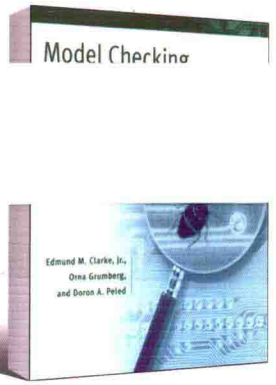

Edmund M. Clarke, Jr.
[美] Orna Grumberg 著
Doron A. Peled

吴尽昭 何安平 高新岩 译

模型检测

Model Checking

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

模 型 检 测

Model Checking

Edmund M. Clarke, Jr.
[美] Orna Grumberg 著
Doron A. Peled

吴尽昭 何安平 高新岩 译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

模型检测是一种用于自动验证有限状态并发系统的技术，与基于模拟、测试和演绎推理的传统技术相比，具有许多方面的优势。本书共分 18 章，涵盖的主要内容包括模型检测的基本知识、系统建模、时序逻辑、符号模型检测技术、SMV 模型检测器、模型检测与自动机理论、偏序约简、抽象解释、有限状态系统的无限簇、实时系统验证等。

本书既适合从事计算机科学、电子科学、电气工程、工业制造等复杂系统研究的科研人员阅读，也适合系统管理、测试部门的企事业单位人员作为参考用书。

版权贸易合同登记号　图字：01-2015-6642

图书在版编目（CIP）数据

模型检测 /（美）埃德蒙 • M. 克拉克等著；吴尽昭等译. — 北京：电子工业出版社，2018.11

书名原文：Model Checking

ISBN 978-7-121-35274-4

I. ①模… II. ①埃… ②吴… III. ①自动检测系统 IV. ①TP274

中国版本图书馆 CIP 数据核字（2018）第 243225 号

策划编辑：冯小贝
责任编辑：冯小贝　　特约编辑：李秦华
印　　刷：三河市鑫金马印装有限公司
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　印张：15　字数：346 千字
版　　次：2018 年 11 月第 1 版
印　　次：2018 年 11 月第 1 次印刷
定　　价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：fengxiaobei@phei.com.cn。

译 者 序

作为一部有关模型检测原理与实践方法的权威性专著，*Model Checking* 一书自面世以来即受到国内外相关领域研究人士的广泛关注，被认为是第一次对模型检测理论到工业实践的一系列研究成果进行了全面整理与汇总。Amir Pnueli 教授在本书序言中已经指出，对“模型检测技术从一个纯理论的学科转变为实际可行的技术”所进行的研究给学界带来了巨大的冲击。回溯这一时期的研究不难发现，对于持续多年的并发系统形式验证技术，特别是模型检测技术的研发和实践应用的探索过程，其实更能准确地描述出当时相关领域的研究趋势与动向。事实上，在过去的几十年间，大量坚韧而敏锐的研究者和探索者对于计算机复杂系统正确运行的实际有效性验证方法的探知与重新发现，带动并促进了新的技术手段与研究方法的产生，重新调整与塑造了研究的议程，转变了历史研究的重点和方向。在计算机系统验证技术的研究领域中，这些变化使解决实际技术问题取得了重要的进展。而在诸多令人欣喜的研究成果中，模型检测技术以其高效的全自动化验证优势、遍布全局空间的完备性验证特性和缺陷轨迹复原的技术独创资本，成为引人瞩目的理论重构与研究新领域。同时，这项技术的实际价值也由学界辐射到产业界，被广泛而成功地应用于计算机硬件、通信协议、时序电路、控制系统、安全认证协议等方面真实的工业设计与分析验证中，推动了形式化验证领域的根本性改变。

