Advances in Applied Mathematics 应用数学进展, 2020, 9(11), 1908-1915
Published Online November 2020 in Hans. http://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2020.911220

Hans汉斯

Gröbner基方法验证乘法器的Maple实现

刘佳姝，张璇思，江建国*

辽宁师范大学数学学院，辽宁 大连
Email: *jjgbox@sina.com

收稿日期：2020年9月25日；录用日期：2020年11月10日；发布日期：2020年11月17日

摘　要

乘法器电路验证问题是一项极其重要而又极具挑战性的难题。当前主流的方法是先将其建模成交换代数中的理想成员问题，然后使用Mathematica、Singular等计算机代数系统中的Gröbner基方法进行判定。本文使用计算机代数系统Maple对这一方法给出了全新实现并在计算机上进行了对比实验。实验结果表明：Maple的实现在某种程度上具有更高的验证效率，可成为乘法器电路强有力的验证平台。

关键词

乘法器，Gröbner基，形式化方法，Maple

Maple Implementation of Verification Multiplier Based on Gröbner Basis

Jiashu Liu, Xuansi Zhang, Jianguo Jiang*

School of Mathematics, Liaoning Normal University, Dalian Liaoning
Email: *jjgbox@sina.com

Received: Sep. 25th, 2020; accepted: Nov. 10th, 2020; published: Nov. 17th, 2020

Abstract

The verification of multiplier circuit is a very important and challenging problem. At present, the mainstream method is to model the ideal member problem in commutative algebra, and then use the Gröbner basis method in Mathematica, Singular and other computer algebra systems to determine. In this paper, a new implementation of this method is given by using the computer algebra system Maple, and a comparative experiment is carried out on the computer. The experimen-

*通讯作者。

文章引用: 刘佳姝, 张璇思, 江建国. Gröbner 基方法验证乘法器的 Maple 实现[J]. 应用数学进展, 2020, 9(11): 1908-1915. DOI: 10.12677/aam.2020.911220

tal results show that the implementation of maple has higher verification efficiency to some extent, and can be a powerful verification platform for multiplier circuits.

Keywords

Multiplier, Gröbner Basis, Formal Method, Maple

1. 引言

乘法器电路是计算机系统、信号处理、密码系统、机器学习等众多现代数字电路系统中的重要组成部分，对其正确性进行验证是必不可少的一个环节[1]。传统的模拟仿真技术与形式化验证方法是乘法器电路验证理论中最重要的两类验证方法。

传统的模拟仿真验证技术原理简单，可操作性强，应用范围广，但无法满足当前乘法器电路设计的巨大验证需求，取而代之的是形式化验证方法[2]。例如，结合 SAT 求解方法与计算机代数方法验证乘法器[3]；使用优化乘数的 SCA 方法验证乘法器[4] [5]。

近几年来，Gröbner 基方法作为形式化验证方法的一种，越来越受到学术界的广泛关注与认同。

Ritirc 等人提出的逐列检测方法[6]验证乘法器是使用 C 语言程序设计出相应的验证工具，并使用列式项序将电路逻辑变成切片，通过验证工具调用计算机代数系统 Mathematica 和 Singular 应用 Gröbner 基方法逐步验证电路每个切片的正确性进而验证乘法器的正确性。该方法在一定程度上降低了归约计算量，同时计算机代数系统 Mathematica 和 Singular 的加入，减少了手工操作，提高了其自动化程度。但在中间的约化过程中单项式数目急剧增加，验证耗费时间较长。因此，Ritirc 等人在此成果之上又提出了加法器重写的方法[7]。加法器重写即提取出电路中的加法器和半加器，约化其 Gröbner 基的内部变量。该方法大大的优化了 Gröbner 基方法，加快了验证速度，并可在有限时间内验证 128 位乘法器。

乘法器电路验证的 Gröbner 基方法逐步优化，但可供选择的计算机代数系统却只有 Mathematica 和 Singular。由于各种操作系统优缺点不同，使用不同种方法搭配不同的代数系统，其验证结果和中间可能出现的情况未知，在一定程度上具有局限性。本文使用计算机代数系统 Maple 对 Gröbner 基方法给出了全新实现并在计算机上进行了对比实验，在某种程度上加快了乘法器验证速度，为后续验证方法的优化提供更多计算机代数系统的选择。

