模型检测研究进展

朱雪阳 张文辉 李广元 吕毅 林惠民

中国科学院软件研究所
计算机科学国家重点实验室
北京 100190

目录

摘要

保证计算机软硬件系统的正确和可靠是计算机科学与应用中的重要问题。在为此提出的诸多理论和方法中，模型检测(model checking)以其简洁明了和自动化程度高而引人注目。模型检测的研究大致涵盖以下内容：模态/时序逻辑、模型检测算法及其时空效率的改进以及支撑工具的研制。这几个方面之间有着密切的内在联系。不同模态/时序逻辑的模型检测算法的复杂性不一样，优化算法往往是针对某些特定类型的逻辑公式。本文将对模型检测的典型应用、主要研究内容及关键技术分别加以阐述，最后介绍国内研究人员在该领域的部分新进展。

Abstract

How to assure the correctness and reliability of computer hardware and software systems is an important problem of computer science and application. Among theories proposed as solutions to this problem, model checking has become a very attractive and appealing approach, because of its simplicity and high level of automation. Research on model checking covers the following subjects: modal/temporal logics, model checking algorithms, efficiency of model checking with respect to time and space, and development of model checking tools. These aspects are closely related. Complexities of model checking algorithms vary very much for different modal/temporal logics; optimizations are often targeted at certain types of logic formulas. We discuss these aspects of model checking as well as some new achievements in this area in China.

1. 引言

随着计算机软硬件系统日益复杂，如何保证其正确性和可靠性成为日益紧迫的问题。对于并发系统，由于其内在的非确定性，这个问题难度更大。在过去三十多年间，各国研究人员为解决这个问题付出了巨大的努力，取得了重要的进展。在为此提出的诸多理论和方法中，模型检测(model checking) [81]以其简洁明了和自动化程度高而引人注目。

1981 年，Clarke 和 Emerson 提出了描述并发系统性质的时序逻辑 CTL，以及检查有穷状态并发系统是否满足 CTL 公式的算法，开创了模型检测这一研究方向[1]。现在模型检测已被应用于计算机硬件、软件、通信协议、控制系统、安全认证协议等领域，取得了令人瞩目的成功，成为分析、验证并发系统性质的最重要的技术，并被 Intel、IBM、微软等公司用于生产实践中[2]。模型检测的工作于 1998 和 2005 年两度获得 ACM Paris Kanellakis Theory and Practice

Award。2007 年，模型检测的创始人 Clarke、Emerson 和 Sifakis 共同获得了国际计算机科学界最高奖--ACM 图灵奖。

模型检测的基本思想是用状态迁移系统（S）表示并发系统的行为，用模态/时序公式（F）描述系统的性质。这样，“系统是否具有所期望的性质？”就转化为数学问题“S 是否是 F 的模型？”。对于有穷状态系统，这个问题是算法可判定的。与其他验证方法相比，模型检测有两个显著的特点：一是可以自动进行，无须人工干预；二是在系统不满足所要求的性质时，可以生成反例，准确地指出错误的位置。这两个特点对模型检测在实际应用中的成功起了至为重要的作用。

本文将介绍模型检测的典型应用、主要研究内容及关键技术、及国内研究人员在该领域的部分进展。

2. 典型应用

早期的模型检测侧重于对硬件设计的验证，随着研究的进展，模型检测的应用范围逐步扩大，涵盖了通讯协议、安全协议、控制系统和一部分软件。电子线路设计验证的例子包括：先进先出存储器的验证，验证的性质包括输入和输出的关系[3]；浮点运算部件的验证，验证的性质包括计算过程所需满足的不变式[4]。

对于复杂的协议和软件，模型检测的验证基于抽象和简化的模型，从验证角度来讲，这并非十分严谨。更为确切的说法应该是协议和软件的分析。协议验证的例子包括：认证协议的验证，验证的性质包括对通话双方的确认[5][6][7]；合同协议的验证，验证的性质包括公平和滥用的可能性[8]；缓存协议的验证，验证的性质包括数据的一致性和读写的活性[9]。

由于软件相对比较复杂，并且软件运行中可能到达的状态个数通常没有实质上的限制，验证通常局限于软件的某些重要组成部分。软件验证的例子包括飞行系统软件的验证，验证的性质包括系统所处的状态和可执行动作之间的关系[11][12]；铁路信号系统软件的验证，验证的性质包括控制信号与控制装置状态的关系[13][14] ；数字信号处理算法的缓存分析[10]。

测试是保证软件产品可靠性和正确性的传统手段。但它的主要局限性在于：对给定数据集通过了测试并不能保证在实际运行中对其它输入不发生错误。模型检测不是针对某组输入，而是面向某类性质来检查系统是否合乎规约。在系统不满足所要求的性质时，模型检测算法会产生一个反例（一般是一条执行路径）来说明不满足的原因。这一功能与测试有异曲同工之处。与测试技术相结合，也是模型检测在软件方面的重要应用[110][111]。

模型检测还可以应用于其它方面，其基本思想是将一个过程或系统抽象成一个有穷状态模型，加以分析验证。这方面的例子包括：化学过程验证，验证的性质包括阀门、管道和容器的状况[15]；公司操作过程分析，分析的内容包括操作过程是否具有所需的功能[16]；电站操作程序的验证，验证的性质包括操作程序的动作前后关系和操作是否安全可靠[17]。

3. 主要研究内容及关键技术

3.1. 有限状态模型检测

模态/时序逻辑

模型检测中最常用的逻辑有计算树逻辑（CTL－Computation Tree Logic）[1]、命题线性时序逻辑（PLTL－Propositional Linear Temporal Logic）[2]和命题 mu-演算[18][19]。其中表示能力最强的是命题 mu-演算。为了实际应用的需要，出现了这些逻辑的各种（如实时、概率）扩展。以下分别介绍 CTL、PLTL 及命题 mu-演算。

计算树逻辑

一个系统的运行可以看成是系统状态的变化。系统状态变化的可能性可以表示成树状结构。比如说一个并发系统从一个初始状态开始运行，由于行为的不确定性，它可以有多个可能的后续状态，每个这样的状态又可以有多个可能的后续状态，依此类推可以产生一棵状态树。

计算树逻辑 CTL 是一种分叉时序逻辑。CTL 可以描述状态的前后关系和分枝情况。描述一个状态的基本元素是原子命题符号。CTL 公式由原子命题，逻辑连接符和模态算子组成。CTL 的逻辑连接符包括：not（非）， or（或），and（与）。它的模态算子包括：E（Exists a path），A（All paths），X（Next-time），U（Until），F（Future），G（Global）。 E 表示对于某一个分枝，A 表示对于所有分枝， X 表示下一状态，U 表示直至某一状态， F 表示现在或以后某一状态，G 表示现在和以后所有状态。前两个算子描述分枝情况，后四个算子描述状态的前后关系。CTL 中描述分枝情况和描述状态的前后关系的算子成对出现，即一个描述分枝情况的算子后面必须有一个描述状态的前后关系的算子。

CTL 公式的产生规则如下：原子命题是 CTL 公式；如果 p，q 是 CTL 公式，则 not p，p or q，p and q，EX p，E(pU q)，EF p，EG p，AX p，A(p U q)，AF p，AG p 是 CTL 公式。例如公式 EF p 表示：一定会沿某条路径达到一个满足 p 的状态。对 CTL 公式存在线性时间的模型检测算法，即算法的最坏时间复杂度与|S||F|成正比，这里|S|是状态迁移系统的大小，|F|是逻辑公式的长度[1]。

命题线性时序逻辑

系统状态变化的可能性既可以看成是一种树状结构，又可以看成是所有可能的系统初始状态经历各种可能变化的集合，也就是把一颗树看成是有限或无限条路径的集合。一条路径所代表的是系统的一次可能的运行情况。

