第35卷 第23期 计 算 机 工 程 2009年12月
Vol.35 No.23 Computer Engineering December 2009
·软件技术与数据库· 文章编号：1000—3428(2009)23—0068—03 文献标识码：A 中图分类号：TP311

计算树逻辑特性模式研究

周 慧
(九江学院商学院，九江 332005)

摘 要：模型检查是系统验证的有效方法，在验证过程中需要对系统待检验特性用时态逻辑公式进行刻画，然后在模型检查工具中进行检验。介绍计算树逻辑的语法及语义，根据计算树逻辑中特性模式的划分及作用范围给出计算树逻辑常见的特性模式，包括缺失性模式、存在性模式、普遍性模式、优先性模式和跟随性模式等。

关键词：计算树逻辑；特性模式；模型检查

Research on Property Patterns of Computation Tree Logic

ZHOU Hui
(Commercial College, Jiujiang University, Jiujiang 332005)

【Abstract】 Model checking is an effective method for system verification and validation. Temporal logic formulas are used to specify system properties to be verified and verified in a model checking tool. This paper introduces the syntax and semantics of Computation Tree Logic(CTL), presents property patterns of CTL based on the partition and scope of CTL, which includes absence model, absence model, university model, precedence model, response model, and so on.

【Key words】 Computation Tree Logic(CTL); property patterns; model checking

模型检查[1]是一种重要的形式化验证方法，在计算机硬件、通信协议[2]、实时系统[3]等方面的分析和验证中取得了极大的成功。系统特性的时态逻辑公式描述是模型检查中的重要一步，本文对计算树逻辑(Computation Tree Logic, CTL)常用的特性[4]进行研究。

1 计算树逻辑

CTL是一种分支时态逻辑。CTL公式的时态操作符由路径算子和时态算子组成。路径算子有2种：A表示“适用于所有的路径”，E表示“存在至少1条以上的路径”。时态算子由X(neXt time)算子、F(Future)算子、G(Global)算子、U(Until)算子组成。

定义1 CTL语法

$\phi ::= p \mid \neg p \mid p_1 \vee p_2 \mid EXp \mid E[p_1 U p_2] \mid A[p_1 U p_2]$

CTL语法定义了2种逻辑操作符($\neg, \vee$)和4种时态操作符EX,E,A,U。逻辑操作符true,false,$\wedge, \rightarrow$等可以由$\neg, \vee$很容易得到，时态操作符EFp,AFp,EGp,AGp,AXp可以用上述操作符表示如下：

$EFp \equiv E[trueUp]$ $AFp \equiv A[trueUp]$ $EGp \equiv \neg AF\neg p$
$AGp \equiv \neg EF\neg p$ $AXp \equiv \neg EX\neg p$

定义2 CTL模型

CTL模型是一个三元组M = (S,R, Label)，其中，S是非空状态集合；$R \subseteq S \times S$是状态转移关系；Label : S $\rightarrow 2^{AP}$是状态标记函数。

模型M也被称为Kripke结构。例如，令AP = { x = 0, x = 1, x ≠0 }是一个原子命题集合，S ={s0, s1, …, s3}是有限状态集合，状态标记函数label有：Label(s0) ={x ≠ 0}, Label(s1) = Label(s2)={x =0}, Label(s3)={x = 1, x ≠ 0}，且状态转移关系R={(s0, s1), (s1, s2), (s1, s3), (s3, s3), (s2, s3), (s3, s2)}。模型M=(S,R, Label)如图1所示。

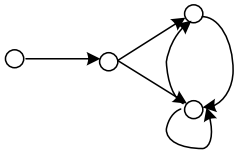

图1 CTL模型示例

定义3 路径

路径path是一个无限状态序列，如果对于任意的$i \in \{0,1,\cdots\}$，$(s_i \rightarrow s_{i+1}) \subseteq R$。

令$\sigma \in S^w$表示一条路径，对于$i \geqslant 0$，用$\sigma|i|$表示路径σ中第i+1个元素，例如$\sigma = t_0 t_1 \cdots$，则有$\sigma|i| = t_i$。

