设备自动巡检控制逻辑的层级时间自动机建模与验证

孙 程[1] 邢建春[1] 杨启亮[1,2] 韩德帅[1]
(解放军理工大学国防工程学院 南京 210007)[1]
(计算机软件新技术国家重点实验室(南京大学) 南京 210093)[2]

摘 要 地下建筑工程中的设备系统经常处于静止状态,为保证其在需要时能安全可靠地运行,需对设备进行定期的自动巡检。在自动巡检的过程中,设备自动巡检控制逻辑起到了举足轻重的作用。为了解决复杂的设备自动巡检控制逻辑造成的一系列问题,之前提出了一种层级有限自动机(HFA)的形式化模型,并利用HFA对设备自动巡检控制逻辑实现了行为建模,但并未添加时间属性,也未验证其正确性与可靠性。现提出一种层级时间自动机形式化模型,并利用它对设备自动巡检控制逻辑进行建模,再利用UPPAAL对其进行分析与形式化验证,分别验证其安全性、可达性、活性及时间约束,以此来确保其时效正确性与可靠性。这种建模与形式化验证方法弥补了之前无时间约束的漏洞,有效确保了设备自动巡检控制逻辑的正确性与可靠性。最终,该模型通过了模拟和验证,这充分证明了设备自动巡检控制逻辑是正确可靠的。

关键词 自动巡检,层级时间自动机,UPPAAL,模型检测
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.014

Modeling and Verifying Device Automatic Polling Control Logic Using Hierarchical Timed Automata

SUN Cheng[1] XING Jian-chun[1] YANG Qi-liang[1,2] HAN De-shuai[1]
(College of Defense Engineering, PLA University of Science and Technology, Nanjing 210007, China)[1]
(State Key Laboratory for Novel Software Technology at Nanjing University, Nanjing 210093, China)[2]

Abstract The device systems in the underground construction are often dormant. In order to ensure they can operate securely and reliably when they are needed, it is necessary to poll automatically on the devices at a regular interval. In the automatic polling process, the device automatic polling control logic is of great importance. To solve these problems which are caused by device automatic polling control logic, one formal model of hierarchical finite automaton(HFA) was set up and discussed previously, and modeled for device automatic polling control logic with HFA. But we didn't add time constraints and verify its accuracy and reliability. This paper proposed a formal model of hierarchical timed automata and modeled with it to the device automatic polling control logic. Then we analyzed and carried out formal verification with UPPAAL to the model. We verified its safety, reachability, activation and time constraint respectively to make sure the accuracy of timeliness and reliability. The method of modeling and formal verification makes up the disadvantage without time constraints in the former and ensures the accuracy and reliability of the device automatic polling control logic effectively. Finally, this model passes the simulation and formal verification, which proves the device automatic polling control logic is correct and reliable.

Keywords Automatic polling, Hierarchical timed automata, UPPAAL, Model checking

在地下建筑工程中,有一些设备系统平时处于静止状态,只有在应用时才运行,而且其运行时间远远少于停止时间,再加上建筑工程内部潮湿的环境极易造成设备锈蚀和短路,因此很难保证设备在真正需要时能正常运行。即使定期进行人工巡查,但由于设备多、安装分散等原因,很容易造成设备漏巡,不能及时发现故障。在前期工作中,我们提出了利用计算机来替代人工巡查的思想,开发出一套自动定期巡查控制软件系统[1],以确保设备随时可用;并利用层级有限自动机进行了分析和建模以及软件的实现[2],提出了一种适合自动巡检表达的基于复合关系的层级有限自动机形式化模型。

到稿日期:2015-11-30 返修日期:2016-03-05 本文受江苏省自然科学基金(BK20151451),装备预先研究基金(9140A06050215JB25091),计算机软件新技术国家重点实验室(南京大学)开放课题(KFKT2014B12)资助。

孙 程(1990—),男,硕士生,主要研究领域为软件安全,E-mail:820808595@qq.com;邢建春(1964—),男,博士,教授,主要研究领域为复杂智能信息系统、信息物理融合系统;杨启亮(1975—),男,博士,副教授,主要研究领域为自适应软件、信息物理融合系统;韩德帅(1990—),男,博士生,主要研究领域为软件自适应建模、软件形式化方法。

但对于自动巡检控制软件而言，不仅要求系统运行结果的逻辑正确性，而且要求时间正确性，时间约束会严重影响工程防护效能的发挥。在我们前期的研究工作中提出了适合自动巡检表达的基于复合关系的层级有限自动机形式化模型，但未验证其正确性与可靠性，而且未能体现出时间属性。因此刻画出设备自动巡检控制逻辑模型的时间属性，并借助 UPPAAL 工具验证其时效正确性显得至关重要。

目前已有的时间自动机建模与检测方法多种多样，各有利弊，如运用离散语义下的时间自动机[3]进行模型检测虽然效率较高，却不适合对异步系统进行检测。连续语义下的时间自动机[4]既可以描述同步系统，也可以描述异步系统，但一般情况下进行模型检测的复杂度很高[5]。而设备自动巡检控制逻辑比较复杂，层级性明显，既要整体考虑，又要兼顾个体。目前的研究方法不适应于设备自动巡检控制逻辑的建模与检测，针对以上这些问题，本文提出层级时间自动机（Hierarchy Timed Automat，HTA）模型，用以进行设备自动巡检控制逻辑的模型检测。

