基于自动机的构件实时交互行为的形式化模型

贾仰理[1] 张振领[1] 李舟军[2]
(聊城大学计算机学院 聊城 252059)[1] (北京航空航天大学计算机学院 北京 100191)[2]

摘 要 采用形式化方法对复杂实时构件系统交互行为进行描述和验证,对于提高系统的正确性、可靠性等可信性质具有重要意义。分析了基于进程代数和自动机的构件交互行为形式化建模方法各自的优缺点,在此基础上提出了基于时间构件交互自动机的建模方法,给出了时间构件交互自动机的相关定义、组合和验证算法。时间构件交互自动机引入了时间限制、时间代价、时间代价计算半环、构件组合层次等概念,既能够描述构件交互情况,又能够清楚地表示出构件系统的体系结构信息和实时信息,便于对系统进行描述和验证。最后,结合具体应用给出了应用示例。

关键词 构件,交互行为,形式化描述,自动机
中图法分类号 TP311 **文献标识码** A

Formal Model of Component Real-time Interaction Behavior Based on Automata Theory

JIA Yang-li[1] ZHANG Zhen-ling[1] LI Zhou-jun[2]
(School of Computer Science & Technology, Liaocheng University, Liaocheng 252059, China)[1]
(School of Computer Science & Engineering, Beihang University, Beijing 100191, China)[2]

Abstract Formal specification and verification on complex real-time component system's interaction behavior have great significance of improving component systems' trustworthy properties such as correctness and reliability. The advantages and disadvantages of using process algebra and automata to model components interaction behavior were analyzed, and timed component interaction automata(TCIA) based modeling methods were presented based on the analysis. The related definition, composition and verification algorithm of TCIA were given. Models based on TCIA can clearly specify components' interaction behavior, architecture and real-time information in detail and are convenient to verify. Finally, an application example was introduced.

Keywords Component, Interaction behavior, Formal specification, Automata

1 引言

当前,构件技术在软件开发中的应用越来越广泛,并逐渐渗透到实时系统的开发中[1-3],已有越来越多的软构件系统部署应用到航空航天、军事过程控制、指挥通讯、交通管理等实时控制领域中。这类生命攸关和安全攸关领域的实时构件系统对正确性、可靠性等可信性质要求较高,一些情况下系统如果无法在要求的时间内完成规定动作,可能会引发巨大的灾难。但是,这类系统往往具有应用规模庞大、复杂性高等特点,系统各组成构件之间往往交互频繁、时序行为复杂,将它们组装到一起时,经常会出现构件间动态行为不相容、行为协议冲突等各种难以预料的错误。保障系统在功能和性能(时间)上的正确性、可靠性,防止巨大灾难的发生,是当前复杂反应式构件计算领域面临的主要挑战之一。

对复杂构件系统的行为进行形式化描述和验证,对于提高系统的正确性、可靠性、安全性具有重要意义。形式化描述使用具有严格数学定义语法和语义的语言来刻画软件系统及其性质,具有简明、无二义性、精确清晰等特点。在形式化描述模型的基础上,我们可以基于模型进行自动的形式化验证。文献[4]指出,在基于构件的软件开发过程中引入轻量级的形式化方法和成熟的自动分析技术,是一个重要的方向。文献[5]也提出,将软件构件化开发方法形式化和科学化,建立利用构件组装并开发出的高可信软件的方法,将是未来软件工程,尤其是 Internet 软件开发的重要方向。

本文在分析基于进程代数和基于自动机的构件交互行为形式化建模优缺点的基础上,针对 SOFA 构件模型[6],提出了基于时间构件交互自动机的建模方法,给出了时间构件交互自动机的相关定义、组合和验证算法。由于在构件交互自动机的基础上引入了时间限制、时间代价、时间代价计算半环、构件组合层次等概念,时间构件交互自动机既能够描述构件交互情况,又能够清楚地表示出构件系统的体系结构信息和实时信息,便于对系统进行描述和验证。

本文第2节介绍了构件交互行为建模的两类方法的优缺点和 SOFA 构件模型;第3节提出了时间构件交互自动机的

到稿日期:2009-10-15 返修日期:2010-01-27 本文受国家自然科学基金项目(90718017,60473057),博士学科点专项科研基金项目(20070006055)资助。

贾仰理(1976—),男,博士,讲师,主要研究方向为形式化方法、高可信软件技术等,E-mail:jyl@cse.buaa.edu.cn;张振领(1977—),女,硕士,讲师,主要研究方向为智能信息处理等;李舟军(1963—),男,博士,博士生导师,主要研究方向为形式化方法、信息安全、智能信息处理等。

相关定义和例子；第 4 节给出了时间构件交互自动机的组合相关定义、组合算法和验证方法；第 5 节给出了时间构件交互自动机的应用示例；最后总结全文并指出了下一步的工作。

