正确性,保证过程定义符合其描述语言或模型的语法.进一步的要求是结构正确性,保证过程定义无结构冲突,如死锁、无同步等[4−9].然而,从工作流管理联盟对过程定义的描述来看,“一些相连的过程或活动,在定义了功能角色及其关系的组织结构中,共同实现一个业务目标”[10],反映出过程定义由控制流(结构)、数据流和资源(包括人力资源和设备资源)三维元信息共同描述,并且三维不是彼此孤立存在的,而是相互协作实现一个“业务目标”.并且,从实际需求来看,使用者真正关心的是能否由正确的用户操作适当的数据最终完成一项业务工作.因此,检查三维元信息共同作用产生的冲突才能确定一个过程能否正确实现其业务目标,而仅对过程进行结构验证是不够的.基于过程目标的正确性称为语义正确性,检查控制流、数据流和资源三维冲突的验证称为语义验证.图 1 总结了过程验证的地位和层次.

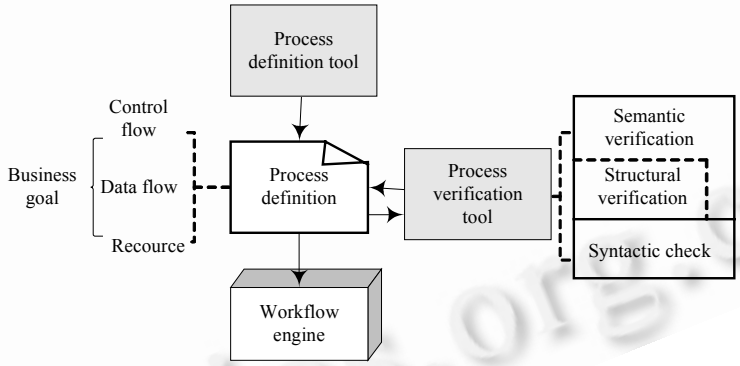

Fig.1 The status and levels of process verification

图 1 过程验证的地位和层次

在工作流建模领域,基于 Petri 网技术的过程描述和验证方法具有较强的形式化优势.在过程描述中,Petri 网变迁表示过程中的活动,网系统标识表示过程状态,弧表示控制流[11].在过程验证中,可按“检测对象”将验证方法分为两类,一类直接验证过程应该具有的特性(如合理性、活性等)[4],另一类使用化简技术检测过程中的冲突[8,9].除了基本 Petri 网的化简技术[12]以外,基于高级 Petri 网[13−17]和工作流图的化简技术[5−7]也得到了相应研究.化简技术的目标,简言之,就是“融合一致、凸现冲突”,其优点在于允许过程定义的可扩展性,而且有助于冲突定位,便于修改,在工作流及其他应用[18,19]的验证中发挥了重要作用.

然而,现有的工作流验证研究大多集中在结构验证的层面上[4−9],控制流与数据流相结合的验证只进行了有限的探索[20],三维元信息相结合的语义验证问题目前还没有得到很好的解决.其中一个原因是在面向过程的建模中,往往将控制流作为整个过程控制、协同和调度的核心,而忽略了三维元信息间的有机联系,忽略了“过程目标”这个中心.另一个原因是缺少动、静结合的观点,认为数据和资源等因素反映的是运行时特征,与其相关的正确性需要通过仿真过程执行来检验.事实上,只将过程建模作为工作流创建时功能已经是过时的看法,现在兴

起的灵活性、可适应性和动态性研究充分证明了这一点.并且,Petri 网是具有可执行特性的模型,完全可以通过丰富模型上的信息来完成语义验证.因此,本文将结合考虑过程三维元信息的共同作用,对过程模型进行语义验证,保证其最终完成业务目标.图 2 所示的语义验证目的和方法可作为全文工作的概括.

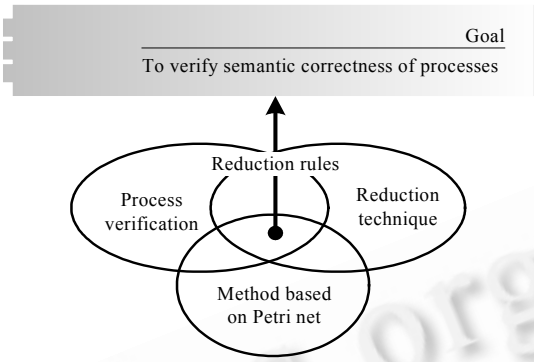

Fig.2 Goal-Method of semantic verification of processes

图 2 过程语义验证的目的-方法

2 基于 Petri 网化简的过程语义验证

2.1 面向工作流的扩展Petri网模型

基于 Petri 网的模型在工作流过程描述中应用广泛[4,8,9],但大多数侧重模拟控制流,在涉及语义验证的问题时就会暴露出一些不足,如:缺少数据和角色的映射,无法表示数据的传递方式等.因此,考虑将基本 Petri 网的五元组(P,T,F,W,M_0)[12]在面向工作流的数据和资源方面进行相应扩充.

定义 1(三维工作流网,3DWFN). 设Σ,K,Ψ分别表示工作流过程中的活动名、数据对象名和资源名的集合,则其上一个三维工作流网是一个 7 元组 $3DWFN=\langle P,T,F,C;Lab,Exp,M_0\rangle$:

P 是库所的有限集合;T 是变迁的有限集合:$P\cap T=\varnothing;P\cup T\neq\varnothing$,且$\exists i,o\in P,{}^{\bullet}i=\varnothing,o^{\bullet}=\varnothing,T=Ta\cup Tp\cup T\tau$,其中 Ta 是原子变迁的集合,Tp 是子网变迁的集合,$T\tau$是内部变迁的集合;

$F\subseteq(P\times T)\cup(T\times P)$是弧集合,且禁止弧集合 $F^{\circ}\subseteq F$;

C 是托肯类型的有限非空集合,且 $C=Cd\cup Cr\cup\{\bullet\}$,其中 Cd 表示数据类型,Cr 表示资源类型,'•'为缺省值;

$Lab:T\rightarrow\Sigma,Cd\rightarrow K,Cr\rightarrow\Psi$是标记函数;

$Exp:F\rightarrow N\times C\times Con$ 是弧表达函数,其中 $N=\{1,2,\ldots\},Con$ 是可以完成计算、比较和逻辑操作的条件表达式,$con\in Con$ 的结果取值为布尔类型,即:$Type(con)=\{\text{TRUE},\text{FALSE}\}$;

$M_0:P\rightarrow\{\varnothing,\mu C\}$是初始网标识,$\mu C$ 是 C 的多集.