本文第 1 节介绍了时间自动机的基本理论和模型检测工具 UPPAAL；第 2 节描述了设备自动巡检控制逻辑的流程；第 3 节提出了层级时间自动机，并用其对设备自动巡检控制逻辑进行建模；第 4 节对模型进行了形式化验证；第 5 节简单讨论了模型检测的相关工作；最后总结全文。

1　预备知识

1.1　时间自动机理论

由 Alur 和 Dill[4]所提出的时间自动机是一种可以描述和分析实时系统行为的计算机模型。它在有限状态自动机的基础上添加了时间约束，用以刻画连续变化的时间，从而可以处理实时系统，而且可以判定任一状态是否可达。时间自动机的状态空间可以划分为有限个状态区域[6]，从而实现将无限问题的求解转换为有限问题的求解。时间自动机使用有限个变量来表示时间，用一个约束条件来注释状态转换图，这个与时间有关的约束条件决定了状态转换发生的时机[7]。

定义 1（时钟约束[4]）　对于一个时钟变量有穷集合 X，时钟约束 Φ 的集合 $\Phi(X)$ 的形式语法为：

$$\Phi ::= x \leqslant c \mid c \leqslant x \mid x < c \mid c < x \mid \phi_1 \wedge \phi_2$$

其中，x 是 X 中的一个时钟，c 是 Q 中的一个常量。

定义 2（时钟解释[4,8]）　一个在 X 上的时钟解释 v 是指给每个时钟分配一个实数值，即 $X \to R$ 是一个从 X 到非负实数集 R 的一个映射。X 的一个时钟解释 v 满足 X 上的一个时钟约束 φ，当且仅当依照 v 给出的值，φ 为真。

定义 3（时间自动机）　时间自动机 A 是一个六元组 $\langle \Sigma, S, S_0, X, I, E \rangle$，其中：$\Sigma$ 是一个有穷标记集合；S 是一个有限状态集合；$S_0 \in S$ 是开始状态集合；X 是一个有限时钟集合；I 是一个映射，它给每个位置 s 指定 $\Phi(X)$ 中的一些时钟约束；$E \subseteq S \times \Sigma \times 2^x \times \Phi(X) \times S$ 是转换集合，一个转换 $(s, \varphi, a, \lambda, s')$ 表示在符号 a 下从位置 s 到位置 s' 的一条边。φ 是一个 X 上的时钟约束，它指定何时可以发生状态转换，集合 $\lambda \subseteq X$ 给出在这次转换中被复位的时钟。时间自动机 A 的语义由一个与它相关的转换系统 S_A 来定义。S_A 的一个状态是一个二元组 (s, v)，s 是 A 的一个位置，v 满足 $I(s)$ 的 X 的一个时钟解释。A 的所有状态的集合表示为 Q_A。如果 s 是 A 的一个初始位置，且对于所有的时钟 x，$v(x)=0$，那么状态 (s, v) 是一个初始状态。有两种类型的转换：

（1）状态可以因为时间流逝而改变。对一个状态 (s, v) 和一个实数值时间增量 $\delta > 0$，如果对于所有的 $0 < \delta' \leqslant \delta$，$v + \delta'$ 满足 $I(s)$，那么 $(s, v) \xrightarrow{\delta} (s', v+\delta)$。

（2）状态可以因为一个位置的转换而改变。对于一个状态 (s, v) 和一个转换 $\langle s, a, \varphi, \lambda, s' \rangle$，如果 v 满足 φ，那么 $(s, v) \xrightarrow{a} (s', v[\lambda := 0])$。

1.2　UPPAAL 及其规范[9-10]

1.2.1　UPPAAL 简介

UPPAAL 是由 Aalborg 大学和 Uppsala 大学于 1995 年开发的基于时间自动机的模型验证工具，比较适用于验证实时系统的安全性和可达性。每个进程都被描述为有限控制结构、实数时钟和变量组成的时间自动机，进程之间通过管道共享变量进而进行通信，管道用于保证不同自动机中的两个转换同时执行。UPPAAL 通过快速搜索机制来验证可达性和时钟约束，不但可以有效地发现设计中的错误，还可以清楚地显示导致错误的判定路径。UPPAAL 的用户界面主要包括 3 个部分：编辑器（Editor）、模拟器（Simulator）和验证器（Verifier）。编辑器用于创建和编辑要分析的系统，模拟器用于模拟系统模型执行过程，以此验证前发现一些错误。验证器通过快速搜索机制来搜索系统的状态空间并检查时钟约束和响应限制性质。

1.2.2　需求规范

UPPAAL 通过验证需求规格来达到模型检测的目的。UPPAAL 所使用的时序逻辑公式是时间分支时序逻辑（TCTL）的子集，其查询语言包括状态公式（State Formulae）和路径公式（Path Formulae）。状态公式描述系统的状态，路径公式量化模型的路径或轨迹。

UPPAAL 为验证提供了一种 BNF 语法：$Prop ::= A[]\phi \mid E\langle\rangle\phi \mid E[]\phi \mid A\langle\rangle\phi \mid \phi \to \psi$。其中 ϕ, ψ 为描述带验证系统性质的逻辑表达式。字符 A 和 E 用来量化路径，路径是系统的一个状态迁移序列。A 表示给定的性质对于所有路径均满足，E 表示至少有一条路径满足给定性质。[]和〈〉用来量化路径上的状态，[]表示路径上的所有状态均满足给定的性质，〈〉表示路径上至少有一个状态满足给定性质。如 $A[]\phi$ 则表示对于所有路径，逻辑表达式 ϕ 在任一路径状态序列中的所有状态均满足要求；$E\langle\rangle\phi$ 则表示至少存在一条路径，使得逻辑表达式 ϕ 在该路径的状态序列中的某一个状态满足要求。