2 构件交互行为形式化描述方法

典型的构件交互行为建模方法可分为基于进程代数的方法和基于自动机的方法两大类。基于进程代数的方法使用 CSP/Timed CSP、Pi 演算、FSP、行为协议(Behavior Protocol，BP)、CCS/Timed CCS 等进程代数语言对构件交互行为建模，这种方法往往可以很方便地集成到构件模型或体系结构描述语言中去。例如，Wright[7] 通过采用 CSP 的子集并进行一定的扩充来描述构件的计算行为和接口的交互行为，其中构件接口上的交互行为是构件计算行为在该接口上的投影。同时，其能够描述连接子的角色规约和角色之间的交互协议。对于实时构件系统，我们可以使用 Timed CSP 等对构件实时行为进行建模。Timed CSP[8] 在 CSP 的基础上引入了延迟、超时和实时中断等特殊操作，以便精确地描述时间行为。但因为 CSP 固有的静态特性，在构件的动态行为描述方面使用它存在不足，对动态软件体系结构描述能力有限。Darwin[9] 则运用 Pi 演算来描述构件的行为语义，但并未对构件的计算行为进行建模，从而不能支持复合构件行为的推导。

SOFA 构件模型则使用行为协议来描述构件行为。行为协议是一种类正则(Regular-like)语言，使用简单方便，可以自底向上分别描述接口、Architecture 和 Frame 上的行为。其中，接口行为协议定义了单个接口上可接受的服务调用次序，Architecture 行为协议定义了 Frame 层上的服务调用与子构件服务调用的关系，Frame 行为协议定义了在 Frame 层次上可以接受的、来自外部环境的、提供服务调用和对外部环境提出的请求服务调用之间的合法次序关系。SOFA 各层次上的行为协议能精确描述构件的行为以及构件行为与内部子构件行为的关系，实现了结构和行为逐层精化的思想。但这种形式化描述方法存在一个缺点，就是构件行为协议是附在接口、构件框架上，它本身无法体现构件的结构信息，也不支持实时构件行为的描述；另外，这种方法不能显式描述构件的交互情况，当构件组合时，不能充分保留构件的交互性质。其它基于进程代数的方法也存在类似的问题。

基于自动机的方法包括使用接口自动机、I/O 自动机以及描述实时行为的时间自动机(Timed Automata，TA)、时段自动机(Duration Automata，DA)、时间 I/O 自动机等来对构件行为进行建模。例如，时间自动机[10] 是有限自动机的一种扩展，它在状态机的基础上加入了时间的约束机制，用于时间系统的形式建模。它是 R. Alur 和 D. Din 首先提出来的，一直是一种用于描述时间系统的标准模型。一个 TA 具有有限个“位置”(Location)和多个实型值时钟。所有的时钟同步，记录了上次重置后流逝的时间。由 TA 的边，即位置的转换来重置时钟，因此时钟实际上记录的是相对于进入某个位置后流逝的时间。每个位置则对原子命题的值和时钟值加以约束，即当这些值满足一定条件时 TA 处于某个位置。接口自动机[11] 可以用来描述构件使用时对外部环境的输入假设和输出保证，包括构件向环境提供的方法被调用的先后次序以及构件对环境输出调用信息或结果的次序。时间接口自动机[12] 为输入/输出动作增加了时间约束扩展接口自动机来对实时交互行为建模。时间接口自动机在接口自动机的基础上增加了时钟变化，每个状态上均有一组时钟条件限制。时钟条件分别对输入/输出动作在该状态上的触发时间进行约束，对应地分别称为输入不变式和输出不变式。同时为接口自动机中的每个输入/输出动作增加一个卫式(guard)，它规定了该输入/输出动作的发生所必须满足的时间约束条件。

这类基于自动机的方法能高度形式化地为构件的交互建模，但很难描述层次构件体系结构中构件间的关系，而这种关系可能会影响构件的行为[13-15]。

另外，也有研究人员使用 Petri 网/时间 Petri 网(Timed Petri Nets)、标记迁移系统/时间迁移系统(Timed Transition Systern)、时序逻辑等方法来描述构件的交互行为及性质。

文献[13-15]提出了一种新的构件交互行为建模工具——构件交互自动机(Component Interaction Automata，CIA)，它为交互中的每个动作绑定了输出和输入构件名字(接口名)，因此允许构件组合中不同构件中动作的输入输出动作集合存在交集；严格定义并区分了构件组合中原输入、输出、内部动作和新产生的输入、输出、内部动作集合，可以根据组合层次需要选取不同层次的动作集合。构件交互自动机既能描述构件交互行为，又可以描述构件的多个层次的体系结构信息。

在 SOFA 构件模型中，构件是构件模板(Component Template)的实例。本质上构件模板 T 就是构件类型，由二元组$\langle F,A\rangle$组成。F(Frame)定义了构件模板的接口，体现了以黑盒的角度观察构件的思想；A(Architecture)定义了构件模板的内部体系结构(子构件以及子构件之间的绑定、委派、包含等关系)，体现了以灰盒的角度观察构件的思想。因此，SOFA 构件模型与大多数构件模型仅仅将构件视为黑盒复用实体不同，它能够在需要的时候给出复合构件的内部结构信息，使用的行为协议也简洁方便，因此一经提出就引起了广泛关注。