3DWFN 区别与已有 Petri 网模型的区别在于:(1) 细化了变迁的含义:工作流过程中的活动包括原子活动与子过程,以及一些起控制作用的内部活动,因此将变迁集合 T 相应地划分为 Ta,Tp 和 $T\tau$,以支持层次的工作流过程模拟.(2) 丰富了托肯的类型:为了描述数据流和资源,将各类数据和资源映射为托肯类型,用集合 C 表示.相应地,网标识也不再是一个正整数向量,而是一个集合的向量,每个集合分量可以为空集或托肯的多集.(3) 扩展了弧表达函数:托肯类型的多样性使工作流过程中受关注的不再仅仅是托肯的数目,因此,弧上的权值函数也相应地扩展为约束数据和资源流动的弧表达函数.这样,3DWFN 网就能够有效地描述数据流和资源,为进行语义验证提供了条件.

定义 2(投影). 设 $Fx:Q\rightarrow P_1\times P_2\times P_3\times\ldots,\exists q\in Q,p_i\in P_i(i=1,2,3,\ldots):Fx(q)=(p_1,p_2,p_3,\ldots),Fx(q)\upharpoonright P_j\times P_k\times P_l\times\ldots$表示$Fx(q)$在 $P_j,P_k,P_l\ldots(j,k,l\in\{1,2,3,\ldots\})$上的投影,且 $Fx(q)\upharpoonright P_j\times P_k\times P_l\times\ldots=(p_j,p_k,p_l,\ldots)$或$\{p_j,p_k,p_l,\ldots\}$.

投影的作用是将多元向量中的任意分量分离出来.这样,就可以利用它取得弧表达式中的任意部分,然后根据需要使用.

定义 3(3DWFN 的点火规则).

变迁 $t\in T$ 的前置条件分为点火条件和控制条件:$\forall s\in{}^{\bullet}t$,若$(s,t)$是普通弧,即:$(s,t)\in F\backslash F^{\circ}$,则 $Pre(s,t)=Exp(s,t)$为 s 到 t 的点火条件,$Pre(t)=\{s\in{}^{\bullet}t|Exp(s,t)\}$ 为这一类前置条件的集合;而当(s,t)是禁止弧,即:$(s,t)\in F^{\circ}$,则 $Prev(s,t)=Exp(s,t)$为 s 到 t 的控制条件.

$\forall s\in t^{\bullet}$,$Post(t,s)=Exp(t,s)$为 s 对 t 的后置条件,$Post(t)=\{s\in t^{\bullet}|Exp(t,s)\}$为这些后置条件的集合.

变迁 t 在标识 M 下是使能的,$\forall s\in{}^{\bullet}t$,iff:

① $(s,t)\in F\backslash F^{\circ}$:$M(s)\geq(Pre(s,t)\upharpoonright N\times C)\wedge Type(con)=\text{TRUE}$,

② $(s,t)\in F^{\circ}$:$M(s)<(Prev(s,t)\upharpoonright N\times C)\vee Type(con)=\text{FALSE}$,

其中,$Pre(s,t)\upharpoonright N\times C$ 表示 $Pre(s,t)$中描述托肯数量和类型的部分,con 表示其中的条件表达式.

变迁 t 点火后产生新标识 M',其中 $M'=M-(Pre(t)\upharpoonright N\times C)+(Post(t)\upharpoonright N\times C)$,记为 $M[t\rangle M'$或 $M\xrightarrow{t}M'$.

满足全部前置条件的变迁才能点火,按前置条件的要求消耗前置库所中的相应托肯,并在点火后按照后置条件的要求向后置库所中添加托肯.这样,3DWFN 网的执行看起来就像是承担某种角色的用户操作并传递相应的数据沿特定的路径穿过网,有效地描述了数据和资源沿控制流的流动.

2.2 用于工作流过程语义验证的化简规则

基于 3DWFN 网的化简规则首先应该遵循特性保持原则,即:支持语义验证的化简规则应不减少过程中原有的各项特性,并且不增加任何新的特性到过程中.

既然是保持“各项”特性,就包括希望有的特性和不希望有的特性(冲突).在结构验证中,化简操作融合无结构冲突的部分,而包含冲突的部分在化简过程中不被处理,因而被保留下来.那么,在语义验证中,就需要融合无语义冲突的部分,也称为语义一致的部分.在此融合操作中,需要注意两点,第一,如何识别语义一致的部分;第二,在操作中如何融合弧表达式.因此,给出如下两个定义.

定义 4(语义一致性).

两个变迁集合 T_1 和 T_2 是语义一致的,表示为 $T_1\ conf\ T_2$:若$\exists T_1,T_2\subset T$,满足$(\forall t\in T_1:{}^{\bullet}({}^{\bullet}t)\subset T_2)\wedge(\forall t\notin T_1:{}^{\bullet}({}^{\bullet}t)\not\subset T_2)$,则$\forall t\in T_1,\forall p\in{}^{\bullet}t$,满足 $Pre(p,t)\upharpoonright N\times C\leq\bigcap_{s\in{}^{\bullet}p}Post(s,p)\upharpoonright N\times C$;

两个库所集合 P_1 和 P_2 是语义一致的,表示为 $P_1\ conf\ P_2$:若$\exists P_1,P_2\subset P,\exists T_1\subset T$,且$\forall p\in P_1,p'\in P_2,t\in T_1$:$(p,t)\in F\wedge(t,p')\in F$,满足$\exists M(P_1)\neq\varnothing\wedge M\rightarrow M'\wedge M'(P_2)\neq\varnothing$,有 $M(P_1)-M'(P_1)=M'(P_2)-M(P_2)$.