2　设备自动巡检控制逻辑的流程

2.1　设备自动巡检的基本流程

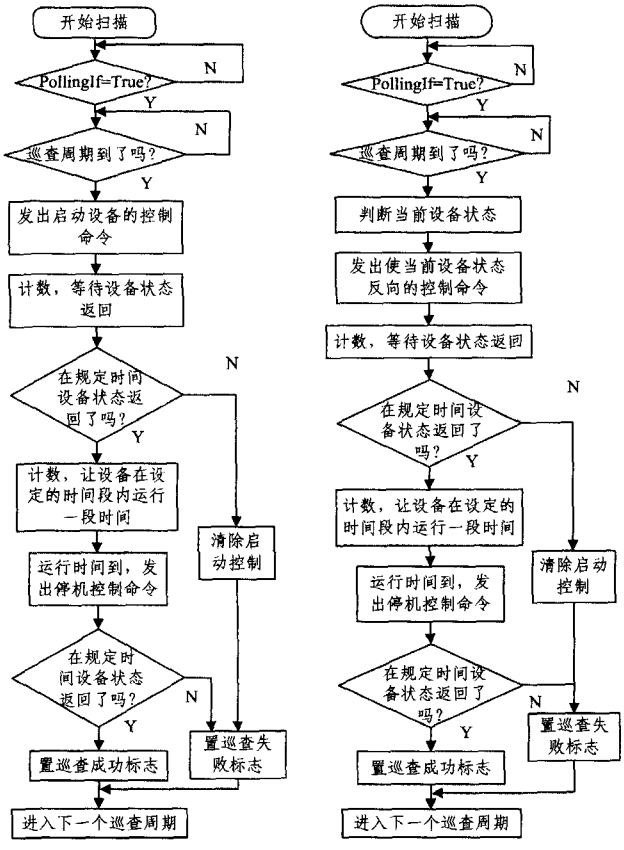
在地下建筑工程中，应用较多的被控设备包括风机、水泵、油泵、风阀、水阀、油阀等，其能否正常运行对地下工程环境的好坏和整体效能的发挥起着至关重要的作用。一般而言，现阶段对机电设备实现控制主要有两种形式：单控制点方式和双控制点方式。对于单控制点设备，只需要一个控制点

就可实现对设备的控制，而设备的运行状态也只需要一个状态反馈点，如风机、水泵等。对于双控制点设备，需要用两个控制点才能实现对设备的控制，而设备状态的反馈也需要两个开关量点，如风阀、水阀等。本文以这两类控制方式为参考，对设备自动巡检控制逻辑进行建模与验证。

具体的设备自动巡检控制逻辑是相当复杂的，为了便于建模与检测，仅抽取出其主要的流程并作适当的假设。设备在巡检时一般处于静止状态，如果设备正在运行，则说明设备正常，无需对其巡检，因此，在对设备的自动巡检开始前，默认设备处于静止状态。对于单控制点设备，直接启动设备，让设备运行一段时间，然后再停止设备，如果在规定的时间内设备状态都能正确反馈到控制器中，则认为巡检成功，设备处于正常状态，否则设备处于故障状态，整个过程结束后系统产生巡检报告。对于双控制点设备的巡检，首先正向启动设备(如电机正转，打开阀门)，让设备运行一段时间，然后反向启动设备(如电机反转，关闭阀门)，再让设备运行一段时间，然后停止设备，在规定的时间运动到位后，则整个巡检过程结束，生成巡检报告。

2.2 设备巡检调度算法控制逻辑流程

设备巡检调度算法主要用于实现对上述两种设备巡检时的调度和管理，确保巡检过程的有序执行。针对自动巡检控制逻辑的行为复杂性特征，设计了单控制点设备巡检调度算法和双控制点设备巡检调度算法。如图 1 所示，图(a)为单控制点设备的巡检控制逻辑流程图，图(b)为双控制点设备的巡检控制逻辑流程图。



(a)单控制点设备 (b)双控制点设备

图 1 设备巡检控制逻辑流程

3 基于层级时间自动机的设备自动巡检控制逻辑建模

3.1 层级时间自动机

由于在设备自动巡检控制逻辑的建模过程中，既要整体考虑某一类设备，又要兼顾某个设备，而且还要体现出时间属性，因此提出一种适用于设备自动巡检控制逻辑建模的层级时间自动机。

定义 4 层级时间自动机 HTA 是一个七元组$\langle A,S,S_0,T,I,E,H\rangle$，其中：$A$ 是动作集合；S 是一个有限状态集合；S_0 是初始状态；T 是时钟变量集合；I 是一个映射，将一个状态映射为一个保卫公式，称为状态不变量；E 是有向边的集合；H 表示层级关系。