定义4 路径集合

在模型M中，以状态s为初始点的路径集合可以定义为

$P_M(s) = \{\sigma \in S^w \mid \sigma[0] = s\}$

对于模型M=(S, R, Label)和状态$s \in S$，有1棵以s为根的计算树，(s′, s″)是计算树中的一条边当且仅当$(s', s'') \in R$。

定义5 CTL语义

令$p \in AP$是一个原子命题，M=(S,R, Label)是一个模型，$s \in S$，p，p_1，p_2是CTL公式。满足关系‘ 定义如下：

s ‘ p iff $p \in Label(s)$
s 慑¬p iff ¬(s p)
s 慑$p_1 \vee p_2$ iff (s p_1)∨(s □p_2)
s 慑EXp iff $\exists \sigma \in p_M(s).\sigma[1]$ **p**
s 慑E [$p_1 U p_2$] iff$\exists \sigma \in p_M(s).(\exists j \geqslant 0.\sigma[j]$ $p_2 \wedge (\forall 0 \leqslant k < j.\sigma[k]$ 蹇p_1))
s ‘ A [$p_1 U p_2$] iff$\forall \sigma \in P_M(s).(\exists j \geqslant 0.\sigma[j]$ 慑$p_2 \wedge (\forall 0 \leqslant k < j.\sigma[k]$ pp$_1$))

作者简介：周 慧(1981－)，女，讲师、硕士研究生，主研方向：电子商务，物流管理

收稿日期：2009-05-06 **E-mail：**sweathui@126.com

2 模型检查中的特性描述

文献[5]对模态逻辑的特性模式进行了详细的划分，见图 2。为简化起见，使用术语“一个给定的状态(或事件)发生”表示如下状态：使给定的状态公式为真；或在给定的析取事件集的一个事件发生。

(1)缺失性(absence)。在指定的范围内，给定的状态(或事件)不会发生，也称为 never。在模型检查中，如果指定的范围是全部范围，称为安全性。

(2)存在性(existence)。在指定范围内，给定的状态(或事件)必然发生。存在性也被称为必然性 eventually。

(3)受限存在性(bounded existence)。在指定的范围内，给定的状态(或事件)必然发生 k 次。这种模式的变体是给定的状态(或事件)在指定的范围内必然最少(最多)发生 k 次。

(4)普遍性(university)。在指定的范围内，给定的状态(或事件)必然处处发生，即其只包含满足特性的状态(或事件)。也称为 henceforth and always。在模型检查中，如果指定的范围是全部范围，称为活性。

(5)状态(或事件)的优先性(precedence)。在指定的范围内，状态(或事件)P 总是优先于状态(或事件)Q。优先性被用于描述状态(或事件)间的关系，状态(或事件)P 是状态(或事件)Q 发生的前置条件。也可以说，状态(或事件)Q 由状态(或事件)P 使能(enable)。

(6)状态(或事件)的跟随性(response)。在指定的范围内，状态(或事件)P 总是尾随于状态(或事件)Q。跟随性被用于描述状态(或事件)间的因果关系，状态(或事件)Q 称为因，状态(或事件)P 称为果。

(7)状态序列(或事件序列)的优先性(precedence chain)。在指定的范围内，状态序列(或事件序列) $P_1\ P_2 \cdots P_n$ 总是先于状态序列(或事件序列) $Q_1\ Q_2 \cdots Q_n$ 。

(8)状态序列(或事件序列)的跟随性(response chain)。在指定的范围内，状态序列(或事件序列) $P_1\ P_2 \cdots P_n$ 总是先于状态序列(或事件序列) $Q_1\ Q_2 \cdots Q_n$ 。

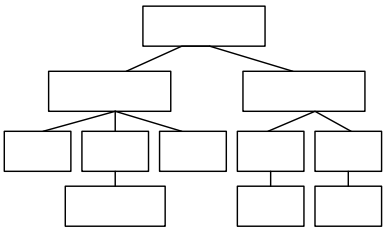

图 2　特性模式的分类

3 特性模式作用范围

每种特性模式都有一定的作用范围，即在该范围内特性为真。文献[5]将特性模式作用范围划分为 5 种：Global(全局)，Before(在…之前)，After(在…之后)，Between(在…和…之间)，after-until(在…之后直到)，见图 3。