2. Gröbner 基理论

本节介绍 Gröbner 基基础理论的关键内容，包括多项式理想、单项式序、Gröbner 基定义和 Gröbner 基判定理想成员归属问题，相关证明和基础知识可以参考[8] [9] [10]。

定义 2.1 设 k 为任意一个数域，$k[x_1,\cdots,x_n]$ 是数域 k 上关于变量 $x_1,\cdots,x_n$ 的 n 元多项式环，如果非空子集 $I\subseteq k[x_1,\cdots,x_n]$ 满足：

(1) $0\in I$。

(2) 对任意的多项式 $f\in I$， $g\in I$，都有 $f+g\in I$。

(3) 对任意的多项式 $f\in I$， $p\in k[x_1,\cdots,x_n]$，都有 $pf\in I$。

则称 I 是一个**多项式理想**。

任给有限个多项式 $f_1,\cdots,f_s\in k[x_1,\cdots,x_n]$，令

$$\langle f_1,\cdots,f_s\rangle=\left\{\sum_{i=1}^{s}h_if_i\mid h_i\in k[x_1,\cdots,x_n]\right\}$$

易证 $\langle f_1,\cdots,f_s\rangle$ 是 $k[x_1,\cdots,x_n]$ 的多项式理想，$f_1,\cdots,f_s$ 是 $\langle f_1,\cdots,f_s\rangle$ 的生成元。

定义 2.2 设N是非负整数集合

$$\text{单项式集合 } T^n=\left\{x^\beta\mid x^\beta=x_1^{\beta_1}x_2^{\beta_2}\cdots x_n^{\beta_n},\beta_i\in N,i=1,2,\cdots,n\right\}$$

$x^\alpha,x^\beta\in T^n$，如果 T^n 上的一个序关系>满足：

(1) >是一个全序。

(2) >是一个良序。

(3) 若 $x^\alpha>x^\beta$，对所有 $x^\gamma\in T^n$，都有 $x^\alpha x^\gamma>x^\beta x^\gamma$。

则序关系>是**单项式序**。

多项式环 $k[x_1,\cdots,x_n]$ 中单项式序概念是一个基本概念。给定序关系后，可以将任一多项式的各项按序的大小惟一排列，保证 n 元多项式除法得到的余式是惟一的。单项式序包括字典序、分级字典序和分级逆字典序等。本文介绍字典序，其他类型项序可以参考[9]。

定义 2.3 对单项式 x^α,x^β，规定 $x^\alpha>x^\beta$，如果 $\alpha-\beta$ 的从左向右第一个非零分量为负数，称这个序为**字典序**，记为 $>_{lex}$。

定义 2.4 设 I 是 $k[x_1,\cdots,x_n]$ 的多项式理想。$lt(g_i)(i=1,2,\cdots,m)$ 表示 g_i 的首项，$\langle lt(I)\rangle$ 是 I 的所有元素关于字典序 $>_{lex}$ 的首项生成的理想，若存在 I 的有限子集 $G=\{g_1,\cdots,g_m\}$ 满足

$$\langle lt(g_1),\cdots,lt(g_m)\rangle=\langle lt(I)\rangle$$

则 G 是 I 的 **Gröbner 基**。

每个多项式理想都有一个 Gröbner 基，给定多项式理想的任意一个基，Buchberge 算法会在有限步之内算出 Gröbner 基，具体参见[10]。

定理 2.1 设 G 为多项式理想 I 的 Gröbner 基，多项式 $f\in k[x_1,\cdots,x_n]$，则 $f\in I$ 当且仅当 f 被 G 除的余式为 0，记为 $remd(f,G)=0$。

