基于半张量积的逻辑综合研究进展

储著飞*[①] 马钺昱[①] 闫 鸣[①] 潘家祥[①] 潘鸿洋[②] 王伦耀[①] 夏银水[①]

[①](宁波大学信息科学与工程学院 宁波 315211)

[②](复旦大学微电子学院 上海 200433)

摘 要：逻辑综合在现代电子设计自动化流程中扮演着至关重要的角色。随着计算能力的不断增强以及新的计算范式的涌现，各种高效的布尔可满足性(SAT)求解器和电路仿真器(Simulator)得以开发，并在逻辑综合的领域取得了显著的应用。该文首先对布尔可满足性问题和电路逻辑仿真器进行了简要介绍；其次回顾了矩阵半张量积的发展历程，并根据半张量积的基本原理深入阐述了其在推理引擎和逻辑综合方面的研究进展；最后，对未来可能对逻辑综合产生重大影响的新技术进行了展望。

关键词：逻辑综合；逻辑优化；推理引擎；半张量积

中图分类号：TN47 **文献标识码**：A **文章编号**：1009-5896(2025)03-0001-13

DOI: 10.11999/JEIT231457

Research Progress in Logic Synthesis Based on Semi-Tensor Product

CHU Zhufei[①] MA Chengyu[①] YAN Ming[①] PAN Jiaxiang[①] PAN Hongyang[②]
WANG Lunyao[①] XIA Yinshui[①]

[①](*Faculty of Electrical Engineering and Computer Science* (*EECS*), *Ningbo University*, *Ningbo* 315211, *China*)

[②](*School of Microelectronics*, *Fudan University*, *Shanghai* 200433, *China*)

Abstract: Logic synthesis plays a crucial role in the modern electronic design automation process. With the continuous enhancement of computational capabilities and the emergence of new computing paradigms, various efficient Boolean SATisfiability (SAT) solvers and circuit simulators have been developed and applied in the context of logic synthesis. First, the overview of the Boolean Satisfiability problem and circuit logic simulator is briefly described. Subsequently, the historical development of the matrix semi-tensor product is reviewed, and based on the fundamental principles of the semi-tensor product, its research progress in inference engines and logic synthesis is expounded. Finally, a prospective analysis is conducted on emerging technologies that may significantly impact logic synthesis in the future.

Key words: Logic synthesis; Logic optimization; Reasoning engine; Semi-Tensor Product(STP)

1 引言

逻辑综合是现代电子设计自动化(Electronic Design Automation, EDA)流程的重要组成部分，被认为是实现高端集成电路的关键步骤之一。逻辑综合是将用户设计的数字电路转换为工艺库网表的过程，输入通常是寄存器传输级硬件描述语言(Register Transfer Level, RTL)、工艺库文件和设计约束文件，经过翻译、逻辑优化和工艺映射等步骤输出工艺库网表。综合出的网表质量直接影响到后续的物理设计过程[1]。逻辑电路仿真器和布尔可满足性(Boolean SATisfiability, SAT)求解器在逻辑综合领域扮演着至关重要的角色。SAT问题被描述为通过推理，判定是否存在1组或多组变量的取值组合，能够使公式取值为真；逻辑电路仿真是通过按拓扑顺序依次访问电路中的所有节点并使用其输入值计算其输出值的过程。它们不仅在数理逻辑和计算机理论等方面有着举足轻重的研究地位，而且作为EDA的底层计算引擎在其各个环节中发挥着重要的作用。

近年来，由中国学者提出和发展的矩阵半张量积(Semi-Tensor Product, STP)理论，打破了维数限制，成为处理多线性数组的有力工具。STP的数

收稿日期：2024-01-09；改回日期：2024-03-15；网络出版：2024-03-21
*通信作者： 储著飞 chuzhufei@nbu.edu.cn
基金项目：国家自然科学基金(62274100, U23A20351)，浙江省重点研发计划(2024C01111)，宁波市重点研发计划(2023Z071)
Foundation Items: The National Natural Science Foundation of China (62274100, U23A20351), The Key R&D Program of Zhejiang Province (2024C01111), The Key R&D Program of Ningbo City (2023Z071)

理逻辑推理与SAT和电路仿真的计算理论高度契合[2,3]，被认为可能实现SAT求解器和电路仿真器设计新思路的突破。同时在逻辑综合的组合电路优化时，随着变量的数量以及约束的数量增多，使用尽可能多的小规模电路来形成输入集的过程十分重要。因此，当涉及到逻辑综合时，不需要功能强大的单实例SAT求解器，一个更灵活、更稳定、更轻量级的基于电路信息的求解器在逻辑综合中产生的一系列基于电路的问题上会表现更好。因此，针对现存的推理引擎在逻辑综合中的应用瓶颈，结合STP的相关理论，基于数理计算的SAT求解器和电路仿真器研究，并应用在逻辑综合的场景具有重要前景。

本文对基于STP的推理引擎及逻辑综合应用的研究进展进行综述，首先对布尔可满足性问题、电路逻辑仿真等背景知识进行介绍；第3节针对矩阵半张量积的发展史、基本原理及其在逻辑推理和逻辑动态系统的应用进行具体阐述；第4节归纳总结了目前国内外常见的基于半张量积的逻辑综合应用场景；最后探讨了研究挑战和未来发展趋势。

2 背景

本节的主要内容是介绍相关研究中涉及的一些基础知识，包括布尔可满足性问题的定义和算法、逻辑电路仿真、k输入查找表(k-input LookUp Table, k-LUT)网络的定义和逻辑优化相关理论以及常见算法。

2.1 布尔可满足性问题

布尔可满足性问题是著名的判定问题，通过推理来确定是否存在一组或多组变量的取值组合，能够使给定问题的布尔公式取值为真。如果存在使布尔公式取值为真的变量取值组合，称该布尔公式是可满足的(SAT)，否则该布尔公式是不可满足的(UNSAT)[4]。一个布尔变量或者其反变量称为文字，若干文字的析取式又称为子句(clause)，所有子句的合取式则构成了描述问题的布尔公式，即合取范式(Conjunction Normal Form, CNF)。

以$x_1 \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_3)$例，该CNF包含3个子句，分别是$x_1$,$(\bar{x}_1 \vee \bar{x}_2 \vee x_3)$,$(\bar{x}_1 \vee x_3)$，要使该CNF为真，则每个子句都必须为真，此例中$x_1$是单文字子句，只有将$x_1$赋值为真才能满足约束。经过推理，$x_3$为真，$x_2$任意赋值即可使CNF为真。将问题转换为CNF描述的过程被称为SAT编码，而求解CNF的过程则被称为SAT求解算法。用于获得SAT判定结果的工具通常被称为SAT求解器[5]。

2.2 电路逻辑仿真

逻辑仿真(logic simulation)是通过为电路的输入分配随机值，并按照拓扑顺序逐一访问电路中的所有节点，利用其输入值计算输出值的过程，以评估电路内部节点或输出的逻辑行为。仿真向量是指为布尔网络的每个主输入(Primary Input, PI)的布尔赋值的集合[6]。如果一组仿真向量包含了所有可能的值分配组合，则被称为穷尽的，而这需要k个PI的2^k个仿真向量。通过仿真穷举向量集生成的仿真信号也被称为真值表，完整地表达了布尔网络中节点所代表的布尔函数。仿真可以在整个逻辑网络中进行全局操作，也可以局部进行，限制在一个小窗口中。在对整个逻辑网络进行仿真时，整个布尔网络的PIs数量通常大于16个，然而穷举超过2^{16}个仿真向量是非常复杂的。因此，为了使用详尽的仿真向量，仿真必须在少于16个叶节点的窗口内进行，通常限制在8～10个。

2.3 k-LUT网络

现场可编程门阵列(Field-Programmable Gate Array, FPGA)的基本逻辑块是k输入查找表，它可以实现多达k个变量的任何布尔函数[7]。基于LUT的FPGA的工艺映射问题是生成一组布尔函数到k-LUTs的映射，通常，这些布尔函数以布尔网络$G=(V,E)$的形式描述，该网络是一个有向无环图(Directed Acyclic Graph, DAG)，其每个顶点都表示为一个布尔函数。

2.4 逻辑优化

逻辑优化以逻辑表达为基础，在逻辑功能一致的前提下，通过优化逻辑表达的逻辑深度、开关活动性、文字数(literals)或节点个数等，分别对应具体工艺下的性能(Performance)、功耗(Power)和面积(Area)，即PPA优化。逻辑综合通常分为工艺无关优化和工艺相关优化。前者旨在简化布尔表达式和逻辑网络，而不考虑制造的目标工艺节点。工艺无关优化主要分为布尔和代数方法，其中代数法又称为多项式代数法，基于普通代数发展而来；布尔法则充分利用布尔代数区别于普通代数的特性，增强了对布尔空间的搜索能力。从算法寻优能力的角度分类，又可分为启发式算法和最优算法。前者通过可接受的成本计算出一个次优解，而后者则不惜成本找到问题的最优解[1]。超大规模数字集成电路在逻辑优化中因受制于规模约束，常采用全局启发式，局部最优算法的模式。