构件交互自动机建模特性与 SOFA 构件模型的黑盒和灰盒思想具有很强的一致性，很适合对 SOFA 构件的交互信息进行建模。因此，我们对 CIA 的理论进行了改进和扩展，提出了时间构件交互自动机(Timed Component Interaction Automata，TCIA)，并将它用于 SOFA 构件模型的构件交互行为建模。

3 时间构件交互自动机

首先给出时间构件交互自动机的相关定义，然后给出一个例子。

3.1 时间构件交互自动机

时间构件交互自动机涉及到时间限制、动作和时间代价、时间代价计算半环等概念，首先定义时间约束。

定义 1(时间约束) 时间约束(Ti)可表示为一个元组$[a,b]$，其中 $a,b\in N,a\leqslant b$。a,b 表示时间。由于计算机处理的时间是离散的，可以用自然数表示 a,b 的值。计时器从 0 开始，每嘀嗒一下，数字加 1；一个完整的处理事件结束，计时器清零。

时间约束将用于限制时间构件交互行为。

定义 2(动作) 我们使用动作对构件交互行为进行建模。动作是指在构件的接口上接受和发出的方法调用事件，

可分为输入动作(input action)、输出动作(output action)、内部动作(τ action)和空动作(ε action)。每个输入(输出)动作都有一个与之关联的接收(发送)该动作的接口及构件名。每个内部动作则有两个构件接口与之对应,一个内部构件在接口上发出动作,一个内部构件在对应的接口上接收该动作。ε 动作反映的不是方法调用事件,而是用来表示一段时间的流逝。

我们用符号'－'和'＋'来分别表示输出和输入动作。一个'－'符号表示一个动作的输出,而'＋'符号表示它的接收。

定义 3(时间代价) 每一个动作可以绑定一个时间代价,用来表示执行该动作需要花费的时间代价。本文用一个自然数来表示其值。时间代价可以用来表示程序运行代价,也可以用来表示网络环境下的动作命令或参数传输需要的时间代价。

定义 4(时间约束动作符号表) 用 $X\times Act\times Ti\times Tc\times(X\cup\{\pm\})$ 来表示时间约束动作符号。其中 X 是构件及接口的名称,$Act\in Actions$,Ti 是时间约束间隔的有限集合,Tc 是时间代价的集合。一个带时间约束信息的动作 $Act[a,b]$ 表示该动作(发出或响应)应不早于时刻 a、不迟于时刻 b 发生。带时间约束的空动作 $\varepsilon[t,t]$ 表示时间流逝 t 个时间单位。特别地,不带时间约束的动作 a 表示对于该动作我们不关心它的时间约束。X 是所涉及的构件名称的集合。时间约束动作符号表用 Σ 表示。

有了时间限制,动作只有在满足时间限制的情况下才能够执行成功。

符号 $(A,a[t_1,t_2][c_1],+)$,$(B,a[t_3,t_4][c_2],+)$,$(A,a[t_5,t_6][c_3],B)\in\Sigma$ 分别表示输出、输入和内部动作。

符号 $(A,a[t_1,t_2][c_1],+)$ 表示在时间间隔 $[t_1,t_2]$ 内构件/接口 A 发出请求动作 a,时间成本为 c_1;

符号 $(B,a[t_3,t_4][c_2],+)$ 表示在时间间隔 $[t_3,t_4]$ 内构件 B 接收到动作 a,时间成本为 c_2;

符号 $(A,a[t_5,t_6][c_3],B)\in\Sigma$ 表示在时间间隔 $[t_5,t_6]$ 内构件 A 发送的动作 a 为输出,构件 B 接收到动作,时间成本为 c_3。

在输出、输入动作匹配的情况下,我们要求 $c_1==c_2$,即动作发送的代价即为接收的代价(通讯成本)。

定义 5(时间代价计算半环) 时间代价计算半环可以用结构 $Cs=(C,|,||,0,1)$ 表示,其中 C 是时间代价值集合,$0,1\in C$,$|$ 和 $||$ 是作用于 C 上的操作符。

- $|$ 操作符用于计算两个顺序执行的动作的执行代价;
- $||$ 操作符用于计算两个并行执行的动作的最大执行代价;
- 0 为 $|$ 操作符的单位元;
- 1 为 $||$ 操作符的单位元。

定义 6(构件组合层次) 构件组合层次采用以下表示形式:

- 元组 $H_i=(n_{i1},n_{i2},\cdots,n_{im})$,$m\in N$,表示 $n_1,\cdots,n_m$ 共 m 个构件在同一层次上组合。
- 元组 $H=(H_1,H_2,\cdots,H_k)$,$k\in N$ 表示 $H_1,H_2,\cdots,H_k$ 共 k 个构件在同一层次上组合。

