广西科学院学报
Journal of Guangxi Academy of Sciences

2008,24(4):320~324,327
Vol. 24, No. 4 November 2008

模型检测研究进展
Research Progress on Mode Checking

王飞明,胡元闯,董荣胜
WANG Fei-ming, HU Yuan-chuang, DONG Rong-sheng

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

摘要:阐述模型检测的基本思想和工作方式,介绍二叉决策图、符号模型检测、偏序规约等几种在模型检测中抑制状态爆炸的优化技术,并分析模型检测在应用上的优势,最后展望模型检测今后的研究热点。

关键词:模型检测 状态爆炸 形式验证 时态逻辑
中图法分类号:TP301.5 **文献标识码**:A **文章编号**:1002-7378(2008)04-0320-05

Abstract: This paper presented the basic ideas and operate mode about mode checking and gave some optimization technology to deal with the state explosion, such as binary decision diagram, symbolic model checking, partial order reduction, abstract technology and combinational logic. It also analyzed the advantage of mode checking in application, and forecasted the hot research about mode checking.

Key words: mode checking, sate explosion, formal verification, temporal logic

随着计算机软硬件系统功能日渐强大,系统也越来越趋复杂,同时也越来越脆弱,一个微不足道的错误,都有可能引发灾难性的后果[1]。具有交互、实时、并发、分布等特征的系统,其行为有一定的不确定性,这也就造成传统的测试手段如跟踪调试、用例覆盖等技术难以达到理想的测试效果。为了从根本上保证软件系统的可靠安全,包括图灵奖得主 A. Pnueli 在内的许多计算机科学家都认为,采用形式化验证方法验证和分析系统,是构建可靠安全软件的一个重要途径[2,3]。

模型检测由 Clarke 等[4]最早提出,它是一种被广泛应用的验证有限状态系统满足规范的形式化方法,主要针对具有逻辑性质的有限状态系统,其思想是先建立待测系统的有限状态模型,然后用算法穷尽检测模型中的状态,判断其是否满足待测属性。由于模型检测器在算法支持下可以自动执行,并能在系统不满足性质时提供反例路径,因此备受工业界关注。Clarke、Emerson 和 Sifakis 3 位科学家因开发模型检测技术,并在硬件和软件工业中使之成为一个广泛应用且非常有效的算法验证技术所做的奠基性贡献获得了 2007 年度的图灵奖。

本文阐述模型检测的基本思想和工作方式,介绍常见的处理状态爆炸问题的优化技术,并将模型检测与其它系统验证方法进行简单的比较,最后对模型检测的研究热点和现状进行展望。

1 模型检测的思想与工作方式

1.1 基本思想

模型检测是一种关于系统性质验证算法的方法,它通过状态空间搜索的方法来检测一个给定的计算模型是否满足某个时序逻辑公式表示的特定的性质。对于有穷状态系统,该问题是可判定的,即用计算机程序可以在有限时间内自动确定。模型检测技术的优点是自动化程度高,不需要使用者掌握大量的逻辑知识。当所设计的系统不满足某个性质时,模型检测工具可以返回一个反例,通过对反例的解读可以得到性质不成立的原因,为系统的修正提供重要线索。

模型检测基于对系统状态空间的穷举搜索,其状态的数目往往随并发分量的增加呈指数增长。因

收稿日期:2008-10-12
作者简介:王飞明(1983-),男,硕士研究生,主要从事网络信息安全技术研究。

此当一个系统的并发分量较多时,直接对其状态空间进行搜索实际上是不可行的,这就是所谓的状态爆炸问题。由于软件涉及无穷数据域上的运算,所以状态爆炸问题十分突出,已成为模型检测应用于软件系统的一个具有挑战性的难题[1]。

模型检测按模型及属性的描述方式不同可以将其分为两类:自动机模型检测和时态逻辑模型检测[1]。前者将系统 p 和属性 f 表示成自动机 A_P 和 A_f,检测各自接受的语言是否存在关系 $L(A_P)\subseteq L(A_f)$,满足则通过验证,检测的途径是计算 $L(A_P)\cap L(A\subseteq f)$ 是否为空集,不为空则反例产生。时态逻辑模型检测用状态迁移系统 S 表示系统的行为,S_0 是初始状态集($S_0\subseteq S$),用模态/时序逻辑公式 F 刻画系统的性质,通过计算找到 S 中所有满足 F 的状态组成的集合 S',若 $S_0\subseteq S'$ 则通过验证。由于状态空间有限,可以确保检测能够终止。一个时态逻辑公式可以转换为一个等价的自动机,Vardi 和 Wolper 于 1986 年实现将时态逻辑检测转变为自动机模型检测,从而把这两种方法关联起来[5]。模型检测相关的工具和技术有 SMV、SPIN、SCR、Nitpick、Murphi、BDDs 等[6]。