命题线性时序逻辑关心的是系统的任意一次运行中的状态以及它们之间的关系。PLTL 公式由原子命题，逻辑连接符和模态算子组成。PLTL 的逻辑连接符包括：not，or，and。它的模态算子包括：U(Until)，◇(Eventually)，□(Always)。◇ 表示现在或以后某一状态（类似 CTL 的 F）， □表示现在和以后所有状态（类似 CTL 的 G）。

PLTL 公式的产生规则如下：原子命题是 PLTL 公式；如果 p，q 是 PLTL 公式，则 not p，p or q，p and q，$\Diamond p$，$\Box p$ 是 PLTL 公式。例如公式$\Diamond\Box p$表示：某个时刻后所有的状态都满足 p。

PLTL 和 CTL 的表达能力不同，各有所长。PLTL 模型检测的常用方法是将所要检测的性质即 PLTL 公式的补转换成 Buchi 自动机，然后求其与表示系统的自动机的交[89][90]。如果交为空，则说明系统满足所要检测的性质；否则生成一个反例，说明不满足的原因[2]。

命题 mu-演算

以上介绍的两种逻辑语言关心的是系统的状态以及它们之间的关系。现在我们介绍一种面向动作的逻辑语言，即 mu-演算。

系统状态的改变总是由某种动作引起的。mu-演算关心的是系统的动作与状态之间的关系。描述动作的基本元素是动作符号。mu-演算公式由原子命题，命题变量，逻辑连接符，模态算子和不动点算子组成。逻辑连接符包括：not，or，and，对于每个动作符号 a，有两个模态算子--[a]和<a>。[a]表示对所有的 a 动作，<a>表示对某个 a 动作。不动点算子有最小不动点算子 μ 和最大不动点算子 ν。

mu-演算公式的产生规则如下：原子命题和命题变量是 mu-演算公式；如果 p，q 是 mu-演算公式，则 not p，p or q，p and q，[a]p，<a>p，μX.p，νX.p 是 mu-演算公式，例如公式 μX.(p or [a]X)表示：在任何无穷的 a-路径上一定会有某个满足 p 的状态。

mu-演算的主要缺点是公式不易读懂，其优点是它的表示能力非常强。其表示能力主要来自于最大与最小不动点的任意交错嵌套。Bradfield 证明了 mu-演算的交错层次是严格的，即对任意的交错深度 d，总存在交错深度为 d+1 的公式与任何交错深度不超过 d 的公式都不等价[54]。CTL 和 PLTL 都可以嵌入到它的真子集中，并且相应的子集具有与 CTL 和 PLTL 相同的复杂度的模型检测算法。因此很多学者将 mu-演算作为模型检测的一般框架加以研究。

模型检测算法及其时空效率的改进

模型检测算法可分为两类：全局算法与局部算法。全局算法通过遍历系统的所有状态空间来计算每一个状态所满足的公式；局部算法则从所给定的公式出发，只搜索为满足该公式所需要查看的那部分状态，因此往往可以提高空间效率。对于 CTL，存在关于系统的状态数和公式长度成线性的检测算法；LTL 模型检测算法的复杂性是多项式空间完备的，目前最好的算法时间复杂度对系统状态数呈线性而对公式长度呈指数；对于 mu-演算，目前已知的最好的检测算法时间复杂度关于公式的交错嵌套深度成指数增长；mu-演算是否存在多项式算法的问题仍然有待解决[53]。接收无穷对象的自动机是研究模型检测的判定问题和复杂性的有力工具。近年来，一些学者将模型检测的过程看成双人博弈，取得了有意义的结果[55]。

模型检测基于对系统状态空间的穷举搜索。对于并发系统，其状态的数目往往随并发分量的增加呈指数增长。因此当一个系统的并发分量较多时，直接对其

状态空间进行搜索在实际上是不可行的。这就是所谓的状态爆炸问题。

为了能够有效地应用模型检测方法，需要研究减少和压缩状态空间的方法。文献上关于这方面的文章很多，其中的很多方法已经实现在不同的模型检测工具中。以下介绍几种常用方法。

符号模型检测的基本原理是将系统的状态转换关系用逻辑公式表示[18]。在这方法中，二叉图（Binary Decision Diagram）是用以表示逻辑公式的重要手段[21]，它能较为紧凑地表示状态转换关系，以降低系统模型所需的内存空间。另外，状态转换的计算可以以集合为单位，以提高搜索的效率。符号模型检测为克服“状态爆炸”问题迈出了关键的一步。

On-the-fly 模型检测的基本原理是根据需要展开系统路径所包含的状态，避免预先生成系统的所有状态[22]。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，我们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。这种方法称为偏序归约[23][24]。

由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。基于对这一问题的认识，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。这种方法称为对称模型检测[25][27]。

一般来讲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，有些方法的目标是以时间换空间。例如在 SPIN[22]中实现的压缩方法和用自动机表示可达状态的方法[28]。

除了在模型检测的过程应用不同方法以增加效率和减少内存空间的需求外，还有许多研究的目标是减少模型本身或验证性质的复杂性。这方面的方法有:不同类型的抽象[29][30][31]，程序切片[32][33][34]，模型分解[35][36]，及验证性质的分解[37]等。抽象的基本想法是抽掉系统中的细节、用尽可能少的状态来刻划系统的运作过程。程序切片的基本想法是将程序中不影响所要验证的性质的语句去掉以减少模型复杂性。模型分解的想法是将一个模型分解成若干部分，或者分别验证，或者提供一个较好的组合方法以降低验证的复杂性。同样，一个需要验证的性质也有可能分解成若干部分，然后分别验证。

克服状态爆炸的另一条途径是限界模型检测（bounded model checking）[57][58]。软件系统状态规模通常极其巨大，应用传统的模型检测方法或符号模型检测方法都难以见效，而限界模型检测在这方面则有相当的优势[59][60]。限界模型检测方法的主要思想是检查一个系统在给定的界内是否满足时序逻辑性质。即，如果要检查 $M|=\phi$，我们构造一系列 M 的近似模型 Mk 并验证 $Mk|=\phi$ 是否成立。所构造的近似模型应具有这样的性质：$Mk|=\phi$ 蕴涵 $M|=\phi$。如果能够在较小的 k 时得到验证，则该方法是十分有效的。对于有些用其他方法均由于状态爆炸而无法处理的问题，限界模型检测方法能够有效验证。目前这方面的研究十分活跃[61][62][63][64][65][66]。

支撑工具的研制

模型检测的优点在于可以完全自动地进行验证，这一方法的成功在很大程度上应归功于有效的软件工具的支持。以下介绍几个验证不同类型逻辑公式的软件工具。

SMV[56]是美国卡耐基梅隆大学开发的模型检测工具。用以检测一个有限状态系统是否满足 CTL 公式。其系统描述语言为 CSML，是一种基于状态转换关系的描述并发状态转换的语言。SMV 的典型应用领域包括电子电路的验证。它的建模方式是以模块为单位，模块可以同步或异步(interleaving)组合。模块描述的基本要素包括非确定性选择，状态转换和并行赋值语句。其模型检测的基本方法是以二叉图表示状态转换关系，以计算不动点的方法检测状态的可达性和其所满足的性质。其后，意大利 FBK-IRST 和卡耐基梅隆大学合作开发了 NuSMV[26]，在原有的符号模型检测的基本原理上进行了多方面的优化和扩展，这些扩展包括限界模型检测等功能。

SPIN[22][38]是美国贝尔实验室开发的模型检测工具。用以检测一个有限状态系统是否满足 PLTL 公式及其它一些性质，包括可达性和循环。其系统描述语言为 Promela，其语法基于进程描述，有类似 C 语言的结构。SPIN 的典型应用领域包括协议验证。它的建模方式是以进程为单位，进程异步组合。进程描述的基本要素包括赋值语句，条件语句，通讯语句，非确定性选择和循环语句。其模型检测的基本方法是以自动机表示各进程和 PLTL 公式，以计算这些自动机的组合可接受的语言是否为空的方法检测进程模型是否满足给定的性质。