构件组合层次用来表示构件组合的层次性,我们用括号表示组合的递进性。最内层的构件最先组合,然后再和其外层的构件组合。

例如,$((C1,C2),C3)$ 表示 $C1$ 构件和 $C2$ 构件先组合,然后再和 $C3$ 构件组合。

特殊情况,$H=(C)$,表示 H 仅有一个元素 C,即是一个原子构件。

用 $N(H)$ 表示构件组合层次涉及的各构件名字的集合。

定义 7(时间构件交互自动机) 一个时间构件交互自动机是元组 $C=(S,Act,Ti,Tc,Cs,\Sigma,\delta,S_0,H)$。其中,

- S 是状态的有限集合;
- Act 是动作的有限集合;
- Ti 是在时间间隔的有限集合;
- Tc 是时间代价的有限集合;
- $Cs=(C,|,||,0,1)$ 是时间代价计算半环;
- $\Sigma\subseteq\{X\times Act\times Ti\times Tc\times(X\cup\{\pm\})\}$,$X\in N(H)$ 为时间约束动作符号表;
- $\delta\in S\times\Sigma\times S$ 为标记迁移的有限集合;
- $S_0\in S$ 为初始状态的非空集合;
- H 是该构件的构件组合层次。

前面在介绍时间构件交互自动机动作时,将动作分为输出、输入和内部动作。相应地,时间构件交互自动机中由输入、输出和内部动作所激发的迁移称为输入、输出和内部迁移,它们的集合记为 $\Delta^I,\Delta^O,\Delta^H$。

3.2 时间构件交互自动机示例

下面我们给出时间构件交互自动机的例子。考虑如图 1 所示的数据库应用系统,数据库构件向应用构件提供数据插入、删除、查询等服务,其内部又由数据处理构件和事务处理构件组成。数据库应用构件发出数据处理请求,等待响应结果。在接收到数据处理请求后,数据处理构件响应请求,启动事务处理,返回处理结果信息。

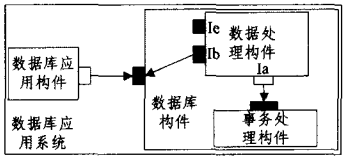

图 1 数据库应用系统构件关系

对系统构件分别用时间构件交互自动机表示如下:

$C1=(\{q_0,q_1\},\{a,b\},\{[0,2],[9,10]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(C1))$,其中 Σ,δ 等的信息如图 2 所示。

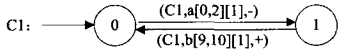

图 2 数据库应用构件时间构件交互自动机

$C2=(\{q_0,q_1,q_2,q_3\},\{a,b,c,d\},\{[1,3],[3,4],[7,8],[8,9]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(C2))$,其中 Σ,δ 等的信息如图 3 所示。

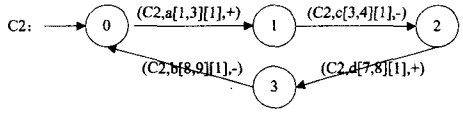

图 3 数据处理构件时间构件交互自动机

$C3=(\{q_0,q_1\},\{c,d\},\{[4,5],[5,6]\},\{1\},Cs,\Sigma,\delta,,\{q_0\},(C3))$,其中 Σ,δ 等的信息如图 4 所示。

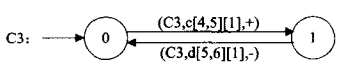

图 4 事务处理构件时间构件交互自动机

上面图示中，$C1$，$C2$ 和 $C3$ 分别为数据库应用构件、数据库处理构件和事务处理构件的时间构件交互自动机表示，a 为数据处理请求动作，b 为请求处理返回结果，c 为事务处理请求动作，d 为事务处理返回结果动作。

如果不考虑数据库构件的内部结构，其时间构件交互自动机可表示如下：

$C4=(\{q_0,q_1,q_2,q_3\},\{a,b,\tau\},\{[1,3],[8,9]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(C4))$，其中 Σ,δ 等的信息如图 5 所示。

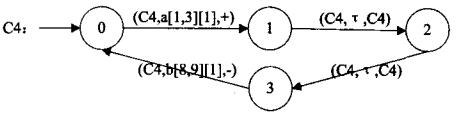

图 5　数据库构件时间构件交互自动机

对应 SOFA 构件模型 Frame 定义的思想，以黑盒角度观察数据库构件的交互行为，则可以用时间构件交互自动机建模如下：

$C4=(\{q_0,q_1\},\{a,b\},\{[1,3],[8,9]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(C4))$，其中 Σ,δ 等的信息如图 6 所示。

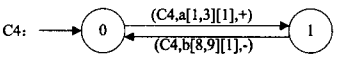

图 6　黑盒角度数据库构件时间构件交互自动机

4　基于时间构件自动机的组合与验证

时间构件自动机可以组合形成更高层次的自动机。我们先给出相关定义，然后给出组合和验证算法。

4.1　时间构件交互自动机的组合相关概念

构件交互自动机 I 和时间构件交互自动机 J 可以组合成更高层次的时间构件交互自动机。我们必须匹配所有的、相对应的输入、输出动作，计算它们新的时间代价，调整组合后迁移空间，以构造新产生的自动机。