定理 2.1 既可以作为 Gröbner 基的定义，也可以作为理想成员判定问题的解决方法。

3. 代数模型

我们验证具有 $2n$ 个布尔输入 $a_0,\cdots,a_{n-1},b_0,\cdots,b_{n-1}$ 和个 n 布尔输出 $s_0,\cdots,s_{n-1}$ 的乘法器电路，电路的每个内部门(门输出)被一个门变量 $g_0,\cdots,g_{k-1}$ 表示。定义电路变量

$$X=a_0,\cdots,a_{n-1},b_0,\cdots,b_{n-1},g_0,\cdots,g_{k-1},s_0,\cdots,s_{n-1}$$

我们将门输出(门变量) u 与门的输入 v，w 用如下多项式(3.1)表示：

$$\begin{aligned}&u=\neg v\Rightarrow -u+1-v=0\\&u=v\wedge w\Rightarrow -u+vw=0\\&u=v\vee w\Rightarrow -u+v+w-vw=0\\&u=v\oplus w\Rightarrow -u+v+w-2vw=0\end{aligned}\tag{3.1}$$

电路中的所有变量均为布尔变量。因此，我们可以得到关系式 $u(u-1)=0$，此关系式被称为域多项式 F。每个门的输入与输出有相对应的变量，电路中的门被转换成这些变量的多项式关系，即通过多元

多项式来模拟电路的运行方式[11]。

4. Gröbner 基方法及其优化

4.1. Gröbner 基方法

定义 4.1 设 C 是一个电路，$G \subseteq k[x_1,\cdots,x_n]$，$G$ 为电路 C 中每一个门对应的多项式(见(3.1))和输入变量的域多项式 $a_i(a_i-1)$ 及 $b_i(b_i-1)$ 构成的多项式集合。由 G 生成的理想记为 $J(C)$。

根据 Gröbner 基定义结合定义 4.1.1 可知按照字典项序 $>_{lex}$ 排列，G 为 $J(C)$ 的 Gröbner 基。

定义 4.2 设 C 是一个电路，多项式 $p \in k[x_1,\cdots,x_n]$，如果将

$$(a_0,\cdots,a_{n-1},b_0,\cdots,b_{n-1}) \in \{0,1\}^{2n}$$

及由 $a_0,\cdots,a_{n-1},b_0,\cdots,b_{n-1}$ 表示的 $g_0,\cdots,g_{k-1},s_0,\cdots,s_{n-1}$ 的值代入到多项式 p 中结果为 0，则称多项式 p 是电路 C 的规范多项式，记为 PPC。电路 C 的所有 PPC 构成的集合定义为 $I(C)$。

定义 4.3 若

$$\sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right)$$

是 PPC，则电路 C 是**乘法器电路**。

推论 4.1 任一多项式 $p \in k[x_1,\cdots,x_n]$，p 是 PPC 当且仅当 $p \in J(C)$。

根据以上定义我们可得：检验一个给定电路是否是乘法器可以归结为通过 Gröbner 基判定理想成员问题进行验证。即给定一个电路 C 和其对应的规范多项式 p，检验 p 是否属于 $J(C)$。若 p 属于 $J(C)$，则该电路为乘法器电路，反之不成立。这一部分的理论知识也可以推广到任意非循环算术电路。

4.2. 逐位检测方法

逐位检测方法是使用列式项序将电路逻辑变成切片，应用 Gröbner 基方法验证每个电路切片的正确性进而验证乘法器的正确性。将乘法器验证问题分为更小更易于解决的子问题，逐步验证这些子问题进而验证乘法器。