3 矩阵的半张量积

本节首先介绍了矩阵半张量积的发展史和研究现状，然后介绍了矩阵半张量积的定义、性质及其在逻辑推理上的应用，即运用矩阵的半张量积把逻辑算子用矩阵和向量的形式表示出来，最后介绍了基于矩阵半张量积的逻辑动态系统。

3.1 STP的发展史

随着科技的快速发展，矩阵理论和方法已经成为现代科学技术研究中不可缺少的工具。概率统计、微分方程、数值分析、电子学、生物学、力学、网络技术等学科都与矩阵理论有着密切联系。近代矩阵理论适合处理线性或双线性数组，但由于矩阵乘法对因子维数的限制，它对3线性乃至多线性数组的处理无能为力。自1997年起，程代展教授及其研究团队开始研究这个问题，在计算机中存放高维数组采用了“指针”的方式表达中得到启示，这也成了矩阵半张量积理论的原始出发点。在多次的尝试与研究之后，程代展教授[8]将矩阵半张量积与非线性系统的Morgan问题数值解相结合，发表了首篇关于矩阵半张量积的文章。随后，程代展教授在文献[9]中，更进一步系统地介绍了半张量积的相关概念。初期研究集中在逻辑方程的矩阵表示，通过半张量积方法，逻辑方程可以用矩阵表达[10]。2008年，程代展教授结合布尔网络理论，首次提出以矩阵半张量积为工具，特别是矩阵的逻辑表示，研究布尔网络的拓扑结构，实现了将逻辑动态系统转化为本质上是离散动态系统的转换，取得了较大的进展。近年来，宁波大学研究团队开发了一款基于STP引擎的EDA开源工具[11]，并将其集成到逻辑综合中，这为深入研究逻辑综合提供了有力的工具。

3.2 矩阵半张量积的基本原理及其逻辑推理

矩阵的半张量积是普通矩阵乘法的一种扩展，它实现了将普通矩阵乘法推广到任意的两个矩阵，同时保持了普通矩阵的所有重要性质，这为应用带来了巨大的方便。更可贵的是，由于维数的推广以及换位矩阵的引入，使矩阵的半张量积具有若干伪交换性，在一定程度上克服了矩阵乘法缺少可交换性的弱点，为应用矩阵的方法提供了有力工具。关于矩阵的半张量积的更多细节，本文建议读者参考文献[12]。

首先，定义维度为$m\times n$的实矩阵为$\boldsymbol{M}^{m\times n}$。考虑两个实矩阵为$\boldsymbol{X}\in\boldsymbol{M}^{m\times n}$和$\boldsymbol{Y}\in\boldsymbol{M}^{p\times q}$，其矩阵相乘的基本条件为$n=p$，这也是大多数矩阵乘法的一个不可逾越的障碍，然而，矩阵的半张量积可以在任意维度下进行矩阵乘法。已知两个实矩阵$\boldsymbol{X}\in\boldsymbol{M}^{m\times n}$和$\boldsymbol{Y}\in\boldsymbol{M}^{p\times q}$，矩阵$\boldsymbol{X}$和$\boldsymbol{Y}$的半张量积，如式(1)，表示为$\boldsymbol{X}\ltimes\boldsymbol{Y}$，定义为

$$\boldsymbol{X}\ltimes\boldsymbol{Y}=(\boldsymbol{X}\otimes\boldsymbol{I}_{t/n})\cdot(\boldsymbol{Y}\otimes\boldsymbol{I}_{t/p}) \tag{1}$$

其中，$\boldsymbol{I}_t$表示维度为t的单位矩阵、t表示n和p的最小公倍数(least common multiples, lcm)、$\cdot$表示基本矩阵乘法、$\otimes$表示两个矩阵的克罗内克积(Kronecker product)。数学上，克罗内克积是两个任意方框矩阵间的运算，表示为$\otimes$[13]。

通过式(1)，可以推导出两个引理：

引理1　已知两个实矩阵$\boldsymbol{X}\in\boldsymbol{M}^{m\times n}$和$\boldsymbol{Y}\in\boldsymbol{M}^{p\times q}$，如果$n=kp$，则有$\boldsymbol{X}\ltimes\boldsymbol{Y}=\boldsymbol{X}\cdot(\boldsymbol{Y}\otimes\boldsymbol{I}_k)$[14]。

引理2　已知实矩阵$\boldsymbol{X}\in\boldsymbol{M}^{m\times n}$，若$\boldsymbol{Z}_r\in\boldsymbol{M}^{1\times t}$是一个行向量，则有$\boldsymbol{X}\ltimes\boldsymbol{Z}_r=\boldsymbol{Z}_r\ltimes(\boldsymbol{I}_t\otimes\boldsymbol{X})$；相反，若$\boldsymbol{Z}_c\in\boldsymbol{M}^{t\times 1}$是一个列向量，则有$\boldsymbol{Z}_c\ltimes\boldsymbol{X}=(\boldsymbol{I}_t\otimes\boldsymbol{X})\ltimes\boldsymbol{Z}_c$。因此，矩阵的半张量积实现了矩阵交换律[15]。

例如，已知$\boldsymbol{A}=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\in\boldsymbol{M}^{2\times 4}$和$\boldsymbol{B}=\begin{bmatrix}0&1\\1&0\end{bmatrix}\in\boldsymbol{M}^{2\times 2}$，其中$t=\mathrm{lcm}(2,4)=4$。矩阵$\boldsymbol{A}$和$\boldsymbol{B}$的半张量积计算如式(2)所示

$$\begin{aligned}\boldsymbol{A}\ltimes\boldsymbol{B}&=\boldsymbol{A}\cdot(\boldsymbol{B}\otimes\boldsymbol{I}_2)\\&=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\cdot\left(\begin{bmatrix}0&1\\1&0\end{bmatrix}\otimes\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)\\&=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\cdot\begin{bmatrix}0\begin{bmatrix}1&0\\0&1\end{bmatrix}&1\begin{bmatrix}1&0\\0&1\end{bmatrix}\\1\begin{bmatrix}1&0\\0&1\end{bmatrix}&0\begin{bmatrix}1&0\\0&1\end{bmatrix}\end{bmatrix}\\&=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\cdot\begin{bmatrix}0&0&1&0\\0&0&0&1\\1&0&0&0\\0&1&0&0\end{bmatrix}\\&=\begin{bmatrix}1&0&1&1\\0&1&0&0\end{bmatrix}\end{aligned} \tag{2}$$

接下来，主要描述了半张量积在逻辑推理中的应用，即逻辑矩阵公式。本文中接下来的所有的矩阵乘法均为矩阵的半张量积，因为在后文中省略符号“$\ltimes$”。

首先，为了研究半张量积在逻辑推理中的具体应用，需要定义逻辑变量。

定义1　逻辑变量在$D=\{1,0\}$中取值，为了得到矩阵表达式，将“1”和“0”用如式(3)的向量表示

$$1\sim\begin{bmatrix}1\\0\end{bmatrix}=\boldsymbol{\delta}_2^1,0\sim\begin{bmatrix}0\\1\end{bmatrix}=\boldsymbol{\delta}_2^2 \tag{3}$$

因此，$D=\{1,0\}\sim\boldsymbol{\Delta}_2=\{\boldsymbol{\delta}_2^1,\boldsymbol{\delta}_2^2\}$。

定义2　已知一个维度为2×2^n的实矩阵$\boldsymbol{M}$，若$\boldsymbol{M}$的每一列均为$\boldsymbol{\Delta}_2$中的元素，则称该实矩阵$\boldsymbol{M}$为逻辑矩阵。

下面，讨论逻辑算子与它的结构矩阵之间的关

系。引入4个常见的2元逻辑运算符，分别为“与”($\wedge$, AND)、“或”($\vee$, OR)、“蕴涵”($\rightarrow$, IMP)、“等价”($\leftrightarrow$, EQU)。

定义3 已知一个逻辑矩阵$\boldsymbol{M}$，若$\boldsymbol{M}$从左到右的每一列与任意的逻辑运算符的真值表一致的话，则称该逻辑矩阵$\boldsymbol{M}$为结构矩阵。

通过定义，可以得到常见的1元逻辑运算符“非”($\neg$, NOT)的结构矩阵为$\boldsymbol{M}_{\rm n}=\boldsymbol{M}_{\neg}=\begin{bmatrix}0&1\\1&0\end{bmatrix}$，和上述的4个常见的2元逻辑运算符的结构矩阵[16]，如式(4)所示

$$\left.\begin{aligned}\boldsymbol{M}_{\rm c}&=\boldsymbol{M}_{\wedge}=\begin{bmatrix}1&0&0&0\\0&1&1&1\end{bmatrix}\\ \boldsymbol{M}_{\rm d}&=\boldsymbol{M}_{\vee}=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\\ \boldsymbol{M}_{\rm i}&=\boldsymbol{M}_{\rightarrow}=\begin{bmatrix}1&0&1&1\\0&1&0&0\end{bmatrix}\\ \boldsymbol{M}_{\rm e}&=\boldsymbol{M}_{\leftrightarrow}=\begin{bmatrix}1&0&0&1\\0&1&1&0\end{bmatrix}\end{aligned}\right\}\tag{4}$$

