基于带数据约束实时系统的互模拟检测方法

李国拯,高 正
(南京航空航天大学 计算机科学与技术学院,江苏 南京 210016)

摘 要:带数据约束的实时系统是指一种既带有时间约束又带有数据变量约束的计算系统,其广泛存在于航空航天、工业控制、国防等安全攸关系统,并发挥着至关重要的作用。针对这类系统的形式化建模与验证是确保其正确性和可靠性的重要途径。文中首先研究了组合接口自动机、Z 语言、时间自动机的形式规范—CT-ZIA,其能同时描述带数据约束的实时系统的时序行为性质和数据结构性质;其次,为了研究该规范上的互模拟形式化验证,给出了 CT-ZIA 上的互模拟关系定义;然后,为了互模拟算法的可判定性,对 CT-ZIA 中的时钟进行等价划分,提出了有限论域 CT-ZIA 的定义;最后,基于有限论域 CT-ZIA 模型,给出了其上互模拟检测算法,并说明其正确性。

关键词:实时系统;接口自动机;Z 语言;时间自动机;互模拟检测
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2016)01-0006-04
doi:10.3969/j.issn.1673-629X.2016.01.002

An Approach of Bisimulation Checking for Real-time System Based on Data Constraints

LI Guo-zheng, GAO Zheng
(School of Computer Science and Technology, Nanjing University of Aeronautics & Astronautics, Nanjing 210016, China)

Abstract: Real-time systems with data constraints refer to computing systems both with time-bound and data variables constraints, which is widely used in safety-critical areas like aerospace, industry control, defense system, playing an important role. Formal modeling and verification for these systems is an important way to ensure the correctness and reliability of the systems. In this paper, study a specification model combining interface automata, timed automata and Z language, named CT-ZIA. This model can be used to describe temporal properties and data properties of real-time systems with data constraints. Second, in order to study formal verification for bisimulation in the specification, the bisimulation definition for CT-ZIA is given. Then, for the decidability of simulation algorithm, each clock of CT-ZIA is partitioning in equivalence, putting forward the definition of limited domain CT-ZIA's. Finally, give an algorithm for checking bisimulation relation between CT-ZIAs with finite domain and demonstrate the correctness of the algorithm.

Key words: real-time system; interface automata; Z notation; timed automata; bisimulation checking

0 引 言

带数据约束的实时系统[1]是指一种既带有时间约束,又带有数据变量约束的计算系统。飞行控制、核反应堆控制以及铁路调度控制等计算机控制系统都属于带数据约束的实时系统。这些系统中许多动作的完成都与时间相关,即要满足一定的时间限制,如某个动作要在一秒钟内完成;同时这些系统中数据变量之间也有一定的约束关系,如飞机的飞行速度可能跟气压、温度等有一定的约束关系。单一的规范技术很难适合这样的应用场景,所以这就要求利用多种能够描述系统行为各个方面的专门技术。基于这一想法,在文献[2]中提出了组合接口自动机、时间自动机和 Z 语言的形式规范。

接口自动机(Interface Automata)[3]是一种轻量级的基于自动机语言的组件规范语言,能够描述基于组件系统中组件及组件间的交互行为。时间自动机(Timed Automata)[4]是使用最为广泛的一种描述实时系统的数学模型,可简单看作带时钟变量的有限自动

收稿日期:2014-11-25 **修回日期**:2015-03-04 **网络出版时间**:2016-01-04
基金项目:航空科学基金(20128052064);中央高校基本科研业务费专项资金(NZ2013306);国家"973"重点基础研究发展计划项目(2014CB744903)
作者简介:李国拯(1990-),男,硕士研究生,研究方向为形式化方法、模型检测。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20160104.1608.084.html

机。Z[5]是基于一阶谓词逻辑和集合论的形式规格语言。

文中研究了一种组合接口自动机、时间自动机和 Z 语言的规范语言,即 CT-ZIA[2]。接口自动机是描述软件组件接口性质的直观模型,时间自动机是描述实时系统的模型,Z 可以描述状态的数据性质以及状态的变迁。为了对复杂实时系统进行规范说明,这里简单介绍下 CT-ZIA 的定义,粗略地讲,CT-ZIA 具有时间自动机风格和特点,但它的状态和操作是用 Z 语言来描述的。文中进一步研究了 CT-ZIA 上的互模拟关系,并给出了有限论域 CT-ZIA 上的互模拟检测算法。