如图 1 所示，在逐位检测方法中，每个电路切片包含且仅包含一个输出位。n 位乘法器电路有 $2n$ 个输出位，即有 $2n$ 个电路切片。

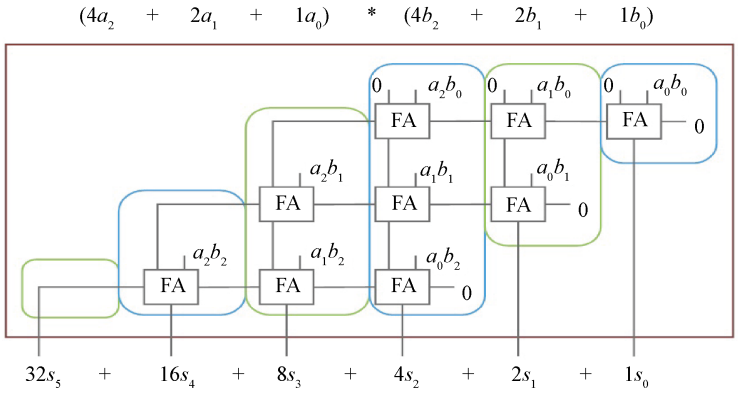

Figure 1. Schematic diagram of bit by bit detection method for 3-bit multiplier

图 1. 3 位乘法器的逐位检测方法示意图

定义 4.4 设 C 是一个 n 位乘法器电路，有如下定义：

(1) 电路 C 的变量 X 上的一列多项式 $C_0,\cdots,C_{2n}$ 是进位序列的**进位多项式**。

(2) 第 i 个电路切片**部分乘积** $P_i=\sum_{k+l=i}a_kb_l,(0\le i\le 2n)$。

(3) 第 i 个电路切片进位**递归关系** $R_i=-C_i+2C_{i+1}+s_i-P_i,(0\le i\le 2n)$。

通过这些定义，我们得到一个抽象定理，可以用来验证经过切片处理的乘法器。

定理 4.1 设 C 是一个所有进位递归关系都成立的电路，若 $C_0-2^{2n}C_{2n}\in I(C)$，则 C 是乘法器。

证明

$$\begin{aligned}&\sum_{i=0}^{2n-1}2^i s_i-\left(\sum_{i=0}^{n-1}2^i a_i\right)\left(\sum_{i=0}^{2n-1}2^i b_i\right)\\&=\sum_{i=0}^{2n-1}2^i\left(P_i+C_i-2C_{i+1}\right)-\sum_{i=0}^{2n-1}2^i P_i\\&=\sum_{i=0}^{n-1}\left(2^i C_i-2^{i+1}C_{i+1}\right)\\&=C_0-2^{2n}C_{2n}\end{aligned}\tag{4.1}$$

由定义 4.3 可知只要证明(4.1)是 PPC 即可证明电路 C 是乘法器。为了得到逐位检测算法，我们逐步定义切片。对每个切片中的输出 s_i，定义它的输入锥即决定输出 s_i 的电路逻辑门(见图 1)集合 $I_i=\{\text{逻辑门}g\mid g\text{是输出}s_i\text{的输入锥}\}$。由输入锥可以定义切片

$$S_0=I_0,\quad S_{i+1}=I_{i+1}\setminus\bigcup_{j=0}^{i}S_j$$

切片 Gröbner 基 G_i 为其切片 S_i 中所有门所对应的多项式集合。综上，即可得逐位检测算法：

算法 1：乘法器逐位检测

输入：切片 Gröbner 基的电路 C
输出：真，C 是乘法器；假，不是乘法器。

1. $C_{2n}=0$;
2. **for** $i\leftarrow 2n-1$ **to** 0 **do**
3. $C_i=Remainder\left(2C_{i+1}+s_i-P_i,G_i\cup F\right)$
4. **end**
5. **return** $C_0=0$

在算法 1 中，C_i 是由给定的第 i 列的部分乘积 P_i、输出位 s_i 和进位多项式 C_{i+1} 唯一确定的，这就保证了进位递归关系 R_i 成立。实际上在算法中简单的取边界进位多项式 $C_{2n}=0$。从最后一个输出位 $s_i\,(i=2n-1)$ 开始，求 $2C_{i+1}+s_i-P_i$ 模切片 Gröbner 基 G_i 和所有域多项式 F 的余式 C_i，通过余式结果检测每个切片的正确性。若每个电路切片所得余式均为 0，则电路 C 是乘法器电路，进而验证了乘法器的正确性。

5. Maple 实现

Maple 是一个具有强大符号计算能力的计算机代数系统，其符号计算能力是 MathCAD 和 Matlab 等软件符号处理的核心[12]。

本文验证两种类型的乘法器，分别是 btor 乘法器和 sp-ar-rc 乘法器。在 Gröbner 基方法验证乘法器的 Maple 实现流程图中(见图 2)，file1.aig 为乘法器通过 AIGER [13]转化得到的 AIG 文件。下面给出一个 AIG 文件的例子：