CWB[39]的不同版本是英国爱丁堡大学和美国北卡洛莱纳州立大学相继开发的模型检测工具。用以检测系统间的等价关系，PRE-ORDER 关系，以及系统是否满足 mu-演算公式。其系统描述语言为进程代数类型的语言，包括 CCS 和 LOTOS。这些工具对电子电路、协议、工作流程和程序的验证起到了很好的作用[3][4][14][17][108]。由于建模语言的局限性和模型检测方法复杂度较高，这个工具的可应用性比前面提到的工具要差一些。

有些模型检测工具直接面向编程语言，比如贝尔实验室开发的 VeriSoft[40]等。模型检测工具很多，其余如牛津大学的 FDR[71] ，斯坦福大学的 Murphi[72]等，不一一列举。另外，需要一提的是模型检测工具与大型软件开发工具的结合。比如， IBM 公司的软件开发工具(Rational Tau)、加州大学的 BLAST[73]，微软的 SLAM[74]和 Terminator[75]，卡内基梅隆大学的 SATABS[76]，以及 NASA 的 Java PathFinder[77]等。这些工具在计算机硬件和协议的验证和检错上取得了很大的成功[78][79]，在验证一些简单的软件性质上也取得了较好的效果[77][80]。

3.2. 带参并发系统的模型检测

非有穷状态系统的典型代表是所谓带参并发系统。带参并发系统实际包含一族并发系统实例，其中以一个（或多个）参数 N 表示每个系统实例的规模。给定 N 的一个具体值，就得到一个具体的并发系统，称为该带参系统的一个实例。带参系统的难点在于：系统的行为应对参数的任意取值都是正确性的。由于参数的

取值可以任意多，因此不可能进行穷尽的测试或验证。

带参并发系统的模型检测问题在最一般的情况下是不可判定的[67]，但这并不妨碍人们对某些特定类别且具有重要实际意义的问题取得积极的成果。到目前为止，国内外学者所提出的带参模型检测方法大体上可分为两类：一是针对某些特定的带参系统得到判定算法，例如基于对称归约（symmetry reduction）的方法；二是针对某些重要的时序性质提出可靠但不完备的方法，例如基于抽象（abstraction）的模型检测技术。第一种方法中最有代表性的是 Emerson 等人提出的 cutoff 方法[68][69][70]。第二种方法中最重要的工作有：基于抽象的模型检测技术（Abstraction-based Model Checking），组合模型检测技术（Compositional Model Checking），以及基于归纳不变量检测（Inductive Invariant Checking）的技术。由于抽象过程往往会损失一些有关系统活性（liveness）的信息，因此抽象的方法通常适用于安全性质（safety property）的验证。

3.3. 实时系统的模型检测

作为模型检测基础的传统模态/时序逻辑可以方便地表示离散事件或状态是否发生以及发生的先后顺序，而在许多实际应用中人们不仅关心某一事件是否发生，而且还要求它在确定的时间间隔内发生，例如要求锅炉的水位一旦落到警戒线时进水阀门必须在 3 秒钟内开启。这类对相关事件发生的时间有明确要求的系统称为实时系统。实时系统涉及飞机控制系统、机器人、工业与军事控制系统等安全性至关紧要的应用领域。

上世纪九十年代以来，实时系统模型检测的研究取得了重大进展，主要表现在三个方面：

1)出现了用于表示实时系统的各种数学模型，如时间 Petri 网[41][42]，时间自动机[43][44]，以及各种进程代数的时间扩充[45][46]。其中以时间自动机的影响和应用最为广泛。

2)提出能描述实时系统的模态/时序逻辑，如 CTL，PLTL 和 mu-演算的实时扩充[47][48][49]。例如实时 PLTL 公式$\Box\Diamond_{\leqslant 3} p$ 表示“在任何时候系统总会在 3 个单位时间内达到满足性质 p 的状态”。

3)针对这些实时系统的数学模型和逻辑设计了各种模型检测的算法，并实现了相应的分析与验证工具，如 Uppall[50]、Kronos[51]和 HyTech[52]等。

4. 研究进展

以下重点对国内在 mu-演算性质的模型检测、组合模型检测、限界模型检测、时间自动机模型检测以及带参并发系统模型检测等方面的理论研究及相关工具研制的进展，以及模型检测应用方面的研究进展作简单介绍。此外模型检测还有一些重要的研究方向如软件模型检测和概率模型检测等，限于篇幅，这些领域的进展不再一一阐述。

4.1.mu-演算的模型检测

为了对传值并发系统进行模型检测，中科院软件研究所的林惠民等人提出了一阶 mu-演算，设计了在符号迁移图上对有穷数据域上的输入变量动态实例化的模型检测算法，并将其推广到“数据无关（data independent）”的无穷域[91][92]。基于这个算法实现了一个面向传值并发系统的模型检测工具,并利用这个工具开展了一系列应用研究。他们还提出了基于谓词的面向移动进程的 mu-演算，并采用对名字空间上的受囿变量动态实例化的策略，首次得到了有穷控制移动进程模型检测的判定性结果[93]。

命题 mu-演算的表示能力强于 CTL 和 LTL。这个较强的表示能力主要来源于最大与最小不动点的任意交错嵌套。与 mu-演算模型检测等价的一个问题是奇偶博弈的求解。迄今为止所有已经公开发表的 mu-演算模型检测和奇偶博弈的求解算法的复杂度都与公式或博弈图的嵌套深数成指数增长。Mu-演算是否存在多项式时间的模型检测算法是 30 年来理论计算机科学中一个尚未解决的难题。中科院软件研究所的林惠民等提出的奇偶博弈取胜位置集可以分层的理论结果，为解决 mu-演算模型检测是否存在多项式时间算法的问题提供了一条新的途径[94]。

国防科学技术大学的刘万伟等提出了一种基于 Game 理论的 mu-演算公式的可满足性的测试方法，该种方法能够将模态 mu-演算公式的可满足性问题转化为 Focus Game 的求解问题.进一步，基于这套 Game 规则，给出了一个新的关于 mu-演算可靠完备的推理系统[95]。

韶关学院的江华等分析了用 Tarski 不动点定理计算不动点交替嵌套深度为 4 的命题 mu-演算公式的计算过程，找到了计算中间结果间具有的两组偏序关系，利用这两组偏序关系设计了一个高效的命题 mu-演算全局模型检测算法。该算法与 Long 等人提出的算法有相似的时间复杂度，但空间复杂度有很大的改进(O(dn)相对于 O(n\+{[d/2]+1}))，其中 n 是变迁系统的状态规模，d 是命题 mu-演算公式中不动点算子的嵌套深度[96]。

4.2.限界模型检测

限界模型检测的研究始于 1999 年图灵奖获得者 Clarke 与合作者提出的基于存在路径型 LTL 公式的限界语义及其在模型检测中应用的工作。之后国内外有许多相关工作。这些工作主要是用同样的思路，发展以限界语义为基础进行较快查错的方法。

中科院软件研究所的张文辉等人近年来开展用限界模型进行验证的工作，取得了一系列进展：

1) 研究了 LTL 和 ACTL 的限界验证问题，提出了一种限界验证 LTL 公式的不完全的方法[86]。利用类似的基本原理，提出了一种建立在 ECTL 公式的限界语义基础上的限界验证 ACTL 公式的方法[87]。在上述基础上，针对这种不完全的问题，进一步研究了 ACTL 公式的限界验证问题。提出了一种建立在 ACTL 公式的限界语义，并给出了一个 ACTL 公式完备的

有界验证 ACTL 公式的方法[66]，比较了基于命题逻辑公式可满足性判定的查错和验证与基于 BDD 的模型检测方法的效率和优缺点[88]。

2) 在[66]的基础上，完成了 CTL 公式全集的限界语义定义[98]。该工作的优点在于能够正确处理限界模型中隐藏的原模型的部分信息，在公式的限界语义定义中恰当反映公式对限界路径的要求。这样的一种定义有别于之前的相关工作，使得该限界语义既适合于验证一个性质是否满足又适合于验证其对偶性质是否满足。同时证明了在一类限制下，不可能定义可靠和完备的 CTL*(LTL 和 CTL 的一种超集)的限界语义，反映了定义全称路径型限界语义的困难、说明了已有思路在定义合适的 CTL*和全称路径型 LTL 限界语义方面的不足。