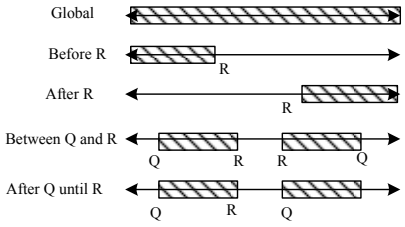

图 3　特性模式作用范围

Global 表示整个程序执行范围内模式为真，Before 表示在给定的事件/状态发生之前特性为真，After 表示在给定的事件/状态发生之后特性为真，Between 表示在给定的 2 个事件/状态之间特性为真，after-until 表示在一个事件/状态发生之后一直到另一特定事件/状态发生特性为真。

4 特性模式的CTL描述

在 CTL 特性模式表述中，为了使特性模式易于理解，使用了 W(weak until)算子，它与 U 算子有如下关系：

A[xWy]=!E[!yU(!x&!y)], E[xUy]=!A[!yW(!x&!y)]

由特性模式的划分方法和作用范围可以得到常见的 CTL 特性模式，见图 4~图 8(因为篇幅缘故，所以本文只列出了 5 种特性模式)。

Global	AG(!P)
(*)Before R	A[(!P|AG(!R))WR]
After Q	AG(Q → AG(!P))
(*)Between Q and R	AG(Q &!R → A[(!P | AG(!R))WR])
(*)After Q until R	AG(Q &!R → A[!P W R])

图 4　缺失性模式(设 P 为 false)

Global	AF(P)
(*)Before R	A[!RW(P &!R)]
(*)After Q	A[!QW(Q&AF(P))]
(*)Between Q and R	AG(Q&!R → A[!RW(P&!R)])
(*)After Q until R	AG(Q &!R → A[!RU(P &!R)])
(*)After Q until R	AG(Q &!R → A[!P W R])

图 5　存在性模式(设 P 为 true)

Global	AG(P)
(*)Before R	A[(P|AG(!R))WR]
After Q	AG(*Q* → AG(*P*))
(*)Between Q and R	AG(*Q*&!*R* → A[(P|AG(!R))WR])
(*)After Q until R	AG(Q &!R → A[PWR])

图 6　普遍性模式(设 P 为 true)

Global	A[!PWS]
(*)Before R	A[(!P|AG(!R))W(S|R)]
(*)After Q	A[!QW(Q&A[!PWS])]
(*)Between Q and R	AG(Q&!R → A[(!P|AG(!R))W(S|R)]
(*)After Q until R	AG(Q&!R → A[!PW(S|R)])

图 7　优先性模式(设 S 优先于 P)

Global	AG(P → AF(S))
(*)Before R	A[((P → A[!RU(S&!R)])|AG(!R))WR]
(*)After Q	A[!QW(Q&AG(P → AF(S)))]
(*)Between Q and R	AG(Q&!R → A[((P → A[!RU(S&!R)]]
(*)After Q until R	AG(Q & !R → A[(P → A[!R U (S & !R)]) W R])
(*)After Q until R	AG(Q&!R → A[!PW(S|R)])

图 8　跟随性模式(设 S 跟随于 P)

5 实例

下面的简单实例描述了 2 个异步进程使用 1 个信号量实现互斥。每个进程有 4 个状态：idle, entering, critical, exiting。状态 entering 表示进程想进入临界区，如果信号量 semaphore 为 0，进程迁移到状态 critical 并且 semaphore 置 1。当进程离开临界区时，semaphore 置 1。

```
MODULE main
VAR
semaphore: boolean;
proc1: process user (semaphore);
proc2: process user (semaphore);
ASSIGN
Init (semaphore):=0;
SPEC
  AG (proc1.state = entering -> AF proc1.state = critical)
SPEC
  AG (!((proc1.state=critical)&(proc2.state=critical )))
```

```
MODULE user (semaphore)
VAR
state:{idle,entering,critical,exiting};
ASSIGN
init(state):=idle;
next(state):=
case
state=idle:{idle,entering};
state=entering&!semaphore:critical;
state=critical:{critical,exiting};
state=exitingg:idle;
1:state;
esac;
next(semaphore):=
case
state=entering:1;
state=exiting:0;
1:semaphore;
esac;
FAIRNESS
Running
```