Edmund M. Clarke, Jr.教授联袂 Orna Grumberg 和 Doron A. Peled 两位世界知名专家共同撰写的这本书，对并发系统自动化验证的研究具有开创性意义，可以说是第一本全面介绍模型检测的理论与实践的专著，也是关于模型检测的更为全面的参考资料之一，它涵盖了大多数用于模型检测的主要技术和实际方法。2007 年 ACM 图灵奖得主之一的 Clarke 教授在计算机软/硬件验证、自动定理证明、形式方法等方面享有崇高的国际声誉。他为并发系统自动化校验开辟了一条新的路径，成为近三十年来计算机科学基础研究的重要热点之一。在这部著作的撰写过程中，Clarke 教授与他的研究团队对于计算机系统自动化验证的基本思想有着非常独特的构思和研究路径。书中除了翔实可据的模型检测基本知识和实际验证方法，还介绍了 SMV 和 SPIN 两种流行的模型检测器，并由此引申出新的理论架构和极富创新的观点，从而在很大程度上解构了基于仿真、测试和演绎推理的传统方法，因而获得了一种不同于既有观念的认知，揭示出模型检测在复杂系统状态空间下的技术属性与研发价值。尽管这部著作写于十余年前，因而并没有也无法包含一些本领域更新的研究进展，但毫无疑问的是，这部著作对于计算机系统自动化验证技术与工业实践研发的深入研究，为当下的研究者和从业者提供了基础性的重要参考与研究构建。该书出版之后，有界模型检测等新的验证技术的不断涌现，也印证了这部著作的写作意义和时代影响力。这

也是我们在时隔十余年之后，承蒙多方襄助之力，将这部著名的学界力作予以翻译并付梓，且与学界同仁共分享的主要初衷。

本书的出版惠蒙学界同仁的鼓励、支持与有关机构的资助，于此一并致谢。鉴于本书广博深入的研究精髓，是以译者不揣浅陋，勉力译成，书中难免存遗、疏、漏之处，尚祈读者不吝指正，也继续恭候学界的使用与更深入的发现。

吴尽昭

2018年9月26日谨识于广西南宁

序　言

目前，大家广泛认为有某种原因阻碍了实践“帮助计算机就会帮助人类更多”的理念，这使得我们更易于将复杂、敏感的系统丢给其他人去实现。这既不是机器的计算速度引起的，也不是计算能力造成的，而是工程师普遍缺乏设计并实现在所有环境中都会正确运行的复杂系统的自信。

这种设计有效性，即尽可能早地保证设计的正确性，是任何系统开发过程中都会遇到的挑战；而且关于这一问题的解决过程，在整个开发周期的成本和时间预算中所占的比例不断增加。

目前，保证设计有效性的方法依旧是持续了多年的模拟和测试技术。工程师已经在这两种技术领域积累了丰富的经验。在调试的早期阶段，这两种技术非常有效，但是经过它们检测后，系统设计仍然可能包含大量错误。随着系统设计越来越精确，这两种方法的工作效率急剧降低，每发现一个小错误都需要花费大量时间。此外，这两种方法还会导致一系列的问题：没有人知道这种技术的查错极限，更没有人能预测出经检测后设计中还剩余多少错误。随着设计复杂度的急剧增加，比如从大约五十万门的芯片设计提高到五百万门的芯片设计，一些有远见的项目经理已经预见到这些传统方法将要崩溃，并且它们将无力再扩充或提升。

形式化验证技术是一个非常具有吸引力的验证方法，可以用于替代模拟与测试，这种方法是本书的主旨。模拟与测试能够检测系统的部分可能行为与情况，但是不能确定系统是否还含有致命错误，而形式化验证却能对系统的全部行为进行彻底的检测。因此当系统设计被形式化验证方法证明为正确时，就蕴含了所有的行为已被检测通过，并且再也不用考虑是否达到足够的覆盖率或者是否含有行为缺失这样的问题。

这些年已经提出了多种形式化验证技术。本书集中介绍的模型检测方法，通过对给定反应系统(模型)中所有可达状态与行为进行(显式的或隐式的)彻底的检测，来验证系统规约的行为特性。

与其他方法相比，模型检测方法有两个显著的优势：

- 自动进行，并且不要求使用者具有专业的数学知识(如定理证明)和经验。对于任何具有模拟检测经验的人，完全能够使用模型检测进行验证。同当前的验证方法相比，模型检测可以被视为一个更高级的模拟工具。
- 如果模型检测得出设计未能满足某些期望的性质，那么将产生一个反例来说明系统违反性质的具体行为。这种缺陷轨迹有助于理解检测失败的真实原因，同时也提供了修复此问题的重要线索。

这两个重要的优势，再加上可以对天文数字般的状态进行彻底的隐式枚举的符号模型检

测方法，引起了形式化验证领域的根本变革，将模型检测技术从一个纯理论的学科转变为实际可行的技术。模型检测技术可以融入许多工业开发流程中，已经成为一种确保设计有效性的有价值的方法。