1　连续时间 ZIA 规范

在文献[2]中,提出了 CT-ZIA 规范,并研究其上的模型检测[6]问题,下面直接给出其定义:

定义 1:一个基于连续时间的 ZIA(CT-ZIA) $P = \langle S_P, S_P^i, A_P^I, A_P^O, A_P^H, V_P^I, V_P^O, V_P^H, F_P^V, F_P^A, X, I, T_P\rangle$ 由如下部分组成:

(1) S_P 是状态的集合;

(2) $S_P^i \subseteq S_P$ 是初始状态的集合。如果 $S_P^i = \varnothing$ 那么 P 就为空;

(3) A_P^I, A_P^O 和 A_P^H 分别是不相交的输入动作集合、输出动作集合和内部动作集合,记所有动作集合为 $A_P = A_P^I \cup A_P^O \cup A_P^H$;

(4) V_P^I, V_P^O 和 V_P^H 分别是不相交的输入变量集合、输出变量集合和内部变量集合,记所有变量集合为 $V_P = V_P^I \cup V_P^O \cup V_P^H$;

(5) F_P^V 是一个映射,把 S_P 中的任意一个状态映射到用 Z 语言描述的状态模式;

(6) F_P^A 是一个映射,把 A_P^I 中的任意一个输入动作映射到用 Z 语言描述的输入操作模式,把 A_P^O 中的任意一个输出动作映射到用 Z 语言描述的输出操作模式,把 A_P^H 中的任意一个内部动作映射到用 Z 语言描述的内部操作模式;

(7) X 为时钟变量的非负实数有限集合, $C(X)$ 为 X 上时钟约束的集合,其语法定义如下:

$\Phi ::= x \prec c \mid c \prec x \mid \phi_1 \wedge \phi_2$

其中, $x \in X$; $\prec \in \{<, \leqslant\}$; c 为非负有理数。

(8)映射 $I: S_P \rightarrow C(X)$ 为每个状态赋以一时间约束,此约束称为节点不变量;

(9) $T_P \subseteq S_P \times A_P \times C(X) \times 2^X \times S_P$ 是状态之间转换关系的集合。如果 $(s, a, \varphi, \lambda, s') \in T_P$,那么表示在满足转换约束条件 φ 的前提下,通过动作 $a \in A_P(s)$,状态 s 可以迁移到新的状态 s',同时 $\lambda \subseteq X$ 中的时钟被重置为 0。并且, $I = ((F_P^V(s) \wedge F_P^A(a)) \setminus (x_1, x_2, \cdots, x_m) \Leftrightarrow F_P^V(t)[y_1'/y_1, y_2'/y_2, \cdots, y_n'/y_n])$。其中, $\{x_1, x_2, \cdots, x_m\}$ 是 $F_P^V(s)$ 上的变量集合; $\{y_1, y_2, \cdots, y_n\}$ 是 $F_P^V(t)$ 上的变量集合; $F_P^A(a)$ 上的变量集合是 $\{x_1, x_2, \cdots, x_m\} \cup \{y_1', y_2', \cdots, y_n'\}$ 的子集。

2　CT-ZIAs 间的互模拟关系

互模拟[7-8]是进程演算里的一个十分重要的概念,用来研究并发系统行为的一种方法。在软件系统里,互模拟通常用来描述抽象规格说明与可执行程序代码的相互转换过程,所以互模拟关系的目的在于形式化说明相同组件的抽象和具体版本之间的关系,例如接口的规范与其实现之间的关系。

对于某个 Z 模式 A,使用 $V^I(A)$ 代表 A 中的输入变量集合, $V^O(A)$ 代表 A 中的输出变量集合, $V^H(A)$ 表示 A 中的内部变量集合。