如果 $T_1\ conf\ T_2$,则表示 T_1 中变迁点火所需要的数据和资源等条件必须曾经由 T_2 中的变迁执行产生出来过;反过来,T_2 中变迁点火产生的数据和资源也将在过程结束之前被 T_1 中的变迁点火消耗掉.这样,既不会因为数据和资源的缺失而造成过程死锁,也不会因为产生未处理的数据和资源而造成冗余.如果 $P_1\ conf\ P_2$,则说明 P_1 和 P_2 两个库所集合之间的数据和资源只是“流动”,在流动中不发生类型和数量变化.因此能够保证既不减少也不增加过程中有关控制流、数据和资源方面的各项特性.

定义 5 给出在语义化简验证中,弧表达式的融合方式.

定义 5(弧表达式运算). 设有弧表达式 $exp_1=\langle n_1c_1,cond_1\rangle$和 $exp_2=\langle n_2c_2,cond_2\rangle$,其中 $n_1,n_2\in N,c_1,c_2\in C,cond_1,cond_2\in Con$,定义运算如下:

$exp_1 X exp_2$:$X\in\{\cup,\cap\}$;

或运算 $exp_1\cup exp_2=\langle n_1c_1+n_2c_2,cond_1\vee cond_2\rangle$,“+”表示托肯之间的“或”,“∨”表示条件之间的析取;与运算 $exp_1\cap exp_2=\langle n_1c_1\cdot n_2c_2,cond_1\wedge cond_2\rangle$,“▪”表示托肯之间的“与”,“∧”表示条件之间的合取;

$\underset{i=1,2,\ldots}{X}exp_i=(((exp_1 X exp_2)X\,exp_3)\ldots)$表示弧表达式运算可以从二元扩展到多元.

将基于 3DWFN 网的化简规则集称为 RR-SV(reduction rules for semantic verification),其中包含 14 条规则.并且,根据特性保持原则,化简规则只对在初始标识下不含托肯的库所进行操作,即满足 $M_0(p)=\varnothing,p\in P$ 条件的库所,防止将含有初始标识的库所化简掉而影响网的活性等性质.为清晰起见,将 RR-SV 进一步划分为 3 个规则子集.下面分别叙述各条规则的含义,并给出其形式化描述和图形表示.

设一个表示工作流过程的 3DWFN 网 PN,化简前为 $PN=\langle P,T,F,C;Lab,Exp,M_0\rangle$,化简后为 $PN'=(P',T',F',C;Lab,Exp,M'_0)$,记作:$PN \xrightarrow{R} PN'$.

第 1 个规则子集处理过程中的一些简单结构,包含 6 条规则.

R1:顺序变迁结合(combination of serial transitions,简称 CST).

如图 3(a)所示:$\exists t_1,t_2\in T,\exists p\in P: t_1^\bullet={}^\bullet t_2=\{p\}\wedge {}^\bullet p=\{t_1\}\wedge p^\bullet=\{t_2\}\wedge\{t_1\}\ conf\ \{t_2\}\mapsto P'=P\backslash\{p\},T'=T\cup\{t\}\backslash\{t_1,t_2\},F'=F\cup\{(s,t),(t,q)|s\in{}^\bullet t_1,q\in t_2^\bullet\}\backslash\{(t_1,p),(p,t_2)\}\backslash\{(s,t_1),(t_2,q)|s\in{}^\bullet t_1,q\in t_2^\bullet\}$.

R2:顺序库所结合(combination of serial places,简称 CSP).

如图 3(b)所示:$\exists p_1,p_2\in P,\exists t\in T: p_1^\bullet={}^\bullet p_2=\{t\}\wedge {}^\bullet t=\{p_1\}\wedge t^\bullet=\{p_2\}\wedge\{p_1\}\ conf\ \{p_2\}\mapsto P'=P\cup\{p\}\backslash\{p_1,p_2\},T'=T\backslash\{t\},F'=F\cup\{(s,p),(p,q)|s\in{}^\bullet p_1,q\in p_2^\bullet\}\backslash\{(p_1,t),(t,p_2)\}\backslash\{(s,p_1),(p_2,q)|s\in{}^\bullet p_1,q\in p_2^\bullet\}$.

R3:并行变迁结合(combination of parallel transitions,简称 CPT).

如图 3(c)所示:$\exists t_i\in T(i=1,\ldots,k),\exists p_1,p_2\in P: {}^\bullet t_i=\{p_1\}\wedge t_i^\bullet=\{p_2\}\wedge\{t_1,\ldots,t_k\}\subseteq p_1^\bullet\wedge\{t_1,\ldots,t_k\}\subseteq {}^\bullet p_2\mapsto P'=P,T'=T\cup\{t\}\backslash\{t_1,\ldots,t_k\},F'=F\cup\{(p_1,t),(t,p_2)\}\backslash\{(p_1,t_i),(t_i,p_2)|i=1,\ldots,k\},Exp'(p_1,t)=\cup Exp(p_1,t_i),Exp'(t,p_2)=\cup Exp(t_i,p_2)$.

R4:并行库所结合(combination of parallel places,简称 CPP).

如图 3(d)所示:$\exists p_i\in P(i=1,\ldots,k),\exists t_1,t_2\in T: {}^\bullet p_i=\{t_1\}\wedge p_i^\bullet=\{t_2\}\wedge t_1^\bullet\supseteq\{p_1,\ldots,p_k\}\wedge {}^\bullet t_2\supseteq\{p_1,\ldots,p_k\}\wedge\forall p_i\in P: Exp(t_1,p_i)\upharpoonright N\times C\geq Exp(p_i,t_2)\upharpoonright N\times C\mapsto P'=P\cup\{p\}\backslash\{p_1,\ldots,p_k\},T'=T,F'=F\cup\{(t_1,p),(p,t_2)\}\backslash\{(t_1,p_i),(p_i,t_2)|i=1,\ldots,k\},Exp'(t_1,p)=\cap Exp(t_1,p_i),Exp'(p,t_2)=\cap Exp(p_i,t_2)$.