3) 在上述工作的基础上，研制了基于 SAT、验证 ACTL 性质的限界模型检测工具 VERBS（http://lcs.ios.ac.cn/~zwh/verbs/）。VERBS 的基本策略是结合 CTL 限界语义中 ACTL 和 ECTL 的语义描述的命题逻辑公式刻画，用命题逻辑公式可满足性求解的方法或工具对所给定的 ACTL 性质进行限界验证或查错。

华南师范大学的杨晋吉等人对验证 G(p)和 G(p→F(q))编码转换公式进行优化。通过分析当验证这些模态算子时有限状态机的状态转移和线性时序逻辑的语义特征，在现有的编码公式的基础上，给出了简洁、高效的递推公式，该公式有利于高效编码成 SAT 实例[99]。

清华大学的骆翔宇等提出了在同步的多智体系统中验证时态认知逻辑的有界模型检测算法。基于同步解释系统语义,在时态逻辑 CTL*的语言中引入认知模态词,从而得到一个新的时态认知逻辑 ECKLn.通过引入状态位置函数的方法获得同步系统的智能体知识,避免了为时间域而扩展通常的时态认知模型的状态及迁移关系编码。ECKLn 的时态认知表达能力强于另一个逻辑 CTLK[100]。

江苏大学的周从华等对线性时态逻辑 SE-LTL 提出了一种基于 SAT 的有界模型检测过程,该过程避免了基于 BDD 方法中状态空间快速增长的问题；进一步提出了一种组合了基于 SAT 的有界模型检测、基于反例的抽象求精、组合推理 3 种状态空间约简技术的并发软件验证策略,降低了验证时间和对内存空间的需求[102]。

国防科学技术大学的虞蕾等提出了 PSL（一种用于描述并行系统的属性规约语言）的限界模型检测方法及其算法框架。他们定义了 PSL 逻辑的有界语义,而后,将有界语义进一步简化为 SAT,分别将 PSL 性质规约公式和系统 M 的状态迁移关系转换为 SAT 命题公式,最后验证这两个 SAT 命题公式合取式的可满足性,这样就将时序逻辑 PSL 的模型检测转化为一个命题公式的可满足性问题[101]。

4.3.组合式模型检测

在 LTL 模型检测技术的研究方面，中科院软件研究所的张文辉等人对一类模型给出了缓解状态爆炸的方法，即通过分析模型、使用 LTL 公式刻画不同模式的运行、将验证问题分解成子问题以降低模型检测空间需求的、基于系统运行路径集合划分的 case-based 组合式验证方法[82]。此后，他们又将该方法扩展到更

广的应用层面，并研究了通过重复使用该方法更好地降低模型检测复杂性的策略[83][84][85]，取得如下进展：

1) 对 case-based 方法做了改进，更好地对模型检测的搜索空间进行划分。新的方法不需要考虑系统运行时相关变量值的改变次数，它适用于更广的条件，从而使 case-based 方法适用于检测更多的系统。进一步考虑了如何结合静态分析来引入新变量，使得系统的 case-basis 不依赖于系统本身的合适的变量。使得 case-based 方法有了进一步的拓广。通过分析还可引入更多的变量来更细地划分已划分的搜索空间，达到进一步降低模型检测空间复杂度的目的。

2) 考虑了如何将 case-based 方法运用到抽象模型检测中。探讨如何在抽象状态空间中进行有效划分。建立了能划分抽象空间的条件，给出了如何结合静态分析来逐步划分抽象状态空间的方法。在结合静态分析方法来不断引入新变量以逐步划分抽象状态空间的基础上，进一步探讨了采用 multiple case-bases 方法进行更好的抽象搜索空间划分。

4.4. 时间自动机的模型检测

基于 zone 的符号化方法是现今最有效的关于时间自动机的模型检测技术，其核心是基于 zone 的符号化抽象，主要有最大界抽象和最大上下界抽象（简称 LU-抽象），其中 LU-抽象是迄今在时间自动机的模型检测工具中实现的最有效的 zone 抽象技术。2004 年，LU-抽象被证明保持时间自动机的可达性，并在模型检测工具 Uppaal 中实现，用于验证时间自动机的可达性。 但是 LU-抽象是否保持时间自动机的活性却一直是一个未解决的问题。

中科院软件研究所的李广元等人研究了这一问题，并给出了肯定的回答，从而使 LU-抽象这一有效抽象技术可以用于实时系统的活性验证。此外他们基于 LU-抽象实现了时间自动机关于线性时序逻辑 LTL 的符号化模型检测工具 CTAV[103]（http://lcs.ios.ac.cn/~ligy/tools/CTAV_LTL/）。这是国际上实现的第一个可对时间自动机的线性时序逻辑性质进行自动验证的符号化模型检测工具。

加权时间自动机（Weighted Timed Automata）和加权度量时序逻辑 WMTL（Weighted Metric Temporal Logic）分别是时间自动机和度量时序逻辑 MTL 关于加权量的扩展，用于表示和分析实时嵌入式系统关于能耗、资源等有关性能方面的性质。中科院软件所李广元与丹麦 Aalborg 大学 Kim Larsen 教授等人合作，实现了从加权度量时序逻辑 WMTL 的一个重要子集 WMTL≤到加权时间自动机的转换工具 Casaal（http://lcs.ios.ac.cn/~ligy/tools/CASAAL/），并通过 Casaal 与另一验证工具 Uppaal-SMC 的连接，实现了加权时间自动机关于加权度量时序逻辑 WMTL≤的统计模型检测。

南京航空航天大学张君华等研究了模型检测基于概率时间自动机（PTA，Markov 链的不确定性和系统时钟的扩展）的反例产生问题。通过在 PTA 上寻找概率之和恰好大于 λ 的 k 条最大概率的路径，并根据这些路径和原 PTA 构造原 PTA 的一个子图，从而快速找到违背性质的具有较少证据的反例。然后精化此结

果——通过逐条加入上述各条最大概率的路径来精确地计算已加入路径所构成的 PTA 子图的最大概率。由于考虑到符号状态交集对概率系统的影响，可以得到证据更少的反例[104]。

4.5. 带参并发系统的模型检测

带参并发系统实际包含一族并发系统实例，其中以一个（或多个）参数表示每个系统实例的规模。由于在实际应用中的重要性，带参并发系统是模型检测研究中的一个热点问题。带参模型检测方法大体上可分为两类：一是针对某些重要的时序性质提出可靠但不完备的方法；二是针对某些特定子类的带参系统得到判定算法。

第一类方法中最有代表性的是基于卫士增强的参数抽象方法，Intel 公司使用该方法完成了迄今为止复杂度最高的的众核处理器 Cache 一致性协议的形式验证，是目前最高效的带参模型检测方法。该方法最有挑战性的部分在于发现合适的辅助不变式，中科院软件研究所的吕毅等人通过计算参考模型的可达状态空间获得所需的辅助不变式，完成了该方法的自动化，在此基础上实现了一个基于启发信息的全自动化验证工具 PaTLV（http://lcs.ios.ac.cn/~lvyi/PaTLV/）。PaTLV 以开源软件 TLV 为基础，能够自动进行辅助不变式的计算和卫士增强，全自动化地进行带参模型检测，包括自动进行系统抽象、自动计算辅助不变式量、自动进行卫士增强等过程。他们还对若干典型的缓存一致性协议开展实例研究，在此过程中，发现了著名的 GERMAN2004 协议存在数据不一致错误并加以改正，第一次形式验证了 GERMAN2004 协议的正确性[105]；提出了把“数据无关”技术应用于缓存一致性协议模型检测的方法，证明了改正后的 GERMAN2004 协议在任意大小的数据域上都是正确的[107]。在此基础上，他们结合 Clarke 等提出的环境抽象方法，利用状态聚类和参数截断技术，以很小的空间开销完成了包括 FLASH，GERMAN 协议的验证。这是 FLASH 这类工业级的复杂协议第一次同时在控制部分和数据部分性质均得到高度自动化的验证[106]。

