ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.11, November 2010, pp.2752−2767
doi: 10.3724/SP.J.1001.2010.03711

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

时间属性序列图:语法和语义*

张鹏程[1,2], 李必信[1+], 李雯睿[2]

[1](东南大学 计算机科学与工程学院,江苏 南京 210096)

[2](河海大学 计算机与信息学院,江苏 南京 210098)

Syntax and Semantics of Timed Property Sequence Chart

ZHANG Peng-Cheng[1,2], LI Bi-Xin[1+], LI Wen-Rui[2]

[1](School of Computer Science and Engineering, Southeast University Nanjing 210096, China)

[2](College of Computer and Information Engineering, Hohai University, Nanjing 210098, China)

+ Corresponding author: E-mail: bx.li@seu.edu.cn; http://cse.seu.edu.cn/people/bx.li/index.htm

Zhang PC, Li BX, Li WR. Syntax and semantics of timed property sequence chart. *Journal of Software*, 2010,21(11):2752−2767. http://www.jos.org.cn/1000-9825/3711.htm

Abstract: In this paper, in order to make property sequence chart have timed expressiveness, the property sequence chart is extended into a timed property sequence chart that gives the semantics of the timed property sequence chart in terms of timed Büchi automaton. Then, the expressive power of timed property sequence chart is measured with the use of a recently proposed real-time specification pattern. Finally, the use of timed property sequence chart is illustrated in a case study, which shows the extensive application prospect of a timed property sequence chart in real-time system.

Key words: property sequence chart; timed property sequence chart; timed Büchi automaton; formal verification

摘 要: 为了表示事件出现的时间约束,扩展属性序列图为时间属性序列图,使其继承属性序列图的优点,并且能够表示时间属性,定义了时间属性序列图的形式语法,并给出基于时间 Büchi 自动机的形式操作语义;用实时规约模式度量了时间属性序列图的表达力.最后,对时间属性序列图进行了实例研究,显示了其广泛的应用前景.

关键词: 属性序列图;时间属性序列图;时间 Büchi 自动机;形式验证

中图法分类号: TP311 **文献标识码**: A

模型检验和其他有限状态验证技术能够自动检测系统模型是否满足给定的时态属性[1,2],通常,这些时态属性是由时态逻辑公式表示的,如 Linear Temporal Logic(LTL)[3],Computation Tree Logic(CTL)[4]等.然而由于内在结构的复杂性,一般的软件工程师很难正确地使用这些时态逻辑公式表示时态属性.所以,正如文献[1,3]所强调的,这阻碍了形式化验证技术从研究领域走向工业实践的步伐.于是,Autili 等人提出了一种基于场景的图形化规约语言,即属性序列图(property sequence chart,简称 PSC)[5,6],以解决时态逻辑公式内在的复杂性问题.与传统

* Supported by the National Natural Science Foundation of China under Grant Nos.60773105, 60973149 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2008AA01Z113 (国家高技术研究发展计划(863)); the Fundamental Research Funds for the Central Universities of China under Grant No.2009B04314 (中央高校基本科研业务费专项资金)

Received 2008-09-10; Revised 2009-05-19; Accepted 2009-07-23

的时态逻辑相比,PSC 具有更加直观和易于理解的优点,一般的软件工程师无须拥有专家知识,就能使用 PSC 正确地表示时态属性.

目前,已有很多工作扩展时态逻辑以表示实时系统的时间属性,如 Metric Temporal Logic(MTL)[7],Timed Computation Tree Logic(TCTL)[8],Timed Propositional Temporal Logic(TPTL)[9].然而,实时属性进一步增加了这些时态逻辑公式的复杂性.为了容易和正确地表示实时系统的属性和行为,并鉴于 PSC 的优点,文献[10,11]将 PSC 非形式地扩展为 Timed PSC(简称 TPSC),图形化表示时间属性,有助于软件工程师描述实时系统的需求.本文在此基础上进一步定义了 TPSC 基于时间 Büchi 自动机(timed Büchi automata,简称 TBA)[12]的形式操作语义.由于一些形式验证技术,如基于 LTL 的模型检验[13],通常需要构建表示行为补的自动机,所以,如果要构建表示 TPSC 的正确行为的 TBA,则需要将其转换为其对应的补的自动机.一个代替的办法就是直接构建表示 TPSC 的补行为的 TBA.并使用实时规约模式(real-time specification patterns)[14]来度量 TPSC 的表达力.

本文第 1 节简要介绍 PSC.第 2 节首先解释对 PSC 进行时间扩展的基本思想,然后定义 TPSC 形式语法和基于 TBA 的操作语义.第 3 节使用实时规约模式度量 TPSC 的表达力.第 4 节进行实例研究.第 5 节给出相关工作的比较.第 6 节是结论和进一步工作的设想.

1 PSC 简介

PSC 是一种基于场景的图形化规约语言,是由 UML 2.0 序列图的子集扩展而来的,用来表示并发执行的构件之间交互行为.关于 PSC 形式语法和语义的详细描述见文献[5,6,15],本节简单介绍 PSC 基本元素.

图 1 显示了 PSC 的图形化元素.首先,PSC 定义两类消息,分别是箭头消息和内部消息,并区分了 3 种不同类型的箭头消息:a) 正则消息(regular message):由符号“*e*”标识.可将正则消息视为下一条消息发生的前置条件,系统没有强制其必须发生.然而,一旦该正则消息发生,则意味着下一条消息的前置条件成立.b) 强制消息(required message):由符号“*r*”标识.如果强制消息的前置条件满足,那么系统强制交换这种类型的消息,由此验证系统的活性.c) 错误消息(fail message):由符号“*f* ”标识.如果系统交换了此类型消息,那么将会发生错误,用来表示系统不能交换的消息,由此验证系统的安全性.

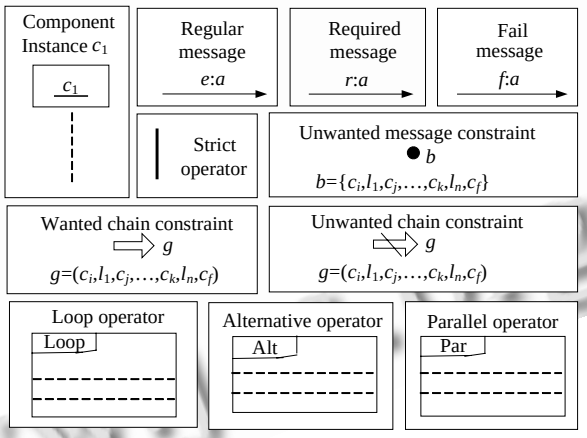


Fig.1 PSC graphical elements

图 1 PSC 的图形化元素

PSC 中还定义了箭头消息的限制,是由强加在单个箭头消息上的内部消息组成的.如图 1 所示,箭头消息的限制分为 unwanted 消息限制和链(chain)限制.unwanted 消息限制表示不期望系统发生的内部消息的集合.链限制表示一个内部消息序列,用来描述单个箭头消息和一个消息序列之间的关系.与正则、强制和错误消息不同的是,链限制是将消息序列作为一个整体进行交换的.链限制可进一步划分为 wanted 和 unwanted 链限制,wanted

链限制表示消息序列必须按照链中规定的顺序完全发生,而 unwanted 链限制表示不期望消息序列完全交换.另外,根据限制和箭头消息在时序上的关系,还可进一步将限制分为 past 限制和 future 限制,分别表示限制在箭头消息之前或之后发生.这样,unwanted 消息限制又可分为 past unwanted 消息限制和 future unwanted 消息限制;而链限制则分为 past wanted,past unwanted,future wanted 和 future unwanted 链限制.

此外,PSC 定义了 4 种操作符:严格(strict)、并发(parallel)、循环(loop)和选择(alternative).严格操作符表示一对消息之间具有严格的顺序,即它们之间不允许其他消息发生,反之则默认为一对消息之间是松散的顺序(loose),允许它们之间有其他消息发生;并发表示多条消息可以按不同的顺序交换;循环表示 1 条或多条消息可以重复发生;选择表示从多个条消息中选择 1 条或多条进行交换.

2 TPSC 形式语法和语义

PSC 已经被成功地用来表示并发构件的交互行为[5],然而 PSC 只能描述事件发生的先后顺序,而不能表示事件具体的时间约束.为了表示实时系统的性质和行为,并考虑到 PSC 图形化的优点,从而引入了 TPSC.本节给出将扩展 PSC 为 TPSC 的基本思想,并给出其形式语法和语义.

2.1 基本思想

PSC 是由 UML 序列图的一个子集扩展而来的,对 PSC 进行时间扩展的思想源于在 UML 序列图中加入时间约束的思想.在 UML 序列图中,两个连续的消息之间都可以添加有关时间约束的上界和下界[11].同样,在 PSC 中需要对交换的消息(包括箭头消息和消息限制)添加时间约束(时间上界和下界).

这里给出了一些对 PSC 中不同类型的箭头消息和消息限制进行时间扩展的实例,并简单解释其非形式化的语义.如图 2(a)所示,一个正则消息 *e*:*a* 扩展为 *e*:*a*;*x*<*t*,*y*:=0,表示在时间约束 *x*<*t* 内 *e*:*a* 发生,这时一个新的时钟变量 *y* 被重置为 0.由于系统没有强制正则消息一定发生,所以,如果在规定的时间内 *e*:*a* 没有发生,那么系统也不会出现错误.同样地,将图 2(a)中的正则消息 *e*:*a* 替换为强制消息 *r*:*a*,其时间扩展形式为 *r*:*a*;*x*<*t*,*y*:=0,表示在时间约束 *x*<*t* 内 *r*:*a* 必须发生;如果在时间约束 *x*<*t* 内 *r*:*a* 不发生,则会发生错误.类似地,错误消息可以扩展为 *f*:*a*; *x*<*t*,表示如果在时间约束 *x*<*t* 内发生了该消息,则系统到达错误状态.