一个逻辑算子σ具有唯一的结构矩阵。例如，$\boldsymbol{M}_1$和$\boldsymbol{M}_2$是两个逻辑矩阵，$a\in D$为一个布尔变量，那么$\boldsymbol{M}_i a(i=1,2)$和$\boldsymbol{M}_1\boldsymbol{M}_2$都是逻辑矩阵。若已知$a\in D$为一个布尔变量，则存在$\neg a=\boldsymbol{M}_n a$；若已知$a\in D$,$b\in D$，且$\sigma_i$为任意的2元逻辑运算符，则存在$a\sigma_i b=\boldsymbol{M}_\sigma ab$。因此，矩阵的半张量积实现了从逻辑推理转换成矩阵乘法[17]。

两个逻辑表达式是逻辑等价的，当且仅当任意变量取任意值时，它们的值都相等。例如，本文使用矩阵的半张量积证明一个逻辑公式$a\rightarrow b=\bar{a}\vee b$。公式的左边的矩阵表达为$\boldsymbol{M}_i ab$，而公式的右边的矩阵表达为$\boldsymbol{M}_{\rm d}\boldsymbol{M}_n ab$。已知$\boldsymbol{M}_{\rm d}\boldsymbol{M}_{\rm n}=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}0&1\\1&0\end{bmatrix}=\begin{bmatrix}1&0&1&1\\0&1&0&0\end{bmatrix}=\boldsymbol{M}_{\rm i}$因此，通过半张量积的计算，可以得到$\boldsymbol{M}_{\rm d}\boldsymbol{M}_{\rm n}=\boldsymbol{M}_{\rm i}$，因此该逻辑公式成立。

通过使用结构矩阵和半张量积的性质，逻辑关系可以被轻松地证明。利用每个逻辑算子的矩阵表达，一个具有k个变量的逻辑表达式$\sigma(A_1,A_2,\cdots,A_k)$总可以表示成$\sigma(A_1,A_2,\cdots,A_k)=\boldsymbol{M}_L A_1A_2\cdots A_k$。若一个逻辑表达式具有多个逻辑变量、且该逻辑变量可能出现不止1次，在STP中，就需要将其进行化简运算。为了重复逻辑变量乘法，如$a\cdot a=a^2$，引入降幂矩阵$\boldsymbol{M}_{\rm r}\in\boldsymbol{M}^{4\times2}$，其定义为

$$\boldsymbol{M}_{\rm r}=\begin{bmatrix}1&0\\0&0\\0&0\\0&1\end{bmatrix}\tag{5}$$

此外，引入变量交换矩阵$\boldsymbol{M}_{\rm w}\in\boldsymbol{M}^{4\times4}$对逻辑变量进行排序，其定义为

$$\boldsymbol{M}_{\rm w}=\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}\tag{6}$$

例如，已知存在两个逻辑变量a和b，可以根据降幂矩阵$\boldsymbol{M}_{\rm r}$实现$a^2=\boldsymbol{M}_{\rm r}a$，或者根据变量交换矩阵$\boldsymbol{M}_{\rm w}$实现$\boldsymbol{M}_{\rm w}ba=ab$[18]。

在一个逻辑表达式中，如果一个逻辑变量的值是预先给定的，称其是常量；如果一个逻辑变量的值是任意的，称其是自由变量。利用上文中的定义和引理，有以下结论：任一个含有自由变量$x_1,x_2,\cdots,x_r$的逻辑表达式$L(x_1,x_2,\cdots,x_r)$都可以唯一地表示成规范型

$$L(x_1,x_2,\ldots,x_r)=\boldsymbol{M}_L x_1x_2\cdots x_r=\boldsymbol{M}_L\ltimes_{j=1}^{r}x_j\tag{7}$$

其中，$\boldsymbol{M}_L$是一个2×2^r矩阵，被称为逻辑表达式$L(x_1,x_2,\cdots,x_r)$的结构矩阵，$x_1,x_2,\cdots,x_r\in D$。

通过利用2元逻辑操作的逻辑矩阵，可以得到任意逻辑函数的代数形式。在定义1的基础上，一个布尔变量$v\in D$可以被认为是一个向量$\boldsymbol{v}\in\boldsymbol{\Delta}_2$，相应的一个有$n$个变量的布尔函数$f:D^n\rightarrow D$可以看作是一个映射$f:(\boldsymbol{\Delta}_2)^n\rightarrow\boldsymbol{\Delta}_2$。因此，可以得出以下性质[10]：$f:D^n\rightarrow D$是一个布尔函数，那么就存在唯一的矩阵$\boldsymbol{F}\in\boldsymbol{M}^{2\times2^n}$使得对于每一个$(\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_n)\in(\boldsymbol{\Delta}_2)^n$有式(8)成立

$$f(\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_n)=\boldsymbol{F}\boldsymbol{v}_1\boldsymbol{v}_2\cdots\boldsymbol{v}_n==\boldsymbol{F}\ltimes_{i=1}^{n}\boldsymbol{v}_i\tag{8}$$

其中，$\boldsymbol{F}$被称为逻辑函数f的结构矩阵。

3.3 基于STP的逻辑动态系统

逻辑系统指自变量只取有限个值的动态系统，包括2值的经典逻辑(或布尔逻辑)、k值逻辑以及有限值逻辑。近年来，借助矩阵半张量积发展起来的逻辑动态系统在代数状态空间方面取得了长足的进展，并受到了广泛关注。通过矩阵半张量积，逻辑动态系统被转化成普通的离散时间代数演化方程。在此背景下，本文以布尔网络为例，着重探讨了布尔网络的代数形式。

当状态、输入和输出都只能取两个值的时候，逻辑网络就是布尔网络。在布尔网络中，所有节点均可用半张量积进行表示，节点的值被限制在二进制值0和1之间。通常情况下，布尔网络中的边表示节点之间的连接关系，这些关系由布尔函数描述。在布尔网络中，节点状态的演化由相应的布尔函数控制，节点状态可以同步或异步更新。具有n个节点的布尔网络的逻辑动态系统可被表达为

$$\left.\begin{aligned} x_1(t+1) &= f_1(x_1(t), x_2(t), \cdots, x_n(t)) \\ x_2(t+1) &= f_2(x_1(t), x_2(t), \cdots, x_n(t)) \\ &\vdots \\ x_n(t+1) &= f_n(x_1(t), x_2(t), \cdots, x_n(t)) \end{aligned}\right\} \tag{9}$$

其中，$x_i \in D, i=1,2,\cdots,n$，为点的状态，$f_i: D^n \to D$，$i=1,2,\cdots,n$，为决定状态演化的逻辑函数。

根据式(8)，对于任意一个逻辑函数，可以得到其对应的结构矩阵。因此对于逻辑函数$f_1, f_2, \cdots, f_n$，可得到它们所对应的结构矩阵，分别记为$\boldsymbol{F}_1, \boldsymbol{F}_2, \cdots, \boldsymbol{F}_n$，那么系统式(9)就转化为

$$\left.\begin{aligned} x_1(t+1) &= \boldsymbol{F}_1 \ltimes_{i=1}^{n} x_i(t) \\ x_2(t+1) &= \boldsymbol{F}_2 \ltimes_{i=1}^{n} x_i(t) \\ &\vdots \\ x_n(t+1) &= \boldsymbol{F}_n \ltimes_{i=1}^{n} x_i(t) \end{aligned}\right\} \tag{10}$$

其中，$\boldsymbol{F}_1, \boldsymbol{F}_2, \cdots, \boldsymbol{F}_n \in \boldsymbol{M}^{2\times 2^n}$。

令$x(t) = \ltimes_{i=1}^{n} x_i(t) \in \boldsymbol{M}^{2^n \times 2^n}$，利用矩阵Khatri-Rao积[19]，可以得到与系统式(9)等价的代数形式

$$x(t+1) = \boldsymbol{F}x(t) \tag{11}$$

其中，$\boldsymbol{F} = \boldsymbol{F_1} * \boldsymbol{F_2} * \cdots * \boldsymbol{F_n} \in \boldsymbol{M}^{2^n \times 2^n}$，这里$*$是矩阵的Khatri-Rao积，定义如下：

定义4　设$\boldsymbol{X} \in \boldsymbol{M}^{p\times r}$和$\boldsymbol{Y} \in \boldsymbol{M}^{q\times r}$，则$\boldsymbol{X}$和$\boldsymbol{Y}$的Khatri-Rao积定义为

$$\begin{aligned} \boldsymbol{X} * \boldsymbol{Y} := [&\mathrm{Col}_1(\boldsymbol{A}) \ltimes \mathrm{Col}_1(\boldsymbol{B}), \cdots, \mathrm{Col}_r(\boldsymbol{A}) \\ &\ltimes \mathrm{Col}_r(\boldsymbol{B})] \in \boldsymbol{M}^{pq\times r} \end{aligned} \tag{12}$$