第二类方法研究针对给定的模型和性质确定参数的上界，将无穷状态验证的问题转化为有界模型检测问题。现有研究工作主要是根据带参并发系统的拓扑结构和性质，确定进程或者线程实例数目的上界；而基于搜索路径上界的模型检测算法主要用于非带参系统的验证，且算法本身是不完备的。基于抽象和逐步求精，中科院软件研究所的杨秋松等提出了基于搜索路径长度上界的完备的带参并发系统前向分析模型检测算法。在上述研究的基础上，区别于传统的向上封闭集合的表示方法，进一步提出基于有界反向可达图的反向逐步求精模型检测算法；并通过整合 Omega 路径和有界可达图，提出新的符号化表示方法[108]。

4.6. 模型检测应用

随着模型检测技术及工具的逐渐成熟，可用性不断提高，使得模型检测的应用范围越来越广，相关应用研究不断涌现。以下就国内学者将模型检测应用于硬件设计、系统安全、系统性能分析、实时调度、Web 服务、构件技术等方面的研究进展作简单介绍。

在将模型检测应用于硬件设计方面，中科院计算技术研究所的陈云霁等将模型检测应用于龙芯 2 号微处理器浮点除法功能部件的形式验证上[112]；赵阳等将模型检测应用于验证的复杂电路属性[113]；北京大学的张良等将模型检测应用于北大众志 UniCore32 定点处理器核的功能验证上[114]。

在将模型检测应用于系统安全方面，中国科学技术大学的程亮等人结合安全操作系统对测试的特殊需求，提出了简并测试集的概念，设计了一种使用模型检测的基于安全状态转移的高效测试集生成方法[115]；北京航空航天大学的王雷等人提出了一种利用模型检测技术对代码中潜在的缓冲区溢出漏洞进行精确检测的方法，并对 6 个开源软件进行检测，发现了一些新漏洞[116][117]。

在将模型检测应用于系统性能分析方面，同济大学的钮俊等人对具有不确定性的复杂系统如网络协议等的性能进行分析，将空间资源分析纳入到性能评估过程，用模型检测技术验证时间或空间性能是否满足期望的需求约束[118]。

在将模型检测应用于实时调度方面，西安电子科技大学的田聪等用 SPIN 对一个任务组在单调速率调度下是否可调度进行分析[119]；电子科技大学的桂盛霖等应用模型检测技术，研制了基于自动机理论的分布式实时调度分析工具[120]。

在将模型检测应用于 Web 服务方面，西安电子科技大学的门鹏等针对 Web 服务组合的特点，对着色 Petri 网模型检测工具进行扩展[121]；清华大学的骆翔宇等提出并实现了一种 Web 服务组合的认知模型检测方法，可以有效验证 Web 服务组合的时序逻辑规范[122]；苏州大学的张广泉等基于软件体系结构描述语言 XYZ/ADL 和精化检验/模型检测方法，提出了一种 Web 服务组合的描述与验证方法[123]。

在将模型检测应用于构件技术方面[124]，上海大学的曾红卫等人针对构件组合的状态爆炸问题，提出了组合式的抽象精化方法，使构件组合的模型检测转化为各成分构件的局部抽象精化，降低了分析的复杂度[125]；南京大学的胡军等人对系统设计阶段实时软件构件的动态行为进行形式化分析与检验[126]。

其它方面的应用研究还有很多，例如将模型检测应用于物联网服务验证[127]、面向服务的企业应用集成系统验证[128]以及 UML 的验证[129][130]，等等，在此不一一列举。

5. 结束语

模型检测方法由于可自动化程度高而引起学术界与工业界越来越多的关注。本文介绍了模型检测的典型应用、主要研究内容及关键技术、以及部分国内研究人员的最新工作进展。还有许多学者在这方面做了出色的工作[131][132][133][134][135][136][137]，由于时间、篇幅所限，本文未能提及国内所有研究进展，疏漏之处，还望相关同行见谅。

参考文献

[1]. E. M. Clarke, E. A. Emerson, A. P. Sistla. Automatic Verification of Finite state Concurrent Systems Using Temporal Logic Specifications. 10th Annual ACM Symposium on Principles of Programming Languages (POPL 83). New York: ACM Press, 1983. 117-126.

[2]. M. Y. Vardi, P. Wolper. An automata-theoretic approach to automatic program verification. 1st IEEE Symposium on Logic in Computer Science (LICS 86). Los Alamitos: IEEE Computer Society, 1986. 322-331.

[3]. Jorg Bormann, Jorg Lohse, Michael Payer, Gerd Venzl. Model Checking in Industrial Hardware Design. 32nd Conference on Design Automation (DAC 95). New York: ACM Press, 1995. 298-303.

[4]. Yirng-An Chen, Edmund M. Clarke, Pei-Hsin Ho, Yatin Vasant Hoskote, Timothy Kam, Manpreet Khaira, John W. O'Leary, Xudong Zhao. Verification of All Circuits in a Floating-Point Unit Using Word-Level Model Checking. Lecture Notes in Computer Science 1166 - 1st International Conference on Formal Methods in Computer-Aided Design (FMCAD 96). Berlin: Springer-Verlag, 1996. 19-33.

[5]. Gavin Lowe. Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR Software - Concepts and Tools, 1996, 17(3): 93-102.

[6]. J. C. Mitchell, V. Shmatikov, U. Stern. Finite-State Analysis of SSL 3.0. 7th USENIX Security Symposium (USENIX 98). Berkeley: USENIX, 1998. 201-215.

[7]. Paolo Maggi, Riccardo Sisto. Using SPIN to Verify Security Properties of Cryptographic Protocols. Lecture Notes in Computer Science 2318 - 9th International SPIN Workshop on Model Checking of Software (SPIN 02). Berlin: Springer-Verlag, 2002. 187-204.

[8]. Vitaly Shmatikov, John C. Mitchell. Finite-state analysis of two contract signing protocols. Theoretical Computer Science, 2002, 283(2): 419-450.

[9]. Edmund M. Clarke, Orna Grumberg, Hiromi Hiraishi, Somesh Jha, David E. Long, Kenneth L. McMillan, Linda A. Ness. Verification of the Futurebus+ Cache Coherence Protocol. Formal Methods in System Design, 1995, 6(2): 217-232.

[10]. Marc Geilen, Twan Basten, and Sander Stuijk. 2005. Minimising buffer requirements of synchronous dataflow graphs with model checking. In Proceedings of the 42nd annual Design Automation Conference (DAC '05). ACM, New York, NY, USA, 819-824. DOI=10.1145/1065579.1065796

[11]. R. J. Anderson, P. Beame, S. Burns, W. Chan, F. Modugno, D. Notkin, J. D. Reese. Model Checking Large Software Specifications. University of Washington, Department of Computer Science and Engineering. Report: TR-96-04-02, April, 1996.

[12]. T. Sreemani, J. M. Atlee. Feasibility of Model Checking Software Requirements. 11th Annual Conference on Computer Assurance (COMPASS 96). Gaithersburg, Maryland: National Institute of Standards and Technology, 1996, 77-.

[13]. E. Cleaveland, G. Luttegen, V. Natarajan, S. Sims. Modeling and verifying distributed systems using priorities: A case study。 Software Concepts and Tools, 1996, 17(2): 50-62.

[14]. A. Cimatti, F. Giunchiglia, G. Mongardi, D. Romano, F. Torielli, P. Traverso. Model Checking Safety Critical Software with SPIN: An Application to a Railway Interlocking System. Lecture Notes in Computer Science 1516 - 17th International Conference on Computer Safety, Reliability and Security (SAFECOMP 98). Berlin: Springer-Verlag, 1998. 284-295.