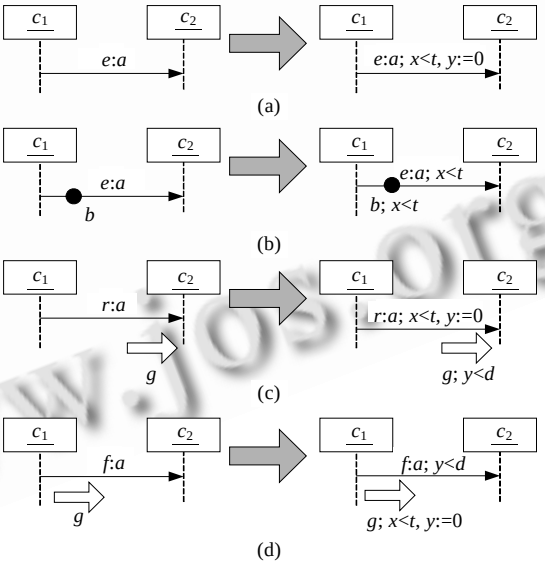


Fig.2 Extending PSC with time constructs

图 2 用时间构造子扩展 PSC

对于不同的消息类型,其限制也需要添加时间约束.根据实际的时间需求,限制可能与箭头消息具有同样的时间约束,或者两者之间也存在新的时间需求.图 2(b)显示了一个带有 past unwanted 消息限制 b 的正则消息 e:a,其中,e:a 和 b 的时间约束都是 $x<t$,表示如果在 $x<t$ 内 e:a 发生,且 b 中的消息在 e:a 发生之前不能发生;但如果在满足 $x<t$ 时 b 中的消息在 e:a 发生之前发生或 e:a 没有发生,系统也不会到达错误状态.对于强制消息,箭头消息和消息限制都必须被时间约束严格限制.图 2(c)显示了一个带有 future wanted 链限制 g 的强制消息 r:a,其中,g 有相对于 r:a 新的时间需求,表示 r:a 必须在 t 时刻之前被交换.接下来,消息链 g 必须在 d 个时间单元内发生,一旦 r:a 没有发生或 g 不能完全交换,系统将发生错误.同样地,对于错误消息,限制和错误消息都可以加入时间约束,表示在时间约束内发生了错误消息,系统到达错误状态.图 2(d)显示了一个带有 past wanted 链限制的错误消息,表示当消息链 g 在 t 时刻前完全发生,时钟变量 y 被重置;而错误消息 f:a 在 $y<d$ 内发生,系统到达错误状态.

2.2 TPSC形式语法

本节给出 TPSC 的形式语法定义.首先引入时钟约束的概念.

定义 1(时钟约束). 对于一个时钟变量集合 X,时钟约束 δ 的集合 $\Phi(X)$ 递归地定义如下:

$$\delta:=x<c|c\leqslant x|\neg\delta|\delta_1\wedge\delta_2,$$

其中,$x\in X$,c 是有理数常量.

时钟约束表示某个消息发生时时钟应满足的条件,需判断当前的时钟值是否满足时钟约束.记 V 是时钟变量值的集合,$v\in V$ 为时钟变量 x 的当前值.当 $v\vDash\delta$ 时,如果:

- $v\vDash x<c$ 当且仅当 $v(x)<c$.
- $v\vDash x\geqslant c$ 当且仅当 $v(x)\geqslant c$.
- $v\vDash\delta_1\wedge\delta_2$ 当且仅当 $v(x)\vDash\delta_1$ 和 $v(x)\vDash\delta_2$.
- $v\vDash\neg\delta$ 当且仅当 $v(x)\nvDash\delta$.

接着,记 $[\![\delta]\!]$ 为时钟变量 x 满足 δ 的所有值的集合,即 $[\![\delta]\!]=\{v\in V|v\vDash\delta\}$.

函数 $pre(\delta)$ 和 $succ(\delta)$ 分别表示时间约束 δ 的前驱时间和后继时间,定义如下:

$pre(\delta)$:$\forall x_1\in[\![pre(\delta(x))]\!]$ 和 $\forall x_2\in[\![\delta(x)]\!]$,使得 $x_1<x_2$;$succ(\delta)$:$\forall x_1\in[\![\delta(x)]\!]$ 和 $\forall x_2\in[\![succ(\delta(x))]\!]$,使得 $x_2>x_1$.

例如,$\delta(x)=\{x|2<x<4\}$,则 $pre(\delta(x))=\{x|x\leqslant 2\}$,而 $succ(\delta(x))=\{x|x\geqslant 4\}$.

由文献[6,15]中定义的 PSC 形式语法和扩展 PSC 为 TPSC 的基本思想,定义 TPSC 的形式语法如下:

定义 2(TPSC). 一个 TPSC 是一个六元组 $P=\langle T,\prec,C,M,Con,Clock\rangle$,其中,

- T 是一个有限的时间线集合 $\{t_0,t_1,\ldots,t_{n+1}\}$.
- $\prec$ 是关于时间线中的元素的一个严格的偏序关系,$t_0\prec t_1\prec\ldots\prec t_{n+1}$,表示时间线集合中的元素之间具有严格的顺序.注意,这里是系统抽象的时间线,和每个消息受到的时间约束不一样.
- C 是有限的构件集合,表示系统中交互的构件.
- M 是一个消息集合,$M=AM\cup IM$,指消息的类型有两种:箭头消息 AM 和内部消息 IM,允许 $AM\cap IM\neq\varnothing$.即某个消息既可以是箭头消息,也可以是内部消息.
- Con 是箭头消息交换时满足的限制集合,包括 $\{pum,pwc,puc,fum,fwc,fuc,ncon\}$.其中,$pum,fum$ 属于 unwanted 消息限制,分别表示 past unwanted 消息限制和 futuer unwanted 消息限制;pwc,puc,fwc,fuc 属于链限制,分别代表 past wanted 链限制、past unwanted 链限制、future wanted 链限制和 future unwanted 链限制.而 $ncon$ 则表示对应的限制为空.对于 unwanted 消息限制和链限制,它们的基本定义如下:$unwantedmessages=\{l_1,l_2,\ldots,l_n\}$.其中,$l_i=c_j.!m_i.c_k(i=1,\ldots,n)$,$m_i\in IM$;$c_j,c_k\in C$,"$!m_i$"表示 m_i 没有被交换.$chain=(l_1,l_2,\ldots,l_n)$,其中,$l_i=c_j.m_i.c_k(i=1,\ldots,n)$,$c_j,c_k\in C,m_i\in IM$.
- $Clock$ 是有限的时钟集合,2^{Clock} 是所有时钟变量的子集的集合,而 Φ(Clock)表示时钟约束的集合.

箭头消息和其限制的形式定义为 $msg=(type,label,sender,receiver,pastlab,futurelab)$.其中,$type\in\{e:,r:,f:\}$,表明箭头消息有 3 种类型,分别是正则、强制和错误类型;$label=(a,\delta,\psi)$ 表示每个箭头消息的标签由箭头消息 a、

时间约束δ和时钟重置变量ψ组成,其中,$a\in AM,\delta\in\Phi(Clock),\psi\subseteq 2^{Clock}$;$sender,receiver\in C$ 分别表示消息的发送和接收构件;$pastlab=(past,\delta',\psi')$表示 *pastlab* 由限制 *past*、时间约束δ'和时钟重置变量ψ'组成,其中,$past\in\{pum,pwc,puc,ncon\},\delta'\in\Phi(Clock),\psi'\subseteq 2^{Clock}$.类似地,$futurelab=(future,\delta'',\psi'')$表示 *futurelab* 由限制 *future*、时间约束δ''和时钟重置变量ψ''组成,其中,$future\in\{fum,fwc,fuc,ncon\},\delta''\in\Phi(Clock),\psi''\subseteq 2^{Clock}$.

额外的函数定义如下:函数 $timeline:AM\leftrightarrow T\setminus\{t_0,t_{n+1}\}$,箭头消息集合中的元素和时间线集合中的元素(去除 t_0 和 t_{n+1})之间存在着一一对应关系,即 $timeline(m_i)=t_i,1\leqslant i\leqslant n$;定义操作符 $timeline(m_i)\times timeline(m_j)\in\{strict,alt,par,loop,nop\},1<i<j<n$,指时间线 t_i 和 t_j 之间的消息可能具有 *strict* 操作符、*alt* 操作符、*par* 操作符、*loop* 操作符或没有操作符.