3.2 设备自动巡检控制逻辑的形式化模型

模型中各个动作、状态、时钟变量等符号的具体含义如表 1 所列。

表 1 模型中符号的含义

状态	含义
Start	启动自动巡检
CheckList	检查设备配置列表
End	结束自动巡检
GetDevice	获取巡检设备
TestFlag	测试设备巡检标志
StartPoll	巡检开始
Wait	等待巡检结束
Judge	判断错误标志
BeingPoll	正在巡检
EndPoll	巡检结束
HandingException	异常处理
JudgeLevel	判断错误等级
Idle	空闲状态
Testcategory	检测设备类别
OneControlPointPoll	单控制点设备开始巡检
TwoControlPointPoll	双控制点设备开始巡检
Reset	时间清零
Keep	设备保持运行
Check	检测设备
其它	含义
DeviceNum	巡检设备数
PollingFlag	巡检标志
ErrFlag	错误标志
PollingCounter	巡检计时
ExceptionLevel	错误等级
DeviceCategory	设备种类
DeviceState	设备状态
ReturnFlag	返回标志
StartPolling!	发出开始巡检消息
EndPolling?	接收结束巡检消息
PollingFlag==0	判断巡检标志是否为 0
DeviceNum>0	判断巡检设备数是否大于 0
DeviceState=0	令设备状态归零
X<=5	时间小于等于 5 秒
flag:int[0,1]	给标志 flag 赋值，取 0 或 1
ReturnFlag=flag	把标志 flag 的值赋给返回标志
ExceptionLevel++	错误等级加 1
$DI1=1,DI2=0$	正向开启设备
$DI1=0,DI2=1$	反向开启设备

整个系统可表示为 AutomaticPollingLogic≡Level0 ‖ Level1StartPoll ‖ Level2BeingPoll1 ‖ Level2BeingPoll2 ‖ 。

顶层形式化模型为 Level0 $=\langle A_0,S_0,S_{00},T_0,I_0,E_0,H_0\rangle$，其中：

$A_0=\{$StartPolling,Return$\}$；

$S_0=\{$Start, End, CheckList, GetDevice, TestFlag, Start-

Poll, Wait, BeingPoll, EndPoll, Judge1, Judge2, Judge3, HandingException, JudgeLevel}；

S_{00}={Start}；

T_0={PollingCounter}；

I_0 为一个映射，将一个状态映射为一个保卫公式，蕴含于图 2 中；

E_0={⟨Start, DeviceNum=dn, CheckList⟩, ⟨CheckList, DeviceNum==0, End⟩, ⟨End, Start⟩, ⟨CheckList, DeviceNum>0, GetDevice⟩, ⟨GetDevice, PollingFlag=pf, TestFlag⟩, ⟨TestFlag, PollingFlag==0, StartPoll⟩, ⟨StartPoll, StartPolling, PollingCounter=0, Wait⟩, ⟨Wait, Return, Judge1⟩, ⟨Judge1, ErrFlag==1, HandingException⟩, ⟨HandingException, JudgeLevel⟩, ⟨JudgeLevel, ExceptionLevel==0, End⟩, ⟨JudgeLevel, ExceptionLevel==1, CheckList⟩, ⟨JudgeLevel, ExceptionLevel>=2, GetDevice⟩, ⟨TestFlag, PollingFlag==1, BeingPoll⟩, ⟨BeingPoll, Judge2⟩, ⟨Judge2, ErrFlag==1, HandingException⟩, ⟨TestFlag, PollingFlag==2, Endpoll⟩, ⟨EndPoll, ErrFlag=ef, Judge3⟩, ⟨Judge3, ErrFlag==1, HandingException⟩, ⟨Judge1, ErrFlag==0, GetDevice⟩, ⟨Judge2, ErrFlag==0, GetDevice⟩, ⟨Judge3, ErrFlag==0, GetDevice⟩}。

H_0={Level0}。

子一层开始巡检形式化模型为 Level1StartPoll=⟨A_1, S_1, S_{01}, T_1, I_1, E_1, H_1⟩，其中：

A_1={StartPolling, Polling1, Polling2, Endpolling1, Endpolling2, Return}；

S_1={Idle, Testcategory, OneControlPointPoll, TwoControlPointPoll, Wait1, Wait2}；

S_{01}={Idle}；

T_1={PollingCounter}；

I_1 为一个映射，将一个状态映射为一个保卫公式，蕴含于图 3 中；

E_1={⟨Idle, StartPolling, PollingCounter>=60, DeviceCategory=dc, TestCategory⟩, ⟨TestCategory, DeviceCategory==1, OneControlPointPoll⟩, ⟨OneControlPointPoll, Polling, Wait1⟩, ⟨Wait1, EndPolling1, End⟩, ⟨End, Return, Idle⟩, ⟨TestCategory, DeviceCategory==2, TwoControlPointPoll⟩, ⟨TwoControlPointPoll, Polling2, Wait2⟩, ⟨Wait2, EndPolling2, End⟩}。