工业界普遍认同模型检测具有巨大的实力和潜力，大量的研究人员也在致力于模型检测技术的开发，所开发的产品已经应用于大型先进半导体电路和处理器公司的研发过程中。

非常幸运能够发现这本关于模型检测的原理与方法的权威性著作，这本书的作者从模型检测思想着手，全面介绍了模型检测中各种令人惊讶的技术，并最终构建出一个成功的技术体系。

我对这本出色的参考书非常有信心。这本书将有助于读者(包括学生与从业人员)理解形式化验证特别是模型检测技术的原理与实现。

Amir Pnueli

前　言[①]

计算机科学中的多个领域都涉及有限状态并发系统，特别是数字电路与通信协议与这种系统关系紧密。这些系统研发过程中出现的逻辑错误对于电路设计者和程序员而言，都是一个非常棘手的问题，可能会推迟新产品的上市时间，也可能导致一些投入使用的重要设备发生故障。目前广泛使用的验证技术是测试和模拟，但当电路或协议的状态规模巨大时，这些技术无疑会遗漏重要的错误。虽然定理证明器、项重写系统和证明检测器都经过了长期和大量的研究，但是这些技术不但耗时，还常常需要许多人工干预。在 20 世纪 80 年代，一个被称为时序逻辑模型检测的验证技术由美国的 Clarke 与 Emerson[61]以及法国的 Quielle 与 Sifakis[219]分别独立提出。这种方法使用命题时序逻辑来表示性质规约，电路与协议被建模为状态变迁系统，并且提出一个用于确定规约是否在变迁系统上为真的高效查找过程，即检测变迁系统是否是性质规约的模型。

同机械化的定理证明或证明检测相比，模型检测在验证电路与通信协议方面有着多个重要的优势，其中最重要的优势是检测过程的全自动化，使用者只需提供被检测的模型与性质规约的高阶描述。模型检测算法要么以结果为真终止，此时模型满足规约；要么给出一个反例指出性质违反规约的原因。在复杂变迁系统中，这种反例非常有益于发现和修正细微错误。模型检测过程相当快，通常大约几秒钟产生一个结果。在检测过程中，由于可以部分地检测规约，所以在获得有用信息之前不必构建完整的系统模型。当性质规约不能满足时，可以通过精心构建与当前规约不同的公式，来检测并定位错误的源头。除此之外，描述性质规约的逻辑能直接表示许多并发系统推理所需的性质。

模型检测的主要缺点是状态爆炸，这种情况发生在由许多系统组件并发演化构成的系统中。在这种情况下，整个系统状态的数目将按组件数量呈指数级增长。由于这个问题，许多形式验证的研究者预言模型检测对于大型系统绝对是不实用的。但是，在 20 世纪 80 年代后期，模型检测技术所能检测的变迁系统的规模显著地增大了。

这种增长归因于高效表示布尔函数的二叉判定图结构的应用，它不但简洁地表示了变迁系统，也提升了布尔运算的速度。符号模型检测方法对同步电路特别有用。在验证异步协议时，可通过偏序约简技术来减少状态空间的规模。偏序约简的基础是，不同顺序事件对应的计算无法被性质规约区分，可以认为是等价的，因此只需为等价类保留一个典型的计算，检测这种约简空间即可。

基于以上这些技术，以及稍后在本书中介绍的其他一些技术，现在模型检测已经作为一种实用的验证技术在工业界中得到了广泛使用。实际上，几家公司正开始把模型检测工具推向市场。

---

① 中文翻译版的一些图示、符号、正斜体沿用了本书英文原版的写作风格，特此说明。

我们认为这本书既可作为模型检测的简介，也可作为研究者的参考。我们试图包含尽可能多的内容而使其完整，但是这个领域的研究进展如此之快，以至于勉强跟上令人兴奋的新研究成果都不可能。这本书的若干部分与其他部分相比更加专业，在第一次阅读时可以跳过它们，这些部分在书中已经标出，它们主要针对从业者与研究者。我们真诚地希望这本书能够激励读者在模型检测领域做出更好的研究。