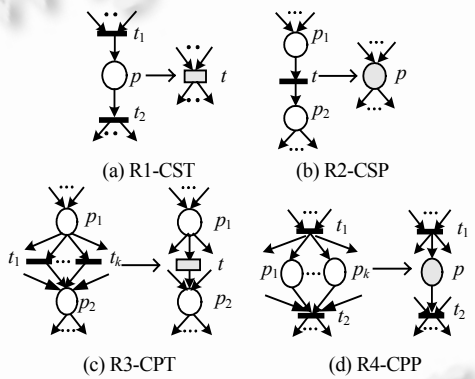

Fig.3 Reduction rules R1~R4

图 3 化简规则 R1~R4

在图 3 及以下各图中,阴影形式的库所和变迁表示结合后的结果.并且,为图示简化起见,图中省略了弧表达式.

R5:重叠变迁结合(combination of overlapped transitions,简称 COT).

如图 4(a)所示:$\exists T_1=\{t_1,t_2,\ldots,t_k\}\subset T,P_1=\{p_1,p_2,\ldots,p_m\}\subset P,P_2=\{p'_1,p'_2,\ldots,p'_n\}\subset P:(\forall t\in T_1: {}^\bullet t=P_1,t^\bullet=P_2)\wedge(\forall p\in P_1: p^\bullet=T_1)\wedge(\forall p\in P_2: {}^\bullet p=T_1)\wedge P_1\ conf\ P_2\mapsto P'=P,T'=T\cup\{t_{1k}\}\backslash\{t_1,t_2,\ldots,t_k\},F'=F\cup\{(p,t_{1k}),(t_{1k},p')|p\in P_1,p'\in P_2\}\backslash\{(p,t),(t,p')|p\in P_1,p'\in P_2,t\in T_1\},Exp'(p,t_{1k})=\bigcup_{t\in T_1}Exp(p,t)\ (p\in P_1),Exp'(t_{1k},p')=\bigcup_{t\in T_1}Exp(t,p')\ (p'\in P_2)$.

R6:重叠库所结合(combination of overlapped places,简称 COP).

如图 4(b)所示:$\exists P_1=\{p_1,p_2,\ldots,p_k\}\subset P,T_1=\{t_1,t_2,\ldots,t_m\}\subset T,T_2=\{t'_1,t'_2,\ldots,t'_n\}\subset T:(\forall p\in P_1: {}^\bullet p=T_1,p^\bullet=T_2)\wedge(\forall t\in T_1: t^\bullet=P_1)\wedge(\forall t\in T_2: {}^\bullet t=P_1)\wedge T_1 conf\ T_2\mapsto P'=P\cup\{p_{1k}\}\backslash\{p_1,p_2,\ldots,p_k\},T'=T,F'=F\cup\{(t,p_{1k}),(p_{1k},t')|t\in T_1,t'\in T_2\}\backslash\{(t,p),(p,t')|p\in P_1,t\in T_1,t'\in T_2\},Exp'(t,p_{1k})=\bigcap_{p\in P_1}Exp(t,p)\ (t\in T_1),Exp'(p_{1k},p')=\bigcap_{p\in P_1}Exp(p,t')\ (t'\in T_2)$.

第 2 个规则子集处理过程中分叉、合并结构的嵌套情况,共 4 条规则.

R7:或分叉结合(combination of OR-Split,简称 COS).

如图 5(a)所示:$\exists p_1,p_2\in P,t_1,t_2,t_3,t_4\in T:p_1^{\bullet}=\{t_1,t_2\}\wedge{}^{\bullet}t_2=\{p_1\}\wedge t_2^{\bullet}=\{p_2\}\wedge{}^{\bullet}p_2=\{t_2\}\wedge p_2\in{}^{\bullet}t_3\wedge p_2\in{}^{\bullet}t_4\wedge\{t_2\}\ conf\ \{t_3,t_4\}\mapsto P'=P\backslash\{p_2\},T'=T\cup\{t_{23},t_{24}\}\backslash\{t_2,t_3,t_4\},F'=F\cup\{(p_1,t_{23}),(p_1,t_{24})\}\backslash\{(p_1,t_2),(t_2,p_2),(p_2,t_3),(p_2,t_4)\},Exp'(p_1,t_{23})=Exp(p_1,t_2)\cap Exp(p_2,t_3),Exp'(p_1,t_{24})=Exp(p_1,t_2)\cap Exp(p_2,t_4)$.

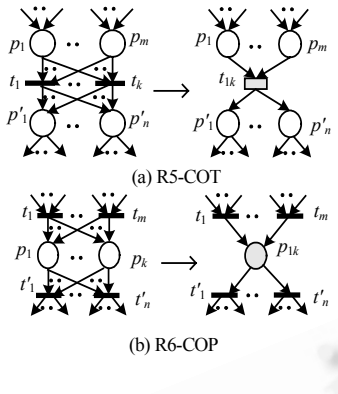

Fig.4 Reduction rules R5~R6

图 4 化简规则 R5~R6

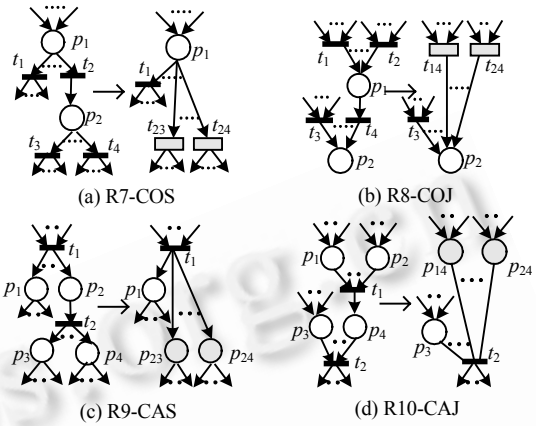

Fig.5 Reduction rules R7~R10

图 5 化简规则 R7~R10

需要说明的是,COS 规则是否可以由 p_1 和 p_2 进行"顺序库所结合"来代替呢?在结构验证中,这似乎是可行的,但在语义验证中,当考虑到数据流和资源流时,情况完全不同.举个小例子可以说明,假设图 5(a)中 $Exp(p_1,t_1)=\langle x,x\geq 5\rangle,Exp(p_1,t_2)=\langle x,x<5\rangle,Exp(t_2,p_2)=\langle x\rangle,Exp(p_2,t_3)=\langle x,x\geq 3\rangle,Exp(p_2,t_4)=\langle x,x<3\rangle$,若$\{p_1\}\ conf\ \{p_2\}$而将 p_1 和 p_2 合并,则 $Exp'(p_{12},t_3)=\langle x,x\geq 3\rangle,Exp'(p_{12},t_4)=\langle x,x<3\rangle$,导致缺失了 $x<5$ 那部分的信息.因此,语义验证中的或分叉结合应如 COS 规则所示,变迁结合后的弧表达式为 $Exp'(p_1,t_{23})=\langle x,3\leq x<5\rangle,Exp'(p_1,t_{24})=\langle x,x<3\rangle$,完整地保持了各部分信息.