H_1={LevelStartPoll}。

其余各子层形式化模型依据顶层模型 Level0 及子一层开始巡检形式化模型 Level1StartPoll 很容易得出，因此不再详述。

3.3　顶层巡检逻辑的建模

基于层级网络建模的基本原理，首先建立顶层逻辑模型，如图 2 所示。

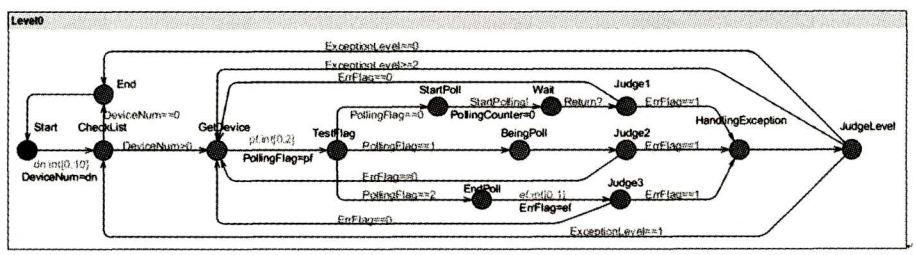


图 2　顶层模型

该模型抽象地表示出了巡检的整个流程，从 Start 开始，随机给定 0～10 个设备，如果 DeviceNum=0，直接结束巡检；如果 DeviceNum>0，则获取任一设备，检测巡检标志，Pollingflag=0 表示设备正准备巡检，Pollingflag=1 表示设备正在巡检，Pollingflag=2 表示设备巡检结束。然后再分别进行错误标志判断，ErrFlag=0 表示没有错误，ErrFlag=1 表示有错误，如果没有错误就返回到 GetDevice 状态，获取下一个设备；如果有错误就进入到 HandingException 状态，进行错误处理。在此时间自动机模型中，默认初始错误等级为零，即 ExceptionLevel=0，每出一次错，错误等级就升一级，然后根据错误的等级返回到不同的状态。为了能比较完整地反映出巡检的过程，以 StartPoll(开始巡检)这一核心状态为例，对其进行子层分解，顶层模型中 StartPolling! 表示在此通道发出“开始巡检”的消息，子层的模型会在相应的通道接受消息，Return? 表示接受子层模型发出的“返回”的消息。

3.4　子层巡检逻辑的建模

为了更详细地描述设备的巡检过程，对 StartPoll 这一状态进行了子层分解，然后又以单控制点和双控制点这两种控制方式为基准，再次进行下一层的分解，建立子层模型。具体情况如图 3－图 5 所示。

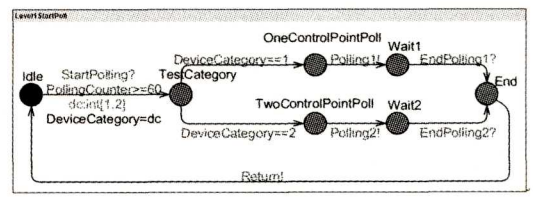


图 3　子一层开始巡检模型

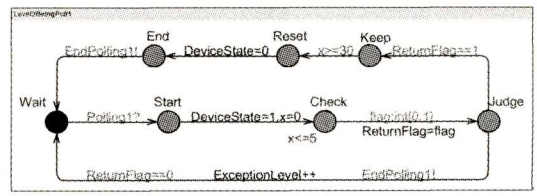


图 4　子二层单控制点巡检模型

其中，Level1StartPoll 为子一层模型，其用于辨别是单控制点设备还是双控制点设备，Level2BeingPoll1 和

Level2BeingPoll2 为子二层模型，其作用是分别对单控制点设备和双控制点设备展开巡检。各个层级之间通过收发消息来进行通信，其具体过程由顶层模型及各位置、消息和通道的含义很容易得知，在此不再赘述。需要指出的是，在单控制点设备中，DeviceState＝0 表示关闭设备，DeviceState＝1 表示开启设备；在双控制点设备中，DI1＝0，DI2＝0 表示关闭设备，DI1＝1，DI2＝0 表示正向开启设备，DI1＝0，DI2＝1 表示反向开启设备。在开始巡检前的等待计时阶段及让设备保持运行等阶段都添加了明确的时间约束，如设备须在 5s 内开启完毕，让其保持运行 25s 以上再关闭，以此来保证设备时间的正确性。

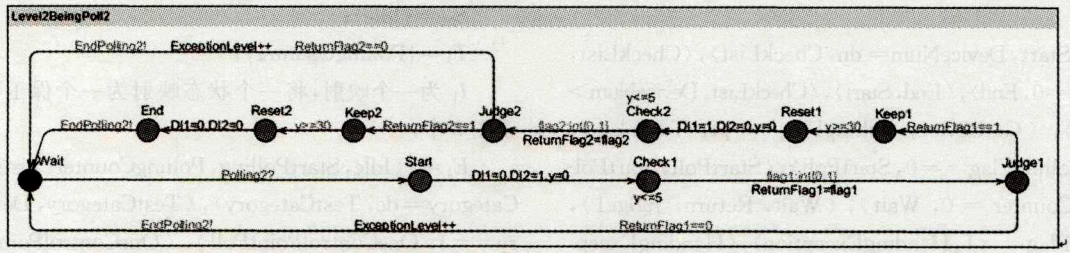


图 5 子二层双控制点巡检模型

4 设备自动巡检控制逻辑的形式化验证

建立好自动巡检控制逻辑的顶层和子层模型后，首先在 UPPAAL 的模拟器中模拟各个层级对象的交互情况，随机得到的巡检时序图如图 6 所示，消息控制序列如图 7 所示。