1.2 工作方式

模型检测通常采用状态空间搜索的方法来检测一个给定的计算模型是否满足某个用时序逻辑公式表示的特定属性,其工作方式如图 1 所示。

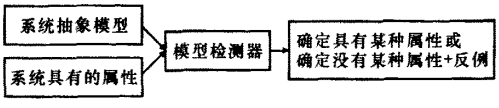

图 1 模型检测的工作方式

模型检测大致可分为以下 3 个步骤:(1)系统建模。在此阶段需要把系统的 Kripke 结构用形式化语言描述出来,即将系统的状态集合及状态迁移关系进行形式化描述。具体使用哪种描述语言要根据模型检测工具来确定,每个不同的模型检测工具都有各自的建模语言。系统建模的关键在于正确地描述好系统的状态迁移关系,这是进行规范验证的前提条件,因为迁移关系描述有误则不能保证规范验证的结果是可靠的。此外,还要注意尽量使用精简的方式来描述系统,避免因为引入过多变量而造成的系统状态爆炸问题。(2)建立规范。通常将需要验证的系统性质表示为时态规范,即时态逻辑表达式。对硬件或软件系统,时态逻辑能够断言系统随着时间演变的行为变化。不同的模型检测工具对规范的描述有不同的要求。如 SPIN 中的规范必须使用 LTL 来描述,SMV 则能同时接受 LTL 和 CTL 描述的规范。虽然建立规范能验证系统是否满足某个规范,但由于模型检测方法是基于语义推导的,因此并不能确定给定的规范是否涵盖了系统应满足的所有性质。(3)系统验证。在理想情况下,系统验证过程是由模型检测工具自动进行的。然而在实际应用中常常需要人工参与,这通常涉及到对验证结果的分析。如果系统不满足规范,模型检测工具往往会把出错的路径呈现给设计者,并能帮助设计者跟踪到出错的地方。产生错误也可能是因为不正确的系统建模或是不正确的规范要求所导致,而错误路径可以用于发现和修补这两类问题。此外,验证任务可能会非正常终止,这是因为模型规模太大以致于不能装进计算机内存。在这种情况下,通常需要更改模型检测工具的参数或调整模型后重新进行验证。

2 模型检测中状态爆炸的优化技术

目前常用的一些减少和压缩状态空间的方法主要有:二叉决策图、符号模型检测、偏序规约、抽象技术和组合逻辑等。这些方法也是目前在模型检测中抑制状态爆炸的主要技术。

2.1 二叉决策图

二叉决策图(BDD)最初由 Bryant 提出[7],它是一种有根无环图,用来存储布尔表达式。该技术极大地减少了存储布尔公式所需的空间,使模型检测工业应用成为可能。二叉决策图是一种表示布尔函数的高效方法,它首先是作为一种简单的形式,即二值判定树(BDT)被提出来的。二叉决策图可以通过对二值判定树进行优化(如对一些计算进行短路、规约)得到。

二叉决策图中每一个非终端结点用布尔变量 $x,y,z,\ldots$ 标记,终端结点用 1 或 0 标记。每一个非终端结点 v 由一个变量 var(v)标识,它有两个后继结点:low(v)对应变量被赋值为 0 的情况,high(v)对应变量 v 被赋值为 1 的情况。每一个终端结点 v 被表示为 value(v),value(v)要么是 0 要么是 1。从初始结点出发到终端结点,我们可以确定该路径上对变量的真值赋值是否使该二值判定树表示的布尔公式为真。如果变量 v 被赋值为 0,那么从根结点到终端结点的路径上的下一个结点将是 low(v)。如果 v 被赋值为 1,那么路径上的下一个结点将会是 high(v)。