定义 8(时间代价计算)　如果动作 a 的时间代价为 c_1，动作 b 的时间代价为 c_2，那么在给定时间代价计算半环 $Cs=(C,|,||,0,1)$下，a 和 b 顺序执行的代价为 $c_1|c_2$，并发执行的代价为 $c_1||c_2$。

定义 9(动作匹配)　如果时间构件交互自动机 I 输出为 $(N_i,a_i[t_{i1},t_{i2}][c_i],-)\in\Sigma_i,i\in N$，和时间构件交互自动机 J 有输入 $(N_j,a_j[t_{j1},t_{j2}[c_j],+)\in\Sigma_j,j\in N$，组合 I 和 J，我们称动作 a_i 匹配 a_j，如果 $c_i==c_j$，$[t_{i1}+c_i,t_{i2}+c_i]$和$[t_{j1},t_{j2}]$有重叠区域，而且能得到一个内部动作符号 $(N_i,a_k[t_{k1},t_{k2}][c_k],N_j)\in\Sigma_k$，其中 $t_{k2}=\min(t_{i2},t_{j2})$，$t_{k2}=\min(t_{i2},t_{j2})$，$c_k=c_i||c_j$。在不需要考虑内部动作细节的情况下，内部动作可以简记为 $(N_i,\tau,N_j)\in\Sigma_k$。

我们记：

- Act_{mo} 是所有的组合后匹配的输出动作的集合；
- Act_{mi} 是所有的组合后匹配的输入动作的集合；
- Act_{ni} 是在组合后新的内部产生的动作的集合；
- Tc_{nti} 是在组合中新产生的所有的时间代价集合；
- Tc_{nti} 是在组合中新产生的所有的时间间隔集合；
- Tc_{mti} 是 Act_{mo} 和 Act_{mi} 中已组合的所有时间间隔的集合。

如果动作已经组合产生新动作，则原输入、输出动作应该从新时间构件自动机中删除，新产生的内部动作等应该添加到相应的集合中。显然，这些变化将影响到 Σ_C 和 δ_C。

定义 10(构件组合宽度)　我们定义构件组合宽度为该层次上参与组合的构件的个数，用 $W(H)$ 表示，$W(H)\geqslant 1$。$W(H)$可以部分反映出单次组合的复杂度。

定义 11(构件组合深度)　我们用 $D(H)$表示构件组合深度，它是构件组合层次数量的度量，$D(H)\geqslant 1$。$D(H)$给出了组合的次数，可以部分反映出组合的复杂性。

特殊情况，$H=(C)$，表示 H 仅有一个元素 C，即是一个原子构件，其层次深度为 1。

设集合 $I\subseteq N$ 包含 k 个元素，对于每一个 $i\in I$，S_i 是一个集合，$\prod_{i\in I}S_i$ 为集合$\{(x_{i_1},x_{i_2},\cdots,x_{i_k})|(\forall j\in\{1,\cdots,k\}:x_{i_j}\in S_{i_j})\wedge\{i_1,i_2,\cdots,i_k\}=I\wedge(\forall j_1,j_2\in\{1,\cdots,k\}:j_1<j_2\Rightarrow i_{j_1}<i_{j_2})\}$。如果 $I=\emptyset$，则 $\prod_{i\in I}S_i=\emptyset$。而集合$(H_i)_{i\in I}$表示构件组合层次$(H_{i_1},H_{i_2},\cdots,H_{i_k})$。

下面给出时间构件交互自动机组合的定义。

定义 12(时间构件交互自动机的组合)　设 $S=\{(S_i,Act_i,Ti_i,Tc_i,Cs_i,\Sigma_i,\delta_i,S_{0_i},H_i)\}$，其中 $i\in n,n\subseteq N$，是一个时间构件交互自动机集合。如果参与组合的各构件名集合 $\{N((H_i)_{i\in I})\}$互不相交，则 $C=(\prod_{i\in n}S_i,Act_C,Ti_C,Tc_C,Cs_i,\Sigma_C,\delta_C,\prod_{i\in n}S_{0_i},(H_i)_{i\in I})$是一个 S 上的组合时间构件自动机：

- $Act_C=\bigcup_{i\in n}Act_i$；
- $Ti_C=\{\bigcup_{i\in n}Ti_i\}\cup Ti_{nti}\backslash Ti_{mti}$；
- $Tc_C=\{\bigcup_{i\in n}Tc_i\}\cup Tc_{ntc}\backslash Tc_{mtc}$；
- $Cs_C=\{\bigcup_{i\in n}Cs_i\}$；
- $\Sigma_C\subseteq\{X\times Act_C\times Ti_C\times Tc_C\times(X\cup\{\pm\})\},X\in N((H_i)_{i\in I})$；
- $\delta_C\subseteq\prod_{i\in n}S_i\times\Sigma_C\times\prod_{i\in n}S_i$ 是新产生的标记迁移的有限集合。