最后，作者要感谢那些帮助创作这本书的人。首先想要对 David Long 表示我们的谢意，他的努力使这本书顺利出版。我们也想要感谢那些审阅初稿的人们，他们是：Eric Allen、Ilan Beer、Armin Biere、Sergey Berezin、Sergio Campos、Ching-Tsun Chou、Allen Emerson、Kousha Etessami、Nissim Francez、Masahiro Fujita、Yair Harel、Wolfgang Heinle、Hiromi Hiraishi、Neil Immerman、Somesh Jha、Irit Katriel、Shmuel Katz、Bob Kurshan、Kim G. Larsen、Yuan Lu、Jan Maluszynski、Will Marrero、Marius Minea、Bud Mishra、Ulf Nilsson、Wojciech Penczek、Amir Pnueli、Toshio Sekiguchi、Subash Shankar、Zeev Shtadler、Prasad Sistla、Frank Stomp、Wolfgang Thomas、Moshe Vardi、Dong Wang、Pierre Wolper、Bwolen Yang、Husnu Yenigün、Yunshan Zhu。如果遗漏了对这本书有所帮助的人们，我们在此表示歉意。Edmund Clarke 感谢 Michael Shostak 在写书的过程中所给予的鼓励。Doron Peled 感谢 Marta Habermann 于 1998 年在 CMU 的春季学期中提供了一个舒适安逸的居所。

目　录

第1章 绪 论

模型检测是一种针对有限状态并发系统的自动验证技术，相比于基于模拟、测试和演绎推理的传统技术，它有许多优势。这种方法已经应用于验证复杂的时序电路设计和通信协议，并取得了成功。模型检测的最大困难来自状态空间爆炸。在验证具有大量交互组件的系统或者大规模数据系统[如数据通路电路(DataPath)]时，全局状态集合具有巨大的数量级，引起了状态空间爆炸问题。在过去十年中，人们已经找到了多种方法来减缓状态空间爆炸。在本章，将从硬件和软件设计两个方面来比较模型检测与其他形式化的验证方法；同时将介绍使用模型检测来验证复杂系统的设计流程；此外，也将回顾各种模型检测算法的发展历程，并讨论若干缓解状态爆炸的有效手段。

1.1 形式化方法的需求

现今，硬件和软件广泛应用于各类故障敏感系统，如电子商务、电话交换网、高速公路和航线控制系统、医疗器械等，也常常听到由微小的硬件或软件缺陷引起故障并最终造成事故的例子。有代表性的一个事故发生在 1996 年 6 月 4 日，Ariane 5 火箭在发射升空不到 40 秒爆炸解体，事故调查委员会最终发现此事故的原因是火箭姿态计算机中的软件错误：在发射过程中，当 64 位长的浮点数转换成 16 位有符号整数时，发生了一个例外，但例外处理代码没有覆盖到这个转换过程，导致该计算机死机，同样的失误也导致备份计算机死机。最终，错误的飞行姿态数据传送到火箭的主计算机，导致了这场灾难。调查出此故障的小组建议了若干避免类似事故的策略，其中就包含对 Ariane 5 火箭的软件进行验证。

很明显，对硬件与软件系统可靠性的需求是急迫的。随着我们对这种系统的依赖与日俱增，确保其正确运行的工作也越来越重要。遗憾的是，现在我们更加依赖连续运转的系统，为了重获安全性而简单关闭故障子系统的方案已不再可行，在一些场合中，关闭设备可能产生更危险的问题。即使故障不会造成生命威胁，替换重要代码或电路以保证系统正确运行，在经济上也可能行不通。

由于在汽车、飞机与其他安全攸关系统中因特网与嵌入式系统的成功应用，未来我们极有可能更加依赖于计算机设备的正确运行。而从目前来看，这种依赖的节奏正在加快。对应于技术的快速提升，增强我们对系统设计正确性的信心的方法，变得更加重要。

1.2 硬件与软件验证

复杂系统的基本验证方法是模拟、测试、演绎验证和模型检测。模拟和测试[202]都是在系统实际使用之前做实验验证，不同的是模拟对系统的抽象或模型进行验证，而测试是对实际

产品进行验证。虽然模拟验证的是电路设计，测实验证电路本身，但是这两种方法的大体流程都是在验证时，对系统的某些点注入信号并在另外的一些点观察生成信号是否符合要求。对于软件验证，模拟和测试的基本方法是为软件提供一些输入而后观察对应的输出。所以对于可能具有大量错误的系统设计而言，这两种方法的费效比可能比较高，而且它们几乎不可能检测所有交互故障和潜在缺陷。