二叉决策图中的结点是有序的,序列对应布尔函数的参数表,图的表示也是正则的。对布尔函数基

本操作(如应用、规约、限定、组合)可以用图的一些基本操作(对图中执行子图的插入、删除、遍历、规约等)来实现,并有相关算法自动完成。因此二叉决策图既能较为紧凑地表示布尔函数,也可以方便地表达布尔运算。

对一个二叉决策图不断使用上述规则进行简化,直到无法再简化为止,这样就得到了二叉决策图的规范形式。Bryant 等[7]给出自底向上化简二叉决策图的过程,该过程对于原二叉决策图的大小来说,是线性时间的。由此得到的图被称之为有序二叉判定图(OBDD),它是布尔函数的一种规范表示形式。对于一个用有序二叉判定图表示的布尔函数 f,当我们规定了其中的某一个变量 x_i 为常量 true 或 false 时,我们就得到一个新函数,求新函数的算法对于原有序二叉判定图的大小来说,是线性时间的。

图 2 为一真值表及其布尔函数的一个决策树表示,其中:左边真值表定义一个函数 $f(x_1,x_2,x_3)$,它的一个二叉决策图如右所示,虚(实)树分支表示决策变量是 0(1)。

x_1	x_2	x_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

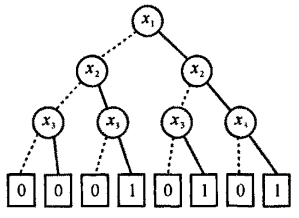

图 2 真值表及其布尔函数的一个二叉决策图表示

OBDD 是以基于图的形式表示的完全描述的布尔函数。对于一个布尔函数,OBDD 是一个有向非循环图。图的顶端是根节点。两个标识为 0 和 1 的终端节点位于底端。其余结点是中间结点。其中内部结点对应于被定义函数的变量,即每一个非终端结点由函数的一个变量标识。对二叉决策图应用如下 3 个归约规则将产生 OBDD 形式的函数的正则表示:(1)删除重复的终端结点。对于每一种标识(0 或 1)的终端结点只保留一个,并把所有指向被删除结点的弧连向剩下的那个结点。(2)删除重复的非终端结点。如果非终端结点 u 和 v 满足 value (u)= value (v), low (u) = low (v), 并且 high (u) = high(v),那么删除这两个结点中的一个并把所有的输入弧连接到另一个结点。(3)删除冗余结点。如果非终端结点 v 满足 low(v)= high(v),那么删除结点 v 并把所有的输入弧连向 low(v)所指向的结点。

对于函数 $f(x_1,x_2,x_3)$的决策树归约过程如图 3 所示。

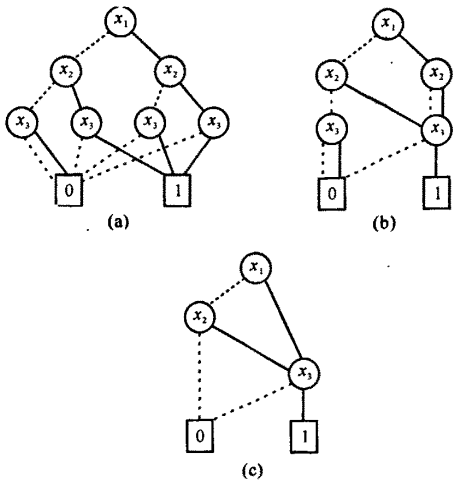

图 3 决策树归约成 OBDD 的过程

(a)删除重复的终端结点;(b)删除重复的非终端结点;(c)删除冗余结点。

OBDD 具有以下优点:(1)根据约减规则,它是可能按指数级增长的布尔函数的正则表示,因此它比对应的真值表表示更简洁。(2)任何关于两个 OBDD 的操作,可以对应成 OBDD 表示的函数的一个布尔操作,因此,它有着更低的上界,是两个结点数的乘积。

基于模型检测方法的规划问题是把规划问题转换成有限状态机问题[1]。由于基于模型检测方法的规划是基于满足检测的,因此采用 OBDD 能够减少随着问题规模的增长而导致的布尔公式的指数级增长所造成的代价,能够比较好的解决状态空间爆炸问题[8]。