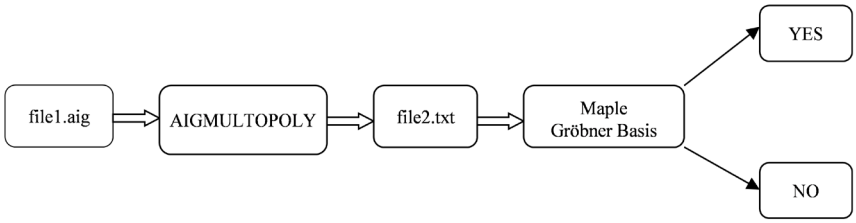

Figure 2. The Maple implementation flow chart of the multiplier verification based on Gröbner basis method
图 2. Gröbner 基方法验证乘法器的 Maple 实现流程图

aag 7 2 0 2 3 //标题
2 //输入 a_0
4 //输入 a_1
6 //输出 b_0
12 //输出 b_1
6 13 15 //与门 0
12 2 4 //与门 1
14 3 5 //与门 2

AIG 文件中第一行为标题行，7 为最大变量索引，2 为输入的个数，0 为锁存的个数，2 为输出的个数，3 为与门的个数。依次列出乘法器的输入，输出和与门。本文所研究的乘法器不包含锁存，故 L 的值为 0。我们重写验证工具 AIGMULTOPOLY 使其读取 AIG 文件后通过脚本文件中的命令行“Aigmultopoly $ aig_file $ Maple_script --maple”产生对应乘法器的 Maple 输出文件，通过脚本文件命令行“maple btori.maplet”使用 Maple 计算检测。AIGMULTOPOLY 输出的n位乘法器 Maple 文件首先定义乘法器位数n和边界进位多项式C_{2n}：

[>] with(Groebner);
[>] Size= n;
[>] Car(2*size)=0;

定义切片 i 的变量集 M[i]和部分乘积集 PP[i]。根据切片 Gröbner 基定义得

$$G_i = \{FI, Fs[i-1], Fs[i], CS[i-1], CS[i], Si\}。$$

用切片 i 的规范多项式 p_i 归约 Gröbner 基 G_i：

[>] Car[i]=NormalForm(s[i]+2Car(i+1)-PP[i],[op(FI),op(Fs[i-1]),
op(Fs[i]),op(CS[i-1]),op(CS[i]), op(Si)], plex(op(MS[i])));

归约通过 Maple 中 Groebner 基库的 NormalForm 函数[14]实现。NormalForm(f, polylist, order)，其中 f 为被归约多项式，polylist 为归约多项式集合，order 为变量的序关系，我们使用的是字典序，对应 Groebner 基库中的 plex 函数。根据逐位检测算法知若电路所有切片归约得到的余式 Car[i]的值均为 0，则输入电路为乘法器电路，反之不成立。

我们使用 Maple、Mathematica 和 Singular 三种计算机代数系统分别对乘法器验证的 Gröbner 基方法和加法器重写(AR)优化的 Gröbner 基方法进行了实验。在我们的实验中，我们将挂钟时间设

置为 1200 秒，主内存限制为 4 GB。所有实验中的时间都以秒为单位。我们从启动验证工具 AIGMULTOPOLY 开始计算到 Maple、Mathematica 和 Singular 完成计算为止，所包含的时间为 AIGMULTOPOLY 生成 Maple、Mathematica 和 Singular 文件的时间和使用相应计算机代数系统计算的时

间。我们通过 TO 标记未完成的实验，MO 标记达到内存限制、EE 标记状态错误。实验结果如下表(见表1)所示：

Table 1. The implementation of Gröbner basis method based on different algebraic systems
表 1. 基于不同代数系统的 Gröbner 基方法的实现

mult	n	Maple		Mathematica		Singular	
			AR		AR		AR
btor	8	0	0	1	0	0	0
btor	16	2	0	4	1	1	0
btor	32	16	2	27	2	20	1
btor	64	172	17	260	18	MO	18
btor	128	TO	402	TO	126	EE	EE
sparrc	8	0	0	1	0	0	0
sparrc	16	3	0	6	1	1	0
sparrc	32	23	2	42	3	19	1
sparrc	64	250	18	450	20	MO	19
sparrc	128	TO	415	TO	129	EE	EE

