广西科学院学报
Journal of Guangxi Academy of Sciences
2006,22(4):382～384
Vol. 22,No. 4 November 2006

简述 Statecharts 的 CASE 工具 SuperState*
CASE Tool SuperState for Verifying Statecharts

钱俊彦
QIAN Jun-yan

(桂林电子科技大学计算机系,广西桂林 541004)
(Department of Computer,Guilin University of Electronic Technology,Guilin,Guangxi,541004,China)

摘要:在介绍基于 Statecharts 语言的验证工具 SuperState 的编辑环境的基础上,简述 SuperState 工具中主要部分的实现思想,即采用静态分析和动态仿真保证设计模型的正确性。

关键词:Statecharts 形式化技术 CASE 工具
中图法分类号:TP301.1 **文献标识码**:A **文章编号**:1002-7378(2006)04-0382-03

Abstract:A modeling and verifying tool SuperState in which Statecharts is integrated into CASE tool is introduced. The idea of keeping a design model to be correct using static and dynamic analysis in the tool is discussed.

Key words:Statecharts,Formal techniques,CASE tool

随着技术的快速发展,需要更快、更可靠的方法开发软件系统去满足用户的需要。在以前的软件开发过程中,改正出现在分析和设计阶段的错误需要花费更多的人力和物力[1]。在系统开发过程中,把形式化技术集成到 CASE 工具,能帮助消除软件早期开发阶段中的设计错误,确信开发人员的意图;也使不熟悉形式化方法的软件工程师容易使用,开发出用户满意的软件。

当前高可靠的反应式系统建模倾向于使用形式化、可视化语言。Statecharts 是一种用以规约复杂反应式系统行为的可视化语言,能被 CASE 工具支持,且有比较完善的语义[2,3]。我们开发了基于 Statecharts 的建模工具 SuperState。本文在介绍基于 Statecharts 语言的验证工具 SuperState 的编辑环境的基础上,简述 SuperState 工具中主要部分的实现思想,即采用静态分析和动态仿真保证模型的正确性。

1 SuperState 的编辑环境

图 1 是使用 SuperState 环境的图形编辑器为十字交通灯建立的 Statecharts 模型。在这个模型中,用户可以根据菜单或工具栏的提示进行图元选择,然后通过键盘或者鼠标器在屏幕上直接定位作图。图 1 中用圆矩形框表示状态来描述系统的操作,NORMAL 状态来描述两个方向交通灯的正常操作。中间用虚线分成两个并发的子状态:南－北(N_S)和东－西(E_W),用来表示交通灯的两个方向。类似于 NORMAL 状态的正交组件称为与状态。状态 N_S 和 E_W 又可以分解为 red,yellow 和 green 3 个状态,表示模型处在这些状态时所显示的颜色。由于模型仅仅能处在 red,yellow 和 green 3 个状态中的一个,所以 N_S 和 E_W 被称为或状态。

大圆矩形框里所有子状态的表示形式形成了状态的层次(深度),状态 NORMAL 是状态 E_W 和 N_S 的祖先状态,反之,状态 E_W 和 N_S 是状态 NORMAL 的后代。如果状态没分解成与状态或者是或状态,则称作基状态。图 1 中有 7 个基状态。

收稿日期:2006-07-07
作者简介:钱俊彦 (1973-),男,浙江嵊县人,副教授,主要从事软件工程、模型检验、嵌入式实时系统方面的研究工作。
* 广西自然科学基金(0542036)资助。

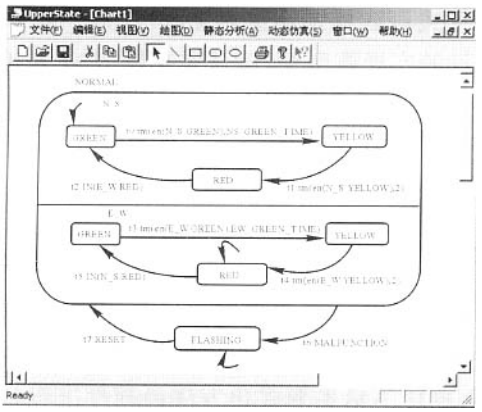

图 1　交通灯 Statecharts 模型

通过如下标签形式的迁移把状态连接起来:

事件[条件]/动作

如果当前状态是迁移标签为 e[c]/a 的源状态,当条件 c 满足且事件 e 发生,则迁移允许。事件和条件组成在一起作为迁移的触发器。条件是一个布尔类型的表达式,包括如 IN(x)的语句,IN(x)的意思是判断当前状态是否是状态 x。如迁移 t5 的标签 IN(N _ S. RED)。

迁移的允许将使系统在状态间传递,执行迁移的意思就是从源状态离开,产生迁移标签上的动作,然后进入目的状态。没有源状态的箭头,我们称为缺省箭头,它使系统进入一系列的基状态。例如迁移 t7 执行,它的目的状态是由两个正交组件组成的与状态 NORMAL,通过缺省箭头,将使系统进入它们的子状态 E _ W. RED 和 N _ S. GREEN。如果迁移执行,标签部分的动作将产生,动作包括产生事件和修改变量。

2　SuperState 的静态分析[4,5]

为了检查用户所建立的模型是否符合 Statecharts 的语法和语义规范,工具 SuperState 提供了静态分析,包括语法分析、一致性分析、完全性分析和确定性分析。

2.1　语法分析

由于计算机难以对图形符号所建立的模型直接进行语法分析,所以为了便于模型的语法分析,必须把图形化的 Stateharts 语言转化为文本的形式化规格说明语言。在 UpperState 环境中,为了验证图形编辑器所建立模型的语法正确性,要求系统能根据图形自动生成其相应 Statecharts 数学表达式;同时,用户可以对已有的 Statecharts 图进行修改,而修改经过认可,系统则必须自动地对其数学表达式作相应的修改。图形的表达式是以 Statecharts 形式化文法中规定的句型为样本,按文法要求的顺序,从结构数组中取出所需的信息,依次生成数学表达式中的各条句子,然后顺序地写入该 Statecharts 数学表达式的文件即可,详见图 2 所示。

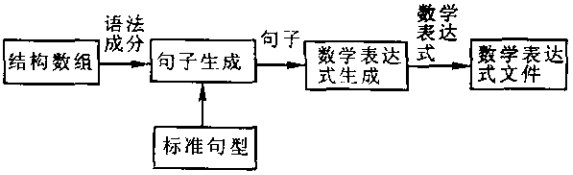

图 2　表达式文件生成

根据文法,用一个编译程序来完成对 Statecharts 的数学表示的语法分析和语义分析。此编译程序采用一遍扫描的形式,从数学表达式文件中依次读出各句子,然后根据上述的文法,分析其语法的正确性;并根据 Statecharts 的语义规则,分析它的合理性,同时做相应的出错处理和识别处理,一次扫描完成后,即形成了相应的结构数组和出错信息。在出错处理时,应尽可能多发现错误和保证错误不扩散的原则,采用局部化处理方法,即出错后跳到下一句,出错信息指错误处相应的文字信息,最后统计错误总数。

2.2　一致性分析

一致性即是迁移的不确定性,指的是在同一状态,对触发事件不会有多个不同的迁移同时得到满足。例如图 3 在,在状态 A 迁移 t1,t2,t3 是否会有两个或多个迁移同时发生,这就是一个一致性问题。如果有两个或多个迁移发生,则会产生不一致性。

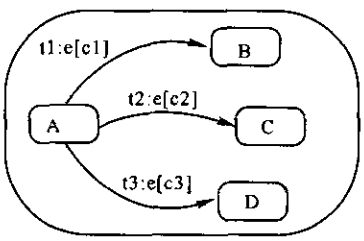

图 3　一致性分析

2.3　完全性分析

完全性是指对系统的每一个输入事件,系统状态都必须有一个相应的响应。当系统有一个外部输入,如果系统没有响应,那么系统将产生不完全性。实际上完全性的检查将与死锁相联系起来,死锁是完全性分析的一种特殊情况。如图 3 所示,当系统处于状态 A 是,事件发生,判断 c1 ∧ c2 ∧ c3 是否是重言式。