演绎验证指使用公理和证明规则来证明系统正确性。在早期的演绎验证研究中，焦点主要集中在证明重大系统的正确性上。可以设想，这种系统的功能正确性如此重要，以至于开发人员或验证专家(通常是数学家或是逻辑学家)将不计成本和时间去验证此系统。在演绎验证的发展初期，所有证明过程都靠人工构造；后来研究者才意识到可以开发软件工具来实现公理和推理的证明过程。使用这些工具也能系统地对从某证明状态开始的不同证明路径进行研究。

计算机科学家广泛认可了演绎验证的重要性，演绎验证也深远地影响了软件发展的各个领域(如不变量的概念始于演绎验证的研究)。但演绎验证是一个耗时的过程，即使单个协议或电路的证明过程也可能持续若干天或多个月，而且这种方法的使用者也局限于一些在逻辑推理方面受过培训并具有相当经验的专家。因此演绎验证的使用机会相当少。它主要应用于高敏感系统(如安全协议验证)，因为在这种系统中必须投入足够多的资源来保证其安全使用。

但仍存在不能被算法代替的数学任务，可计算性[142]理论中描述了算法的局限性，它指出不存在能够判定任意计算机程序(如使用 C 或 Pascal 书写的程序)是否终止的算法。这也就直接限制了可自动验证问题的范围，比如程序是否能正确终止的问题，一般而言是不能自动验证的，所以说，大部分证明系统不可能完全自动运行。

演绎验证的一个优点是它能被用于无限状态系统的推理，此时有一部分推理工作可以自动进行。但即使待验证的性质是真的，也无法确定到底需要多少时间和内存来实现推理过程。

模型检测限定在验证有限状态并发系统上，这种限定保证了验证工作可以自动进行。模型检测算法通常对系统状态空间进行穷尽搜索来确定性质的真假。如果资源充足，检测过程总能以是或否的验证结果终止。除此之外，这种技术能够用高效的算法实现，从而可以在中等规模的计算机(但不是通常的台式计算机)上运行。

虽然限制于有限状态系统可能是模型检测技术的一个主要缺点，但是它非常适用于若干种重要系统的验证，比如硬件控制器是有限状态系统，并且许多通信协议也是有限状态系统。在非有限状态系统中，也可以把模型检测与抽象和归纳方法结合起来进行验证，不仅如此，在许多情况下都可以把无约束数据结构限制到特殊的有限状态系统上进行验证，如可以把包含无约束消息队列程序的队列个数限制到 2 或 3 这样小的数来调试。

因为模型检测能自动进行，并且具有很广泛的实际应用，所以它比演绎验证更优越，但是完全使用定理证明来验证一些极端重要系统的情况也是存在的。一个激动人心的新方向[220]研究如何把演绎验证与模型检测结合起来，因此复杂系统的有限状态部分也许能够完全自动地进行验证。

1.3 模型检测的流程

使用模型检测技术来进行系统设计的验证包含以下三个步骤，每一步骤都将在后文详细叙述。

建模 第一步需要将设计转化为能被模型检测器接受的形式。在许多情况下这只是个简单的编译过程，但在一些时候，由于验证时间和计算机内存的限制，可能还需要使用抽象技术约简不相关或不重要的细节来得到设计的形式化模型。

规约 在验证之前，需要声明设计必须满足的性质。性质规约通常是以某种逻辑的形式表示。对硬件与软件系统验证而言，通常使用时序逻辑规约系统的性质，这种逻辑体系能表示系统行为随时间的变化。性质规约过程中最重要的问题是完备性。虽然模型检测提供了检测模型是否满足给定性质的一套方法，但是这套方法并不能保证性质规约确切地表达了待验证系统所需满足的所有性质。

验证 理想中验证过程应该是完全自动的。但实际上它常常需要人的协助，其中之一就是分析验证结果。当得到失败结果后，通常可以给用户提供一个错误轨迹，可以把它看成所检测性质的一个反例，从而使设计者能够跟踪错误发生的具体位置。当分析错误轨迹并改正系统设计后，需要再次进行模型检测，重新验证，直到验证通过。

错误轨迹也可能由建模或刻画性质规约过程的失误导致(常常称为假否定)，错误轨迹也能用于确定和修复这两类错误。另外，由于计算机的内存限制，当验证过程需要大量内存时，验证可能不会在有限时间内正常终止而产生错误轨迹。这种情况下，需要改变模型检测器的若干参数或直接约简模型(比如使用抽象技术)，然后重做验证。