[15]. Adam L. Turk, Scott T. Probst, Gary J. Powers. Verification of Real Time Chemical Processing Systems. Lecture Notes in Computer Science 1201 - International Workshop on Hybrid and Real-Time Systems (HART 97) . Berlin: Springer-Verlag, 1997. 259-272.

[16]. Henk Eertink, Wil Janssen, Paul Oude Luttighuis, Wouter B. Teeuw, Chris A. Vissers. A Business Process Design Language. Lecture Notes in Computer Science 1708 - World Congress on Formal Methods (FM 99). Berlin: Springer-Verlag, 1999. 76-95.

[17]. Wenhui Zhang. Model Checking Operator Procedures. Lecture Notes in Computer Science 1680 - Theoretical and Practical Aspects of SPIN Model Checking (SPIN 99a, 99b). Berlin: Springer-Verlag, 1999. 200-215.

[18]. E. Allen Emerson, Chin-Laung Lei. Efficient Model Checking in Fragments of the Propositional Mu-Calculus. 1st IEEE Symposium on Logic in Computer Science (LICS 86). Los Alamitos: IEEE Computer Society, 1986. 267-278.

[19]. Colin Stirling, David Walker. Local Model Checking in the Modal Mu-Calculus. Lecture Notes in Computer Science 351 - 3rd International Joint Conference on Theory and Practice of Software Development (TAPSOFT 89). Berlin: Springer-Verlag, 1989. 369-383.

[20]. J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, J. Hwang. Symbolic model checking: 10^{20} states and beyond. 5th IEEE Symposium on Logic in Computer Science (LICS 90). Los Alamitos: IEEE Computer Society, 1990, 428-439.

[21]. Randal E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. IEEE Transactions on Computers, 1986, 35(8): 677-691.

[22]. G. J. Holzmann. Design and Validation of Computer Protocols. New Jersey: Prentice Hall, 1991.

[23]. S. Katz, D. Peled. Verification of distributed programs using representative interleaving sequences. Distributed Computing, 1992, 6: 107-120.

[24]. Gerard J. Holzmann, Doron Peled. An improvement in formal verification. 7th IFIP WG6.1 International Conference on Formal Description Techniques VII (FORTE 1994). London: Chapman and Hall, 1995. 197-211.

[25]. E. A. Emerson, A. P. Sistla. Symmetry and Model Checking. Lecture Notes in Computer Science 697 - 5th International Conference on Computer Aided Verification (CAV 93). Berlin: Springer-Verlag, 1993. 463-478.

[26]. A. Cimatti, E. M. Clarke, F. Giunchiglia, M. Roveri. NuSMV: A New Symbolic Model Verifier. CAV 1999: 495-499.

[27]. A. Prasad Sistla, Viktor Gyuris, E. Allen Emerson. SMC: a symmetry-based model checker for verification of safety and liveness properties. ACM Transactions on Software Engineering and Methodology, 2000, 9(2): 133-166.

[28]. Gerard J. Holzmann, Anuj Puri. A Minimized Automaton Representation of Reachable States. International Journal on Software Tools for Technology Transfer, 1999, 2(3): 270-278.

[29]. Patrick Cousot, Radhia Cousot. Systematic Design of Program Analysis Frameworks. 6th Annual ACM Symposium on Principles of Programming Languages (POPL 79). New York: ACM Press, 1979. 269-282.

[30]. \bibitem{ClaGL94} E. M. Clarke, O. Grumberg, D. E. Long. Model Checking and Abstraction. ACM Transactions on Programming Languages and Systems, 1994, 16(5): 1512-1542.

[31]. \bibitem{LoiGS+95}C. Loiseaux, S. Graf, J. Sifakis, A. Bouajjani, S. Bensalem. Property preserving abstractions for the verification of concurrent systems. Journal of Formal methods in System Design, 1995, 6: 1-35.

[32]. Mark Weiser. Program Slicing. IEEE Transactions on Software Engineering, 1984, 10(4): 352-357.

[33]. Jingde Cheng. Slicing Concurrent Programs - A Graph-Theoretical Approach. Lecture Notes in Computer Science 749 - 1st International Workshop on Automated and Algorithmic Debugging (AADEBUG 93). Berlin: Springer-Verlag, 1993. 223-240.

[34]. L. I. Millett, T. Teitelbaum. Issues in Slicing PROMELA and Its Applications to Model Checking, Protocol Understanding, and Simulation. International Journal on Software Tools for Technology Transfer, 2000, 2(4): 343-349.

[35]. Eugene W. Stark. A Proof Technique for Rely/Guarantee Properties. Lecture Notes in Computer Science 206 - 5th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 85). Berlin: Springer-Verlag, 1985. 369-391.

[36]. E. M. Clarke, D. E. Long, K. L. McMillan. Compositional Model Checking. 4th IEEE Symposium on Logic in Computer Science (LICS 89). Los Alamitos: IEEE Computer Society, 1989, 353-362.

[37]. K. Stahl, K. Baukus, Y. Lakhnech, M. Steffen. Divide, abstract, and model-check. Lecture Notes in Computer Science 1680 - Theoretical and Practical Aspects of SPIN Model Checking (SPIN 99a, 99b). Berlin: Springer-Verlag, 1999. 57-76.

[38]. G. J. Holzmann. The Model Checker SPIN. IEEE Transactions on Software Engineering, 1997, 23(5): 279-295.

[39]. R. Cleaveland, J. Parrow, B. Steffen. The Concurrency Workbench: A semantics-based verification tool for the verification of concurrent systems. ACM Transactions on Programming Languages and Systems, 1993, 5(1): 36-72.

[40]. Patrice Godefroid. VeriSoft: A Tool for the Automatic Analysis of Concurrent Reactive Software. Lecture Notes in Computer Science 1254 - 9th International Conference on Computer Aided Verification (CAV 97). Berlin: Springer-Verlag, 1997. 476-479.

[41]. Bernd Walter. Timed Petri-Nets for Modelling and Alalyzing Protocols with Real-Time Characteristics. 3rd IFIP WG 6.1

International Workshop on Protocol Specification, Testing, and Verification (PSTV 83). Amsterdam: North-Holland, 1983. 149-159.

[42]. James E. Coolahan Jr., Nick Roussopoulos. A Timed Petri Net Methodolgoy for Specifying Real-Time System Timing Requirements. International Workshop on Timed Petri Nets (PNPM 85). Los Alamitos: IEEE Computer Society, 1985. 24-31.

[43]. David L. Dill. Timing Assumptions and Verification of Finite-State Concurrent Systems. Proceedings. Lecture Notes in Computer Science 407 - International Workshop on Automatic Verification Methods for Finite State Systems (89). Berlin: Springer-Verlag, 1990. 197-212.

[44]. Rajeev Alur, Costas Courcoubetis, David L. Dill. Model-Checking for Real-Time Systems. 5th IEEE Symposium on Logic in Computer Science (LICS 90). Los Alamitos: IEEE Computer Society, 1990. 414-425.

[45]. Wang Yi. CCS + Time = An Interleaving Model for Real Time Systems. Lecture Notes in Computer Science 510 - 18th International Colloquium on Automata, Languages and Programming (ICALP 91). Berlin: Springer-Verlag, 1991. 217-228.

[46]. Steve Schneider, Jim Davies, D. M. Jackson, George M. Reed,

[47]. E. Allen Emerson, Aloysius K. Mok, A. Prasad Sistla, Jai Srinivasan. Quantitative Temporal Reasoning. Lecture Notes in Computer Science 531 - 2nd International Workshop on Computer Aided Verification (CAV 90). Berlin: Springer-Verlag, 1991. 136-145.

[48]. Jonathan S. Ostroff. Composition and Refinement of Discrte Real-Time Systems. ACM Transactions on Software Engineering and Methodology, 1999, 8(1): 1-48.