基于逻辑变量和逻辑向量之间的等价性，任何逻辑函数都可以表示为代数形式。在这个框架下，利用矩阵STP，逻辑动态系统研究有了突破性的发展。文献[20]探讨了逻辑动态系统的基础理论问题，特别是布尔控制网络的能观性与可重构性。通过构建一个辅助系统，将布尔控制网络状态的控制转化为辅助系统状态的控制。利用集合能控性方法和控制吸引子的概念，将辅助系统的集合能控性问题等效转化为原系统的能观性和可重构性验证问题。同时，利用矩阵半张量积作为工具，并采用逻辑动态系统的代数状态空间表示方法。文献[21]提出了一种新的逻辑控制系统(Logical Control System, LCS)，被称为非完全逻辑控制系统(Incomplete LCS)。它利用矩阵的半张量积来表达非完全逻辑控制系统的代数形式。通过代数状态空间技术，成功将两个经典的智力问题转化为带有限制状态空间的非完全LCS的可控性问题，其中包括“狼、羊和菜”问题以及“传教士和食人族”问题。文献[21]还提出了相关的解决方案和算法。另外，文献[22]提出逻辑系统理论与控制理论的结合产生了混合系统；文献[23]提出控制论与博弈论的结合产生了博弈论控制；计算机科学与数值方法的结合产生了学习算法。最后，混合系统理论可用于人工智能建模；基于博弈的控制用于控制人工智能，而学习算法则用于实现对人工智能的控制。因此，半张量积在未来的人工智能中将扮演关键角色。随着半张量积理论在布尔网络研究中的不断发展，已逐步形成了布尔网络控制理论的完整框架[24]，为深入研究布尔网络的动态行为提供了有力工具，进一步推动了该方法在多领域的发展和应用。

4　半张量积在逻辑综合中的研究进展

本节主要介绍半张量积在推理引擎、逻辑优化和形式化验证中的应用。

4.1　基于STP的推理引擎

在逻辑综合中，SAT和Simulator被广泛用作推理引擎。

4.1.1　基于STP的SAT求解器

SAT问题是一个在实际应用中具有广泛应用的经典问题，目前已有多种SAT求解器用于求解SAT问题，如MiniSAT[25]。应用于SAT求解器的算法最有名的是20世纪60年代提出的用于解决SAT问题的戴维斯-普特南-洛格曼-洛弗兰德(Davis-Putnam-Logemann-Loveland, DPLL)算法[26]和目前所有SAT求解器的底层算法冲突驱动子句学习(Conflict Driven Clause Learning, CDCL)算法[27]。全解SAT (All solutions SAT, AllSAT)问题是SAT问题的一个变体，其目标是获得给定布尔公式φ的所有可满足的变量赋值。对于AllSAT问题的研究涵盖了无界模型检测[28]、网络验证和数据挖掘[29]等多个领域。为了支持解的枚举，SAT求解器通常通过添加阻塞子句(blocking clauses)来防止已有解的再次出现。然而，由于枚举过程是递增的，这会导致多次重新启动，大大消耗了SAT求解器的资源。目前，对AllSAT问题的求解方法包括阻塞子句方法[30]、按时间顺序回溯方法和2元决策图(Binary Decision Diagram, BDD)方法[31]。最近的研究提出了一种有效地解决AllSAT问题的方法，即将CNF中的公式转换为易处理的析取范式(Disjunctive Normal Form, DNF)表示[32]。尽管已经存在许多AllSAT求解器，但AllSAT问题仍然是一个有待进一步探索的领域[33]。Pan等人[34]针对这个问题提出了一种基于STP的AllSAT求解器，将逻辑推理与SAT问题相结合，并运用矩阵计算来处理SAT问题和AllSAT问题。与传统的使用启发式算法的CNF求解器不同，这种方法将CNF文字视为STP的逻辑变量，以

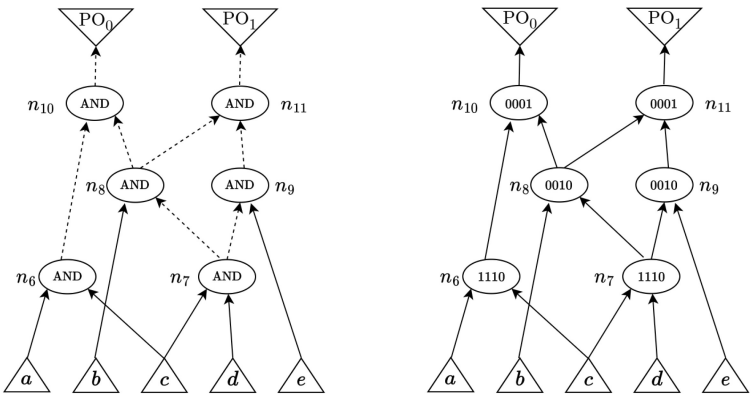
更精确地解决SAT问题。在CNF表达式中，每个子句由若干文字的析取组成。在这一背景下，作者将CNF子句转换为等效的STP形式，并通过计算所有文字的赋值来解决问题。逻辑矩阵将逻辑推理转变为矩阵计算，由于矩阵计算的完备性，该求解器能够高效地一次性计算出所有可满足的解，即AllSAT。从STP理论的角度来看，SAT问题和AllSAT问题的复杂度是一致的。这种方法不仅提供了更精确的解决方案，而且在解的数量较大时表现出更高的效率。

然而，许多SAT问题通常涉及将电路功能和约束转化为CNF描述，然后使用通用的基于CNF的SAT求解器进行求解[35]。除了需要研究电路描述到CNF公式的转换方法之外，改进SAT算法本身也是至关重要的。基于CNF的方法是一种常见的低级编码方式，但将问题转化为CNF形式可能会增加编码的复杂性[29]。与基于CNF的方法不同，基于电路的方法采用更高层次的编码，具有更紧凑的结构，但也更具挑战性。基于电路的SAT求解方法有着悠久的历史，冲突驱动的SAT求解方法[36]是在基于CNF的SAT求解方法的背景下发展起来的，这促使了基于电路的SAT求解器的发展[37]。在进行逻辑综合和组合电路优化时，由于涉及大量的变量和约束，以最小规模电路形成输入集合的过程变得至关重要。因此，当涉及到逻辑综合时，不仅需要强大的单一实例SAT求解器，还需要更加灵活、稳定和轻量级的基于电路信息的SAT求解器，这种求解器在处理基于电路的问题时表现更为出色。文献[2]结合STP知识与电路信息，提出一个可以有效解决逻辑综合问题的电路信息SAT求解器。作者基于CDCL算法提出了一个冲突驱动的电路子句学习算法(Conflict Driven Circuit Clause Learning, CDCCL)，支持电路形式作为输入。通过将逻辑矩阵定义为电路原语，用于将逻辑网络转化为矩阵计算，并保留电路之间的拓扑信息，图1(a)是电路的与非图(And Inverter Graph, AIG)表示，图1(b)是将电路中的每个节点转换成真值表表示，其中n表示电路的节点索引，n_6中的1110表示这个节点的真值表。他们使用基于LUT描述的BENCH文件作为输入，将每个LUT看成一个子电路，对PO和PI之间的逻辑关系进行拓扑排序，通过将每个PO中包含的逻辑关系转换成相应的STP形式，并利用矩阵的STP运算性质，计算出每个PI的所有可满足赋值，解决了AllSAT问题。

4.1.2 基于STP的电路仿真器

常见的逻辑电路仿真器采用快速位并行仿真来减少基于SAT的等价性检查。其基本原理是在预处理阶段为给定逻辑网络计算一组仿真向量(simulation pattern)，然后通过简单地比较仿真特征(simulation signature)来有效地排除大多数不等价性，从而缩短各种计算的运行时间。在文献[38]中，随机仿真和引导仿真被用于识别等价节点并将它们合并以进行函数约简(functional reduction)。文献[39]中，仿真用于寻找两个逻辑网络之间的切入点(cut-points)，以验证输出之间是否存在等价性。此外，文献[40,41]还结合了随机仿真和SAT求解来计算布尔网络窗口中的无关项(don't-cares)，以及计算重替代中的依赖函数(dependency function)。

现代的算术逻辑单元支持按位逻辑运算，以提高仿真效率[6]，然而，对于k-LUT网络的仿真是困难的。这是因为它必须逐位获取每个仿真向量的信息，并按照拓扑顺序访问k-LUT网络中的所有节点，然后使用节点的输入信息来计算其输出信息。根据文献[42]中提出的基于半张量积的逻辑运算方法，k-LUT网络可以很容易地用STP矩阵表示和仿真。不同于文献[6]中逐位逻辑运算仿真k-LUT网络，



(a) 逻辑网络 (b) 真值表

图 1 逻辑网络转换成真值表形式

文献[3]提出了一种基于半张量积的k-LUT逻辑电路仿真器，支持对网络中的所有节点或仅对原始输出(Primary Outputs, POs)进行仿真。通过引入了一种逻辑矩阵编码技术，按照拓扑顺序访问并计算每个节点的真值表，可以加速所有节点的仿真，如图2(a)示例逻辑电路所示，其中n表示节点的索引，0111是节点的真值表；另外提出基于多扇出节点的电路切割(cut)算法，通过将电路切割成多个cut来减少电路的节点数量，并按照拓扑顺序访问并计算每个cut节点的真值表，以做到最大程度的信息复用和对电路PO仿真的速度提升，如图2(b)所示，其中n_8的11010101表示一个三输入节点的真值表。

4.2 基于STP的逻辑优化

随着计算机科学和数学的发展，结合矩阵和逻辑值的半张量积概念引起研究者们的广泛关注，被视为可能实现逻辑优化的重要突破。本节简介矩阵半张量积在逻辑函数简化、精确综合、逻辑重写、SAT扫描(sweeping)的应用。