在组合需考虑构件间的输入输出动作匹配的情况下，匹配后不再使用的时间代价和时间间隔值应该删除，而组合中新产生的时间代价和时间间隔 Ti_{nti} 和 Tc_{ntc} 等应该添加到新构件模型中。

以前面数据库构件为例，该构件由数据处理构件和事务处理构件组成，其时间构件交互自动机模型可以由 $C2$，$C3$ 组合而成：

$C4=(\{q_0,q_1,q_2,q_3\},\{a,b,c,d\},\{[1,3],[3,5],[5,8],[8,9]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(C2,C3))$，其中 Σ,δ 等的信息如图 7 所示。

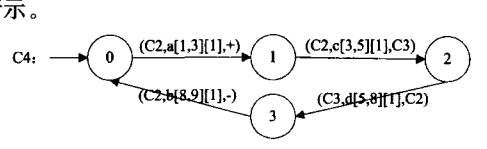

图 7　数据库构件时间构件交互自动机

构件时间构件交互自动机模型 $C4$ 能够给出构件的内部组合层次等信息，体现了 SOFA 构件模型的灰盒思想。

4.2　时间构件交互自动机的组合算法

我们给出基于时间构件自动机的组合算法。

输入：两个 TCIA，$C1=(S_1,Act_1,Ti_1,Tc_1,Cs_1,\Sigma_1,\delta_1,S_{01},H_1)$，和 $C2=(S_2,Act_2,Ti_2,Tc_2,Cs_2,\Sigma_2,\delta_2,S_{02},H_2)$

输出：TCIA1 和 TCIA2 的组合结果

$C=(S,Act,Ti,Tc,Cs,\Sigma,\delta,S_0,H)$

(1)由 S_{01} 和 S_{02} 生成初始状态 S_0

(2)将 Ti,Tc 中的所有时钟信息都置 0

(3)取 $SList1:=(S_1,ACT_1,\delta_1)$,$SList2:=(S_2,ACT_2,\delta_2)$

(4)While(SList 且 SList2 存在未标记状态) do

(5) 从 SList1 中标记状态

$(S_1,act_1,\delta_1)\in(S_1,ACT_1,\delta_1)$

(6) for(每个以 S_1 为起点的迁移 $\delta_1:S_1\times\Sigma_1\times S_1'$)

(7) 取 SList2 表中状态

$(S_2,act_2,\delta_2)\in(S_2,ACT_2,\delta_2)$

(8) for(每个以 S_2 为起点的迁移 $\delta_2:S_2\times\Sigma_2\times S_2'$)

(9) if(匹配 act_1 和 act_2)and 两个动作的时钟限制区域计算后有重叠

(10) 计算 a 的时间代价,加入 Tc,Cs,

(11) 计算 a 的时钟限制加入 Ti

(12) 生成内部动作并加入到 ACT

(13) 将新迁移关系加入 δ

(14) 在 SList2 中标记相应节点

(15) 转(19)

(16) else if(匹配 act_1 和 act_2)and 两个动作的时钟限制区域无重叠,时间限制匹配错误,返回时限错误信息

(17) else 取 SList2 中下一未标记元素,$S_2\leftarrow S_2'$ 转(8)

(18)将 S_1 对应的 TCIA1 中的信息直接加入 TCIA 中

(19)$S_1\leftarrow S_2$ 转(6)

(20)$H=(H_1,H_2)$

4.3 时间构件交互自动机的组合验证

我们可以方便地将上面的算法直接改为组合相容性错误检测算法。如果组合中的动作属于给定的动作集合,则需要在保证动作先后序列的前提下从另一 TCIA 中找到匹配动作。否则,如该动作没有匹配动作,或匹配动作不满足时限要求,则判定出现相容性错误,给出错误提示。

当然,我们也可以将匹配的动作直接转换为内部 τ 动作。如果在给定的组装动作集合上的动作存在没有转换的元素,则存在组装相容性错误。

5 基于时间构件自动机的应用实例

前面例子中用到实时数据库系统,下面我们考虑另一个使用 TCIA 对一个基于构件的实时反应系统建模的例子。铁路道口控制系统首次由 Rajeev Alur 和 Devid L. Dill 提出,并使用时间自动机对其进行了建模模型[10]。我们扩展这个例子,并使用 TCIA 对它进行建模。如图 8 所示,实时铁路道口控制系统由火车控制器(tc)、控制器(ga)、门控(gc)和报警系统(lm)4 个构件组成。当接近岔道时,tc 构件必须发出一个信号给 ga,然后等待响应。为简单起见,假设在这个例子中所有动作的时间成本值为 1 个时间单位。ga 一接收到 tc 发送的信号,就分别向门控系统 gt 和警报系统 lm 发送 lower 和 warnalarm 信号。然后,当门控系统得到减低栏杆的信号时,关闭岔道门(降低岔道栏杆),警报系统响应 warnalarm 信号。ga 给 tc 发出确认信号。火车经过岔道后,控制器会向岔道门发送信号,岔道门打开(升高栏杆)。