定义 3(合法 TPSC). 一个 TPSC 是合法的,要满足如下 3 个条件:1) 单个箭头消息一致性;2) 多个箭头消息一致性;3) 操作一致性.其中,

1) 设对于任意箭头消息和其限制形式为 $msg=(t,l,s,r,p,f)$,其中,$l=(a,\delta,\psi),p=(past,\delta',\psi'),f=(future,\delta'',\psi'')$,单个箭头消息一致性指:
   a) 单个箭头消息的限制一致性:如果 $t=f$:,则 $future=ncon$,即如果消息是错误类型,则不能有 *future* 限制.
   b) 单个箭头消息的时间约束一致性:如果$\delta(x),\delta'(x)$和$\delta''(x)$分别为 *a*,*past* 和 *future* 限制的时间约束,且$x\notin\psi'\cup\psi''$,则$\forall x'\in[\![\delta'(x)]\!],\exists x\in[\![\delta(x)]\!]$,满足 $x\geqslant x';\forall x\in[\![\delta(x)]\!],\exists x''\in[\![\delta''(x)]\!]$,满足 $x''\geqslant x$.即当 *a*,*past* 和 *future* 有相同的时间约束并且没有时钟重置变量时,则它们约束的时间上界应是不递减的.
2) 存在两个箭头消息及其限制 $msg_i=(t_i,l_i,s_i,r_i,p_i,f_i)$和 $msg_j=(t_j,l_j,s_j,r_j,p_j,f_j)$,其中,$l_i=(m_i,\delta_i,\psi_i),p_i=(past_i,\delta_i',\psi_i'),f_i=(future_i,\delta_i'',\psi_i''),l_j=(m_j,\delta_j,\psi_j),p_j=(past_j,\delta_j',\psi_j'),f_j=(future_j,\delta_j'',\psi_j'')$,并且$1\leqslant i<j\leqslant n$,多个箭头消息一致性是指:
   a) 两个连续箭头消息之间限制一致性:对于两个连续的箭头消息 msg_i 和 msg_j(令 $j=i+1$),msg_i 的 *future* 限制和 msg_j 的 *past* 限制不能同时存在,即如果 $future_i\neq ncon$,则 $past_j=ncon$;如果 $past_j\neq ncon$,则 $future_i=ncon$.否则,导致两个连续箭头消息的限制在语义上的不一致.
   b) 两个箭头消息时间约束一致性:记 $\varphi_1(x)\in\{\varphi_1,\varphi_1',\varphi_1''\}$,而 $\varphi_2(x)\in\{\varphi_2,\varphi_2',\varphi_2''\}$,并且在 msg_i 和 msg_j 之间没有对时钟变量 x 进行重置,则$\forall x_1\in[\![\varphi_1(x)]\!],\exists x_2\in[\![\varphi_2(x)]\!]$,满足 $x_2\geqslant x_1$.当同一个时间变量约束两个时间线递增的箭头消息,且该时间变量在这期间没有重置时,它们约束的时间上界也应是不递减的.
3) 操作一致性,对于两个消息 msg_i 和 msg_j,由定义 2 得 $timeline(msg_i)=t_i,timeline(msg_j)=t_j$.若它们之间存在 *strict*,*par*,*alt* 或 *loop* 操作时,则要求:
   a) *strict* 操作一致性:如果$\langle t_i,t_j\rangle=strict$,并且 $j=i+1$.当严格操作符作用在两个连续的箭头消息 msg_i 和 msg_j 之间时,需满足 $future_i=ncon$ 和 $past_j=ncon$,即 msg_i 无 *future* 限制且 msg_j 无 *past* 限制.同时,$s_i=s_j$ 或者 $s_i=r_j$ 或者 $r_i=r_j$.指这两个连续消息必须要有一个相同的发送者或接受者.
   b) *par* 操作一致性:如果$\langle t_i,t_j\rangle=par$,则对于任意 $msg_k,i\leqslant k\leqslant j,past_k=ncon,future_k=ncon$.即 *par* 操作中的所有消息都不允许具有 *past* 或 *future* 限制,否则,存在语义混乱.
   c) *par* 操作时间约束一致性:对于两个消息 msg_k 和 $msg_l,i\leqslant k\leqslant j,i\leqslant l\leqslant j$,则$\varphi_k=\varphi_l$.即,并发操作要求两个消息可以按照任意顺序执行.故为了避免时间冲突,它们所受到的时间约束也应保持一致.
   d) *loop* 操作一致性:$\langle t_i,t_j\rangle=loop$,则如果 $past_i\neq ncon$,则 $future_j=ncon$;如果 $future_j\neq ncon$,则 $past_i=ncon$.即由于循环操作需要最后一个消息和第 1 个消息相连,故在这两者之间,最后一个消息的 *future* 限制和第 1 个消息的 *past* 限制不能同时存在.
   e) *loop* 和 *alt* 操作和限制时间约束一致性:$\langle t_i,t_j\rangle\in\{loop,alt\},msg_k$ 和 $msg_l,i\leqslant k\leqslant j,i\leqslant l\leqslant j$,记$\Phi$为消息或限制受到所有时间约束的集合,对$\forall\varphi_1\in\Phi$和$\forall\varphi_2\in\Phi$,则$\varphi_k=\varphi_l$.由于选择、循环操作的语义将导致消息可能以不同的顺序进行交换,选择表示从多个条消息中选择 1 条或多条进行交换;而作用在多个消息的循环操作允许重复执行多个消息,故为了避免时间冲突,对于选择、循环中的消息和其相应的限制的时间约束都必须相同,即它们有相同的上界和下界.

2.3 TPSC的操作语义

本节将给出基于 TBA 的 TPSC 的操作语义,在给出 TPSC 的操作语义之前,首先定义 TBA,这里的 TBA 是根据需要扩展而来的.

定义 4(TBA). Timed Büchi Automaton(TBA)是一个七元组$\langle \Sigma,S,s_0,F,G,Clock,T\rangle$,其中,

- Σ是系统待交换的消息集合.定义转换标签集合 L,对于每个 $l\in L$,定义如下:$l=1|a|\neg a|a\wedge b$.其中,$a,b\in\Sigma$.1 表示允许任何消息发生;a 表示系统交换消息 a,$\neg a$ 表示系统不能交换 a,而交换了其他消息.最后,l 是由Σ中的元素组成的布尔表达式.
- S 是有限的状态集合.
- $s_0\in S$ 是初始状态.
- F 指有限的接受状态集合.注意,由于用 TBA 表示 TPSC 补的行为,故接受状态 s_{accept} 表示 TPSC 的错误状态.
- $G\subseteq S$ 指粘合状态的集合,一个 TBA 的粘合状态 $s_{glue}\in G$ 用来与下一个 TBA 的初始状态进行粘合而生成更复杂的 TBA.
- $Clock$ 是有限的时钟集合.
- $T:S\times S\times L\times\Phi(C)\times 2^C$ 是一个转换集合.一个转换 $t\in T$,记 $t=\langle s,s',l,\delta,\psi\rangle$表示从状态 s 到 s'的转换标签为 l,$\delta\in\Phi(C)$是时钟约束,而$\psi\subseteq Clock$ 是转换过程中时钟重置变量集合.

下面定义形式语义规则将 TPSC 转化为对应的 TBA.其中,规则分为基本语义规则和组合语义规则两类.基本规则讨论如何将基本的含时间约束的 TPSC 转化为 TBA;而组合规则则讨论如何将基本的 TPSC 生成的 TBA 粘合成一个复杂的 TBA,或者当 TPSC 中存在复杂的结构化操作(如选择、并发、循环等)的情况.

2.3.1 基本语义规则

下面定义 3 类带有时间约束的箭头消息的基本语义规则.对于每种类型的箭头消息,考虑单个箭头消息是松散的,或者是严格的操作,或者带有 unwanted 消息限制或 unwanted/wanted 链限制的情况.

对于每个规则,用操作语义来表示其生成的 TBA 的过程.每个规则的形式为

$$\frac{(t,l,s,r,p,f,op)\wedge l=(a,\delta,\psi)\wedge p=(past,\delta',\psi')\wedge f=(future,\delta'',\psi'')}{\text{具体的转换规则}}.$$

横线上方是指的带限制的消息(t,l,s,r,p,f,op),与定义 2 的区别是加入 op 表示单个箭头消息上是否有操作符.由于是单个箭头消息,故主要区分为是 *strict* 和无操作符(nop)两种.横线下方则是 TBA 具体的转换规则,即表示从初始状态 s_0 如何转换到粘合状态 s_{glue} 或接受状态 s_{accept}.根据 TBA 的定义,每个转换规则的形式为 $t=\langle s,s',l,\delta,\psi\rangle$.当箭头消息或限制没有时间约束和无时钟重置变量时,分别用λ和τ来表示.当箭头消息带有 *past* 或 *future* 限制时,一般从两个角度来考虑:一是箭头消息和限制具有相同的时间约束;二是箭头消息和限制之间存在额外的时间需求,这时需要在它们之间重置一个时钟变量.

下面给出定义具体转换规则时用到的一些其他变量,其中,v 是时钟变量的当前值.例如,$v(x)$和 $v(y)$分别表示时钟变量 x 和 y 的当前值,而 $v(x\mapsto 0)$和 $v(y\mapsto 0)$分别表示将时钟变量 x 和 y 重置为 0,$v(\tau)$表示转换过程中没有重置任何时钟变量.

“|”是一个外部选择符号,表示只能从中选一.例如,$v(x\mapsto 0)|v(\tau)$表示根据需求选择重置时钟变量或者是不对任何时钟变量重置.

图 3 显示的是在时间约束δ下正则消息的语义规则,简写为 ER(regular rules).ER 1 表示带有时间约束δ的 e:a 语义规则.其中,情形 1)表示当满足δ时,a 未发生而允许其他消息发生,TBA 依然停留在 s_0;情形 2)表示当满足δ时 e:a 发生,则 TBA 从初始状态 s_0 转换到粘合状态 s_{glue},这时可能有一个新的时钟变量 x 重置为 0.注意,对正则消息来说,如果当e:a在满足δ时没有交换,则系统不会发生错误,所以生成的TBA中没有表示错误行为的接受状态 s_{accept}.ER 2 表示在时间约束δ下带有 *strict* 操作符的正则消息 e:a 的语义规则,与 ER 1 不同的是,当满足δ时,在 e:a 发生之前不允许其他消息发生.