[49]. Thomas A. Henzinger, Xavier Nicollin, Joseph Sifakis, Sergio Yovine. Symbolic Model Checking for Real-time Systems. 7th IEEE Symposium on Logic in Computer Science (LICS 92). Los Alamitos: IEEE Computer Society, 1992. 394-406.

[50]. Kim G. Larsen, Paul Pettersson, Wang Yi. UPPAAL in a Nutshell. International Journal on Software Tools for Technology Transfer, 1997, 1(1-2): 134-152.

[51]. Marius Bozga, Conrado Daws, Oded Maler, Alfredo Olivero, Stavros Tripakis, Sergio Yovine. KRONOS: A Model-Checking Tool for Real-Time Systems. Lecture Notes in Computer Science 1486 - 5th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT 98). Berlin: Springer-Verlag, 1998. 298-302.

[52]. Thomas A. Henzinger, Pei-Hsin Ho, Howard Wong-Toi. HYTECH: A Model Checker for Hybrid Systems. Lecture Notes in Computer Science 1254 - 5th International Conference on Computer Aided Verification (CAV 97). Berlin: Springer-Verlag, 1997. 460-463.

[53]. E.M. Clarke, O. Grumberg, and D. Peled, *Model checking*. 1999, Cambridge, Mass.: MIT Press.

[54]. J.C. Bradfield, *The modal mu-calculus alternation hierarchy is strict.* THEORETICAL COMPUTER SCIENCE -AMSTERDAM-, 1998. **195**(2): p. 133-154.

[55]. M. Jurdzinski, M. Paterson, and U. Zwick, *A deterministic subexponential algorithm for solving parity games*, in *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. 2006, ACM: Miami, Florida.

[56]. K.L. McMillan, *Symbolic model checking*. 1993, Boston: Kluwer Academic.

[57]. A. Biere, et al., *Symbolic Model Checking using SAT Procedures Instead of BDDs.* DESIGN AUTOMATION CONFERENCE, 1999(36): p. 317-320.

[58]. 16. A. Biere, et al., *Symbolic Model Checking without BDDs.* Lecture notes in computer science., 1999(1579): p. 193.

[59]. M.R. Prasad, A. Biere, and A. Gupta, *A survey of recent advances in SAT-based formal verification.* INTERNATIONAL JOURNAL ON SOFTWARE TOOLS FOR TECHNOLOGY TRANSFER, 2005. **7**(2): p. 156-173.

[60]. A. Biere, et al., *Bounded Model Checking.* ADVANCES IN COMPUTERS, 2003. **58**: p. 118-149.

[61]. E. Clarke, et al., *Computational challenges in bounded model checking.* INTERNATIONAL JOURNAL ON SOFTWARE TOOLS FOR TECHNOLOGY TRANSFER, 2005. **7**(2): p. 174-183.

[62]. M. Awedh and F. Somenzi, *Termination Criteria for Bounded Model Checking: Extensions and Comparison.* ELECTRONIC NOTES IN THEORETICAL COMPUTER SCIENCE, 2006. **144**(1): p. 51-66.

[63]. K. Heljanko, et al., *Bounded Model Checking for Weak Alternating Buchi Automata.* Lecture notes in computer science.,

2006(4144): p. 95-108.

[64]. P.B. Jackson and D. Sheridan, *A Compact Linear Translation for Bounded Model Checking.* ELECTRONIC NOTES IN THEORETICAL COMPUTER SCIENCE, 2007. **174**(3): p. 17-30.

[65]. J.A. Navarro-Perez and A. Voronkov, *Encodings of Bounded LTL Model Checking in Effectively Propositional Logic.* Lecture notes in computer science., 2007(4603): p. 346-361.

[66]. W. Zhang, *Model Checking with SAT-Based Characterization of ACTL Formulas.* Lecture notes in computer science., 2007(4789): p. 191-211.

[67]. K.R. Apt and D.C. Kozen, *Limits for automatic verification of finite-state concurrent systems.* Inf. Process. Lett., 1986. **22**(6): p. 307-309.

[68]. E. Emerson and V. Kahlon, *Reducing Model Checking of the Many to the Few*, in *Automated Deduction - CADE-17*. 2000. p. 236-254.

[69]. A. Bouajjani, P. Habermehl, and T. Vojnar, *Verification of Parametric Concurrent Systems with Prioritized FIFO Resource Management.* Lecture notes in computer science., 2003(2761): p. 174-190.

[70]. E.A. Emerson and V. Kahlon, *Exact and Efficient Verification of Parameterized Cache Coherence Protocols.* Lecture notes in computer science., 2003(2860): p. 247-262.

[71]. A. Roscoe and Z. Wu, *Verifying Statemate Statecharts Using CSP and FDR*, in *Formal Methods and Software Engineering*. 2006. p. 324-341.

[72]. U. Stern and D.L. Dill, *Parallelizing the Murphi verifier*. Lecture notes in computer science., 1997(1254): p. 256.

[73]. T. Henzinger, et al., *Software Verification with BLAST*, in *Model Checking Software*. 2003. p. 624-624.

[74]. B. Thomas and K.R. Sriram, *The SLAM project: debugging system software via static analysis*, in *Proceedings of the 29th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. 2002, ACM: Portland, Oregon.

[75]. B. Cook, A. Podelski, and A. Rybalchenko, *Terminator : Beyond Safety*, in *Computer Aided Verification*. 2006. p. 415-418.

[76]. E. Clarke, et al., *SATABS: SAT-Based Predicate Abstraction for ANSI-C*, in *Tools and Algorithms for the Construction and Analysis of Systems*. 2005. p. 570-574.

[77]. W. Visser, et al., *Model Checking Programs.* Automated Software Engineering, 2003. **10**(2): p. 203-232.

[78]. X. Chen, S.M. German, and G. Gopalakrishnan. *Transaction Based Modeling and Verification of Hardware Protocols*. in *Formal Methods in Computer Aided Design*. 2007.

[79]. H. Jain, et al., *VCEGAR: Verilog CounterExample Guided Abstraction Refinement.* Lecture notes in computer science., 2007(4424): p. 583-586.

[80]. M.B. Dwyer, et al., *Building Your Own Software Model Checker Using the Bogor Extensible Model Checking Framework.* Lecture notes in computer science., 2005(3576): p. 148-152.

[81]. 林惠民，张文辉, *模型检测：理论、方法与应用.* 电子学报, 2002. **30**(12): p. 1907-1912.

[82]. Wenhui Zhang. Combining static analysis and case-based search space partitioning for reducing peak memory in model checking. Journal of Computer Science and Technology 18(6):762-770, 2003.

[83]. Bai Su, Wenhui Zhang. Search Space Partition and Case Basis Exploration for Reducing Model Checking Complexity. Lecture Notes in Computer Science 3299 (ATVA'04):34-48. Springer-Verlag. 2004.

[84]. Fei Pu, Wenhui Zhang, Shaochun Wang. An Improved Case-Based Approach to LTL Model Checking. Lecture Notes in Computer Science 3943 (RISE'05):190-202. Springer-Verlag. 2006.

[85]. Fei Pu and Wenhui Zhang. Combining search space partition and abstraction for LTL model checking. Science in China, Series F, vol. 50(6):793-810. 2007.

[86]. Wenhui Zhang. SAT-Based Verification of LTL Formulas. Lecture Notes in Computer Science 4346 (FMICS/PDMC'06):277-292. Springer-Verlag. 2007.

[87]. Wenhui Zhang. Verification of ACTL Properties by Bounded Model Checking. Lecture Notes in Computer Science 4739 (EUROCAST'07):556-563. Springer-Verlag. 2007.

[88]. Yanyan Xu, Wei Chen, Liang Xu, Wenhui Zhang: Evaluation of SAT-based Bounded Model Checking of ACTL Properties. Proceedings of the 1st Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering (TASE'07):339-348. IEEE Computer Society Press. 2007.

[89]. Jin Yi, Wenhui Zhang. Efficient State Space Reduction for Automata by Fair Simulation. Lecture Notes in Computer Science 4767 (FSEN'07):380-387. Springer-Verlag. 2007.