R8:或合并结合(combination of OR-Join,简称 COJ).

如图 5(b)所示:$\exists p_1,p_2\in P,t_1,t_2,t_3,t_4\in T:{}^{\bullet}p_1=\{t_1,t_2\}\wedge p_1^{\bullet}=\{t_4\}\wedge{}^{\bullet}t_4=\{p_1\}\wedge t_4^{\bullet}=\{p_2\}\wedge\{t_3,t_4\}\subseteq{}^{\bullet}p_2\wedge\{t_1,t_2\}\ conf\ \{t_4\}\mapsto P'=P\backslash\{p_1\},T'=T\cup\{t_{14},t_{24}\}\backslash\{t_1,t_2,t_4\},F'=F\cup\{(t_{14},p_2),(t_{24},p_2)\}\backslash\{(t_1,p_1),(t_2,p_1),(p_1,t_4),(t_4,p_2)\},Exp'(t_{14},p_2)\cup Exp'(t_{24},p_2)=Exp(t_4,p_2)$.

同理,COJ 规则也不能执行为 p_1 和 p_2 的"顺序库所结合",因为,$Exp(t_4,p_2)$表示的内容将影响 p_2 后继变迁的点火,若在 p_1,p_2 融合中将弧(t_4,p_2)化简掉,则将导致数据或资源信息的缺失.

R9:与分叉结合(combination of AND-Split,简称 CAS).

如图 5(c)所示:$\exists t_1,t_2\in T,p_1,p_2,p_3,p_4\in P:t_1^{\bullet}=\{p_1,p_2\}\wedge{}^{\bullet}p_2=\{t_1\}\wedge p_2^{\bullet}=\{t_2\}\wedge{}^{\bullet}t_2=\{p_2\}\wedge t_2\in{}^{\bullet}p_3\wedge t_2\in{}^{\bullet}p_4\wedge Exp(t_1,p_2)\upharpoonright N\times C=Exp(p_2,t_2)\upharpoonright N\times C\mapsto P'=P\cup\{p_{23},p_{24}\}\backslash\{p_2,p_3,p_4\},T'=T\backslash\{t_2\},F'=F\cup\{(t_1,p_{23}),(t_1,p_{24})\}\backslash\{(t_1,p_2),(p_2,t_2),(t_2,p_3),(t_2,p_4)\},Exp'(t_1,p_{23})=Exp(t_2,p_3),Exp'(t_1,p_{24})=Exp(t_2,p_4)$.

R10:与合并结合(combination of AND-Join,简称 CAJ).

如图 5(d)所示:$\exists t_1,t_2\in T,p_1,p_2,p_3,p_4\in P:{}^{\bullet}t_1=\{p_1,p_2\}\wedge t_1^{\bullet}=\{p_4\}\wedge{}^{\bullet}p_4=\{t_1\}\wedge p_4^{\bullet}=\{t_2\}\wedge\{p_3,p_4\}\subseteq{}^{\bullet}t_2\wedge Exp(t_1,p_4)\upharpoonright \boldsymbol{N}\times\boldsymbol{C}=Exp(p_4,t_2)\upharpoonright \boldsymbol{N}\times\boldsymbol{C}\mapsto P'=P\cup\{p_{14},p_{24}\}\backslash\{p_1,p_2,p_4\},T'=T\backslash\{t_1\},F'=F\cup\{(p_{14},t_2),(p_{24},t_2)\}\backslash\{(p_1,t_1),(p_2,t_1),(t_1,p_4),(p_4,t_2)\},Exp'(p_{14},t_2)=Exp(p_1,t_1),Exp'(p_{24},t_2)=Exp(p_2,t_1)$.

CAS 和 CAJ 规则(图 4 所示 R7,R8)也不能由 t_1 和 t_2 的"顺序变迁结合"来实现,因为并不一定存在$\{t_1\}\ conf\ \{t_2\}$,t_1 产生的托肯还可能由 t_2 以外的其他变迁消耗.

另外,在如上所示规则 R7~R10 中,两个分支的或/与结构可以扩展到多个分支,两级的或/与分支/合并结构也可以扩展到多级.

RR-SV 中最后一个规则子集处理过程中的几种特殊结构,共 4 条.

R11:自环变迁删除(deletion of self-loop transition,简称 DST).

如图 6(a)所示:$t\in T,\exists p\in{}^{\bullet}t:{}^{\bullet}t=t^{\bullet}\wedge|{}^{\bullet}t|=|t^{\bullet}|=1\wedge\{{}^{\bullet}p\cup\{t\}\}\ conf\ \{p^{\bullet}\cup\{t\}\}\mapsto P'=P,T'=T\backslash\{t\},F'=F\backslash\{(p,t),(t,p)|p\in{}^{\bullet}t\}$.

R12:自环库所删除(deletion of self-loop places,简称 DSP).

如图 6(b)所示:$p\in P:{}^{\bullet}p=p^{\bullet}\wedge|{}^{\bullet}p|=|p^{\bullet}|=1\wedge M_0(p)\neq\varnothing\wedge(\exists t\in{}^{\bullet}p,Exp(p,t)\upharpoonright N\times C=Exp(t,p)\upharpoonright N\times C)\mapsto P'=P\backslash\{p\},T'=T,F'=F\backslash\{(p,t),(t,p)|t\in{}^{\bullet}p\}$.