使用 SMV 为系统建立模型后，使用 CLT 描述该系统应具备的特性，在主模块中验证如下特性：

跟随性：AG (proc1.state = entering -> AF proc1.state = critical)，表示进程不会无休止地等待进入临界区。

缺失性：AG(! ((proc1.state=critical) & (Proc2.state= critical))),表示不能同时有 2 个进程位于临界区内。

最后使用 NuSMV-2.1 运行 SMV 程序，验证结果如图 9 所示，检测结果表明：系统不满足跟随性，满足缺失性。

Context	Index	Select	Value	Trace	Type	Property
Main						
	0	☑	False	1	CTL	AG (proc1.state = entering -> AF proc1.state = critical)
	1	☑	True		CTL	AG (!(proc1.state = critical & proc2.state = critical))

图 9　验证结果

6　结束语

本文在文献[5]的研究基础上，根据时态逻辑的划分方法和作用范围对计算树逻辑的特性模式进行了研究，给出了计算树逻辑常见的特性模式。这些特性模式的提出为系统特性的 CTL 刻画提供了较大的帮助。

参考文献

[1] Clarke E M, Grumberg O, Peled D A. Model Checking[M]. London, UK: MIT Press, 1999.

[2] Perrig A, Szewczyk R, Tygar J D, et al. SPINS: Security Protocols for Sensor Networks[J]. Wireless Networks, 2002, 8(5): 521-534.

[3] Havelund K, Mike L J. Formal Analysis of a Space-craft Controller Using SPIN[J]. IEEE Transactions on Software Engineering, 2001, 27(8): 749-765.

[4] Huth M, Ryan M. Logic in Computer Science: Modeling and Reasoning about Systems[M]. New York, USA: Cambridge University Press, 2004.

[5] 黎升洪, 缪淮扣, 张新林. 线性时态逻辑中的特性模式[J]. 计算机应用, 2006, 26(8): 1912-1915.

编辑　张正兴

（上接第 67 页）

口业务流程，多式联运部使用了 9 个绩效指标从 5 个维度评价其日常运营绩效，如图 4 所示。这一流程的行业最佳实践是与海关建立电子数据交换(Electronic Data Interchange, EDI)，从而加快货物通关速度。

图 3　集装箱航运供应链流程模型

港口物流：进口	
轮班抵港、货物卸船至产品内陆运输安排完毕。	
绩效维度	绩效指标
时间	海关干涉延迟，海关清关周期，承运商报价响应时间
质量	运单履约率，完美运单履行
协同性	运输柔性
成本	总配送成本，运输管理成本
资金利用	流动资金

图 4　“港口物流：进口”流程的绩效指标和最佳实践

软件工具可以为此流程指定绩效指标，并为其指定最佳实践参考，如图 5 所示。

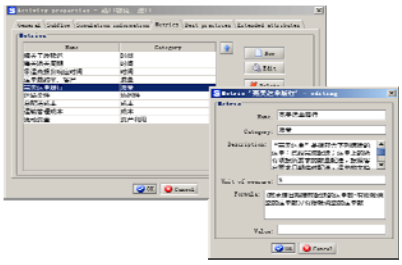

图 5　流程绩效指标的指定

4　结束语

本文从流程分解的视角建立了一个基于流程分解的至下而上的供应链绩效评估模型，并在建模工具的基础上系统实现了对供应链的绩效评估。进一步的研究主要是多收集行业供应链的流程案例和最佳实践数据，从而形成供应链绩效案例库，为企业构建自身的供应链绩效评估模型提供决策支持。

参考文献

[1] Malone T W, Crowston K, Lee J, et al. Tools for Inventing Organizations: Toward a Handbook of Organizational Processes[J]. Management Science, 1999, 45(3): 425-443.

[2] Kobayashi T, Tamaki M, Komoda N. Business Process Integration as a Solution to the Implementation of Supply Chain Management Systems[J]. Information & Management, 2003, 40(8): 769-780.

[3] Chomsky N. Three Models for the Description of Language[J]. IRE Transactions on Information Theory, 1956, 2(3): 113-124.

[4] 吴　琼, 陈云波, 曾广平. 业务流程管理中的大规模整数规划问题求解[J]. 计算机工程, 2008, 34(15): 89-91.

编辑　张　帆