为了定义 Z 模式之间的互模拟关系,需要如下的符号。

定义 2:考虑两个 Z 模式 A 和 B,其中 $V^I(A) = V^I(B)$, $V^O(A) = V^O(B)$, $V^H(A) = V^H(B) = \varnothing$。如果对 $V^I(A) \cup V^O(A)$ 上的任意赋值 ρ,有 $\rho \mid = A \Leftrightarrow \rho \mid = B$,则模式 A 和模式 B 是互模拟的,记为 $A \simeq B$。

直观上, $A \simeq B$ 意味着模式 A 和模式 B 有相同的输入变量和相同的输出变量,并且模式 A 和模式 B 的输入变量范围和输出变量范围一样。

现在给出 Z 模式之间的互模拟关系,其描述了状态的数据结构属性间的互模拟关系。粗略地讲,对于模式 A 和模式 B,说 A 和 B 是互模拟的,就是两者的输入输出变量一样,并且两者的输入变量范围和输出变量范围也一样。

定义 3:考虑两个 Z 模式 A 和 B,使用符号 $A \cong B$,如果

(1) $V^I(A) = V^I(B)$, $V^O(A) = V^O(B)$;

(2) $A(x_1, x_2, \cdots, x_m) \simeq B(y_1, y_2, \cdots, y_n)$, 其中 $\{x_1, x_2, \cdots, x_m\} = V(A) - V^I(A) - V^O(A)$, $\{y_1, y_2, \cdots, y_n\} = V(B) - V^I(A) - V^O(A)$。

接下来给出 CT-ZIAs 间的互模拟关系。对 CT-ZIAs 来说,状态不仅有数据属性,还有行为属性。因此,CT-ZIAs 间的互模拟关系既包含数据属性间的互模拟关系,又包含行为属性间的互模拟关系。

定义 4:给定一个 CT-ZIA P,定义有序对 $(s, D_P) \in S_P \times \mathbb{R}$ 为 P 在某个时刻的状态,其中 $\mathbb{R}$ 是实数集合。下面定义 P 中的状态变迁:

(1)时延变迁: $(s, D_P) \xrightarrow{d} (s, D_P + d)$,其中 $d \in \mathbb{R}^+$,表示对任意的 $0 \leqslant e \leqslant d$,不变式 $I(s)$ 在 $(s, D_P +$

e) 都成立;

(2)动作变迁: $(s,D_P) \xrightarrow{a} (s',D_P')$,其中 $a \in A_P$,表示存在变迁 (s,a,φ,λ,s'),(s,D_P) 满足 φ,集合 λ 中的时钟重置为0。

定义5:考虑 CT-ZIA P 和 CT-ZIA Q,称二元对称关系 $R \subseteq (S_P \times \mathbb{R}) \times (S_Q \times \mathbb{R})$ 是互模拟关系,需对 $(s,D_P) \in S_P \times \mathbb{R}$,$(t,D_Q) \in S_Q \times \mathbb{R}$,$(s,D_P)R(t,D_Q)$ 推出如下条件满足:

(1) $F_P^S(s) \cong F_Q^S(t)$;

(2)对于时延 $d \in \mathbb{R}^+$,如果 $(s,D_P) \xrightarrow{d} (s,D_P+d)$,那么存在时延 $d' \in \mathbb{R}^+$,使得 $(t,D_Q) \longrightarrow (t,D_Q+d')$ 以及 $(s,D_P+d)R(t,D_Q+d')$;并且如果 $(t,D_Q) \xrightarrow{d} (t,D_Q+d)$,那么存在时延 $d' \in \mathbb{R}^+$,使得 $(s,D_P) \longrightarrow (s,D_P+d')$ 以及 $(t,D_Q+d)R(s,D_P+d')$;