[90]. Jin Yi, Wenhui Zhang. Enhancing Simulation for Checking Language Containment. Lecture Notes in Computer Science 4484 (TAMC'07):374-385. Springer-Verlag. 2007.

[91]. Huimin Lin. Model Checking Value-passing Processes. Invited talk, 8th Asia-Pacific Software Engineering Conference (APSEC'2001), Macau, December 4-7, 2001.

[92]. Huimin Lin. A Graphical mu-Calculus and Local Model Checking. Journal of Computer Science and Technology.Vol.17, No.6, 2002.

[93]. Huimin Lin. A Predicate mu-Calculus for Mobile Ambients. Journal of Computer Science and Technology, Vol. 20, No.1, pp. 85-104. 2005.

[94]. Huimin Lin: Stratifying Winning Positions in Parity Games. LNCS 5062 (APTN 2008):9-11. 2008.

[95]. 刘万伟,王戟,陈火旺. 基于 Game 理论的μ-演算公理化.计算机研究与发展.2007 年第 11 期,1896~1902

[96]. 江华. 命题μ-演算全局模型检测的高效算法设计.计算机研究与发展,2010 年,第 8 期,1424~1433 页.

[97]. Wenhui Zhang: Model Checking with SAT-Based Characterization of ACTL Formulas. LNCS 4789 (ICFEM 2007):191-211. 2007.

[98]. Wenhui Zhang: Bounded Semantics of CTL and SAT-based Verification. LNCS 5885 (ICFEM 2009):286-305. 2009.

[99]. 杨晋吉,苏开乐,骆翔宇,林瀚,肖茵茵.有界模型检测的优化.软件学报,2009,20(8):2005-2014

[100]. 骆翔宇,苏开乐,杨晋吉.有界模型检测同步多智体系统的时态认知逻辑.软件学报,2006,17(12):2485-2498

[101]. 虞蕾,赵宗涛. PSL 的有界模型检验.电子学报.2009 37 (3): 614-621

[102]. 周从华.一种基于满足性判定的并发软件验证策略.软件学报,2009,20(6):1414-1424

[103]. Guangyuan Li: Checking Timed Büchi Automata Emptiness Using LU-Abstractions. LNCS 5813 (FORMATS 2009):228-242. 2009.

[104]. 张君华, 黄志球, 曹子宁.模型检测基于概率时间自动机的反例产生研究.计算机研究与发展 2008 年第 10 期，1638~1645 页。

[105]. Yi Lv, Huimin Lin, Hong Pan: Computing Invariants for Parameter Abstraction. Proceedings of the 5th ACM/IEEE International Conference on Formal Methods and Models for Co-Design (MEMOCODE 2007):29-38. 2007.

[106]. Hong Pan, Yi Lv, Huimin Lin: Environment Abstraction with State Clustering and Parameter Truncating. Proceedings of the 3rd IEEE International Symposium on Theoretical Aspects of Software Engineering (TASE 2009):73-80. 2009.

[107]. Hong Pan, Huimin Lin & Yi Lv. Model Checking Data Consistency for Cache Coherence Protocols. JCST，Vol.21 No.5,2006, pp. 765-775.

[108]. Qiusong Yang, Mingshu Li: A cut-off approach for bounded verification of parameterized systems. Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering (ICSE 2010):345-354.

[109]. A. Valmari. A stubborn attack on state explosion. Formal Methods in System Design 1(4):297-322, December 1992.

[110]. Richard H. Carver, Kuo-Chung Tai. Use of Sequencing Constraints for Specification-Based Testing of Concurrent Programs. IEEE Transactions on Software Engineering, 1998, 24(6): 471-490.

[111]. Fraser, G., Wotawa, F. and Ammann, P. E. (2009), Testing with model checkers: a survey. Software Testing, Verification and Reliability, 19: 215–261. doi: 10.1002/stvr.402

[112]. 陈云霁,马麟,沈海华,胡伟武.龙芯 2 号微处理器浮点除法功能部件的形式验证.研究与发展,2006(10):1835-1841

[113]. 赵阳,吕涛,李华伟,李晓维.无界模型检验中融合电路信息的 SAT 算法研究.计算机学报, 2009(6):1110-1118

[114]. 张良,易江芳,佟冬,程旭,王克义.使用局部建模的微处理器测试程序自动生成方法.电子学报,2011, 20 (7): 1639-1644

[115]. 程亮,张阳,冯登国.一种基于安全状态转移的简并测试集生成方法.软件学报,2010,21(3):539-547

[116]. 王雷,李吉,李博洋.缓冲区溢出漏洞精确检测方法研究.电子学报, 2008, 36 (11): 2200-2204

[117]. 王雷,陈归,金茂忠. 基于约束分析与模型检测的代码安全漏洞检测方法研究.计算机研究与发展,2011 年第 9 期，1659-1666

[118]. 钮俊,曾国荪,王伟.基于模型检测的时间空间性能验证方法. 计算机学报,2010(9): 1621-1633.

[119]. 田聪,段振华.基于命题投影时序逻辑的单调速率调度算法模型检测.软件学报,2011,22(2):211-221

[120]. 桂盛霖,罗蕾,李允,于淼,徐建华.基于自动机理论的分布式实时调度分析工具.软件学报,2011,22(6):1236-1251

[121]. 门鹏,段振华.着色 Petri 网模型检测工具的扩展及其在 Web 服务组合中的应用.计算机研究与发展,2009 年第 8 期，1294-1303

[122]. 骆翔宇,谭征,苏开乐,吴立军.一种基于认知模型检测的 Web 服务组合验证方法.电子学报,2011(6): 1041-1061

[123]. 张广泉,戎玫,朱雪阳,何亚丽,石慧娟. 基于 XYZ/ADL 的 Web 服务组合描述与验证. 电子学报, 2011, 39 (3A): 86-93

[124]. 贾仰理,李舟军,邢建英,陈石坤.基于模型检验的构件验证技术研究进展.计算机研究与发展.2011(6):，913-922

[125]. 曾红卫,缪淮扣.构件组合的抽象精化验证.软件学报,2008,19(5):1149-1159

[126]. 胡军,于笑丰,张岩,李宣东,郑国梁.基于场景构件式实时软件设计的一致性检验.软件学报,2006,17(1):48-58

[127]. 李力行,金芝,李戈.基于时间自动机的物联网服务建模和验证.计算机学报,2011(8): 1365-1377

[128]. 张广胜,蒋昌俊,汤宪飞,徐岩.面向服务的企业应用集成系统描述与验证.软件学报,2007,18(12):3015-3030

[129]. 张琛,段振华,田聪.基于事件确定有限自动机的 UML2.0 序列图描述与验证.软件学报,2011,22(11):2625-2638

[130]. 张岩,梅宏.UML 类图中面向非功能属性的描述和检验.软件学报,2009,20(6):1457-1469

[131]. Li Li, XiaoYu Song, Ming Gu and XiangYu Luo. Competent predicate abstraction in model checking. SCIENCE CHINA Information Sciences, Volume 54, Number 2, 258-267.

[132]. 陈振宇,陶志红,KLEINE BüNING Hans,王立福.变量极小不可满足在模型检测中的应用.软件学报,2008,19(1):39-47

[133]. 文中华,黄巍,刘任任,姜云飞.模型检测规划中的状态分层方法.软件学报,2009,20(4):858-869

[134]. 黄卫平.程序重构预处理在提高软件模型检测效率中的应用.计算机研究与发展,2008(8):1417-1422

[135]. 刘万伟,王戟,王昭飞.ETL 的符号化模型检验.软件学报,2009,20(8):2015-2025

[136]. 钱俊彦,徐宝文.基于完备抽象解释的模型检验 CTL 公式研究.计算机学报,2009(5): 992-1001

[137]. 卜磊,李游,王林章,李宣东.BACH:线性混成系统有界可达性模型检验工具.软件学报,2011,22(4):640-658