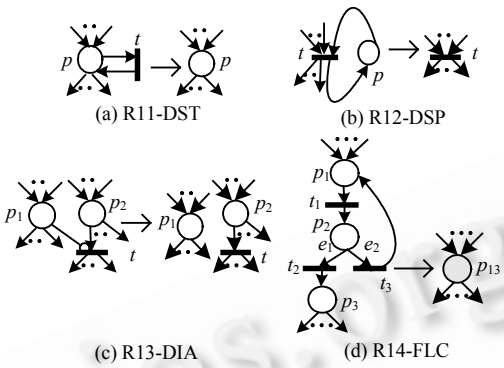

Fig.6 Reduction rules R11~R14

图 6 化简规则 R11-R14

R12 描述的情况允许自环库所的初始标识不为空,只要从该库所中消耗的托肯能够被再次产生出来,就不影响系统其他部分的性质.

R13:禁止弧删除(deletion of inhibitor arc,简称 DIA).

如图 6(c)所示:$\exists p\in P,t\in T:(p,t)\in F^{\circ}\mapsto P'=P,T'=T,F'=F\backslash\{(p,t)\}$.

由于禁止弧的作用主要在于控制,而并不影响托肯在网中的流动,因此化简中将其直接删除.

R14:回路融合(fusion of loop-circuit,简称 FLC).

如图 6(d)所示:$\exists p_1,p_2,p_3\in P,\exists t_1,t_2,t_3\in T:p_1{}^{\bullet}=\{t_1\}\wedge{}^{\bullet}t_1=\{p_1\}\wedge t_1{}^{\bullet}=\{p_2\}\wedge{}^{\bullet}p_2=\{t_1\}\wedge p_2{}^{\bullet}=\{t_2,t_3\}\wedge{}^{\bullet}t_2={}^{\bullet}t_3=\{p_2\}\wedge t_2{}^{\bullet}=\{p_3\}\wedge t_3{}^{\bullet}=\{p_1\}\wedge{}^{\bullet}p_2=\{t_1\}\wedge\{p_1\}\ conf\ \{p_3\}\wedge Type(Exp(p_2,t_2)\upharpoonright Con)=\neg Type(Exp(p_2,t_3)\upharpoonright Con)\mapsto P'=P\cup\{p_{13}\}\backslash\{p_1,p_2,p_3\},T'=T\backslash\{t_1,t_2,t_3\},F'=F\backslash\{(p_1,t_1),(t_1,p_2),(p_2,t_2),(p_2,t_3),(t_2,p_3),(t_3,p_1)\}$.

3 分析与举例

分析以上的化简规则可以得到一些结论.

结论 1(化简规则的完备性). RR-SV 能够处理一个 3DWFN 描述的工作流过程定义中的所有结构.并且,若该定义是语义正确的,它最终可由 RR-SV 化简为从开始库所经一个变迁连接到结束库所的简单网.

首先,RR-SV 不对含有初始标识的库所进行操作(DSP 规则除外),保证了含有初始标识的一个开始库所在化简中保持不变.其次,由于 RR-SV 覆盖了工作流过程的基本结构:顺序、与/或分叉/合并和循环以及结构之间的连接方式,因此,可以操作到除开始库所以外的其余所有部分.详细地说,R1,R2 是顺序结构化简,R3,R4 是对或/与分叉后随即合并结构的化简,R5,R6 是或/与结构交叉情况的化简,R7~R10 是或/与结构嵌套情况的化简,R11,R12 和 R14 化简循环结构,包括自环和有条件循环.并且,3DWFN 中允许使用禁止弧,R13 处理禁止弧化简.对于除开始库所外的其余部分,若其中无语义冲突,RR-SV 将基于变迁、库所融合把它们逐一融合为一个变迁和一个库所.这样,一个 3DWFN 描述的过程定义就化简为从开始库所经一个变迁连接到结束库所的简单网.

结论 2. 若一个由 3DWFN 描述的工作流过程定义到不能再应用 RR-SV 中的化简规则为止,仍存在除开始、结束库所及其间的一个连接变迁以外的结构,则说明过程定义中存在语义冲突.

结论 3. 若一个由 3DWFN 描述的工作流过程定义经化简规则集 RR-SV 的作用最终演化为一个空网($P=T=F=\varnothing$),则说明过程模型的初始标记为空,过程不具有活性.

下面,通过电子政务应用中一个简化的例子说明 RR-SV 的有效性.

例 1(修改计划流程):项目组成员提出计划修改申请,然后在组长的监督下进行修改,修改后,在记录员的监

督下填写修改记录,最后由记录员将修改后计划存档.

图 7(a)表示修改计划过程的初始 3DWFN 网模型 $ModifyPlan=(P,T,F,C;Lab,Exp,M_0)$,其中 $P=\{i,o\}\cup\{p_i|i=1,\ldots,8\}$,$T=\{ti,to\}\cup\{t_i|i=1,\ldots,8\}$,$C=\{*\}\cup A\cup B\cup J$,“*”表示被修改的计划,$A$,$B$ 和 J 是不同的角色.各变迁含义如下:

ti:开始;$t1$:项目组成员提出修改项目计划申请;$t2$:提出申请者本人修改计划;$t3$:其他组成员修改计划;$t4$:项目组长监督;$t5$:内部变迁;$t6$:填写修改记录;$t7$:记录员监督;$t8$:记录员将修改的计划存档;to:结束.

各角色定义如下:

$A=\{a,b,c,d,e\}$为项目组成员,变量 x:A;$B=\{f\}$为项目组长,变量 y:B;$J=\{g\}$为记录员,变量 z:J.另外,$Actor(t_1)=a$ 表示 t_1 的实际执行者是 a.