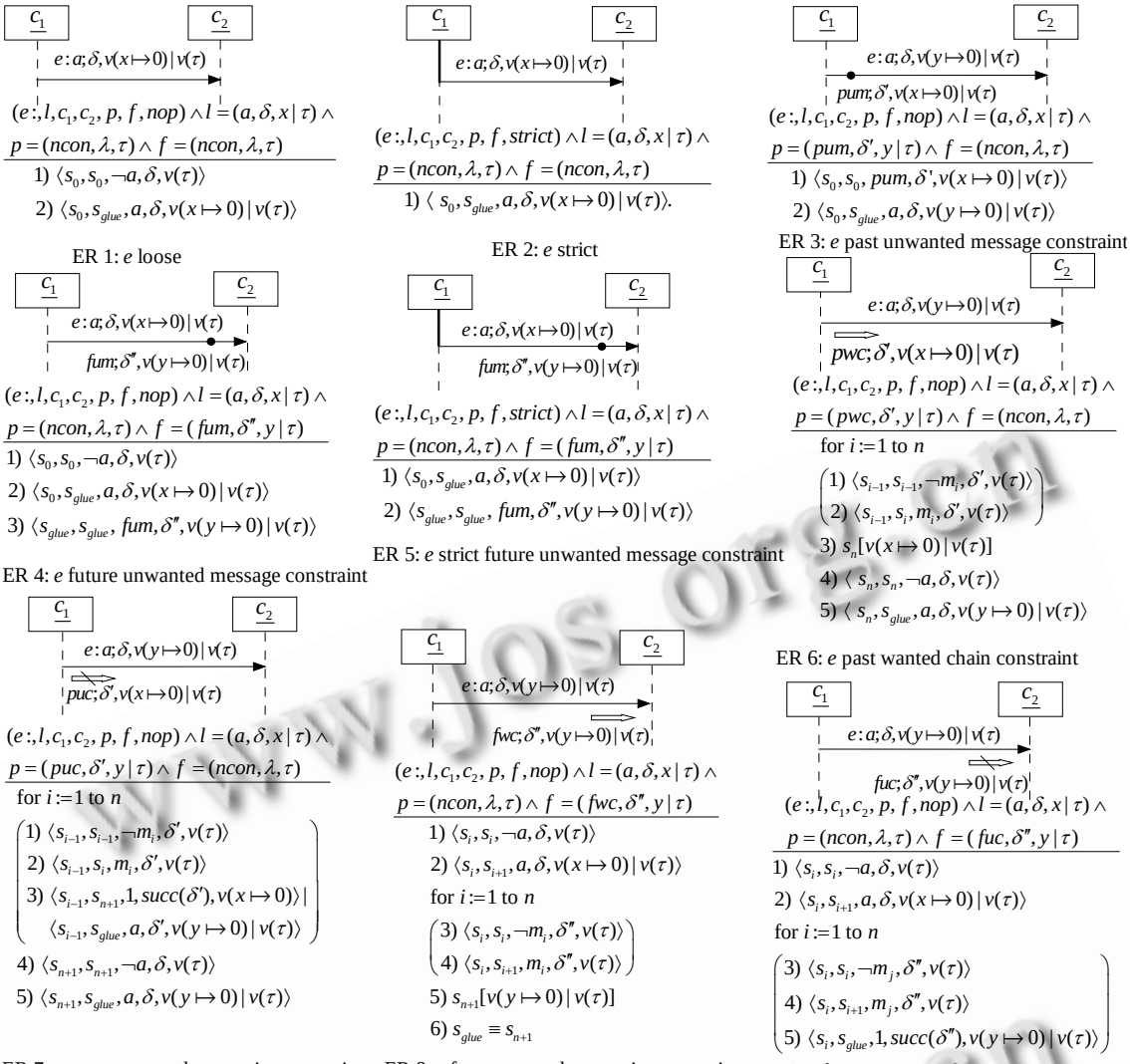


Fig.3 Semantics rules for regular messages

图 3 正则消息的语义规则

ER 3 表示在时间约束δ下带有 past unwanted 消息限制 *pum* 的 *e*:*a* 的语义规则,表示不期望 *pum* 中的消息发生.其中,情形 1)表示当满足δ'时 *pum* 中的消息没有发生,则 TBA 停留在 s_0.注意,当 *pum* 和 *e*:*a* 有相同的时间约束即$\delta'=\delta$时,无须置时钟变量;如果当 *pum* 和 *e*:*a* 之间有新的时间需求时,*pum* 中的消息在规定的时间内没有发生,则需重置时钟变量 *x*,这时,*e*:*a* 的时间约束是$\delta=\delta(x)$.情形 2)表示当满足δ时 *e*:*a* 发生,则 TBA 到达 s_{glue}.

ER 4 表示在时间约束δ下带有 future unwanted 消息限制 *fum* 的 *e*:*a* 的语义规则.其中,情形 1)和情形 2)表示当满足δ时 *e*:*a* 发生,则 TBA 到达 s_{glue},否则 TBA 停留在 s_0;情形 3)表示在 s_{glue} 时,*fum* 中的消息在规定的时间内没有发生,则 TBA 停留在 s_{glue}.ER 5 表示在时间约束δ下带有 future unwanted 消息限制 *fum* 的 *e*:*a* 的语义规则,并有严格操作符.与 ER 4 的区别是,在 *e*:*a* 发生之前不允许其他消息交换.

ER 6 表示在时间约束δ下带有 past wanted 链限制 *pwc* 的 *e*:*a* 的语义规则.用 for 循环来判断链限制是否完全发生,并期望在满足δ'时链限制完全发生.其中,情形 1)表示当满足δ'时链限制 *pwc* 中 m_i 之前的 *i*–1 条消息已

交换,而 m_i 没有交换,则 TBA 从初始状态 s_0 转换到中间状态 s_{i-1},并停留在原状态 s_{i-1}.情形 2)表示在满足δ'时链限制 *pwc* 中 m_i 交换,则 TBA 到达下一个状态 s_i.情形 3)表示链限制 *pwc* 完全交换后,TBA 到达 s_n.如果 *pwc* 和 *e*:*a* 之间存在新的时间需求,则重置时钟变量 *x*,否则不需重置.情形 4)和情形 5)表示如果在满足δ时消息 *a* 没有发生,TBA 将停留在状态 s_n,否则到达 s_{glue}.

ER 7 表示在时间约束δ下带有 past unwanted 链限制 *puc* 的 *e*:*a* 的语义规则.for 循环表示在满足δ'时不期望链限制完全发生.其中,情形 3)表示两种情况:其一,当链限制 *puc* 和 *e*:*a* 之间有新的时间需求时,*puc* 在到达δ'的后继时间时仍然没有完全发生,TBA 将转移到 s_{n+1},并且一个新的时钟 *x* 重置;其二,当链限制 *puc* 和 *e*:*a* 有相同时间约束δ时,在满足δ时,*e*:*a* 在 *puc* 链限制没有完全发生之前就发生了,TBA 将直接到达 s_{glue}.情形 4)和情形 5)表示在状态 s_{n+1} 时,消息 *e*:*a* 在满足δ时发生了,TBA 到达粘合状态,否则将停留在状态 s_{n+1}.

ER 8 表示在时间约束δ下带有 future wanted 链限制 *fwc* 的 *e*:*a* 的语义规则.其中,情形 1)和情形 2)表示当 *e*:*a* 满足δ时交换,TBA 到达下一个状态,否则停留在初始状态;情形 3)和情形 4)表示判断当满足δ''时,*fwc* 链限制是否完全交换完毕;情形 5)和情形 6)表示当满足δ'时,*fwc* 链限制完全交换,则 TBA 到达 s_{glue}.

ER 9 表示在时间约束δ下带有 future unwanted 链限制 *fuc* 的 *e*:*a* 的语义规则,其语义转换过程与 ER 8 类似,不同的是,在 for 循环中,在满足δ''时不期望链限制完全发生,即在时间约束满足 *succ*(δ'')时,链限制 *fuc* 没有完全发生,TBA 才能到达 s_{glue}.

图 4 表示在时间约束δ下强制消息的语义规则,简写为 RR(required rules).RR 1 表示强制消息 *r*:*a* 具有时间约束δ的语义规则.与 *e*:*a* 不同之处是,情形 3)表示当满足δ的前驱时间 *r*:*a* 发生,或满足δ的后继时间 *r*:*a* 仍然没有发生(由∨表示这两种情况的并),则 TBA 将到达 s_{accept}.RR 2 表示在时间约束δ下带有严格操作符的 *r*:*a* 的语义规则.与 RR 1 的区别是,不允许在 *r*:*a* 之前有其他消息发生,否则情形 1)表示 TBA 到达 s_{accept}.

RR 3 表示在时间约束δ下带有 past unwanted 消息限制 *pum* 的 *r*:*a* 的语义规则.其中,情形 1)表示当满足δ'时 *pum* 中的消息发生,则 TBA 到达 s_{accept}.注意,¬*pum* 表示 *pum* 不成立,即 *pum* 中的消息发生了;否则,情形 2)~情形 4)表示当满足δ'时 *pum* 中的消息未发生,在满足δ时 *r*:*a* 发生了,则 TBA 到达 s_{glue}.情形 5)表示在δ的前驱时间 *r*:*a* 发生或到达δ的后继时间 *r*:*a* 依然没有发生,则 TBA 到达 s_{accept}.

RR 4 表示在时间约束δ下带有 future unwanted 消息限制 *fum* 的 *r*:*a* 的语义规则.其中,情形 1)~情形 3)表示当满足δ时 *r*:*a* 发生,则 TBA 到达 s_{glue},否则到达 s_{accept};情形 4)表示在粘合状态在满足δ''时限制 *fum* 中的消息没有发生,否则,情形 5)表示 TBA 到达 s_{accept}.RR 5 表示在时间约束δ下带有 future unwanted 消息限制 *fum* 的 *r*:*a* 的语义规则,并有严格操作符.与 RR 4 的区别是,不允许在 *r*:*a* 之前有其他消息发生,否则将进入接受状态.