(3)对于任意动作 $a \in A_p$,如果 $(s,D_P) \xrightarrow{a} (s',D_P')$,那么存在 $(t',D_Q') \in S_Q \times \mathbb{R}$ 有 $(t,D_Q) \xrightarrow{a} (t',D_Q')$,使得 $F_P^A(a) \cong F_Q^A(a)$,$F_P^S(s') \cong F_Q^S(t')$ 以及 $(s',D_P')R(t',D_Q')$;并且如果 $(t,D_Q) \xrightarrow{a} (t',D_Q')$,那么存在 $(s',D_P') \in S_P \times \mathbb{R}$ 有 $(s,D_P) \xrightarrow{a} (s',D_P')$,使得 $F_Q^A(a) \cong F_P^A(a)$,$F_Q^S(t') \cong F_P^S(s')$ 以及 $(t',D_Q')R(s',D_P')$。

如果 (s,D_P) 和 (t,D_Q) 是互模拟的,记作 $(s,D_P) \sim (t,D_Q)$。

定义6:如果存在 CT-ZIA P 和 CT-ZIA Q 间的互模拟关系 $\sim$、状态 $(s,0) \in S_P^i$ 以及状态 $(t,0) \in S_Q^i$ 使得 $(s,0) \sim (t,0)$,则称 P 和 Q 是互模拟的,并记为 $P \sim Q$。

3 有限论域 CT-ZIA

考虑一个 CT-ZIA P 以及其上的状态有序对 $(s,D_P) \in S_P \times \mathbb{R}$,显然,$P$ 是一个无限状态系统,为了互模拟算法的可判定性,需要首先将无限状态转为有限状态。为了获得 CT-ZIA 无限状态空间的有限描述,下面给出有限论域 CT-ZIA 的定义。

Alur,Courcoubetis 和 Dill 在文献[9-10]中提出一种时钟等价方法,把时间自动机等价为域自动机,但是按照 Alur 时钟等价方法构造出的域自动机,存在状态空间迅速膨胀爆炸的问题。在文献[2]中,笔者采用一种优化的时钟等价方法,并在此基础上定义了适合于优化时钟等价规则的域自动机,使等价后的域自动机状态数尽量少[11],这样就将 CT-ZIA 等价为一种带 Z 的域自动机。

大体思想如下:在构建时钟等价规则时,主要考虑时钟关键点的相互比较。例如,对状态 s 来说,约束条件 $x \geqslant 2$ 中的约束常量2是一个关键点,对于 x 的赋值 $v(x)$ 来说,当 $v(x) > 2$ 或 $v(x) < 2$ 时,无论取何值都是不相关的,所以仅需要考虑约束常量这个关键点就可以了。

综上所述,对时钟 x 而言,可分为三种情况:

(1) $v(x) < c_x$ 则 $v'(x) < c_x$;

(2) $v(x) = c_x$ 则 $v'(x) = c_x$;

(3) $v(x) > c_x$ 则 $v'(x) > c_x$。

类似的,对于 $y-x$ 的取值范围,只需要考虑时钟关键点的比较。

如上方法将 CT-ZIA 等价为一种带 Z 的域自动机,下面对 CT-ZIA 中 Z 模式再加以适当的约束,就得到一类特殊的 CT-ZIAs,其上的互模拟检测算法是可判定的。

定义7:给定一个 Z 模式 $S \triangleq [v_1:T_1;\cdots;v_m:T_m \mid P_1;\cdots;P_n]$,如果模式中的每个变量 v_i 有有限多可能的值,即变量的类型 T_i 有有限多元素,那么称 S 为有限域 Z 模式。考虑一个 CT-ZIA $P = \langle S_P,S_P^i,A_P^I,A_P^O,A_P^H,X_P,V_P^I,V_P^O,V_P^H,C_P,F_P^S,F_P^A,I_P,T_P\rangle$,称其为有限论域 CT-ZIA,需满足如下条件:

(1)对于每个 $s \in S_P$,$F_P^S(s)$ 是有限域 Z 模式;

(2)对于每个 $a \in S_P$,$F_P^A(a)$ 是有限域 Z 模式。

4 互模拟检测算法

给定一个 CT-ZIA P,可以将 P 转为等价的带 Z 域自动机,进一步约束 P 中 Z 模式的变量取值,就得到有限论域 CT-ZIA,记为 $F(P)$。这部分给出针对有限论域 CT-ZIAs 的互模拟检测算法 BC。假定 $F(P)$ 和 $F(Q)$ 是两个有限论域 CT-ZIAs,具体算法如下:

BC(P,Q) =

for each $p \in S_{F(P)}^i$, $q \in S_{F(Q)}^i$

$R_{p,q}$: = BCS(p,q)

return ($\bigvee_{p,q} R_{p,q}$)

BCS(p,q) =

$B_{p,q}$: = BCZ ($F_{F(P)}^S(p)$,$F_{F(Q)}^S(q)$)

if $\bigwedge_{a \in A(p,q)} (p \xrightarrow{a}\!\!\!\!\!\!/\;\;)$ or $\bigwedge_{a \in A(p,q)} (q \xrightarrow{a}\!\!\!\!\!\!/\;\;)$ then

return($B_{p,q}$)

else B: = $\bigwedge_{a \in A(p,q)}$ Match$_a$(p,q)

return ($B \wedge B_{p,q}$)

Match$_{a \in A(p,q)}$(p,q) =

if ($p \xrightarrow{a}$) and ($q \xrightarrow{a}\!\!\!\!\!\!/\;\;$) then

return(false)

if ($q \xrightarrow{a}$) and ($p \not\xrightarrow{a}$) then

return(false)

C_a: = BCZ($F^A_{F(P)}(a)$,$F^A_{F(Q)}(a)$)

for each ($p \xrightarrow{a} p_i$) and ($q \xrightarrow{a} q_j$)

$C_{i,j}$: = BCZ($F^S_{F(P)}(p_i)$,$F^S_{F(Q)}(q_j)$)

$D_{i,j}$: = BCS(p_i,q_j)

return ($\wedge_i(\vee_j(C_a \wedge C_{i,j} \wedge D_{i,j}))$)

BCZ(S,T)=

if ($V^I(S) \neq V^I(T)$) or ($V^O(S) \neq V^O(T)$)

then return(false)

else

V_S: = $V(S) - V^I(S) - V^O(S)$

V_T: = $V(T) - V^I(S) - V^O(S)$

E: = BCL($S \backslash V_S$,$T \backslash V_T$)

return(E)

BCL(M,N)=

return ($TV(\forall v^I_1:V^I_1;\cdots;v^I_m:V^I_m;v^O_1:V^O_1;\cdots;v^O_n:V^O_n \cdot (M \Leftrightarrow N)$

where $V^I(M) = \{v^I_1, v^I_2, \cdots, v^I_m\}$,

$V^O(M) = \{v^O_1, v^O_2, \cdots, v^O_n\}$, the type of v^I_k is V^I_k, and the type of v^O_k is V^O_k.

TV(LS)=

rewrite schema LS to an equivalent first order logical formula LF

if (LF is always true for any assignment on variables) then return(true)

else return(false)

假定 p 和 p' 是 $F(P)$ 中两个状态,a 是 $F(P)$ 中动作,用符号 $p \xrightarrow{a} p'$ 表示 $(p,a,p') \in T_{F(P)}$。$A(p) = \{a \mid \exists p' \cdot p \xrightarrow{a} p' \text{ and } a \text{ is an action}\}$。$A(p) \cup A(q)$ 简记为 $A(p,q)$。用 $p \xrightarrow{a}$ 表示存在状态 p',使得 $p \xrightarrow{a} p'$ 其中 $a \in A(p,q)$。因为 CT-ZIA 中状态是有限的,所以,$p \xrightarrow{a}$ 是可判定的。用 $p \not\xrightarrow{a}$ 表示由状态 p 执行动作 a 后,不存在任何后继状态。

在上面的算法中,如果对任意赋值 LS 总是返回 true,那么函数 TV(LS)返回 true。一般,因为一阶逻辑的重言式问题是不可判定的,所以函数 TV(LS)不能被实现,但是如果只考虑某些可判定的子逻辑,比如,逻辑公式的每个变量有有限多可能的值,这样的子逻辑的重言式问题就是可判定的。因为 LS 是有限域 Z 模式,每个变量仅有有限多的可能的取值。

不难证明上文提出的互模拟检测算法的正确性。下面给出上面算法的正确性说明:

引理 1:算法 BCZ(S,T)可以终止,并且是正确的,即算法返回 true 当且仅当 $S \cong T$。

证明:由 Z 模式的精化关系定义可得引理 1 是正确的。

引理 2:算法 BCS(p,q)可以终止,并且是正确的,即算法返回 true,当且仅当 $p \sim q$。

证明:函数 BCS(p,q)从有限划分后的系统初始状态有序对(p,q)出发,通过匹配从它们出发的变迁来检测 p,q 之间的相似性。在遍历变迁图的过程中,算法根据 CT-ZIA 的变迁从每个状态有序对产生接下来的变迁以及状态,然后通过模拟来匹配变迁,如果匹配成功,算法就进入了下一对状态有序对。函数 Match_a 在两个变迁图的乘积图上执行深度优先的算法,如果一个状态不能匹配另外一个状态的变迁,那么它们间就不是互模拟关系,函数返回 false,否则返回 true。将 P 和 Q 进行了有限划分得到有限状态系统 $F(P)$ 和 $F(Q)$,这就保证了仅会有限次调用函数 $\text{Match}_a(p,q)$,所以,函数 BCS(P,Q)总是会终止。

通过上面的引理,即可得出下面的命题:

命题 1:算法 BCS(P,Q)可以终止,并且是正确的,即算法返回 true 当且仅当 $P \sim Q$。

5　结束语

为了研究带有数据约束的实时系统,文献[2]中提出了组合接口自动机、时间自动机和 Z 语言三种形式规范说明技术的模型(CT-ZIA),研究了其上的模型检测问题,但是未对该模型进行进一步研究;文献[12]提出了一种基于离散时间的 ZIA 规范,但是基于离散时间的模型较适用于同步系统;文献[13]提出了混成 ZIA 模型,并给出了其上的近似精化关系定义,由于混成 ZIA 多数子集的不可判定性,未给出精化检测算法。

文中给出了 CT-ZIAs 间互模拟关系的定义,并对 CT-ZIAs 中的时钟进行等价划分,给出有限论域 CT-ZIA 的定义,在其上的互模拟检测算法是可以判定的,这对自动化验证组件的规范与实现的关系有重大意义。

参考文献:

[1] 李广元,唐稚松. 带有时钟变量的线性时序逻辑与实时系统验证[J]. 软件学报,2002,13(1):33-41.

[2] 倪水妹,曹子宁,李心磊. 带数据约束实时系统的模型检测[J]. 计算机科学,2014,41(5):254-262.

[3] de Alfaro L, Henzinger T A. Interface automata[C]//Proc of ACM SIGSOFT software engineering notes. [s. l.]: ACM, 2001:109-120.

(下转第 17 页)

任务页面上完成配置。产品代码在编译过程中可以查找开发工程师在编码过程中没有关注到的编译错误。编译为下一步的出包步骤生成进程文件。

5.3.4　出　包

出包任务在编译任务完成之后,根据全部编译任务提供的进程文件,在 ICP-CI 的任务页面上完成对出包的 ANT 脚本配置,生成版本包。将版本包部署到相应的目录下。

5.3.5　测试用例

在 ICP-CI 页面完成对测试用例任务的配置。在测试用例环境下对版本包进行自动化测试,完成版本包的初步测试。

6　典型案例

某公司有一个软、硬件结合的大型软件开发项目,采用的 ClearCase 版本是 7.0.1。持续集成主控服务器操作系统为 Windows 2003 Server,代理服务器操作系统为 Windows 7。参与持续集成的服务器共 20 台,1 台主控服务器和 19 台代理服务器。采用批处理脚本和 ANT 脚本编程完成软件的持续集成构建工作(完成进程编译、静态检查和版本出包等任务)。工作实践表明,采用基于 ClearCase 的配置管理和持续集成有助于提高软件质量,也便于项目经理和开发工程师及时了解工作进度和解决存在的问题。

7　结束语

长期的工作实践表明,配置管理在软件开发过程中占有重要地位。采用持续集成进行自动化构建可以使配置管理工作更有效率,有助于减少开发中埋藏 BUG 的风险,从而提高软件质量;同时也便于项目经理和开发工程师及时了解工作进度和解决存在的问题。应用软件的配置管理工作做好了,将很大程度上提高软件质量,降低软件开发成本,推动软件产业的健康发展。

参考文献:

[1] 刘文红. CMMI 项目管理实践[M]. 北京:清华大学出版社,2013.