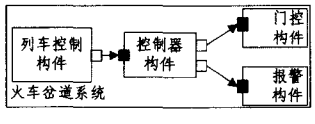

图 8 铁路道口控制系统构件图示

下面各图分别给出各构件的时间构件交互自动机模型。图中每个模型的事件后面都给出了时间限制信息。

$A1=(\{q_0,q_1\},\{pass,done\},\{[0,3],[10,11]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(A1))$,其中 Σ,δ 等的信息如图 9 所示。

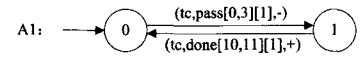

图 9 列车控制构件时间构件交互自动机

$A2=(\{q_0,q_1,q_2,q_3,q_4,q_5\},\{pass,done,alarm,lower,stop,raise\},\{[1,4],[5,6],[9,10],[30,50]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(A2))$,其中 Σ,δ 等的信息如图 10 所示。

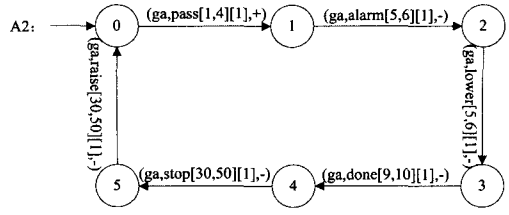

图 10 控制器构件时间构件交互自动机

$A3=(\{q_0,q_1\},\{alarm,stop\},\{[7,8],[31,51]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(A3))$,其中 Σ,δ 等的信息如图 11 所示。

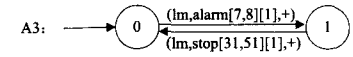

图 11 报警器构件时间构件交互自动机

$A4=(\{q_0,q_1\},\{lower,raise\},\{[7,8],[31,51]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(A4))$,其中 Σ,δ 等的信息如图 12 所示。

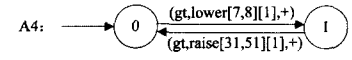

图 12 门控制器时间构件交互自动机

要得到 4 个构件的组合,需要对构件内的输入、输出动作进行组合匹配,计算组合后动作的时间间隔区间。显然,这里的发出和接收动作匹配,各动作的时间代价即通讯代价。例如$(tc,pass[0,3][1],-)$和$(ga,pass[1,4][1],+)$组合后得到$(tc,pass[0,4][1],ga)$。

组合后的 TCIA 模型如下:

$A5=(\{q_0,q_1,q_2,q_3,q_4,q_5\},\{pass,done,alarm,lower,stop,raise\},\{[0,4],[5,8],[9,11],[30,51]\},\{1\},Cs,\Sigma,\delta,\{q_0\},(A1,A2,A3,A4))$,其中 Σ,δ 等的信息如图 13 所示。

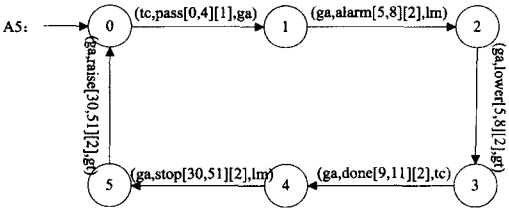

图 13 组合后时间构件交互自动机

为简单起见,仅给出了内部动作,匹配掉的输入和输出动作则没有表示出来。

如果将$(ga,done[9,10][1],-)$改为$(ga,done[12,13][1],-)$,它和$(tc,done[10,11][1],+)$在组合中匹配,则会出现时限错误。由于 ga 构件在系统启动后时间区间[12,13]内发出事件请求,而 tc 构件要求在时间区间[10,11]内接收到请求事件,显然发送时间迟于要求的接收时间,组合会出现时限匹配错误。

TCIA 可以用于描述和验证构件系统设计阶段构件交互的情况。如前面的例子所示，TCIA 可以描述复合构件系统的行为和时间约束信息，根据描述的信息，可以方便地验证系统的安全性、实时性及其它可信特性。

结束语 本文介绍了构件实时交互行为的形式化建模方法，分析了构件交互行为建模的两类方法及优缺点，提出了时间构件交互自动机的相关概念，给出了时间构件交互自动机的组合、验证方法。时间构件交互自动机结合了进程代数与自动机的优点，既能够详细描述构件交互情况，又能够清楚地描述构件系统的体系结构信息和实时信息，便于对系统进行描述和验证。我们将进一步开展构件实时行为形式化验证等相关技术的研究和实用工具的开发实现。

参 考 文 献

[1] Vasu A, Mubarak M. A component model for trustworthy real-time reactive systems development [C] // Formal Aspects of Component Software(FACS'07). Sophia-Antipolis, France: ENTCS, Elsevier, Sept 2007: 1-15

[2] Crnkovic I, Larsson M. Building reliable component-based software systems [M]. Norwood, MA: Artech House Publishers, 2002: 3-21