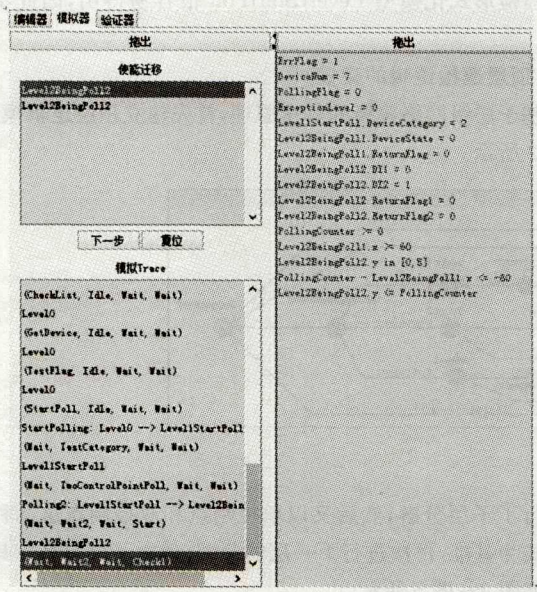


图 6 巡检时序图

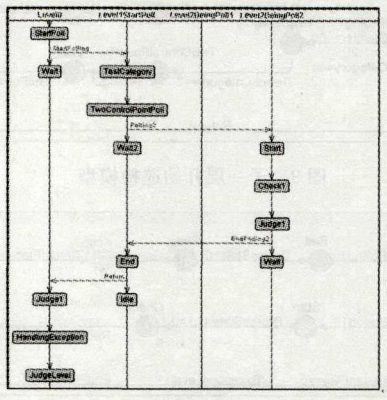


图 7 消息控制序列

在巡检时序图中详细地示出了每一层中各个对象之间的交互情况以及每个时间自动机模型的运行情况，在模拟 Trace 列表中，偶数行表示各对象之间的通信情况，奇数行表示各个时间自动机状态的变化情况。其中括号内第一项记录了 Level0 的状态，第二项记录了 Level1StartPoll 的状态，第三项记录了 Level2BeingPoll1 的状态，第四项记录了 Level2BeingPoll2 的状态。消息控制序列以图文的形式表示出了各对象的运行及交互情况，纵向表示每个时间自动机的运行状况，横向表示相互之间的通信情况以及在同一时间每个时间自动机的状态，这对于分析设备自动巡检逻辑系统起到了极大的辅助作用。

系统的模拟运行在一定程度上是操作人员有目的地选择系统的运行路线，不具有一般性与全面性，不能完全保证系统自动正确地运行，因此还需要用 UPPAAL 提供的验证器对系统的时效正确性进行验证。将设备自动巡检逻辑的时效正确性分解为安全性、可达性、活性和时间约束[9]。安全性表示整个设备自动巡检逻辑不期望的事件永不发生；可达性表示在自动巡检系统中，存在从初始状态到期望到达的某个状态的一条运行轨迹；活性表示系统期望的事件最终能发生；时间约束指的是至少存在一个时间变量用于判断巡检行为是否在规定的时间内完成任务。

(1)安全性验证

A [] not deadlock. 系统无死锁。

(2)可达性验证

E<>Level0. StartPoll. 顶层模型中，开始巡检状态可达，即巡检系统能启动开始巡检功能。

E<>Level0. HandlingException. 顶层模型中，处理错误状态可达，即在巡检过程中会检测到发生故障的设备。

E<>Level1StartPoll. OneControlPointPoll. 子一层开始巡检模型中，单控制点设备巡检状态可达，即设备中存在单控制点设备。

E<>Level1StartPoll. TwoControlPointPoll. 子一层开始巡检模型中，双控制点设备巡检状态可达，即设备中存在双控制点设备。

E<>Level2BeingPoll2. Keep2. 子二层正在巡检模型中，双控制点设备第二次保持运行状态可达，即在检测双控制点方式控制的设备时能让其反向保持运行一段时间。

(3)活性验证

E<>Level0. StartPoll imply Level1StartPoll. TestCategory. 顶层模型中开始巡检启动,则子一层开始巡检模型检测设备种类会启动,即如果巡检系统开始巡检,肯定会判断是单控制点还是双控制点的设备。

E<>Level1StartPoll. TwoControlPointPoll imply Level2BeingPoll2. Start. 子一层开始巡检模型中双控制点设备巡检发生,则子二层正在巡检模型会启动,即如果判断出设备为双控制点设备,那么巡检系统会按照对双控制点设备巡检的规则对其检测。

(4)时间约束

E<>Level2BeingPoll1. Check imply Level2 BeingPoll1. $x<=5$. 在子二层正在巡检模型中,单控制点设备在 5s 内完成检测,即要在 5s 内开启设备。

E<>Level2BeingPoll1. Keep imply Level2 BeingPoll1. $x>=30$. 在子二层正在巡检模型中,单控制点设备保持运行 25s 以上,即在巡检设备时,不但要求设备能启动,还要保证让其运行一段时间。

在 UPPAAL 验证器内逐个对以上的语句进行验证,语句均通过验证,如图 8 所示。通过对设备自动巡检逻辑模型的模拟和形式化验证,证明了系统的正确性与可靠性,该系统满足需求设定。

E<>Level0. StartPoll.
E<>Level0. HandlingException.
E<>Level1StartPoll. OneControlPointPoll.
E<>Level1StartPoll. TwoControlPointPoll.
E<>Level2BeingPoll2. Keep2.
E<>Level0. StartPoll imply Level1StartPoll. TestCategory.
E<>Level1StartPoll. TwoControlPointPoll imply Level2BeingPoll2. Start.
E<>Level2BeingPoll1. Check imply Level2 BeingPoll1. x<=5.
E<>Level2BeingPoll1. Keep imply Level2 BeingPoll1. x>=30.