4.2.1 逻辑函数简化

逻辑函数简化是数字电路设计和布尔代数中的重要概念。它涉及将复杂的逻辑表达式或布尔函数简化为更为紧凑和易于理解的形式，同时保持其等效性。逻辑函数简化的目的是减少电路的复杂性，降低成本，提高性能，并更容易进行逻辑设计和分析。Feng等人[43]的研究专注于逻辑函数的简化问题，并采用矩阵方法来处理这一挑战。作者基于将逻辑矩阵分解为矩阵的克罗内克积的思想，提出一种逻辑函数的简化标准和相应算法。这些工具旨在识别和消除逻辑函数中的冗余变量，从而实现逻辑函数的简化。这种简化过程不仅适用于电路设计，还可用于简化构建状态反馈控制器的逻辑形式，从而提高电路设计和控制器构建的效率。

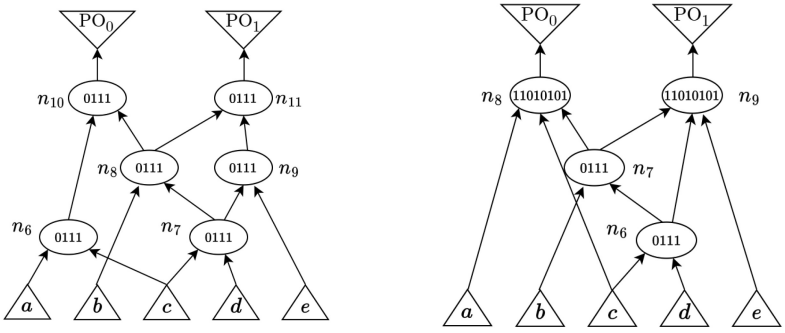
4.2.2 精确综合

在逻辑综合中，基于最优算法寻找最优逻辑表达式的过程被称为精确逻辑综合(exact logic synthesis)，最优综合和精确综合常互换使用[44]。最优逻辑表达式指的是布尔逻辑函数的逻辑表达用到的节点数最少或者逻辑深度最小等。随着算力的提升，逻辑优化越来越追求最优解而不满足于次优解。精确综合采用的最优算法常通过约束求解器完成决策，常用的求解器包括定理证明器、整数线性规划求解器、SAT求解器、可满足性模理论(Satisfiability Modulo Theory, SMT)求解器等。精确综合得到的结果可用于大规模数字电路的局部优化以及新兴计算范式的设计，提升了综合优化的结果质量(Quality of Results, QoR)。

基于SAT的精确综合实际上解决了两个任务：寻找逻辑拓扑结构和分配有向无环图(Directed Acyclic Graph, DAG)顶点上的逻辑运算符，一个典型的目标是寻找具有最少节点或逻辑深度的网络。基于SAT的精确综合可以隐式地利用解空间，而不是显式地枚举所有候选情况。为了加快精确综合的速度，人们尝试提出了可供选择的CNF编码，证明了CNF编码之间的数量差异，并将DAG拓扑结构集成到SAT求解器中，以加速精确综合的过程[45]。文献[46]在基于STP的电路SAT求解器基础上，同时考虑了STP方法在逻辑矩阵计算和电路连通性方面的优势，提出了一种基于STP的精确综合方法。作者将待综合的布尔函数编码为STP形式，然后使用STP分解来筛选无效候选项，并通过基于STP的电路SAT求解器进行求解验证，同时使用DAG拓扑族和基于反例的抽象优化(Counter-Example-Guided Abstraction Refinement, CEGAR)来减少综合运行时间和解空间。结果表明这一算法具有良好的优化效果和适用性。

4.2.3 逻辑重写

在逻辑优化中，逻辑重写(logic rewriting)一般通过切割枚举(cut enumeration)算法将给定逻辑网络划分为更小的子网络，并通过预先计算大量可能的替换子网络，对未优化逻辑网络进行模板模式匹



(a) 示例逻辑电路　　(b) 计算所有cut的真值表

图 2 电路节点真值表

配，用等效的最优逻辑网络进行替换，以达到优化节点数或逻辑深度的目的[47]。Heinz等人[48,49]提出了一种广义的DAG感知重写算法，以k-LUT作为布尔逻辑网络的节点，通过使用支持布尔无关项的精确综合引擎来即时计算替换项，并将所有可能的替换候选存储在冲突图中，从中计算出可兼容替换网络的最大子集。

Pan等人[50]在文献[48]基础上，将基于STP的精确综合应用在逻辑重写算法上，介绍了一种基于工艺映射的逻辑重写算法，重点讨论了面积优化问题，并将其直接应用于k有界逻辑网络(k-bounded logic network)。作者使用基于STP的精确综合在线计算每个k-LUT的潜在网络替换项，通过读取目标实现工艺来计算每个节点的成本cost，减少总体网络大小的替换存储在冲突图中，最终从当前拓扑约束的最优布尔链中选择最低成本的实现。冲突图的一个节点表示一个可能的重写，标记为其实现的增益G和节点化简中的实现成本C[51]。两个节点之间的边表示两个替换之间的冲突，以便只能应用其中一个替换。从冲突图中选择一个最大的非冲突替换集，使G最大，C最小，并应用于布尔网络的重写。图3给出了替换的G和C计算。北卡罗来纳州微电子中心(Microelectronics Center of North Carolina, MCNC)通用标准单元库是目标实现工艺。图3(a)显示了两个功能等价的结构，并计算了C，以红色表示。右边的候选已经包含在图3(b)显示的网络中，同时还给出了初始计数count，其等于每个节点的扇出大小和每个节点的成本cost。接着，移除网络中要被替换的子结构，递归地递减移除节点的扇入中的所有计数，并计算移除子结构的计数和成本，得到count = 2, cost = 8，如图3(c)所示。在图3(d)中插入了等价替代候选项，并初始化新增网络节点的count = 0，所有其它计数保持不变。最后，利用引用计数，计算出替换候选项的count = 2,cost = 6，如图3(e)所示。因此，通过这个例子可以得出，用另一个等价替代候选网络将节省$8-6=2$的实现成本。

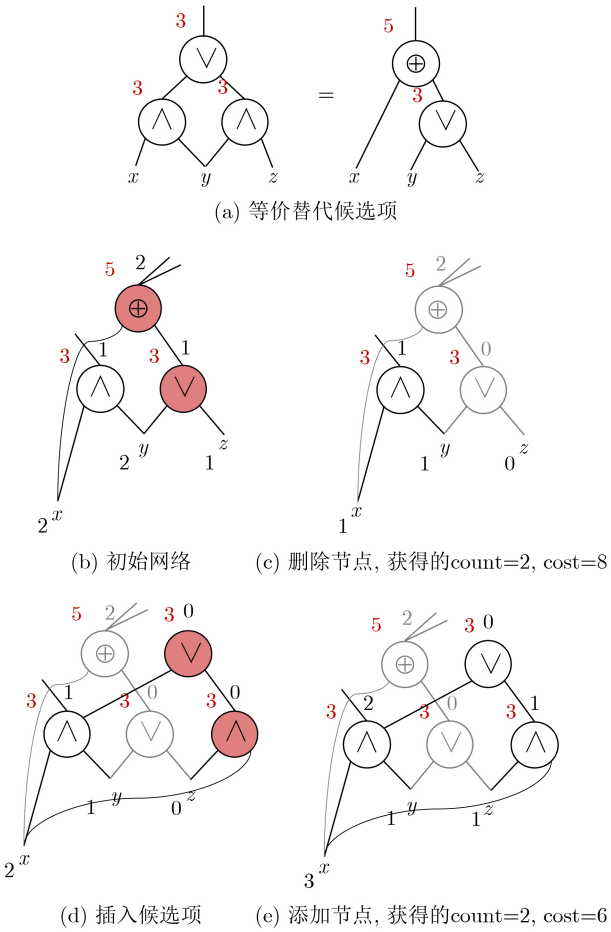


图 3 评估插入替换项的示例

4.2.4 SAT Sweeping

当前的逻辑优化方法通常会将问题编码成SAT问题，以便利用更高效的SAT求解器来提高优化质量。在这个过程中，仿真器的作用是帮助减少候选项的数量，从而降低SAT求解器的调用次数。此外，SAT求解器还可以找到反例(Counter-Example, CE)，而仿真器可以使用这些反例来验证其它属性，从而节省未来的SAT调用[52]。这种相互集成的方法在多个应用中都有潜在的用途，如SAT Sweeping[53]。SAT Sweeping是一种用于简化逻辑网络的方法。它结合了结构哈希(structural Hashing)、电路仿真(circuit simulation)和SAT查询(SAT queries)技术，以合并逻辑网络中具有功能等价性的节点，来减少冗余[54]。当节点无法合并时，SAT求解器提供了CE，即一组输入赋值，允许仿真的两个节点具有不同的值。