[3] Moller A, Akerholm M, Fredriksson J, et al. Evaluation of component technologies with respect to industrial requirements [C] // Proceedings of the 30th EUROMICRO Conference (EUROMICRO'04). Los Alamitos, CA, USA: IEEE Computer Society, 2004: 56-63

[4] Hatcliff J, Deng W, Dwyer M, et al. Cadena: An integrated development, analysis, and verification environment for component-based systems [C] // Lecture Notes in Computer Science. Berlin / Heidelberg: Springer, Volume 2984/2004, 2004: 160-164

[5] 杨芙清，梅宏，吕建，等. 浅论软件技术发展[J]. 电子学报，2002，30(12A)：1901-1906

[6] Plasil F, Visnovsky S. Behavior Protocols for Software Components [J]. IEEE Transactions on Software Engineering, 2002, 28 (11): 1056-1076

[7] Allen R, Garlan D. A Formal Basis for Architectural Connection [J]. ACM Transactions on Software Engineering and Methodology, 1997, 6(3): 213-249

[8] Reed G M, Roscoe A W. A timed model for communicating sequential processes [J]. Theoretical Computer Science, 1988, 58 (13): 249-261

[9] Magee J, Kramer J, Giannakopoulou D. Behaviour analysis of software architectures [C] // Donohoe P. editor. Proceedings of the 1st Working IFIP Conference on Software Architecture (WICSA1). San Antonio, Texas, USA: Kluwer Academic Pub, 1999: 35-50

[10] Alur R, Dill D L. A theory of timed automata [J]. Theoretical Computer Science, 1994, 126(2): 183-235

[11] de Alfaro L, Henzinger T A. Interface automata [C] // Proceedings of the Ninth Annual Symposium on Foundations of Software Engineering (FSE). ACM Press, 2001: 109-120

[12] de Alfaro L, Henzinger T A, Stoelinga M. Timed Interfaces [C] // Proceedings of the Second International Workshop on Embedded Software. LNCS 2491. Springer-Verlag, Berlin: SprinRer Verlag, 2002: 108-122

[13] Brim L, Cerna I, Varekova P, et al. Component-interaction automata as a verification-oriented component-based system specification [J]. ACM SIGSOFT Software Engineering Notes, 2006, 31(2): 1-8

[14] Varekova P, Zimmerova B. Component-interaction automata for specification and verification of component interactions [C] // IFM 2005 Doctoral Symposium on Integrated Formal Methods. Eindhoven, The Netherlands: Technische Universiteit Eindhoven, 2005: 71-75

[15] Zimmerova B, Varekova P, Benes N, et al. The Common Component Modeling Example: Comparing Software Component Models [C] // Component-Interaction Automata Approach (CoIn). LNCS, Springer-Verlag, August 2008, 5153: 146-176

(上接第 123 页)

用户、智能卡和 TC/TPM 间的相互认证，保证了用户使用平台的合法性和终端平台的完整性，有效地保护了信息的使用安全。新方案采用离线式用户认证，有效地提高了认证效率，在实现了用户身份的统一管理和认证的同时，降低了用户支持管理的复杂度。本文仅对单域的单点登录进行了研究，下一步的研究工作将考虑跨域单点登录方案的实施，以及跨域方案中用户访问的匿名实现。

参 考 文 献

[1] 黄琛，李忠献，杨义先，等. 一种新的兼容多种身份认证方式的 Web 单点登录方案[J]. 北京邮电大学学报，2006，29(5)：130-134

[2] Jason G. Single sign-on for your Web applications with Apache and Kerberos [EB/OL]. http://www.onlamp.com/pub/a/onlamp/2003/09/11/kerberos.html, 2008-10-22

[3] Dunne C. Build and implement a single sign-on solution [EB/OL]. http://www-106.ibm.com/developerworks/Web/library/wasinglesign/30, 2008-09-15

[4] Lin C H, Lai Y Y. A flexible biometrics remote user authentication scheme [J]. Computer Standards & Interfaces, 2004, 27 (1): 19-23

[5] Zheng Y, He D K, Yu W C, et al. Trusted computing-based security architecture for 4G mobile networks [C] // Proceedings of the IEEE Conference PDCAT. Dalian, China, 2005: 251-255

[6] Trusted Computing Group. TCG specification architecture overview [EB/OL]. https:// www.trustedcomputinggroup.org, 2008-11-25

[7] Trusted Computing Group. TPM main part 1 design principles [EB/OL]. https://www.trusted-computinggroup.org, 2008-12-11

[8] George P. User authentication with smart cards in trusted computing architecture [C] // Proceedings of the International Conference on Security and Management. Las Vegas, Nevada, USA, 2004: 25-31

[9] Pashalidis A, Mitchell C J. Single sign - on using trusted platforms [C] // Information Security. LNCS 2851. Berlin: Springer, 2003: 54-68

[10] Yoshihama S, Ebringer T, Nakamura M, et al. WS-attestation: efficient and fine-grained remote attestation on Web services [C] // Proceedings of the International Conference on Web Services. Florida: IEEE Press, 2005: 10-17