1.4 时序逻辑与模型检测

时序逻辑能够在不引入时间细节的情况下描述事件序列，已经证实这种逻辑对并发系统的刻画非常成功。这种逻辑最初由哲学家在研究自然语言的时间参数[145]时得出。目前学术界提出的时序逻辑种类繁多，但是大部分都含有类似 $\mathbf{G}f$ 的运算符，表示如果此公式为真仅当子公式 f 总为真(即 f 全局为真)。比如刻画两个事件 e_1 和 e_2 不能同时发生，可以记为 $\mathbf{G}(\neg e_1 \vee \neg e_2)$。一般可以根据线性和分支的时间假设来对时序逻辑进行分类。本书中涉及的时序逻辑公式的语义将由标记状态变迁图给出，由于历史的原因，这种结构被称为 Kripke 结构[145]。

Burstall [48]，Kröger[158]与 Pnueli [216]等一些研究者提议使用时序逻辑推导计算机程序。Pnueli [216]第一个使用时序逻辑推导了并发系统，即通过描述程序语句的公理集合来证明我们关心的程序性质。Bochmann [25]，Malachi 与 Owicki [184]将这个方法应用到了时序电路系统。因为这种方法的证明过程是手工构造的，所以此技术常常很难在实际中应用。

在 20 世纪 80 年代早期，Clarke 与 Emerson [61,103]提出了时序逻辑模型检测方法以期实现上述方法的自动执行，因为检验具体模型是否满足公式比证明公式对所有模型有效要简单得多，并且这个技术完全可以通过高效算法实现。Clarke 与 Emerson 提出的 CTL（分支时序逻辑）模型检测算法的复杂度无论对于待验证系统的抽象模型规模，还是时序逻辑刻画的性质公式长度，都是多项式时间的，他们也给出了在不改变算法复杂度的情况下，处理公正性[120]问题的方式。在许多以公正性假设为前提的并发程序正确性验证中，公正性考虑是非常重要的，例如，在互斥算法中为了避免出现“饥饿”，要求每个进程无限次出现。

几乎在同时，Quielle 与 Sifakis [219]为 CTL 的一个子集给出了模型检测算法，但是他们没有分析算法的复杂度。稍后 Clarke，Emerson 与 Sistla[63]提出了改进的算法，这个算法相对于公式长度与状态变迁图(Kripke)规模的积是线性的。此算法被集成在 EMC 模型检测器中，而 EMC 被广泛用于检验网络协议与时序电路系统[28,29,30,31,63,98,197]。早期的模型检测系统在检测给定时序公式时，能以每秒 100 个状态的速度来验证拥有 $10^4 \sim 10^5$ 状态规模的状态变迁图。尽管限制性很大，但模型检测器仍然在若干已经上市的电路设计中成功检测出了隐含的错误。

Sistla 与 Clarke[232,233]分析了不同时序逻辑系统中的模型检测问题，指出对于线性时序逻辑(LTL)系统而言，模型检测问题是 PSPACE 完全的(PSPACE-complete)。Pnueli 与 Lichtenstein[173]重新分析了检验线性时序公式的复杂度，发现虽然复杂度似乎相对于公式长度呈指数增长，但对于状态变迁图规模却呈线性增长。基于这个观察，相对于短的线性时序公式而言，模型检测的高复杂度也是可接受的。同一年，Fujita[119]实现了基于 tableau 结构的 LTL 模型检测器，并给出了采用这种工具验证硬件系统的方法。

CTL*是一种表达能力更强的逻辑体系，它把分支时间与线性时间的算子结合在一起。这种逻辑的模型检测问题首先在 Clarke，Emerson 与 Sistla[62]的论文中提及，他们指出 CTL*模型检测是 PSPACE 完全的，并确定其与 LTL 的模型检测有相同的复杂度。这个结果指出 CTL*与 LTL 模型检测在状态图规模与公式长度两方面，有着相同的算法复杂度，所以在进行实际的模型检测时，将问题限制到线性时序逻辑并不会有效地降低模型检测的复杂度[106]。

另一些研究者提出了检验并发系统的其他方法，大多数方法使用自动机描述系统规范和系统实现，通过检测系统实现的行为是否与规范一致来进行验证。因为实现与规范使用相同的模型来表示，某一层的实现也可以直接作为下一层改进的规范。Kurshan[1]在语言包含方面做了大量工作，开发了一个称为 COSPAN[132,133,162]的高效检验器。Vardi 与 Wolper[245]首次将 ω 自动机(定义在无限字串上的自动机)用于自动验证，并且给出了如何依据 ω 自动机的语言包含来实现线性时序逻辑模型检测问题。此外，自动机间的一致性问题也得到了研究，包括观测的等价性[77,196,224]与各种各样的精化关系[77,195,223]。