2.2 符号模型检测

符号模型检测(SMC)最早由 Carnegie Mellon University 在读博士生 McMillan 提出[9],其主要思想是使用状态集相关的属性来表示集合,用布尔公式来刻画属性,并用二叉决策图在计算机内实现这些布尔公式及其运算。该方法自 20 世纪 90 年代出现以来已经普遍应用在大规模集成电路系统的设计中,它能验证的状态规模已经达到 10^{120},从而能很好地抑制状态爆炸问题。

通常在符号模型检测中的操作对象是一个状态集,而非单个状态。状态集可以用命题逻辑公式表示,而其基数与规模并没有直接联系。例如:一个系统用有限状态机表示如图 4 所示。

用 CTL 时序逻辑公式表示其属性为:$AG(p \rightarrow AFq)$。

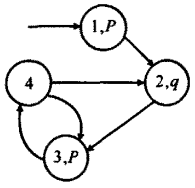

图 4 有限状态机

应用符号化模型检测思想推导公式 $AG(P\rightarrow AF\ q)$成立的状态集:(1)创建一个使 $AG(p\rightarrow AFq)$成立的状态集;(2)由自底向上构建 $AG(p\rightarrow AFq)$;(3)q, AFq, p, $p\rightarrow AF\ q$, $AG(p\rightarrow AF\ q)$。其中 $AF\ q$ 是 q,$AX\ q$,$AX\ AX\ q$,...的合并。

构建 $AG(p\rightarrow AFq)$的状态如图 5 所示。

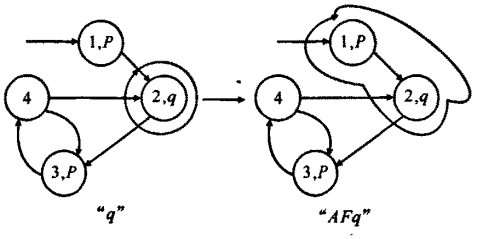

图 5 $AG(p\rightarrow AFq)$状态

状态转换过程如图 6 所示。由图 6 可知,公式 $AG(P\rightarrow AF\ q)$成立的状态集是空集。

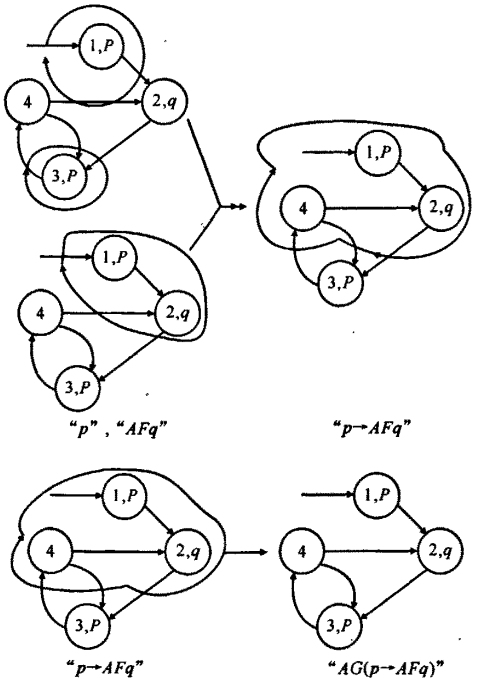

图 6 $AG(p\rightarrow AFq)$状态转换

符号化模型检测技术是对传统模型检测技术的巨大改进,它通过二叉决策图来表示系统模型,并通过 μ-演算来计算出满足系统规范的状态集合,从而可以方便地在超大规模的状态空间中用来验证系统的性质。

目前最常见的符号化模型检测工具是 SMV 及其增强版本 NuSMV。SMV 由 McMillan 于 1987 年开发,而 NuSMV 是在 SMV 基础上加入一些新特性,如对有界模型检测也能有很好的支持效果[10]。

2.3 偏序规约

一个系统可以由多个进程组成,并发执行使得不同进程的动作可以有许多不同的次序。基于对这一问题的认识,可以将某些状态的次序固定,以减少重复验证本质上相同的路径,这种方法被称为偏序归约。

最初,偏序规约是为解决因并发异步模型交替执行导致的空间爆炸而提出的一种技术[11]。并发系统的一次执行可以看作是各事件按执行顺序"插入"得到的一个插入序列,这样,对 n 个事件可产生 $n!$ 个插入序列。有些事件可能与执行次序无关,若把这些事件预先插入到序列中的固定位置,便可以避免验证一批本质上相同的路径。