传统的SAT Sweeping需要测试所有可能的节点对，但为了提高效率，仿真广泛用于SAT Sweeping，以减少SAT求解器的调用次数。在SAT Sweeping中，仿真器可以通过给定逻辑网络计算一组仿真向量，然后通过比较仿真特征来有效地排除大多数不等价性情况[38]。最近，Pan等人[55]将基于STP的电路仿真器集成到SAT Sweeping框架中，如图4所示。通过结构哈希、电路仿真和SAT查询的组合，SAT Sweeper系统地将输入到输出的等价节点合并，以简化逻辑网络。为了提高效率，作者采用了基于STP的穷举仿真(exhaustive simulation)，这显著减少了错误等价类候选(false equivalence class candidates)的数量，从而减少SAT调用的次数，以提高电路设计的计算效率和可靠性。

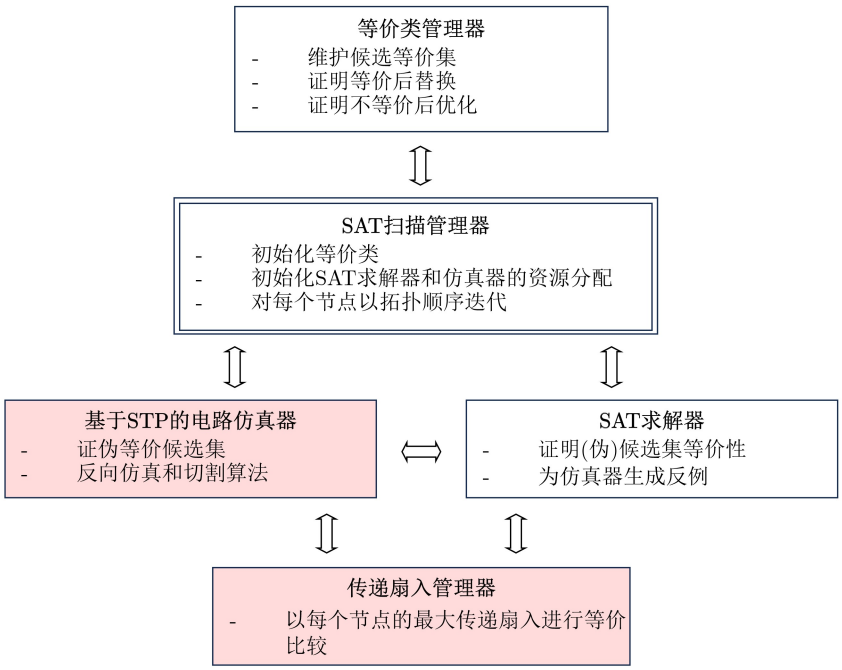


图 4 SAT Sweeping 框架

4.3 基于STP的形式化验证

在实际应用中，逻辑综合和验证起着至关重要的作用，通过逻辑综合，可以将高层次的抽象设计转化为底层的具体电路设计，从而实现对系统性能和功耗的优化，而验证则是确保设计的正确性和可靠性，避免在实际应用中出现故障和错误。目前为止，验证的方法主要有3种，分别是模拟、仿真和形式化验证[56]。形式化验证包括3类：等价性检验(equivalence checking)、定理证明(theorem proving)和模型检验(model checking)。

Lu等人[57]在已有的模型检验(model checking)方法的基础上，结合矩阵的半张量积理论，提出了一种新的基于半张量积的模型检验方法，如图5所示。文献[58]立足于半张量积理论现有的一些应用，针对半张量积在解决问题过程中的不足之处，利用布尔网络中的结构矩阵计算、结构矩阵的乘方计算、结构矩阵逆向推导做出了改进，在参考一般电路模型的基础上，提出了基于半张量积的建模方法。在用半张量积对组合电路建模时，将逻辑运算转换成逻辑矩阵乘积，将每个电路元件换成对应的矩阵，通过矩阵间的运算可以得到整个电路的描述。文献[59]参照布尔网络模型，对同步时序逻辑电路建模，提出了基于半张量积的初步检验方法。将属性描述转化为能够带入矩阵计算的向量形式，通过结构矩阵与向量的乘积得到次态，再与属性所要求的状态迁移进行对比，证明是否满足条件。

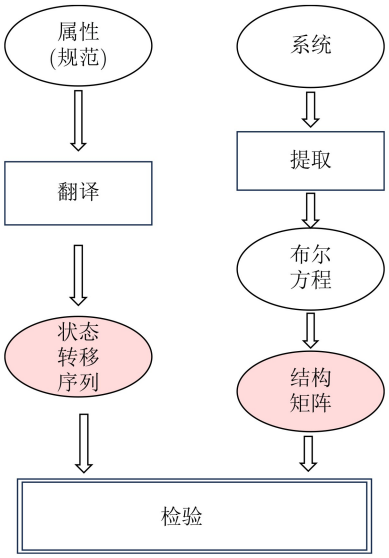


图 5 验证过程图

5 展望

综合上述对推理引擎和逻辑优化的研究现状分析，对未来可能对逻辑综合产生重大影响的新技术进行了讨论，同时对未来的发展趋势做出以下两点展望：

5.1 人工智能

随着人工智能(Artificial Intelligence, AI)技术的发展，AI在逻辑综合方面的应用逐渐受到广泛关注。逻辑综合涉及将各种逻辑优化算法组合成为优化序列并迭代应用于电路，实现对结果质量的优化，例如面积、延迟等。目前，许多研究采用AI的

方法来找到一个高效的优化序列。LSOracle[60]依靠深度神经网络(Deep Neural Networks, DNN)来动态决定将不同优化器应用于电路的不同部分，该框架将电路的DAG表示应用超图分区，并利用AIG和多数反相器图(Majority Inverter Graph, MIG)这两个优化器，动态优化电路。文献[61]结合图卷积网络的强化学习技术来探索设计空间。文献[62]使用了强化学习近端策略优化来自动调整优化序列。他们采用了具有边缘特征聚合能力的图同构网络来学习电路表示，并将电路表示作为强化学习代理的状态表示。此外，为了使智能体能够从历史操作中学习，他们还将长短期记忆(Long Short-Term Memory, LSTM)嵌入到强化学习中。另外，他们还利用人工智能技术从布尔网络中推理出高级抽象。文献[63]则使用神经网络和GPU加速来从网表中推理出高级功能块。他们展示了在超过3.3×10^7个节点的布尔网络中展现出的强大可扩展性，并且在从简单到复杂设计的泛化能力方面表现出色。这项研究在功能验证、逻辑最小化、数据路径综合等应用场景中具有显著作用。此外，逻辑系统理论对AI很重要，而STP在逻辑系统的控制中起着基础性的作用[64]，预计STP将成为AI发展的有力工具。因此，结合AI和STP的方法去解决逻辑综合问题是潜在有效途径之一。

5.2 工艺映射

逻辑优化是和工艺无关的，其目标主要是对逻辑表达进行化简。逻辑综合的最后一步要将优化的逻辑表达用工艺库中的逻辑单元去实现，即工艺映射。目前现有工艺中主要面向的是基于标准单元(standard cell)的专用集成电路(Application Specific Integrated Circuit, ASIC)，和基于LUT的FPGA。随着工艺节点的持续演进，逻辑优化需要考虑更多的物理信息反馈，选择合适的映射算法就会在很大程度上优化FPGA映射结果的面积和延迟。目前，已经有许多用于k-LUTs的FPGA映射算法被提出[65]。基于k-LUTs的FPGA映射问题将门级电路转换为由k-LUTs实现的功能等效电路[66]。在文献[67]中，多值逻辑函数通过半张量积表示其代数形式。然后，通过分解多值逻辑函数的结构矩阵，获得了在不相交对分分解(non-disjoint bi-decomposition)情况下的充分必要条件。在此基础上，文献[68]提出了一种基于半张量积的对分分解(Bi-decomposition)算法，并应用在FPGA映射上。首先通过半张量积技术引入了布尔函数的结构矩阵。然后，通过对结构矩阵进行多次对分分解，将布尔函数分解成具有k个或少于k个输入的子函数，直到这些子函数都被k-LUTs替换，即FPGA映射完成。最后，作者将这一算法应用于一个九输入的组合逻辑电路上，做了基于4-LUTs的FPGA映射，如图6所示，并通过实验结果验证了该算法的有效性。在未来的工作中，可以继续探索基于矩阵半张量积的分解算法，来优化基于k-LUTs的FPGA映射的面积和延迟。

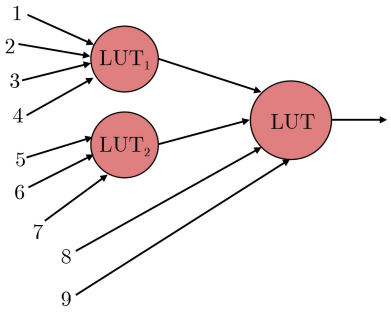


图 6 对应的4-LUTs网络

6 结束语

随着矩阵半张量积理论的不断发展，基于这一技术的推理引擎取得了显著进展，并作为EDA的底层计算引擎在各个环节中发挥着重要的作用。本文详细阐述了基于半张量积的推理引擎以及逻辑综合的研究进展，面对多样化的应用场景，未来的研究需要在推理引擎、逻辑优化、工艺映射、人工智能等方面持续深入研究。通过结合矩阵半张量积理论，本文有望在逻辑综合领域实现更为灵活、高效的设计方法，这不仅有助于提升芯片性能，同时也为电子设计自动化领域带来了全新的可能性。

参 考 文 献

[1] 储著飞, 潘鸿洋. 基于布尔可满足性的精确逻辑综合综述[J]. 电子与信息学报, 2023, 45(1): 14–23. doi: 10.11999/JEIT220391.
CHU Zhufei and PAN Hongxiang. Survey on exact logic synthesis based on boolean SATisfiability[J]. *Journal of Electronics & Information Technology*, 2023, 45(1): 14–23. doi: 10.11999/JEIT220391.