图 8　模型检测结果

5　相关工作

在与模型检测相关的研究工作中,国内外学者针对自动机、建模方法及软件的形式化验证做了大量的研究工作,不断提出新型的自动机,致力于设计适用于描述实时多任务系统的形式化语言,简化建模过程,对模型进行全面的可靠性与正确性验证。如文献[3-4]讨论了离散语义下的时间自动机和连续语义下的时间自动机的利弊。文献[11-12]提出了中断时间自动机和任务自动机。文献[13]提供了由周期性任务组成的实时系统模型库。文献[14]提出了具体的体系结构建模语言 AADL(the Architecture Analysis and Design Language)。文献[15]利用模型检验获得了系统的可控性和可见性。文献[16-17]提出了将统计模型检验应用于复杂系统的思想,解决了状态爆炸问题。文献[18-21]基于严格的形式化语言对软件自适应性质进行了验证。

同上述的建模与检测方法相比,本文提出的层级时间自动机将层级建模的思想应用于时间自动机,进而用 UPPAAL 进行检测,这更适应于对设备自动巡检控制逻辑的建模与验证,既能方便、简洁地建立模型,体现出模型的时间属性,还可以检测模型的正确性与可靠性。

结束语　为了保持地下建筑工程具有良好的环境,充分发挥防效能,不仅要求设备满足运行结果逻辑上的正确性,还要满足时间上的正确性。因此,对于应用于地下建筑工程中的设备自动巡检系统,其时效正确性显得尤为重要。基于以上观点,本文采用层级时间自动机对自动巡检控制逻辑进行建模,并以 UPPAAL 为工具对其进行形式化分析与验证,从而保证设备自动巡检控制逻辑的正确性和可靠性。

本文虽然对设备自动巡检控制逻辑进行了层级时间自动机的建模与验证,但所验证的系统属性并不全面,仍然存在没考虑到的属性有待进一步研究。

参 考 文 献

[1] YANG Q L, XING J C, WANG P. Implementation and research of device automatic polling system in protective engineering [J]. Protective Engineering, 2008, 30(4): 59-64. (in Chinese)
杨启亮,邢建春,王平. 防护工程设备自动巡检系统研究与实现[J]. 防护工程,2008,30(4):59-64.

[2] YANG Q L, XING J C, WANG P, et al. Hierarchical finite automation modeling of device automatic polling logic and its implementation [J]. Journal of PLA University of Science and Technology, 2009, 6(10): 528-535. (in Chinese)
杨启亮,邢建春,王平,等. 设备自动巡检逻辑的层级有限自动机建模与实现[J]. 解放军理工大学学报,2009,6(10):528-535.

[3] ALUR R, HENZINGER T A. A really temporal logic[J]. Journal of the ACM, 1989, 41(1): 164-169.

[4] ALUR R, DILL D L. A theory of timed automata[J]. Theoretical Computer Science, 1994, 126(94): 183-235.

[5] FAL H. Formal Verification of Timed Systems: A Survey and Perspective[J]. Proceedings of the IEEE, 2004, 92(8): 1281-1282.

[6] BENGTSSON J, WANG Y. Timed automata: Semantics, algorithms and tools[M]// Desel J, ed. Proc. of the Lectures on Concurrency and Petri Nets: Advances in Petri Nets. Berlin: Springer-Verlag, 2004: 87-124.

[7] VARDI M Y, WOLPER P. An automata-theoretic approach to automatic program verification[C]// Proceedings of the First Annual IEEE Symp on Logic in Computer Science Lics, 1986: 322-331.

[8] ALUR R. Timed Automata[M]// Computer Aided Verification, Springer Berlin Heidelberg, 1999: 8-22.

[9] LARSEN K G, PETTERSSON P, YI W. Uppaal in a nutshell [J]. International Journal on Software Tools for Technology Transfer, 1997, 1(1/2): 134-152.

[10] BEHRMANN G, DAVID A, LARSEN K G. A tutorial on UPPAAL[M]// Bernardo M, ed. Proc. of the Formal Methods for the Design of Real-Time Systems. Springer-Verlag, 2004: 200-236.

[11] FERSMAN E, KRCAL P, PETTERSSON P, et al. Task automata: Schedulability, decidability and undecidability[J]. Information & Computation, 2007, 205(8): 1149-1172.

(下转第 78 页)

采用 1 和 20 作为头文件类别差异权重和文件 trace. txt 的特殊关注点权重对 MPEG4 聚类，聚类后期，165 个实体被划分为 5 个簇，经合并及拆分，簇的稳定度由 5 个稳定簇对的 0.865，0.906，0.563，0.563 和 1 提高为 4 个稳定簇对的 0.865，0.906，1 和 0.922 或 3 个簇对的 0.656，1 和 0.875，且得到相应的系统划分。

(4)聚类相似度

图 3(c)为 LIMBO 聚集 j2wap 的结果，C_1''，C_2''和 C_3''与 C_1，C_2 和 C_3 级别相同，合并 C_1''，C_2''和 C_3''中任意两个簇，(C_1''，$C_2''\cup C_3''$)的聚类相似度最大。HCSAR 和 LIMBO 的不同系统划分的聚类相似度区间分别为[0.811，0.9]和[0.744，0.844]。由数据可知，HCSAR 的聚类相似度优于 LIMBO 的，其相应的系统划分更准确。