1.5 符号算法

在早期的模型检测算法中，变迁关系经常被显式地表示成邻接表。对包含少量进程的并发系统而言，其状态的规模通常也相当小，所以邻接表方式很实用。但是在包含大量并发组

件的系统中，全局状态变迁图的状态数目会大到难以处理的程度。1987 年秋天，还是卡耐基·梅隆大学研究生的 McMillan 认识到采用符号方法表示状态变迁图可以验证更大规模的系统[46,191]。这种新的符号表示方法基于 Bryant 的有序二叉判定图(OBDD)[34]，这种采用 OBDD 布尔方程的方法比合取或析取范式更简洁，并且高效 OBDD 算法也已经存在。由于符号方法符合待验证的电路或协议所确立的状态空间的若干规律，所以符号方法可以验证具有大量状态的系统——超过状态图遍历算法可处理规模的若干数量级。而且将 Clarke 和 Emerson 的早期 CTL 模型检测算法[61]与状态变迁图的 OBDD 表示相结合，甚至可以验证状态数超过 10^{20} 规模的系统[46,191]。后来其他研究者又不断推出了各种基于 OBDD 的改进技术，使得可以验证的状态超过了 10^{120} 量级[43, 44]。

对时序电路和协议而言，使用符号方法隐式建模是非常自然的：先将电路和协议中的状态变量看成一个集合，对此集合的一组布尔赋值表示一个状态编码，变迁关系可以表示成定义在两个变量集合上的布尔公式，这两个集合一个是现态变量集合，另一个是次态变量集合，现在此方程就可以方便地转化为二叉判定图。而基于符号方法的模型检测算法对应着计算从变迁关系中得到的谓词变换的不动点，不动点本质上表示了并发系统各时序性质对应的状态集合。在这种新的模型检测方法中，谓词变换和不动点都用 OBDD 来表示，这样就可以避免显式地构造并发系统的状态图。

McMillan 开发了符号模型检测系统 SMV[191]，在他的博士论文中有对应的介绍章节。SMV 使用一种层次化的有限状态并发系统描述语言，这种语言支持时序逻辑语法的系统规约公式。SMV 模型检测器从输入语言中提取变迁系统的 OBDD 表示，然后使用基于 OBDD 的搜索算法来判断系统是否满足性质规约。如果变迁系统不满足某性质规约，检测器会产生执行路径来表明性质规约不能满足的原因。目前 SMV 系统使用广泛，已验证了大量的实例，这些应用表明 SMV 可以胜任工业级的应用。

采用模型检测成功验证了 IEEE 的 Futurebus+标准(IEEE 896.1-1991 标准)中的高速缓存一致性协议，使人们深刻地认识到这种方法的强大能力。Futurebus+高速缓存一致性协议的研发始于 1988 年，但是之前所有对该协议验证的尝试都是基于非形式化的技术。1992 年夏天，卡耐基·梅隆大学的研究人员用 SMV 语言构造了该协议的精确模型，然后使用 SMV 验证此变迁系统模型是否满足高速缓存一致性的形式化规范[66,179]。结果他们发现了以前从未发现的错误，以及协议设计本身的潜在缺陷。这是第一次成功应用自动验证工具检测 IEEE 标准的案例。

在验证规模越来越大的电路或协议实例时，符号模型检测方法的 CPU 时间需求增长率体现了其强大的处理能力。将其用于已经被各团队研究过的实例后，观察得知符号检测方法下的 CPU 时间需求增长率相对于电路组件个数而言只是一个小规模的多项式[18,43,44]。

许多学者独立研究发现 OBDD 可用于表示大规模的状态变迁系统。Coudert，Berthet 和 Madre 通过实现基于自动机状态空间的宽度优先搜索，提出了确定性有限状态自动机的等价算法[81]。在此算法中，他们用 OBDD 表示两个自动机的变迁函数。Pixley 提出的算法[213,214,215]也与此类似。另外，Bose 和 Fisher[26]，Pixley[213]，Coudert，Madre 和 Berthet[82]等人的小组都实验了使用 OBDD 的模型检测算法。