[2] 刘江华,王　立,马　玲,等. 软件开发过程与配置管理:基于 Rational 的敏捷方案设计与应用[M]. 北京:电子工业出版社,2011.

[3] 袁肃蓉,王　萍,黄万民,等. 基于 ClearCase 的软件配置管理环境的规划和实施[J]. 海南大学学报:自然科学版,2009,27(1):54-59.

[4] Bellagio D,Milligan T. Software configuration management strategies and IBM rational ClearCase:a practical introduction[M]. 2nd ed. USA:IBM Press,2005.

[5] Lee K A. IBM rational ClearCase,ant and CruiseControl[M]. USA:IBM Press,2006.

[6] Tykal J. Best practices for using composite baselines in UCM[R]. [s. l.]:PrentieeHall,2004.

[7] McConnell S. Daily build and smoke test[J]. IEEE Software,1996,13(4):143-144.

[8] 罗时飞. 敏捷持续集成(CruiseControl 版):高效研发之道[M]. 北京:电子工业出版社,2008.

[9] 韩万江,姜立新. 软件项目管理案例教程[M]. 第 2 版. 北京:机械工业出版社,2009.

[10] Duvall P M,Matyas S,Glover A. 持续集成软件质量改进和风险降低之道[M]. 北京:电子工业出版社,2012.

[11] 李　进. 某公司软件持续集成改进的分析设计及实施[D]. 北京:北京邮电大学,2012.

[12] 徐仕成,杨邦荣. 基于 CruiseControl 的持续集成实现方案[J]. 计算机与数字工程,2007,35(4):169-171.

[13] 吴　奕. 软件配置管理工具在大型网站开发中的应用[D]. 上海:复旦大学,2011.

[14] 杨宏英,林长松. 基于 IBM Rational ClearCase 的配置管理及应用:中国软件行业协会[C]//CSSPI2006 第五届中国系统与软件过程改进年会论文集. 北京:出版地不详,2006:156-162.

(上接第 9 页)

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

[5] Spivey J M. The Z notation:a reference manual[M]. UK:Prentice Hall International Ltd,1992.

[6] 戎　玫,张广泉. 模型检测新技术研究[J]. 计算机科学,2003,30(5):102-104.

[7] 朱维军,刘保罗,周清雷. 时间自动机与信号自动机的互模拟算法[J]. 华南理工大学学报:自然科学版,2008,36(5):38-42.

[8] 李　娜,姚从军. 互模拟的一些基本性质[J]. 云南师范大学学报:哲学社会科学版,2010,42(5):68-73.

[9] Alur R. Techniques for automatic verification of real-time systems[D]. Stanford:Stanford University,1991.

[10] Alur R,Courcoubetis C,Dill D. Model-checking for real-time systems[C]//Proceedings of fifth annual IEEE symposium on logic in computer science. [s. l.]:IEEE,1990:414-425.

[11] 钱俊彦,赵岭忠,古天龙. 一种基于时间自动机的时钟等价性优化方法[J]. 计算机工程,2005,31(18):71-73.

[12] 狄杨思. 形式规范的自动验证算法的研究[D]. 南京:南京航空航天大学,2012.

[13] Cao Z,Wang H. Hybrid ZIA and its approximated refinement relation[C]//Proceedings of the 6th international conference on evaluation of novel approaches to software engineering. Beijing,China:[s. n.],2011:260-265.

基于带数据约束实时系统的互模拟检测方法

作者： 李国拯， 高正， LI Guo-zheng， GAO Zheng
作者单位： 南京航空航天大学 计算机科学与技术学院,江苏 南京,210016
刊名： 计算机技术与发展 ISTIC
英文刊名：
年，卷(期)： 2016(1)