2.4　确定性分析

确定性是指对于由同一事件触发的并行迁移的

动作，不会互相冲突，即不会对同一些变量进行跟操作次序有关的操作。否则会引起冲突。如图4中，由于t1和t2可以并行执行，当t1和t2并发，可能会产生不确定的结果。

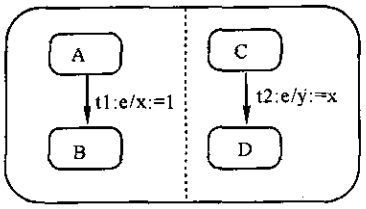

图4 确定性分析

3 SuperState的动态分析

当模型符合Statecharts的规范时，用户更关心所建的模型是否满足所期望的性质，包括安全性等。工具SuperState提供了动态仿真和模型检验。

3.1 动态仿真

工具SuperState对模型的静态分析完成后，在保证模型在语法上正确的基础上，还继续做对模型进行动态仿真。在动态仿真执行中，也将对模型进行自动的一致性、完全性、确定性分析和可达性分析，一经发现错误，立即提示给开发人员，直到错误修改完成后，才能继续向下执行，从而保证设计模型的正确性。

工具SuperState执行动态仿真，是先对环境进行初始化，给出测试用例，根据我们的选择可以自动检验、单步执行以及设置断点，在执行的过程中，根据执行情况对相应的状态进行着色，从而使用户对执行有一个明确的了解，检验出设计模型是否正确，是否符合开发人员的意图。

3.2 模型检验

由于模型仿真不可能穷尽所有可能的输入，故难以发现隐藏在设计深处的错误。为了消除模拟和仿真的缺陷，可以采用模型检验的方法来验证模型的正确性。

模型检验[6]是一种重要的形式化自动验证技术，通过状态空间搜索来保证设计的正确性。它采用一种形式语言描述系统性质φ，构造一种算法来遍历设计的实现模型M，确认实现模型M是否满足系统性质φ，简单的表示为M|φ。模型检验有如下益处：完全自动；当设计不满足系统所期望的规格时，将生成反例。

4 结束语

把形式化技术集成到CASE工具，能帮助消除软件早期开发阶段中的设计错误，确信开发人员的意图；也使不熟悉形式化方法的软件工程师容易使用，并开发出用户所满意的软件，所以CASE工具和形式化方法联系起来将是非常有益的。

把Statecharts规格语言集成到CASE工具SuperState中，为反应式系统的设计建模提供了一种形式化的工具，并保证所建模型的正确性。下一步工作将扩展到带连续时间的Statecharts.

参考文献：

[1] NACY G Leveson. Requirements specification for process-control systems [J]. IEEE Transactions on Software Engineering, 1994, 20(9): 684-707.

[2] HAREL D, PNUELI A, SCHMIDT J P. On the formal semantics of statecharts: in proceedings of the 2nd IEEE symposium on Logic in Computer Science, Ithaca [C]. New York: [s. n.], 1987: 54-64.

[3] HAREL D. Statecharts: A Visual Formalism for Complex Systems [C]. Sciene of Computing, 1987: 231-274.

[4] 钱俊彦. Statecharts的抽象语法分析研究[J]. 计算机工程, 2004, 30(16): 167-168.

[5] 钱俊彦，古天龙，赵岭忠. Statecharts的形式化验证研究[J]. 计算机工程, 2005, 31(18): 19-21.

[6] Miiuller-Olm M, SCHMIDT D, STEFFEN B. Model-Checking A Tutorial Introduction [M]. LNCS 10359. Berlin: Springer, 1998.

（责任编辑：邓大玉）