[2] PAN Hongyang and CHU Zhufei. A semi-tensor product based all solutions boolean satisfiability solver[J]. *Journal of Computer Science and Technology*, 2023, 38(3): 702–713. doi: 10.1007/s11390-022-1981-4.

[3] ZHANG Ruibing, PAN Hongyang, and CHU Zhufei. Logic circuit simulation based on semi-tensor product[C]. Proceedings of 2023 China Semiconductor Technology International Conference, Shanghai, China, 2023: 1–3. doi: 10.1109/CSTIC58779.2023.10219157.

[4] FROLEYKS N, HEULE M, ISER M, *et al*. SAT competition 2020[J]. *Artificial Intelligence*, 2021, 301:

103572. doi: 10.1016/j.artint.2021.103572.

[5] NADEL A. Introducing intel(R) SAT solver[C]. The 25th International Conference on Theory and Applications of Satisfiability Testing, Haifa, Israel, 2022: 8: 1–8: 23. doi: 10.4230/LIPIcs.SAT.2022.8.

[6] LEE S Y, RIENER H, MISHCHENKO A, *et al*. A simulation-guided paradigm for logic synthesis and verification[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022, 41(8): 2573–2586. doi: 10.1109/TCAD.2021.3108704.

[7] MAGYARI A and CHEN Yuhua. Review of state-of-the-art FPGA applications in IoT networks[J]. *Sensors*, 2022, 22(19): 7496. doi: 10.3390/s22197496.

[8] CHENG Daizhan. Semi-tensor product of matrices and its application to Morgen's problem[J]. *Science in China Series:Information Sciences*, 2001, 44(3): 195–212. doi: 10.1007/bf02714570.

[9] CHENG Daizhan. Matrix and Polynomial Approach to Dynamic Control Systems[M]. Beijing: Science Press, 2002.

[10] 程代展, 齐洪胜. 矩阵的半张量积理论与应用[M]. 北京: 科学出版社, 2007.
CHENG Daizhan and QI Hongsheng. Semi-Tensor Product of Matrices—Theory and Applications[M]. Beijing: Science Press, 2007.

[11] CHU Zhufei. Semi-tensor product (STP) engine for electronic design automation (EDA)[EB/OL]. https://gitee.com/zfchu/stp, 2023.

[12] CHENG Daizhan, QI Hongsheng, and ZHAO Yin. An Introduction to semi-Tensor Product of Matrices and its Applications[M]. Singapore: World Scientific, 2012.

[13] VAN LOAN C F. The ubiquitous Kronecker product[J]. *Journal of Computational and Applied Mathematics*, 2000, 123(1/2): 85–100. doi: 10.1016/S0377-0427(00)00393-9.

[14] 程代展, 齐洪胜, 赵寅. 布尔网络的分析与控制—矩阵半张量积方法[J]. 自动化学报, 2011, 37(5): 529–540. doi: 10.3724/SP.J.1004.2011.00529.
CHENG Daizhan, QI Hongsheng, and ZHAO Yin. Analysis and control of boolean networks: A semi-tensor product approach[J]. *Acta Automatica Sinica*, 2011, 37(5): 529–540. doi: 10.3724/SP.J.1004.2011.00529.

[15] HORN R A and JOHNSON C R. Matrix Analysis[M]. 2nd ed. Cambridge: Cambridge University Press, 2012.

[16] RÅDE L and WESTERGREN B. Mathematics Handbook for Science and Engineering[M]. 4th ed. Berlin: Springer, 1999. doi: 10.1007/978-3-662-03556-6.

[17] CHENG Daizhan. On logic-based intelligent systems[C]. 2005 International Conference on Control and Automation, Budapest, Hungary, 2005: 71–76. doi: 10.1109/ICCA.2005.1528094.

[18] CHENG Daizhan and QI Hongsheng. Matrix expression of logic and fuzzy control[C]. The 44th IEEE Conference on Decision and Control, Seville, Spain, 2005: 3273–3278. doi:10.1109/CDC.2005.1582666.

[19] KHATRI C G and RAO C R. Solutions to some functional equations and their applications to characterization of probability distributions[J]. *Sankhyā: The Indian Journal of Statistics, Series A*, 1968, 30(2): 167–180.

[20] ZHANG Xiao, MENG Min, WANG Yuanhua, *et al*. Criteria for observability and reconstructibility of Boolean control networks via set controllability[J]. *IEEE Transactions on Circuits and Systems II:Express Briefs*, 2021, 68(4): 1263–1267. doi: 10.1109/TCSII.2020.3021190.

[21] ZHANG Xiao, WANG Yuanhua, and CHENG Daizhan. Incomplete logical control system and its application to some intellectual problems[J]. *Asian Journal of Control*, 2018, 20(2): 697–706. doi: 10.1002/asjc.1579.

[22] CHENG Daizhan, WU Yuhu, ZHAO Guodong, *et al*. A comprehensive survey on STP approach to finite games[J]. *Journal of Systems Science and Complexity*, 2021, 34(5): 1666–1680. doi: 10.1007/s11424-021-1232-8.

[23] MARDEN J R, ARSLAN G, and SHAMMA J S. Cooperative control and potential games[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2009, 39(6): 1393–1407. doi: 10.1109/TSMCB.2009.2017273.

[24] LI Rui and CHU Tianguang. Synchronization in an array of coupled Boolean networks[J]. *Physics Letters A*, 2012, 376(45): 3071–3075. doi: 10.1016/j.physleta.2012.08.037.

[25] EÉN N and SÖRENSSON N. An extensible SAT-solver[C]. The 6th International Conference on Theory and Applications of Satisfiability Testing, Santa Margherita Ligure, Italy, 2003: 502–518. doi: 10.1007/978-3-540-24605-3_37.

[26] DAVIS M, LOGEMANN G, and LOVELAND D. A machine program for theorem-proving[J]. *Communications of the ACM*, 1962, 5(7): 394–397. doi: 10.1145/368273.368557.

[27] MARQUES-SILVA J, LYNCE I, and MALIK S. Conflict-driven clause learning SAT solvers[M]. BIERE A, HEULE M, VAN MAAREN H, *et al*. Handbook of Satisfiability. 2nd ed. Amsterdam: IOS Press, 2021: 133–182.

[28] CABODI G, CAMURATI P E, PALENA M, *et al*. Interpolation with guided refinement: Revisiting incrementality in SAT-based unbounded model checking[J]. *Formal Methods in System Design*, 2022, 60(2): 117–146. doi: 10.1007/s10703-022-00406-Proceedings of the. doi: 10.1007/s10703-022-00406-Proceedingsofthe.

[29] SUNHARE P, CHOWDHARY R R, and CHATTOPADHYAY M K. Internet of things and data mining: An application oriented survey[J]. *Journal of King Saud University-Computer and Information Sciences*, 2022, 34(6): 3569–3590. doi: 10.1016/j.jksuci.2020.07.002.

[30] ZHANG Yueling, PU Geguang, and SUN Jun. Accelerating All-SAT computation with short blocking clauses[C]. The 35th IEEE/ACM International Conference on Automated Software Engineering, Melbourne, Australia, 2020: 6–17.

[31] TODA T and SOH T. Implementing efficient all solutions SAT solvers[J]. *ACM Journal of Experimental Algorithmics*, 2016, 21(1.12): 1–44. doi: 10.1145/2975585.

[32] YU Yinlei, SUBRAMANYAN P, TSISKARIDZE N, *et al*. All-SAT using minimal blocking clauses[C]. 2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems, Mumbai, India, 2014: 86–91. doi: 10.1109/VLSID.2014.22.

[33] FRIED D, NADEL A, and SHALMON Y. AllSAT for combinational circuits[C]. The 26th International Conference on Theory and Applications of Satisfiability Testing, Alghero, Italy, 2023: 9: 1–9: 18. doi: 10.4230/LIPIcs.SAT.2023.9.

[34] PAN Hongyang and CHU Zhufei. A semi-tensor product based all solutions boolean satisfiability solver[C]. 2021 CCF Integrated Circuit Design and Automation Conference, Wuhan, China, 2020.

[35] MASINA G, SPALLITTA G, and SEBASTIANI R. On CNF conversion for disjoint SAT enumeration[C]. The 26th International Conference on Theory and Applications of Satisfiability Testing, Alghero, Italy, 2023: 15: 1–15: 16. doi: 10.4230/LIPIcs.SAT.2023.15.

[36] CAI Shaowei and ZHANG Xindi. Deep cooperation of cdcl and local search for sat[C]. The 24th International Conference on Theory and Applications of Satisfiability Testing, Barcelona, Spain, 2021: 64–81. doi: 10.1007/978-3-030-80223-3_6.

[37] KULIKOV A S, PECHENEV D, and SLEZKIN N. SAT-based circuit local improvement[J]. arXiv: 2102.12579, 2021.