一般来说,能够减少搜索空间的方法能同时节省时间和内存空间的需求。由于内存空间在某些情况下比时间更为重要,偏序规约便是以时间换空间的典型例子。

2.4 抽象技术

抽象技术是另一种除符号模型检测方法外有效抑制状态爆炸的重要手段[12,13]。传统的模型检测方法主要适用于面向控制的系统,而不太适合与数据路径有关的电路系统或具有复杂数据结构的反应系统。符号模型检测方法虽然可以处理一些与数据处理有关的系统,但是其验证的复杂性往往较高。因此,对于这类系统的验证,常常需要采用抽象技术,即在系统的精确数据值和一个小的抽象数据值之间建立一个映射关系,通过扩展状态和转换之间的映射,产生一个比实际系统小得多的抽象系统。

此外,状态合并是另一种重要的抽象技术,为了压缩状态空间,它通过消除一些不影响规范的变量状态,得到简化的自动机模型,通过验证简化模型的性质来降低模型检测的复杂性。

2.5 其他的优化技术

组合推理。组合推理是一种基于检测局部状态空间的方法。对于大型系统的验证,组合推理方法利用"分而治之"策略,根据系统模块结构,先分别验证系统各个局部模块的性质,再由局部模块的性

质推断整个系统的性质。如果系统满足每一局部性质，并且局部性质的合取蕴涵了整个规范，那么完整的系统也必定满足这个规范。总而言之，组合推理是借助分治策略，从本地属性推导出全局属性，从而减小状态空间。

对称模型检测。由于在多进程组成的系统中，某些进程可能完全类似，并发执行的结果可能产生许多相同或相似的路径。此时，我们可以只搜索在对称关系中等价的一种情形，以避免重复搜索对称或相同的系统状态，这种方法被称为对称模型检测[14]。对称模型检测通过划分等价类来达到简化状态空间的目的。它只考虑等价类中的一种情形，以此为基础，类推同类的其他情形。

值得一提的是，在实际应用中，针对不同验证对象（包括待验模型和待测属性），选取的优化措施也不同。即使是同一种优化技术，在不同工具中实现的策略和方法也不尽相同。

3 模型检测的优势

目前，已经有许多测试方法来保证系统正确可靠，如传统测试方法黑盒、白盒测试法等，这些方法尽管简便易于掌握，但其检测与路径覆盖程度有关，对于具有实时、并行、分布、反应性等特征的系统，测试应用较困难[15]。根据 Floyd 和 Hoare 的思想，用逻辑公式刻画程序语句执行前后的状态，证明在给定输入后，通过程序执行，观察是否能得到期望的输出结果[15]。循环不变式是传统测试方法成功的关键，然而目前尚无获取循环不变式的有效途径。

定理证明是另一种系统测试方法，它需要形式刻画系统模型及待测性质，再通过公理或推理规则来证明系统具有该性质，并可以使用归纳推理来验证无穷状态系统[16]。但是该测试方法应用需要很专业的数学知识，虽有相关工具支持（如 PVS），但是自动化程度低，验证周期长，难以被工业界接受。

与上述方法相比，模型检测具有以下优势：(1)自动执行，检测周期短；(2)发现错误时能给出反例，便于纠错；(3)耗费小（时间、金钱），可用于开发的各个阶段。如在芯片制造的设计阶段使用模型检测，可大幅节约成本；(4)检测完全，模型检测是路径全覆盖，而传统测试一般只能覆盖部分路径。

4 展望

目前，软件模型检测以模型检测的基本原理为基础，融合多种分析、抽象、推理技术，发挥着综合的优势，该领域的实用验证工具研究已成为工业界和学术界共同关注的焦点[17]。今后的发展将主要在以下几个方面进行：(1)集成不同的建模技术或研究新的建模技术；(2)扩展推理系统，如时序扩展等；(3)集成不同的工具，扩展工具的应用范围；(4)形成自动化验证的支撑环境；(5)研究一种新的技术，结合模型检测和定理证明技术各自的优势；(6)由于模型检测将朝着自动化方向发展，在状态压缩、数据结构、编码技术方面将提出更高的要求，因此这方面的研究也将是一个热点。

参考文献：

[1] Clarke E M, Grumberg O, Peled D A. Model checking [M]. Cambridge, Massachusetts: The MIT Press, 1999.