RR 6 表示在时间约束δ下带有 past wanted 链限制 *pwc* 的 *r*:*a* 的语义规则.for 循环表示期望在δ'时链限制 *pwc* 完全发生,如果链限制完全发生,则情形 4)重置一时钟变量 *y*.情形 3)表示两种情况:其一,当在δ'的前驱时间内 *pwc* 中的前 *j* 条消息发生了,或者在当到达δ'的后继时间仍然没有完全发生,则 TBA 到达 s_{accept};其二,*r*:*a* 和 *pwc* 链限制有相同的时间约束δ',在满足δ'时链限制没有完全发生而 *r*:*a* 发生了,则 TBA 到达 s_{accept}.情形 5)~情形 7)表示当满足δ时判断 *r*:*a* 是否发生:*r*:*a* 发生,则 TBA 到达 s_{glue};否则,到达 s_{accept}.

RR 7 表示在时间约束δ下带有 past unwanted 链限制 *puc* 的 *r*:*a* 的语义规则.其中,情形 1)和情形 2)表示在满足δ'时不期望 *puc* 链限制完全发生;情形 3)表示在满足δ'时 *puc* 没有完全发生,TBA 到达 s_{n+1};当 *puc* 和 *r*:*a* 之间有新的时间需求时,重置时钟 *x*;或者当 *puc* 和 *r*:*a* 有相同的时间约束时,在满足δ'时 *puc* 没有完全发生,*a* 已经交换,则 TBA 到达 s_{glue};否则,情形 4)表示 *puc* 在δ'内完全发生,则 TBA 到达 s_{accept}.情形 5)~情形 7)表示当满足δ时,*r*:*a* 发生,则 TBA 到达 s_{glue};否则,到达 s_{accept}.

RR 8 表示在时间约束δ下带有 future wanted 链限制 *fwc* 的 *r*:*a* 的语义规则.其中,情形 1)、情形 2)表示 *r*:*a* 在δ内是否发生,TBA 决定是否进入下一个状态;否则,情形 3)表示 TBA 到达 s_{accept}.接下来,for 循环表示期望 *fwc* 链限制在δ''内完全发生,如果没有完全发生,则情形 6)表示 TBA 到达 s_{accept};否则,情形 7)表示链限制完全发生,TBA 进入 s_{glue}.RR 9 表示在时间约束δ下带有 future unwanted 链 *fuc* 的 *r*:*a* 的语义规则.与 RR 8 不同之处是,RR 9 在满足δ''时不期望链限制完全发生.

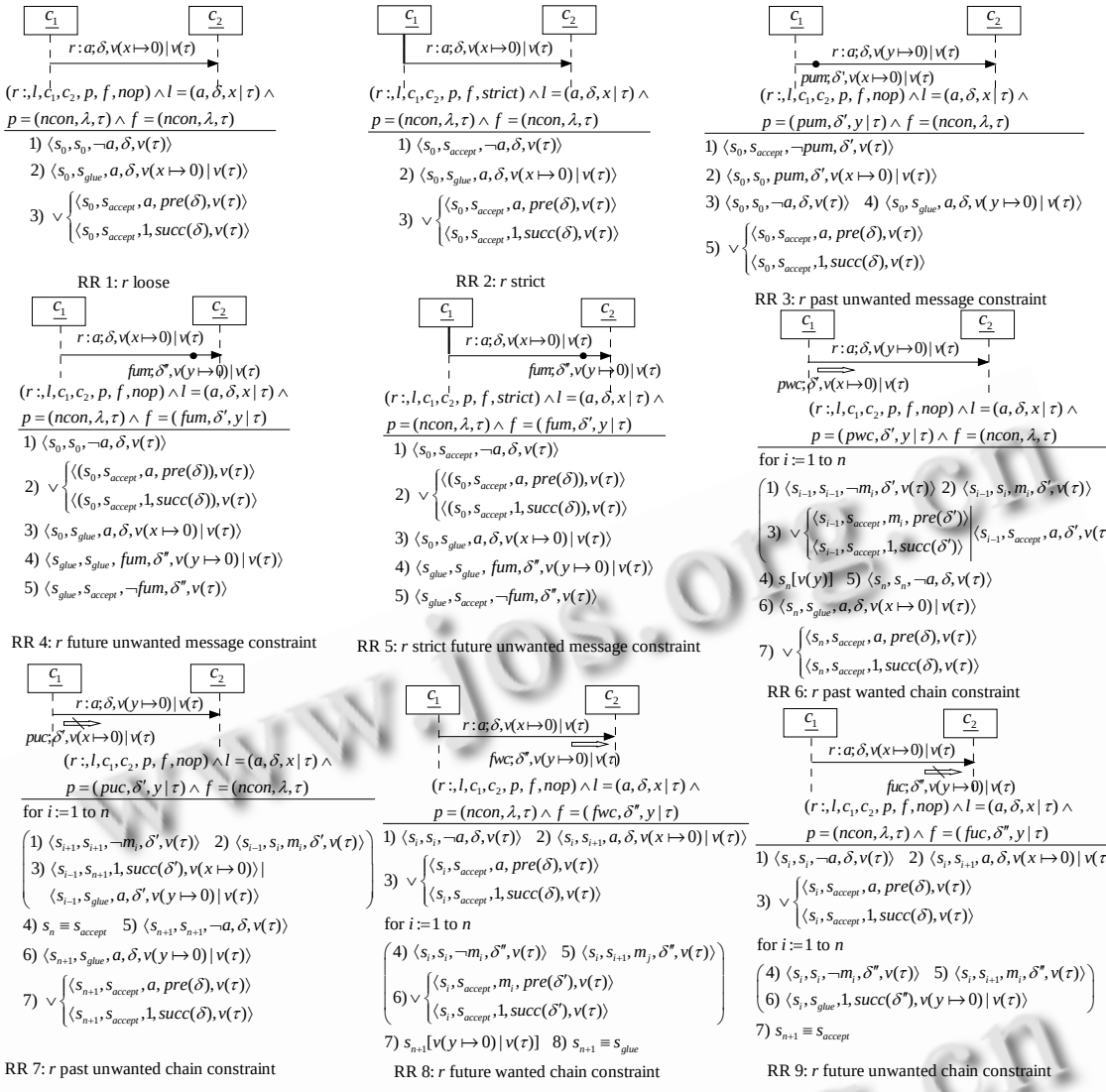


Fig.4 Semantics rules for required messages

图 4 强制消息的语义规则

图 5 表示在时间约束δ下错误消息的语义规则,简写为 FR(fail rules).即当 *f*:*a* 发生时,TBA 进入 s_{accept}.当 *f*:*a* 带有 past 限制时表示,当 past 限制在满足时间约束时成立;而当 *f*:*a* 在规定时间内发生时,则进入接受状态.由于一旦 *f*:*a* 发生了,系统就进入错误状态,故 *f*:*a* 没有 future 限制.FR 1 表示错误消息 *f*:*a* 的语义规则.其中,情形 1)、情形 2)表示当满足δ时 *f*:*a* 发生,则 TBA 到达 s_{accept};否则,情形 3)表示当在δ的前驱时间 *f*:*a* 发生或到达δ的后继时间 *f*:*a* 仍然没有发生,则 TBA 到达 s_{glue}.FR 2 表示在时间约束δ下带有严格操作符的 *f*:*a* 语义规则,故当有其他消息在 *f*:*a* 发生之前交换时,TBA 到达 s_{glue}.其他两条规则与 RR 1 相同.

FR 3 表示在时间约束δ下带有 past unwanted 消息限制的 *f*:*a* 的语义规则.其中,情形 1)表示当满足δ'时 *pum* 中的消息发生,则 TBA 到达 s_{glue};否则,情形 2)和情形 3)表示当满足δ'时 *pum* 中的消息没有发生,并且在满足δ时 *f*:*a* 发生了,则 TBA 到达 s_{accept};情形 4)表示 *pum* 中的消息没有发生,并且 *f*:*a* 在δ内也没有发生,则 TBA 到达 s_{glue}.

FR 4 表示在时间约束δ下带有 past wanted 链限制 *pwc* 的 *f*:*a* 语义规则.for 循环表示如果满足δ'时链限制没

有完全发生,则 TBA 到达 s_{glue};情形 4)~情形 6)如果满足δ'时链限制完全发生,并且 f:a 在δ内也发生,则 TBA 到达 s_{accept};否则,情形 7)TBA 将到达 s_{glue}.

FR 5 表示在时间约束δ下带有 past unwanted 链限制 *puc* 的 f:a 的语义规则.其中,情形 1)、情形 2)和情形 4)表示当 *puc* 中的消息在时间约束下发生完毕了,则 TBA 到达 s_{glue};情形 3)表示如果 *puc* 的消息在后继时间依然没有发生,则 TBA 到达 s_{n+1};或者当两者时间约束相同时,f:a 在 *puc* 没有发生完毕时交换,则 TBA 到达 s_{accept};情形 5)、情形 6)表示在 s_{n+1} 时,当 f:a 在新的时间约束下发生了,则 TBA 到达 s_{accept};否则,情形 7)表示 TBA 到达 s_{glue}.