[38] MISHCHENKO A, CHATTERJEE S, JIANG R, *et al*. FRAIGs: A unifying representation for logic synthesis and verification[R]. ERL Technical Report, 2005.

[39] MISHCHENKO A, CHATTERJEE S, BRAYTON R, *et al*. Improvements to combinational equivalence checking[C]. The 2006 IEEE/ACM International Conference on Computer-Aided Design, San Jose, USA, 2006: 836–843. doi: 10.1145/1233501.1233679.

[40] MISHCHENKO A, ZHANG J S, SINHA S, *et al*. Using simulation and satisfiability to compute flexibilities in Boolean networks[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2006, 25(5): 743–755. doi: 10.1109/TCAD.2005.860955.

[41] LEE S Y, RIENER H, and DE MICHELI G. External don't cares in logic synthesis[M]. DRECHSLE R and HUHN S. Advanced Boolean Techniques: Selected Papers from the 15th International Workshop on Boolean Problems. Cham: Springer, 2023: 33–47. doi: 10.1007/978-3-031-28916-3_3.

[42] LIU Zequn and CHENG Daizhan. Canonical form of Boolean networks[C]. 2019 Chinese Control Conference, Guangzhou, China, 2019: 1801–1806. doi: 10.23919/ChiCC.2019.8865729.

[43] FENG June, ZHAO Rong, and CUI Yanjun. Simplification of logical functions with application to circuits[J]. *Electronic Research Archive*, 2022, 30(9): 3320–3336. doi: 10.3934/era.2022168.

[44] HAASWIJK W, SOEKEN M, MISHCHENKO A, *et al*. SAT-based exact synthesis: Encodings, topology families, and parallelism[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020, 39(4): 871–884. doi: 10.1109/TCAD.2019.2897703.

[45] HAASWIJK W, SOEKEN M, AMARÙ L, *et al*. A novel basis for logic rewriting[C]. The 22nd Asia and South Pacific Design Automation Conference, Chiba, Japan, 2017: 151–156. doi: 10.1109/ASPDAC.2017.7858312.

[46] PAN Hongyang and CHU Zhufei. Exact synthesis based on semi-tensor product circuit solver[C]. 2023 Design, Automation & Test in Europe Conference & Exhibition, Antwerp, Belgium, 2023: 1–6. doi: 10.23919/DATE56975.2023.10137287.

[47] MISHCHENKO A, CHATTERJEE S, and BRAYTON R. Improvements to technology mapping for LUT-based FPGAs[C]. 2006 ACM/SIGDA 14th International Symposium on Field Programmable Gate Arrays, Monterey, USA, 2006: 41–49. doi: 10.1145/1117201.1117208.

[48] RIENER H, HAASWIJK W, MISHCHENKO A, *et al*. On-the-fly and DAG-aware: Rewriting Boolean networks with exact synthesis[C]. 2019 Design, Automation & Test in Europe Conference & Exhibition, Florence, Italy, 2019: 1649–1654. doi: 10.23919/DATE.2019.8715185.

[49] RIENER H, LEE S Y, MISHCHENKO A, *et al*. Boolean rewriting strikes back: Reconvergence-driven windowing meets resynthesis[C]. The 27th Asia and South Pacific Design Automation Conference, Taipei, China, 2022: 395–402. doi: 10.1109/ASP-DAC52403.2022.9712526.

[50] PAN Hongyang, XIA Yinshui, WANG Lunyao, *et al*. Semi-tensor product based exact synthesis for logic rewriting[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023. doi: 10.1109/TCAD.2023.3337279.

[51] MISHCHENKO A, CHATTERJEE S, and BRAYTON R. DAG-aware AIG rewriting a fresh look at combinational logic synthesis[C]. The 43rd Annual Design Automation Conference, San Francisco, USA, 2006: 532–535. doi: 10.1145/1146909.1147048.

[52] MISHCHENKO A and BRAYTON R. Integrating an AIG package, simulator, and SAT solver[C]. The International Workshop on Logic and Synthesis, Linz, Austria, 2018: 11–16.

[53] AMARÚ L, MARRANGHELLO F, TESTA E, *et al*. SAT-sweeping enhanced for logic synthesis[C]. The 57th

ACM/IEEE Design Automation Conference, San Francisco, USA, 2020: 1–6. doi: 10.1109/DAC18072.2020.9218691.

[54] ZHU Qi, KITCHEN N, KUEHLMANN A, *et al*. SAT sweeping with local observability don't-cares[C]. The 43rd ACM/IEEE Design Automation Conference, San Francisco, USA, 2006: 229–234. doi: 10.1109/DAC.2006.229206.

[55] PAN Hongyang and CHU Zhufei. Semi-tensor product based circuit simulation for SAT sweeping[C]. The International Workshop on Logic and Synthesis, Lausanne, Switzerland, 2023.

[56] KRICHEN M. A survey on formal verification and validation techniques for internet of things[J]. *Applied Sciences*, 2023, 13(14): 8122. doi: 10.3390/app13148122.

[57] 卢山. 基于半张量积的模型检验方法的研究与实现[D]. [硕士论文], 电子科技大学, 2014.
LU Shan. Research and implementation of model checking method based on semi-tensor product[D]. [Master dissertation], University of Electronic Science and Technology of China, 2014.

[58] ZHAN Jinyu, LU Shan, and YANG Guowu. Improved calculation scheme of structure matrix of Boolean network using semi-tensor product[C]. The 3rd International Conference on Information Computing and Applications, Chengde, China, 2012: 242–248. doi: 10.1007/978-3-642-34038-3_33.

[59] ZHAN Jinyu, LU Shan, and YANG Guowu. Analysis of boolean networks using an optimized algorithm of structure matrix based on semi-tensor product[J]. *Journal of Computers*, 2013, 8(6): 1441–1448. doi: 10.4304/jcp.8.6.1441-1448.

[60] NETO W L, AUSTIN M, TEMPLE S, *et al*. LSOracle: A logic synthesis framework driven by artificial intelligence: Invited paper[C]. 2019 IEEE/ACM International Conference on Computer-Aided Design, Westminster, USA, 2019: 1–6. doi: 10.1109/ICCAD45719.2019.8942145.

[61] HAASWIJK W, COLLINS E, SEGUIN B, *et al*. Deep learning for logic optimization algorithms[C]. 2018 IEEE International Symposium on Circuits and Systems, Florence, Italy, 2018: 1–4. doi: 10.1109/ISCAS.2018.8351885.

[62] YANG Chenghao, XIA Yinshui, CHU Zhufei, *et al*. Logic synthesis optimization sequence tuning using RL-based LSTM and graph isomorphism network[J]. *IEEE Transactions on Circuits and Systems II:Express Briefs*, 2022, 69(8): 3600–3604. doi: 10.1109/TCSII.2022.3168344.

[63] WU Nan, LI Yingjie, HAO Cong, *et al*. Gamora: Graph learning based symbolic reasoning for large-scale boolean networks[C]. The 60th ACM/IEEE Design Automation Conference, San Francisco, USA, 2023: 1–6,doi: 10.1109/DAC56929.2023.10247828.

[64] 程代展, 齐洪胜, 刘挺, 等. 从矩阵半张量积到逻辑控制系统: 献给陈翰馥教授80华诞[J]. 中国科学:数学, 2016, 46(10): 1401–1424. doi: 10.1360/N012015-00381.
CHENG Daizhan, QI Hongsheng, LIU Ting, *et al*. From semi-tensor product of matrices to logical control systems[J]. *Scientia Sinica:Mathematica*, 2016, 46(10): 1401–1424. doi: 10.1360/N012015-00381.

[65] FAN Longfei and WU Chang. FPGA technology mapping with adaptive gate decomposition[C]. The 2023 ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, USA, 2023: 135–140. doi: 10.1145/3543622.3573048.

[66] KUBICA M, OPARA A, and KANIA D. Technology mapping for LUT-Based FPGA[M]. Cham: Springer, 2021: 119–132. doi: 10.1007/978-3-030-60488-2.

[67] CHENG Daizhan and XU Xiangru. Bi-decomposition of multi-valued logical functions and its applications[J]. *Automatica*, 2013, 49(7): 1979–1985. doi: 10.1016/j.automatica.2013.03.013.

[68] LIU Fengqiu, YAN Ming, MAO Yuxin, *et al*. Application of semi-tensor product-based Bi-decomposition to FPGA mapping[C]. The 41st Chinese Control Conference, Hefei, China, 2022: 5945–5949. doi: 10.23919/CCC55666.2022.9901693.

储著飞：男，教授，研究方向为集成电路电子设计自动化，逻辑综合.

马铖昱：男，硕士生，研究方向为集成电路电子设计自动化，逻辑综合.

闫　鸣：男，硕士生，研究方向为集成电路电子设计自动化，逻辑综合.

潘家祥：男，硕士生，研究方向为集成电路电子设计自动化，逻辑综合.

潘鸿洋：男，硕士生，研究方向为集成电路电子设计自动化，逻辑综合.

王伦耀：男，教授，研究方向为集成电路电子设计自动化，逻辑综合.

夏银水：男，教授，研究方向为集成电路电子设计自动化，集成电路设计.

责任编辑：余　蓉