相应的弧表达函数给出如下:

$Exp(i,ti)=\langle*\rangle$,$Exp(ti,p_1)=Exp(p_1,t_1)=Exp(t_1,p_2)=\langle x\cdot*\rangle$,$Exp(t_1,p_3)=\langle y\cdot*\rangle$,$Exp(p_2,t_2)=\langle x\cdot*,cond_1\rangle$,

$Exp(p_2,t_3)=\langle x\cdot*,cond_2\rangle$,$Exp(t_2,p_4)=Exp(t_3,p_4)=\langle x\cdot*\rangle$,$Exp(p_4,t_5)=\langle x\cdot*,cond_3\rangle$,$Exp(p_4,t_6)=\langle x\cdot*,cond_4\rangle$,

$Exp(t_5,p_2)=Exp(t_6,p_6)=\langle x\cdot*\rangle$,$Exp(p_3,t_4)=\langle y\cdot*\rangle$,$Exp(t_4,p_5)=Exp(p_5,t_7)=Exp(t_7,p_7)=\langle z\cdot*\rangle$,$Exp(p_6,t_8)=\langle x\cdot*\rangle$,

$Exp(p_7,t_8)=\langle z\cdot*\rangle$,$Exp(t_8,p_8)=Exp(p_8,to)=Exp(to,o)=\langle*\rangle$,其中 $cond_1$:$x=Actor(t_1)$,$cond_2$:$x\neq Actor(t_1)$,$cond_3$:‘再次修改?’=‘Y’,$cond_4$:‘再次修改?’=‘N’.

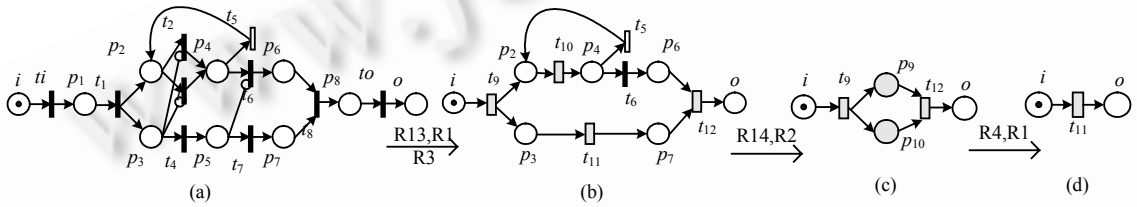

Fig.7 Reduction of a process model

图 7 化简过程模型

一些控制流的含义解释如下:$(p_3,t_2),(p_3,t_3)\in F^\circ$表示在项目组长得到通知监督修改之前任何人不得修改项目计划;$(p_5,t_6)\in F^\circ$表示在记录员得到通知并监督之前,任何人不得以填写记录的方式确认修改.

该过程首先(图 7(a))由 WF_R13,1 和 3 化简为图 7(b);再由 WF_R14 和 2 化简为图 7(c);再由 WF_R4 和 1 最终化简为图 7(d),根据结论 2 可知,原过程定义无语义冲突.图中阴影库所和变迁表示经化简规则合并所得.

例 2(有语义冲突的修改计划流程):假设如上的过程定义中包含一个语义冲突:$Exp(p_7,t_8)=\langle x\cdot*\rangle$,即:修改后的计划存档本应由“记录员”完成,这里,却误设为“项目组成员”.那么,在图 7(c)中,$Exp(t_9,p_9)=Exp(p_9,t_{12})=Exp(p_{10},t_{12})=\langle x\cdot*\rangle$,$Exp(t_9,p_{10})=\langle z\cdot*\rangle$.因为 $Exp(t_9,p_{10})\neq Exp(p_{10},t_{12})$,所以 p_9 和 p_{10} 不能合并,图 7(c)即为最终化简结果,其中存在资源冲突.

4 相关工作的比较

我们选择两项具有一定代表性的工作与 RR-SV 进行比较.Murata 等人工作[12]的代表性体现在提出时间较早且具有一般性,并因此成为一些后续研究工作的基础,如 Aalst[8]就在工作流过程验证中使用了它.李建强等人工作[9]的特点是面向工作流应用提出了化简规则.下面,将 RR-SV 与这两项工作的主要区别总结如下:

(1) 过程定义模型的差异:Murata 使用基本 Petri 网,李使用自由选择 Petri 网,都是侧重控制流描述的模型.而 RR-SV 的模型是三维工作流网 3DWFN,能够完整描述过程的三维元信息,表达过程的语义,并在结构方面允许使用禁止弧,使模型的描述能力更强、应用范围更广.

(2) 验证层次的差异:Murata 和李的工作在结构验证的层面上.RR-SV 是语义验证层面的化简规则.结构验证不能完成语义验证的功能,因为它不考虑任何与数据和资源相关的信息;而反过来,语义验证不仅可以涵盖结构验证的功能,而且可以检测出数据、资源方面的冲突(如例 2 所示).

(3) 具体化简技术的差异:RR-SV 遵循 Murata 规则所使用的基本化简技术——库所融合和变迁融合,李的

规则有些不基于这两种基本化简技术,而是进行了路径删除.对于语义验证来说,这是不适合的,例如,被删除的某个变迁产生的托肯种类,可能会在后续流程中使用,这样,路径删除就会造成死锁.

(4) 具体规则的差异:见表 1,RR-SV 在比较中可看做 3 类.第 1 类与文献中规则完成相同的目的,但是对被合并对象的结构要求有差别,这类规则包括 R1,R2,R3 和 R4.第 2 类也是与文献中规则完成相同的目的,对被合并对象的结构要求也类似,主要区别是验证层次不同,包括 R3~R6,R11 和 R12.第 3 类是新的规则,包括 R7~R10 和 R13,R14.

Table 1 Comparison of reduction rules

表 1 化简规则比较

	RR-SV													
	1 CST	2 CSP	3 CPT	4 CPP	5 COT	6 COP	7 COS	8 COJ	9 CAS	10 CAJ	11 DST	12 DSP	13 DIA	14 FLC
Murata	× R2	× R1	○ R4	○ R3	✓	✓	✓	✓	✓	✓	○ R6	○ R5	✓	✓
Li	× R2	× R1	× R4	× R3	○ R5	○ R6	✓	✓	✓	✓	○ R8	○ R7	✓	✓

备注: ×表示合并对象有差别;○表示扩充到语义验证层次;✓表示新规则.

5 总 结

过程验证是工作流领域中作用重大却取得较少有效成果的研究课题之一,本文工作致力于使用基于 Petri 网的化简方法解决过程的语义验证问题,总结如下:(1) 定义了描述工作流过程的 3DWFN 网模型,同时体现过程定义中的三维元信息,其中,托肯的多类型和弧表达函数的提出为过程语义验证创造了条件;(2) 提出了基于 3DWFN 网的语义验证化简规则,这些规则整合了控制流、数据流和资源三维元信息,从而可以对过程定义进行实用而有效的语义验证;(3) 分析和比较了与结构验证化简规则的区别和联系,说明语义层面的验证是含义更丰富、角度更全面的验证.