HCSAR 和 LIMBO 聚集 MPEG4 的结果与 j2wap 相似，聚类相似度区间为[0.83，0.907]，LIMBO 仅为 0.663。由此可知，相对稳定的簇对构造的系统划分的聚类相似度更高，HCSAR 能辅助构造准确度更高的系统划分。

结束语 本文针对软件聚类欠缺考虑实体和特征的特性的问题，提出了一种基于层次聚类的软件架构恢复方法(HCSAR)。该方法针对软件系统类型选取实体和特征，首次提出特征的多重加权策略，采用信息丢失度来衡量实体相似度，并选取和设计软件聚类的客观和主观评估准则。与目前效果较好的方法对比，HCSAR 能增加聚类中期簇的数目，提高结果的内聚度；能降低主观判定数，提高效率和准确度；能灵活地调整聚类关注点，获得不同的聚类结果；使用设计的评估准则分析关注点不同的聚类结果，能辅助构造准确度更高的系统划分。

参 考 文 献

[1] BIBI M, MAQBOOL O. Version information support for software architecture recovery[C]// Proceedings of the 7th International Conference on Emerging Technologies, 2011. Islamabad: IEEE, 2011: 1-6.

[2] LI Q S, CHEN P. Implementing architecture recovery by using improved genetic Algorithm[J]. Journal of Software, 2003, 14(7): 1221-1228. (in Chinese)
李青山，陈平. 用改进的遗传算法实现架构恢复[J]. 软件学报，2003，14(7)：1221-1228.

[3] YANG Q, ZHANG L P. The research on software architecture recovery based on pattern matching[J]. Computer Application and Software, 2006, 23(4): 27-29. (in Chinese)
杨清，张礼平. 基于模式匹配的软件架构恢复的研究[J]. 计算机应用与软件，2006，23(4)：27-29.

[4] SIDDIQUE F, MAQBOOL O. Enhancing comprehensibility of software clustering results[J]. Software, IET, 2012, 6(4): 283-295.

[5] MAQBOOL O, BABRI H. Hierarchical clustering for software architecture recovery[J]. IEEE Transactions on Software Engineering, 2007, 33(11): 759-780.

[6] CHOT S, CHA S, TAPPERT C. A survey of binary similarity distance measures[J]. Journal of Systemics, Cybernetics and Informatics, 2010, 8(1): 43-48.

[7] SAEED M, MAQBOOL O, BABRI H, et al. Software clustering techniques and the use of the combined algorithm[J]. Journal of Software Maintenance and Evolution: Research and Practice, 2003, 16(4-5): 301-306.

[8] MAQBOOL O, BABRI H. The weighted combined algorithm: a linkage algorithm for software clustering[C]// Proceedings of the 8th European Conference on Software Maintenance and Reengineering, 2004. Los Alamitos, Calif.: IEEE Computer Society, 2004: 15-24.

[9] ANDRITSOS P, TZPERPOS V. Information theoretic software clustering[J]. IEEE Transactions on Software Engineering, 2005, 31(2): 150-165.

[10] NASEEM R, BIN M, MAQBOOL O. Software modulariza-tion using combination of multiple clustering[C]// Proceedings of IEEE 17th International Conference on Multi-Topic Conference, 2004. Karachi: IEEE, 2014: 277-281.

(上接第 71 页)

[12] BÉRARD B, HADDAD S, SASSOLAS M. Interrupt Timed Automata: verification and expressiveness[J]. Formal Methods in System Design, 2012, 40(1): 41-87.

[13] FLORIAN M, GAMBLE E, HOLZMANN G. Logic Model Checking of Time-Periodic Real-Time Systems[C]// Infotech@aerospace. 2013: 1-43.

[14] SENN E, LAURENT J, JUIN E, et al. Refining Power Consumption Estimations in the Component-based AADL Design Flow[C]// Forum on Specification, Verification and Design Languages, 2008(FDL 2008). IEEE, 2008: 173-178.

[15] GLUCK P R, HOLZMANN G J. Using SPIN model checking for flight software verification[C]// Aerospace Conference Proceedings, 2002. IEEE, 2002: 105-113.

[16] CLARKE E M, ZULIANI P. Statistical Model Checking for Cyber-Physical Systems[C]// International Conference on Automated Technology for Verification and Analysis. Springer-verlag, 2011: 1-12.

[17] HÅKAN L, YOUNES S. Ymer: A Statistical Model Checker [M]// Computer Aided Verification. Springer Berlin Heidelberg, 2005: 429-433.

[18] DE I I D G, WEYNS D. Guaranteeing robustness in a mobile learning Application using formally verified MAPE loops[C]// Proceedings of the 2013 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). IEEE Computer Society, 2013: 83-92.

[19] BARTELS B, KLEINE M. A CSP-based framework for the specification, verification, and implementation of adaptive systems[C]// Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. ACM, 2011: 158-167.

[20] RAMIREZ A J, CHENG B H C. Verifying and Analyzing Adaptive Logic through UML State Models[C]// International Conference on Software Testing, Verification, and Validation. IEEE, 2008: 529-532.

[21] LUCKEY M, THANOS C, GERTH C, et al. Multi-Staged Quality Assurance for Self-Adaptive Systems[C]// 2013 IEEE 7th International Conference on Self-Adaptation and Self-Organizing Systems Workshops. IEEE, 2012: 111-118.