如表 1 所示，验证 8~64 位乘法器时，Maple 计算速度较 Mathematica 相比更快且验证 64 位乘法器时，Maple 不会出现 Singular 中超出内存限制的情况。验证 128 位乘法器时，环变量数超过 32707 个，Singular 会出现错误[15]，使用 Maple 可以进行计算验证。但是由于 C 语言编程的多样性以及不同计算机代数系统不同版本的差异性使得 Maple 和 Mathematica 的验证时间略有差异。将来，我们将进一步重写优化验证工具以提高乘法器验证的效率。实验结果表明：Gröbner 基方法验证乘法器通过 Maple 实现，在某种程度上加快了验证速度，为乘法器电路提供了强有力的验证平台。

6. 结束语

本文通过 Maple 实现乘法器验证的 Gröbner 基方法，为后续乘法器电路验证方法提供更多的计算机代数系统选择。随着验证方法的逐步优化，期望可以有更多的计算机代数系统可以加入到乘法器电路验证中，进一步提高验证的自动化程度。

参考文献

[1] 许琪, 原巍, 沈绪榜. 一种新的树型乘法器的设计[J]. 西安电子科技大学学报, 2002, 29(5): 805-812.

[2] 韩俊刚, 杜敏慧. 数字硬件的形式化验证[M]. 北京: 北京大学出版社, 2001.

[3] Kaufmann, D., Biere, A. and Kauers, M. (2019) Verifying Large Multipliers by Combining SAT and Computer Algebra. *FMCAD*, San Jose, 22-25 October 2019, 28-36. https://doi.org/10.23919/FMCAD.2019.8894250

[4] Mahzoon, A., Große, D. and Drechsler, R. (2018) PolyCleaner: Clean Your Polynomials before Backward Rewriting to Verify Million-Gate Multipliers. *ICCAD*, California, 5-8 November 2018, 129:1-129:8. https://doi.org/10.1145/3240765.3240837

[5] Mahzoon, A., Große, D. and Drechsler, R. (2020) Towards Formal Verification of Optimized and Industrial Multipliers. *DATE*, Grenoble, 9-13 March 2020, 544-549. https://doi.org/10.23919/DATE48585.2020.9116485

[6] Ritirc, D., Biere, A. and Kauers, M. (2017) Column-Wise Verification of Multipliers Using Computer Algebra. *FMCAD*, Vienna, 2-6 October 2017, 23-30. https://doi.org/10.23919/FMCAD.2017.8102237

[7] Ritirc, D., Biere, A. and Kauers, M. (2018) Improving and Extending the Algebraic Approach for Verifying

Gate-Level Multipliers. *DATE*, Dresden, 19-23 March 2018, 1556-1561. https://doi.org/10.23919/DATE.2018.8342263

[8] 王东明, 夏壁灿. 计算机代数[M]. 北京: 清华大学出版社, 2003.

[9] Becke, T. and Eispfenning, V. (1993) Gröbner Bases. Springer-Verlag, New York. https://doi.org/10.1007/978-1-4612-0913-3_6

[10] Cox, D., Little, J. and O'Shea, D. (1997) Ideals, Varieties, and Algorithms. Springer-Verlag, New York. https://doi.org/10.1007/978-1-4757-2693-0

[11] Keim, M., Dreschlerc, R. and Becker, B. (2003) Polynomial Formal Verification of Multipliers. *Formal Methods in System Design*, **22**, 39-58. https://doi.org/10.1023/A:1021752130394

[12] 刘辉, 李海. MAPLE 符号处理及应用[M]. 北京: 国防工业出版社, 2000.

[13] Biere, A. (2017) The AIGER And-Inverter Graph (AIG) Format Version.

[14] 何青, 王丽芬. Maple 教程[M]. 北京: 科学出版社, 2006.

[15] Decker, W., Greuel, G.M., Pfifister, G. and Schonemann, H. (2016) SINGULAR. http://www.singular.uni-kl.de