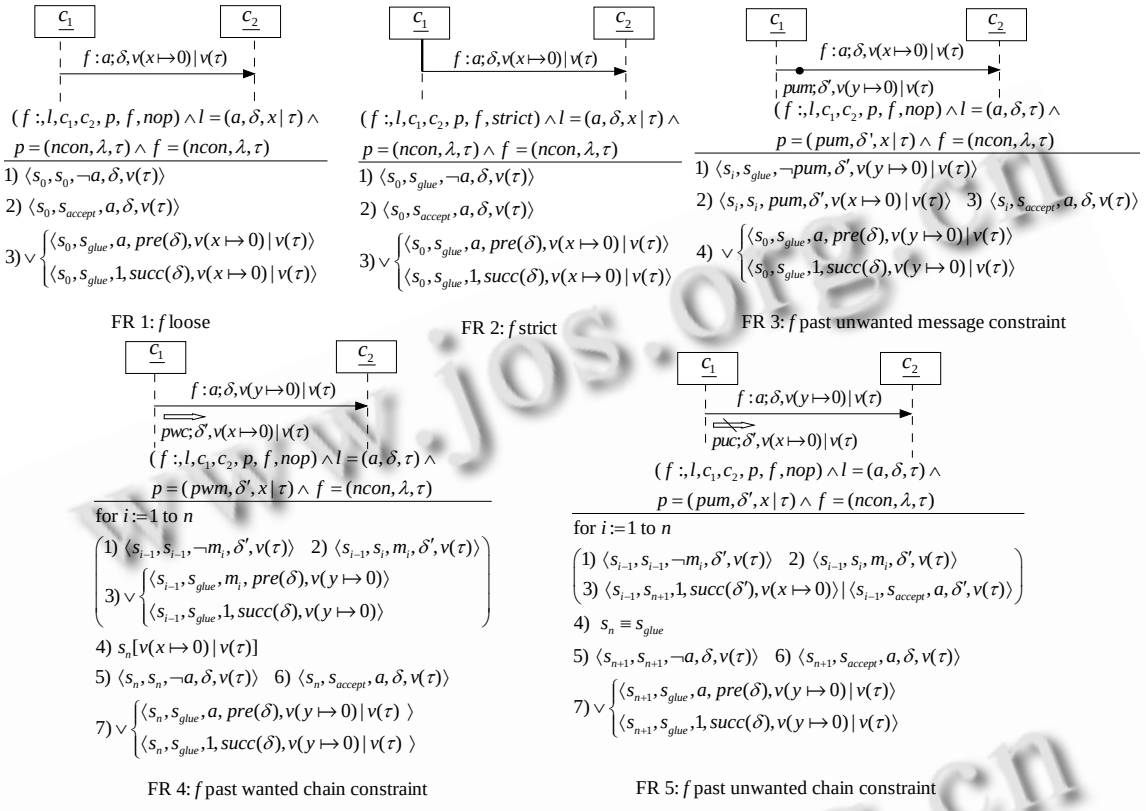


Fig.5 Semantics rules for fail messages

图 5 错误消息的语义规则

2.3.2 组合规则

通过使用基本规则可以将基本 TPSC 转换为对应的 TBA.然而,对比较复杂的实时属性,需用组合规则来形成更加复杂的 TPSC.这里有两种类型的组合规则:第 1 种情况,当存在两个基本的 TPSC 时,如何通过顺序组合产生复杂的 TPSC 的 TBA 的过程;第 2 种情况,当在 TPSC 图形中存在选择(alt)、并(par)和循环(loop)操作时,如何产生对应的 TBA.

(1) 基本规则之间的粘合

通常情况下,一个实时属性可有多个基本的 TPSC 以顺序的方式组成.这时,可通过 Merge 操作将这些基本 TPSC 以顺序的方式组合起来.下面的 Merge 组合将两个基本 TPSC 表示的 TBA(记为 M_1 和 M_2)进行组合,产生一个按顺序组合的 TBA(记为 M).下面给出组合 TBA 的形式化描述:

定义 5(merge 组合). 给定两个 TBA:M_i=⟨Σ,S,s_0,F,G,Con,$Clock$,T⟩,其中,i∈{1,2},一个新的 TBA,记为 M=⟨Σ,S,s_0,F,G,Con,$Clock$,T⟩是 M_1 和 M_2 的组合自动机,M=M_1·M_2.定义新生成的 M 中的各个元素如下:

- $M.\Sigma=M_1.\Sigma\cup M_2.\Sigma$;
- $M.S=M_1.S\cup M_2.S-M_2.s_0$;
- $M.s_0=M_1.s_0$;
- $M.F=M_1.F\cup M_2.F$;
- $M.G=M_2.G$;
- $M.Con=M_1.Con\cup M_2.Con$;
- $M.Clock=M_1.Clock\cup M_2.Clock$.

对于转换关系 $M.T$,记 $t=\langle s,s',l,\delta,\psi\rangle$,根据状态类型分以下几种情况进行讨论:

1. 对于属于 M_1 但不在 M_1 的粘合状态上的转换关系,属于新粘合的 TBAM 的转换关系.即如果 $t\in M_1.T$,并且 $s\notin M_1.G\vee s'\notin M_1.G$,那么 $t\in M.T$.
2. 对于属于 M_2 但不在 M_2 的初始状态上的转换关系,属于新粘合的 TBAM 的转换关系.即如果 $t\in M_2.T$,并且 $s\neq M_1.s_0\vee s'\neq M_1.s_0$,那么 $t\in M.T$.
3. 对于 M_1 的粘合状态和 M_2 的初始状态上的转换关系,分如下几种情况进行讨论:
   a) 如果 M_1 的粘合状态不存在转换关系,M_2 的初始状态也不存在转换关系,则该状态在 M 中也不存在转换关系.即如果 $s_1\in M_1.G,\neg\exists t\in M_1.T$,并且 $t.s=t.s'=s_1$,同时$\neg\exists t\in M_2.T$,并且 $t.s=t.s'=M_2.s_0$,则$\neg\exists t\in M.T$,并且 $t.s=t.s'=s_1$,其中,$s_1\in M.S$.
   b) 如果 M_1 的粘合状态或 M_2 的初始状态之一存在转换关系,则该状态在 M 中的转换关系为该转换关系.即如果 $s_1\in M_1.G,\exists t_1\in M_1.T$,并且 $t_1.s=t_1.s'=s_1$,或者$\exists t_2\in M_2.T$,并且 $t_2.s=t_2.s'=M_2.s_0$,则 $t=t_1\in M.T$ 或者 $t=t_2\in M.T$,并且 $t.s=t.s'=s_1$,其中,$s_1\in M.S$.
   c) 如果 M_1 的粘合状态存在转换关系,M_2 的初始状态也存在转换关系,则该状态在 M 中的转换关系为这两个转换关系相交.即如果 $s_1\in M_1.G,\exists t_1\in M_1.T$,并且 $t_1.s=t_2.s'=s_1$,同时$\exists t_2\in M_2.T$,并且 $t_2.s=t_2.s'=M_2.s_0$,则 $\exists t=t_1\wedge t_2\in M.T$,并且 $t.s=t.s'=s_1$,其中,$s_1\in M.S$.对于两个转换的交 $t=t_1\wedge t_2$,其中,t_1 是 M_1 中粘合状态的转换,t_2 是 M_2 中初始状态的转换,新生成的转换 t 的各个元素定义如下:

$$t.s=t.s'=t_1.s,$$
$$t.l=t_1.l\wedge t_2.l,$$
$$t.\delta=t_1.\delta\wedge t_2.\delta,$$
$$t.\psi=t_1.\psi\cup t_2.\psi.$$

(2) 选择、并行和循环组合

由于选择、并行和循环需要保持各个消息之间的时间约束相同,这样可以重用文献[6,14]中提出的关于 PSC 组合操作来自动生成各个组合操作的自动机,这里不再重复.

3 TPSC 的表达能力

TPSC 能够表示实时系统的关键时间属性,比如,通过指定强制类型的消息在规定时间内发生来表示实时系统的活性,或者是指定错误类型的消息在规定时间内不允许发生来表示实时系统的安全性,也可通过指定箭头消息在规定时间内的限制来表示实时系统的安全性.

然而,TPSC 的表达能力还需进一步细化的评估.Konrad 等人[13]提出了实时规约模式,以帮助用户表示实时系统的需求,提供了实时属性的模板,可用来表示规约中大部分的时间属性.可分为 3 类:持续时间(最小持续时间和最大持续时间)、时间周期(循环约束)和实时次序(有界响应和有界不变式).每个模式对应 5 种时间范围(globally,before r,after q,between q and r,after q until r).在文献[11]中我们已经证明,TPSC 能够表示 Konrad 等人提出的所有的实时规约模式.下面以有界响应模式为例来说明 TPSC 的表达能力.有界响应模式是指“当一个消息发生后,直到另一个消息发生的最大时间”.也就是说,当消息 p 发生后,消息 s 必须在 k 时间单位内发生.图 6 显示了使用 TPSC 表示的有界响应模式对应的 5 种时间范围,并转换成相应的 TBA.在生成具体的 TBA 中,用“!”

表示“¬”,“&”表示“∧”,“||”表示“∨”.在接受状态中,有一个标签为“1”的自我转换,表示一旦 TBA 进入接受状态(错误状态),系统就可能接受所有的输入而停滞不动.