[2] Pnueli A. A Future profession[R]. Sixteenth Annual ACM Symposium on Principles of Distributed Computing, 1997.

[3] Vardi M Y, Wolper P. An automata-theoretic approach to automatic program verification. proc of 1st IEEE Symp on Logic in Computer Science[C]. 1986: 322-331.

[4] Clarke, Burch, Jerry R. Symbolic model checking for sequential circuit verification[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1994(4): 401-424.

[5] Vardi, Moshe Y. On ω-automata and temporal logic: proceedings of the Twenty First Annual ACM Symposium on Theory of Computing[C]. 1989: 127-137.

[6] Bultan, Tevfik. Applying infinite state model checking and other analysis techniques to tabular requirements specifications of safety-critical systems [J]. Design Automation for Embedded Systems, 2008(7): 97-137.

[7] Bryant, Randal E. Symbolic Boolean manipulation with Ordered Binary-Decision Diagrams [J]. ACM Computing Surveys, 1992: 293-318.

[8] Randal E. Graph-based algorithms for Boolean function manipulation [J]. IEEE Transactions on Computers, 1986, 35(8): 677-691.

[9] McMillan, Burch, Jerry R. Symbolic model checking for sequential circuit verification [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1994(4): 401-424.

（下转第 327 页）

晰的递阶结构,然后引入测度理论,通过两两比较,用相对标度使人的判断标量化,并逐层建立判断矩阵,然后求解各判断矩阵的权重,最后计算方案的综合权重并排序。简单地说,即首先根据多目标决策问题的性质和总体目标,按层次分解问题,构成一个由下而上的递阶层次结构。最高层为总目标,称为"目标层";若干中间层是实现总目标所涉及的准则,称为"准则层";最底层是解决问题所选用的各种方案,称为"方案层"。相邻上下层元素之间存在着特定的逻辑关系,通过使用层次或网络结构的方法对层次元素进行成对比较,进而建立比分式关系,最终确定优先次序[3]。

2.2.5 *后台通信模块*

在试验过程中通过串口与上位机进行数据通信,将上位机的数据由 ASCII 码转变为对应的十进制数,并转化为字符类型显示到窗体上,再将 8 位确认码转化成 ASCII 码通过串口发送给上位机。

3 结束语

蚕种选优试验季节性要求很高 强度很大,如果使用原来的装置,就必须在试验完成后,才能进行人工评定蚕种的等级,花的时间长,效率非常低;而采用该系统后,蚕种的评级选优工作可以通过时实数据采集与试验同步完成,不但能够提高评级效率,而且可以避免人为造成的失误,提高蚕种选优的准确性。

参考文献:

[1] 浙江省丝绸公司. 制丝手册[M]. 北京:中国纺织出版社,1994.

[2] 岑咏霆. 质量管理教程[M]. 上海:复旦大学出版社,2005.

[3] 吴育华. 管理科学基础[M]. 天津:天津大学出版社,2001.

(责任编辑:尹 闯)

(上接第 324 页)

[10] McMillan. Symbolic model checking: an approach to the state explosion problem [R]. Camegie-Mellon University, Department of Computer Science, Report CMU-CS-92-131, 1992.

[11] Basten, Twan Bosnacki. Cluster-based partial-order reduction[J]. Automated Software Engineering, 2004 (10):365-402.

[12] Clarke, Grumberg. Model checking and abstraction [J]. ACM Transactions on Programming Languages and System(TOPLAS),1994 ,16 (5):1512-1542.

[13] Clarke, Grumberg. Counterexample guided abstraction refinement for symbolic model checking [J]. Journal of the ACM, 2003,50 (5):752-794.

[14] Sistla A. Symmetry and reduced symmetry in model checking [J]. ACM Transaction on Programming Languages and Systems, 2004, 26(4):702-734.

[15] 古天龙. 软件开发的形式化方法[M]. 北京:高等教育出版社,2005.

[16] Michael Huth, Mark Ryan. 面向计算机科学的数理逻辑系统建模与推理[M]. 何伟,樊磊,译. 北京:机械工业出版社,2005.

[17] 古天龙,刘华东. 基于符号有序二叉决策图的装配序列生成技术[J]. 计算机集成制造系统, 2008 (2): 321-328.

(责任编辑:韦廷宗)