References:

[1] Reichert M, Bauer T, Dadam P. Enterprise-Wide and cross-enterprise workflow management—challenges and research issues for adaptive workflows. In: Dadam P, Reichert M, eds. Proc. of the Workshop on Enterprise-Wide and Cross-Enterprise Workflow Management. Ulm: Ulmer Informatik Berichte, 1999. 56−64.

[2] Workflow Management Coalition. The workflow reference model. WFMC-TC00-1003, 1995.

[3] Aalst W, Hofstede A, Weske M. Business process management: A survey. In: Aalst W, *et al*., eds. Business Process Management 2003. Berlin: Springer-Verlag, 2003. 1−12

[4] Aalst W. Verification of workflow nets. In: Azema P, Balbo G, eds. Proc. of the 18th Int'l Conf. Berlin: Springer-Verlag, 1997. 407−426

[5] Sadiq W, Orlowaka M. Analyzing process models using graph reduction techniques. Information Systems, 2000,25(2):117−134.

[6] Lin H, Zhao ZB, Li HC, Chen ZG. A novel graph reduction algorithm to identify structural conflicts. In: Sprague R, ed. Proc. of the 35th Hawaii Int'l Conf. on System Sciences. Washington: IEEE Computer Society, 2002. 289.

[7] Li PW, Lu ZD. Reduction techniques of workflow verification and its implementation. In: Han YB, Shi ML, eds. Proc. of the Int'l Workshop on Grid and Cooperative Computing. Beijing: Publishing House of Electronics of Industry, 2002. 703−710.

[8] Aalst W, Hirnschall A, Verbeek H. An alternative way to analyze workflow graphs. In: Pidduck A, Mylopoulos J, Woo C, *et al*., eds. Proc. of the 14th Int'l Conf. of Advanced Information Systems Engineering. Berlin: Springer-Verlag, 2002. 535−552.

[9] Li JQ, Fan YS. Research of Petri net based workflow model reduction methods. Information and Control, 2001,30(6):492−497 (in Chinese with English abstract).

[10] Workflow Management Coalition. Terminology & Glossary. WFMC-TC-1011, 1999.

[11] Aalst W. The application of Petri nets to workflow management. The Journal of Circuits, Systems and Computers, 1998,8(1): 21−66.

[12] Murata T. Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 1989,77(4):541−580.

[13] Haddad S. A reduction theory for coloured nets. In: Jensen K, Rozenberg G, eds. High Level Petri Nets, Theory and Application. Berlin: Springer-Verlag, 1991. 399−425.

[14] Sloan R, Buy U. Reduction rules for time Petri nets. Acta Informatica, 1996,33(7):687−706.

[15] Schmidt K. Applying reduction rules to algebraic Petri nets. Technical Reports, Report No.44, Espoo: Heisinki University of Technology, Digital Systems Laboratory, 1997.

[16] Xu AG, Jiang CJ. The reduction operations and their properties for P/T nets. Journal of Software, 1997,8(7):493−504 (in Chinese with English abstract).

[17] Juan E, Tsai J, Murata T, Zhou Y. Reduction methods for real-time systems using delay time Petri nets. IEEE Trans. on Software Engineering, 2001,27(5):422−448.

[18] Shatz S, Tu S, Murata T. An application of Petri net reduction for Ada tasking deadlock analysis. IEEE Trans. on Parallel and Distributed Systems, 1996,7(12):1307−1322.

[19] Wang SG, Yan GF, Jiang JP. Application of net reduction to feedback controller design of Petri nets. Journal of Software, 2003,14(6):1038−1042 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/1038.html

[20] Sadiq S, Orlowska M, Sadiq W, Foulger C. Data flow and validation in workflow modelling. In: Schewe K, Williams H, eds. Proc. of the Conf. in Research and Practice in Information Technology. Darlinghurst: Australian Computer Society, Inc., 2004. 207−214.

附中文参考文献:

[9] 李建强,范玉顺.基于 Petri 网化简方法的工作流模型验证.信息与控制,2001,30(6):492−497.

[16] 许安国,蒋昌俊.P/T 网的化简运算及其性质研究.软件学报,1997,8(7):493−504.

[19] 王寿光,颜刚锋,蒋静坪.网化简技术在 Petri 网反馈控制器设计中的应用.软件学报,2003,14(6):1038−1042. http://www.jos.org.cn/1000-9825/14/1038.html

****

2005 年全国开放式分布与并行计算学术会议

征 文 通 知

由中国计算机学会开放系统专业委员会主办、上海大学计算机学院承办、上海计算机学会协办的“2005 年全国开放式分布与并行计算学术会议”将于 2005 年 10 月 27 日－29 日在上海召开，有关信息如下：

一、征文范围(包括但不限于)

开放式分布与并行计算模型及体系结构；

下一代开放式网络、数据通信、网络与信息安全、业务管理技术；

开放式海量数据存储与 Internet 索引技术，分布与并行数据库及数据/Web 挖掘技术；

开放式机群计算、网格计算、Web 服务、P2P 网络及中间件技术；

开放式移动计算、自组网与移动代理技术；

分布式人工智能、多代理与决策支持技术；

分布与并行计算算法及其在科学与工程中的应用；

开放式虚拟现实技术与分布式仿真；

开放式多媒体技术，包括媒体压缩、内容分送、缓存代理、服务发现与管理技术。

二、征文要求

详见会议主页：http://www.cs.shu.edu.cn/DPCS2005 可查询进一步的会议信息。

三、重要日期

会议时间：2005 年 10 月 27～29 日　　截稿日期：2005 年 7 月 15 日　　录用通知：2005 年 7 月 30 日

四、联系方式

投稿地址：200072 上海延长路 149 号上海大学计算机学院 缪淮扣 收(请在信封上注明 DPCS2005)

电　　话：021-56331669;　　电子邮件: bfzhang@staff.shu.edu.cn (请在邮件主题中注明 DPCS2005)