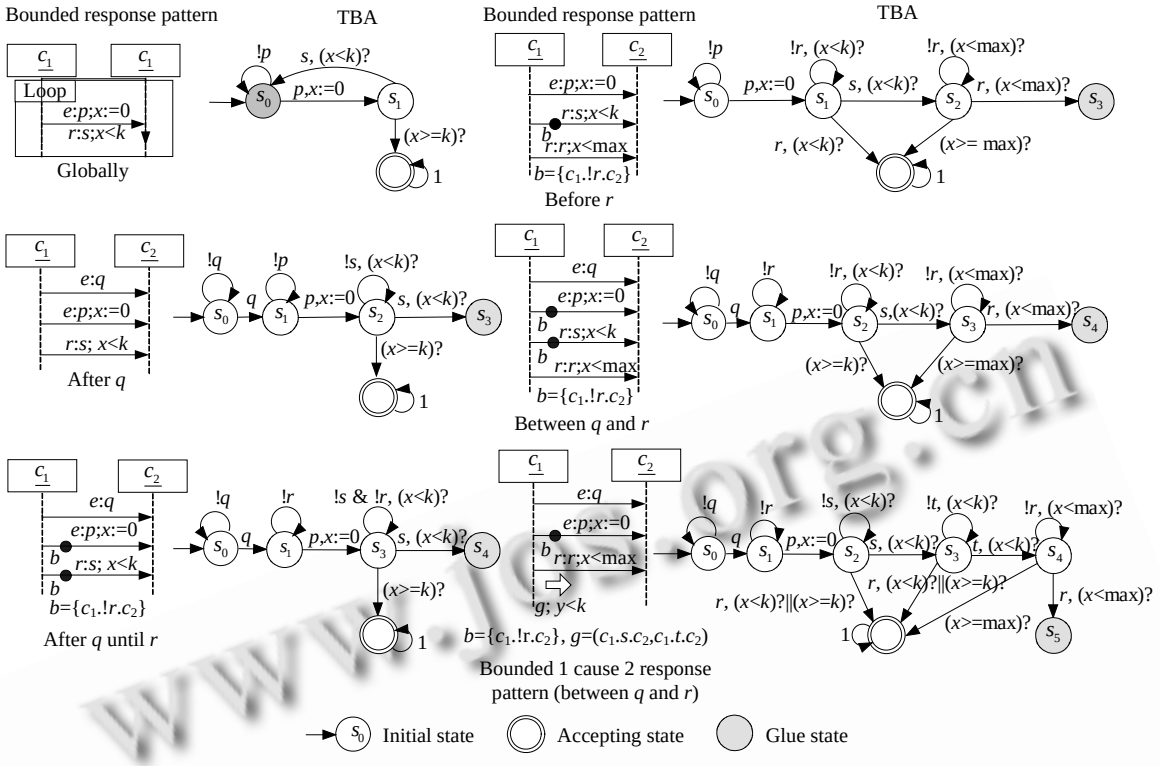


Fig.6 TPSC representation of bounded response real-time pattern

图 6 有界响应实时规约模式的 TPSC 表示

“Globally”表示有界响应模式在全局成立,由正则消息 *e*:*p* 和强制消息 *r*:*s* 表示,并应用 Loop 操作符.当 *e*:*p* 发生后,*r*:*s* 需在 *k* 时间单元内发生,如果 *r*:*s* 在 *x*<*k* 时没有发生,则发生错误.根据语义规则 ER 1 和规则 RR 1,生成的 TBA 如图 6 所示.其中,Loop 操作符表示这种条件是一直成立的.由于存在一个无限的循环,故开始循环时需重置一个时钟变量 *x* 为 0.

“Before *r*”表示“有界响应模式”在一个消息 *r* 之前发生,由正则消息 *e*:*p*、带有 past unwanted 消息限制 *b* 的强制消息 *r*:*s* 和强制消息 *r*:*r* 表示.当 *e*:*p* 发生以后,*r*:*s* 必须在 *k* 时间单元内发生,但是 *r*:*r* 必须在系统允许的将来某个时刻发生.其中,max 表示系统的最大时间约束,past unwanted 消息限制 *b* 表示 *r*:*r* 不能在 *r*:*s* 之前发生.根据对应的语义规则,同样可以生成相应的 TBA.

“After *q*”表示“有界响应模式”在一个消息 *e*:*q* 之后发生,由两个正则消息 *e*:*q* 和 *e*:*p*、一个强制消息 *r*:*s* 表示.当 *e*:*q* 和 *e*:*p* 发生后,*r*:*s* 必须在 *k* 个时间单元内发生.根据语义规则,可以转换成相应的 TBA.

“Between *q* and *r*”表示“有界响应模式”在消息 *e*:*q* 和 *r*:*r* 之间发生.即,当 *e*:*q* 和 *e*:*p* 发生时,*r*:*s* 必须要在接下来 *k* 个时间单元内发生,并且 *r*:*r* 必须要在将来发生.TPSC 由一个正则消息 *e*:*q*、一个带有 past unwanted 消息限制 *b* 的正则消息 *e*:*p*、一个带有 past unwanted 限制 *b* 的强制消息 *r*:*s* 和一个强制消息组成 *r*:*r*.其中,past unwanted 消息限制 *b* 表示 *r*:*r* 不能在 *e*:*p* 和 *r*:*s* 之前发生.

“After *q* until *r*”表示“有界响应模式”在消息 *e*:*q* 和 *r*:*r* 之间发生,其中,*r*:*r* 可以在将来不发生.所以,与“Between *q* and *r*”的不同的是,*r*:*r* 不一定在将来发生.

另外,除了 Konrad 等人提出的实时规约模式以外,TPSC 还能表达其他实时规约模式.如图 6 中的最后一个

是使用 TPSC 表示“有界的 1 原因 2 响应(bounded 1 cause 2 response)模式”中的“Between q and r”时间范围,由一个正则消息 e:q、一个带有 past unwanted 消息限制的正则消息 e:p 和一个带有 past wanted 链限制的强制消息 r:r 组成.它表示当 e:p 发生后,消息链 g(由消息 s 和 t 组成)必须在 k 时间单元内发生,并且必须在 e:q 和 r:r 之间发生.其中,past unwanted 消息限制 b 表示 r:r 不能在 e:p 之前发生;past wanted 链限制 g 表示内部消息 s 和 t 必须在 r:r 之前发生,并且满足时间约束 $y<k$.

4 实例分析

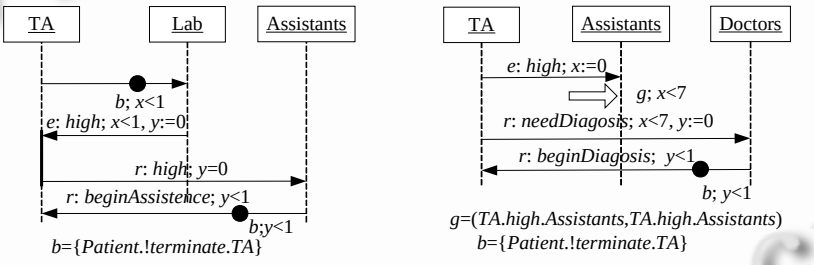
这一节结合应用实例,详细介绍如何使用 TPSC 来表示实时系统的实时属性需求.

TA(telecommunication assistance)是一个实时组合服务,用来帮助需要护理或诊断的病人与护士或医生进行非直接的交互.首先,病人发送请求来初始化一个服务过程 TA.接下来,根据病人的需求,TA 选择接受如下 3 种类型的消息:一是病人将他的关键参数(vitalParameters)发送给 TA,TA 再将关键参数发送给实验室(Lab)进行分析.由关键参数分析病人的警报等级(alarmLevel),结果有 4 种:高(high)、中(middle)、低(low)和需要诊断(needDiagonsis).根据不同的分析结果,TA 通知医生进行新的诊断或者通知最靠近病人的护士来进行护理,这里暗含了对时间的要求;二是 TA 能够直接接收到病人发送的护理请求;三是如果病人不再需要护理和诊断,他将中断该 TA 过程.考虑该过程中一些隐含的时间属性如下:

Prop 1:当 TA 将某个病人的关键参数发送到 Lab 后,且该病人没有中断该 TA 过程的前提下,Lab 必须在 1 个小时内将分析结果发送给 TA.如果分析结果为“高”,则 TA 必须马上通知附近的护士进行护理,并且护士必须在 1 个小时内响应,并开始护理.在此期间,病人不得中断 TA 过程.

Prop 2:如果在 1 周内有关某个病人的分析结果为“高”的次数超过 3 次(包括 3 次),则 TA 通知医生必须在一天内响应进行诊断.

图 7 显示了使用 TPSC 表示的两个属性.正如前述的,这些属性可以由多个消息以顺序的方式组合而成.



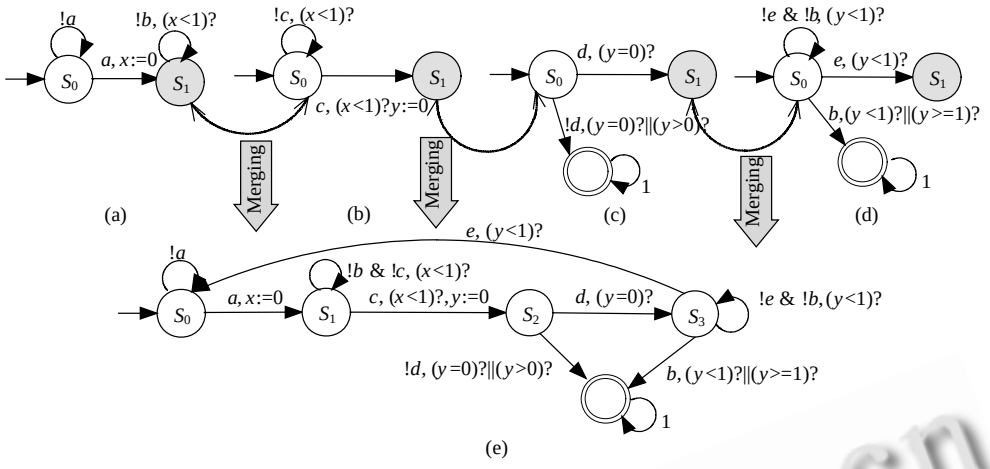
(a) TPSC representation of Prop 1 (b) TPSC representation of Prop 2

(a) 关于 Prop 1 的 TPSC 表示 (b) 关于 Prop 2 的 TPSC 表示

Fig.7 TPSC specifications of the two properties

图 7 两个属性的 TPSC 规约

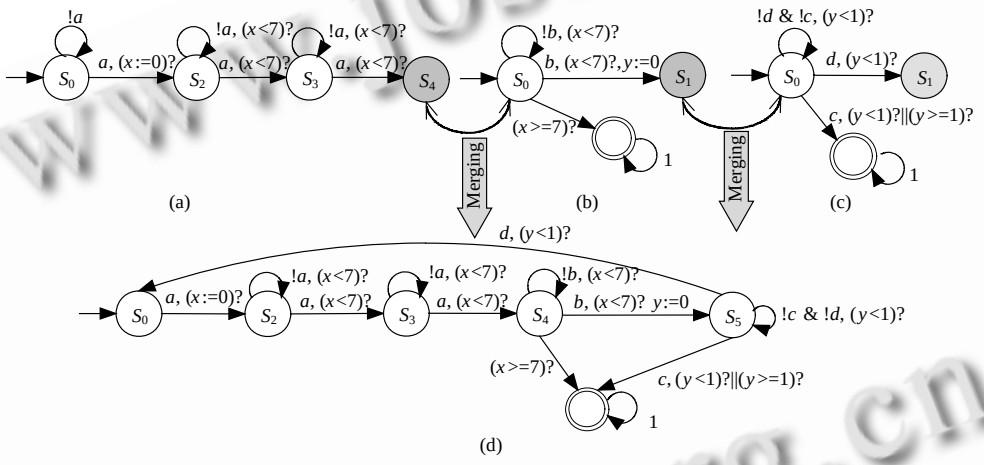
Prop 1 由 4 条消息组成,分别是:带有 future unwanted 消息限制的正则消息、正则消息、强制消息和带有 past unwanted 消息限制的强制消息.Prop 2 是由带有 future unwanted 链限制的正则消息、强制消息和带有 past unwanted 消息限制强制消息组成.注意,对于 Prop 2,需要加入一个空消息 e:*empty* 来表示属性;对于 Prop 1,时间单位是小时,而对于 Prop 2,时间单位是天.图 8 和图 9 分别显示了使用规则生成的最终 TBA.注意,对于每个由基本规则生成的组合 TBA,还需要将其中组合的最后一个自动机的粘合状态与第 1 个自动机的初始状态进行粘合,这样生成的自动机能够识别系统的无限输入.



(a) *TA.vitalParameters.Lab* (b) *Patient.terminate.TA* (c) *Lab.high.TA*
(d) *TA.high.Assistants* (e) *Assistants.beginAssistence.TA*

Fig.8 TBA for Prop 1

图 8 关于 Prop 1 的 TBA



(a) *Lab.high.Tele* (b) *Tele.needDiagosis.Doctor* (c) *Patient.terminate.Tele* (d) *Doctor.beginDiagosis.Tele*

Fig.9 TBA for Prop 2

图 9 关于 Prop 2 的 TBA

5 相关工作

当前,用于描述实时系统属性的时态逻辑通常是对原有的时态逻辑的时间扩展.最初,Koymans 等人[7]提出了量化的时态逻辑来表示时间属性.类似地,Alur 等人提出了两个表达力很强的时间规约 TCTL[7]和 TPTL[8],分别是对 CTL 和 PTL 的时间扩展.但由于这些逻辑公式的内在的复杂性,它们对应的时间扩展的时态逻辑同样很复杂,难以理解.如,对于有界回应实时模式 Between q and r 范围,可用如下的 TCTL 公式表示:

$$AG(q\wedge\neg r)\rightarrow A[((p\rightarrow((A[\neg r\ U_{\leqslant c}(s\wedge\neg r)]))\vee AG(\neg r))Wr)].$$

上述的 TCTL 公式很复杂,不易理解.而 TPSC 具有图形化表示属性的特点,则相对直观并且接近其自然语言的描述.

另外,Konrad 等人[13]扩展规约模式为实时规约模式,并提供了其结构化的语法,用于创建实时系统属性的自然语言表示,以帮助理解时态逻辑属性的含义.这些预定义的模式和结构化的英语语法能够在某种程度上帮助一般软件工程师手工书写时间属性.然而,他们没有为这些文本的模式提供图形化的接口,仍然不够直观.Gruhn 等人[16]用时间可观察自动机来扩展规约模式,同样没有提供直观的方式来表示这些模式,也没有讨论如何从需求中自动生成时间可观察自动机.

与本文的工作类似,文献[17]中的 Real-time graphical interval logic(RTGIL)对图形化的逻辑 GIL 进行了时间扩展.RTGIL 表达能力很强,这种逻辑与时态逻辑很类似,从而使它的公式很难理解.尽管 RTGIL 有图形化接口,对于一般的软件工程师来说仍然很难使用它,而不像 TPSC 一样有友好的用户界面.

6 结束语

本文对基于场景的规约(PSC)进行了时间扩展,从而使 TPSC 能够用来表示实时系统的时间属性.同时,定义了 TPSC 的形式语法和语义,可以根据语义规则自动生成对应的 TBA.然后,用实时规约模式来评估 TPSC 的表达能力.最后,还对 TPSC 进行了实例研究.

下一步的工作包括:开发支持 TPSC 的完全自动化的工具以帮助一般的软件工程师来表示实时系统中的时间属性;还考虑将 TPSC 应用在不同的形式验证环境中;在文献[18]中,我们已经提出一种方法——用 TPSC 来运行时验证 Web 服务组合的时间属性,进一步将考虑将 TPSC 整合到现有的模型检验工具中.

References:

[1] Dwyer MB, Avrunin GS, Corbett JC. Property specification patterns for finite-state verification. In: Proc. of the 21st Int'l Conf. on Software Engineering (ICSE'99). 1999. 411−420.

[2] Lin HM, Zhang WH. Model checking: Theories, techniques, and applications. Acta Electronica Sinica, 2002,12(30):1906−1912 (in Chinese with English abstract).

[3] Manna Z, Pnueli A. The Temporal Logic of Reactive and Concurrent Systems. New York: Springer-Verlag, 1992.

[4] Clarke EM, Emerson EA, Sistla AP. Automatic verification of finite-state concurrent systems using temporal logic specifications. ACM Trans. on Programming Languages and Systems, 1986,8(2):244−263.

[5] Autili M, Inverardi P, Pelliccione P. A scenario based notation for specifying temporal properties. In: Proc. of the 5th SCESM 2006 (ICSE 2006). 2006. 21−27.

[6] Autili M, Inverardi P, Pelliccione P. Graphical scenarios for specifying temporal properties: An automated approach. Automated Software Engineering, 2007,14(3):293−340. [doi: 10.1007/s10515-007-0012-6]

[7] Koymans R. Specifying real-time properties with metric temporal logic. Real-Time Systems, 1990,2(4):255−299. [doi: 10.1007/BF01995674]

[8] Alur R. Techniques for automatic verification of real-time systems [Ph.D. Thesis]. Stanford University, 1991.

[9] Alur R, Henzinger TA. A really temporal logic. Journal of the ACM, 1994,41(1):181−204. [doi: 10.1145/174644.174651]

[10] Zhang PC, Li BX, Sun MJ. A timed extension of property sequence chart. In: Proc. of the 11th IEEE High Assurance Systems Engineering Symp. (HASE 2008). IEEE Computer Society Press, 2008. 179−206.

[11] Zhang PC. Researches on the modeling and verification techniques for Web service compositions [Ph.D. Thesis]. Nanjing: Southeast University, 2010 (in Chinese with English abstract).

[12] Alur R, Dill DL. A theory of timed automata. Theoretical Computer Science, 1994,126(2):183−235. [doi: 10.1016/0304-3975(94)90010-8]

[13] Gerth R, Peled D, Vardi M, Wolper P. Simple on-the-fly automatic verification of linear temporal logic. In: Proc. of the 5th Int'l Symp. on Protocol Specification Testing and Verification (IFIP WG6.1). 1996. 3−18.

[14] Konrad S, Cheng BHC. Real-Time specification patterns. In: Proc. of the Int'l Conf. on Software Engineering (ICSE 2005). 2005. 372−381.

[15] Zhang PC, Zhou Y, Li BX, Xu BW. Property sequence charts: Formal syntax and semantic. Journal of Computer Research and Development, 2008,45(2):318−328 (in Chinese with English abstract).

[16] Gruhn V, Laue R. Patterns for timed property specifications. Electronic Notes in Theoretical Computer Science, 2006,153(2): 117−133.

[17] Moser LE, Melliar-Smith PM, Ramakrishna YS, Kutty G, Dillon LK. The real-time graphical interval logic toolset. In: Pro. of the Conf. on Computer-Aided Verification (CAV 1996). LNCS 1102, Springer-Verlag, 1996. 446−449.

[18] Zhang PC, Li BX, Su ZY, Sun MJ. Extending PSC for monitoring the timed properties in composite services. In: Proc. of the 15th Asia-Pacific Software Engineering Conf. (APSEC 2008). IEEE Computer Society Press, 2008. 335−342.

附中文参考文献:

[2] 林惠民,张文辉.模型检测:理论、方法与应用.电子学报,2002,12(30):1906−1912.

[11] 张鹏程.Web 服务组合建模和验证技术研究[博士学位论文].南京:东南大学,2010.

[15] 张鹏程,周宇,李必信,徐宝文.属性序列图:形式语法和语义.计算机研究与发展,2008,45(2):318−328.

张鹏程(1981－),男,江苏滨海人,博士,讲师,CCF 会员,主要研究领域为软件建模、分析和验证.

李雯睿(1981－),女,博士,讲师,CCF 会员,主要研究领域为形式化方法,Web 服务建模、分析和验证,Web 服务组合.

李必信(1969－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件分析、测试、验证,实证软件工程.