Challenges in Verifying Arithmetic Circuits Using Computer Algebra

Armin Biere Manuel Kauers Daniela Ritirc
Johannes Kepler University Linz, Altenbergerstr. 69, 4040 Linz, Austria
armin.biere@jku.at manuel.kauers@jku.at daniela.ritirc@jku.at

***Abstract*—Verifying arithmetic circuits is an important problem which still requires considerable manual effort. For instance multipliers are considered difficult to verify. The currently most effective approach for arithmetic circuit verification uses computer algebra. In this approach the circuit is modeled as a set of pseudo-boolean polynomials and it is checked if the given word-level specification is implied by the circuit polynomials. For this purpose the theory of Gröbner bases is used. In this paper we give a summary of two recent papers on this work. We reword the theory and illustrate the results of these papers by examples. We also present a new technical theorem which allows to rewrite local parts of the Gröbner basis. Rewriting the Gröbner basis has tremendous effect on computation time.**

I. Introduction

Even more than 20 years after the famous Pentium FDIV bug the problem of arithmetic circuit verification, especially multipliers, is still considered to be hard [13]. Up to now several techniques have been used for circuit verification. One approach models the verification problem as a satisfiability (SAT) problem, where the circuit is translated into a conjunctive normal form (CNF). In the SAT 2016 competition a large set of such CNF-encodings was introduced for multiplier circuits [1]. However the results of the competition show that even small multipliers produce very hard SAT problems.

Alternative verification techniques use decision diagrams [2], [5], more specifically binary decision diagrams (BDDs) and binary moment diagrams (BMDs). The drawback of BDDs is their high usage of memory [2]. This issue is prohibited by BMDs which remain linear in size, but require explicit structural knowledge of the multiplier [5].

The currently most effective approach for the verification of gate-level arithmetic circuits uses computer algebra [6], [11], [12], [13], [14], [15], [16], [18]. In this approach each circuit gate output is represented by a variable. For each gate a polynomial is introduced which represents the relation of the gate output and the inputs of the gate. To ensure that variables in the circuit are restricted to boolean values, additional so-called "field polynomials" are introduced.

Furthermore the word-level specification of the multiplier is modeled as a polynomial. Note, that we use the field $\mathbb{Q}$ instead of $\mathbb{Z}_2$ as coefficient domain, because it is unclear how to model a word-level specification in $\mathbb{Z}_2$. If the circuit variables are ordered according to their reverse topological appearance in the circuit, i.e., a gate output variable is greater than the input variables of the gate, then the gate polynomials and field polynomials form a Gröbner basis. As a consequence, the question if a gate-level circuit implements a correct multiplier can be answered by reducing the multiplier specification polynomial by the circuit Gröbner basis. The multiplier is correct if and only if the reduction returns zero. In [13] we showed soundness and completeness of this verification approach for integer multipliers. In this paper we only summarize this approach but also provide more detailed examples.

Related work [6], [18] uses a similar algebraic approach. On the circuit so-called function extraction is applied. The word-level output of the circuit is rewritten to derive a unique polynomial representation of the gate inputs, which is then compared to the circuit specification. This rewriting method is essentially the same as Gröbner basis reduction. The authors of [11], [12] focus on verification of Galois field multipliers using Gröbner bases. We focus in our work [13], [14] on integer multipliers as the authors of [6], [15], [16], [18]. In this recent work [15], [16] the authors propose a reduction scheme which rewrites and simplifies the Gröbner basis and thus has a substantial effort on the computation time. Inspired by these ideas we presented in [13] an incremental column-wise verification technique for integer multipliers where a multiplier can be decomposed into columns as depicted in Fig. 1. In each column the partial products can be uniquely identified and we can define a distinct specification for each slice which relates the partial products, incoming carries, slice output and outgoing carries. Then we incrementally apply Gröbner basis reduction to verify the circuit.

We improved the incremental column-wise checking algorithm in [14]. The idea in that work was to simplify the Gröbner basis by introducing linear adder specifications. We search for full- and half-adder structures in the gate-level circuit and eliminate internal gates, with the effect of including adder specifications in the Gröbner basis. Reducing by these linear polynomials leads to substantial improvements in terms of computation time. In this paper we present a new theorem which allows to rewrite local parts of the Gröbner basis, and thus provides a formal proof for the correctness of such rewriting techniques.

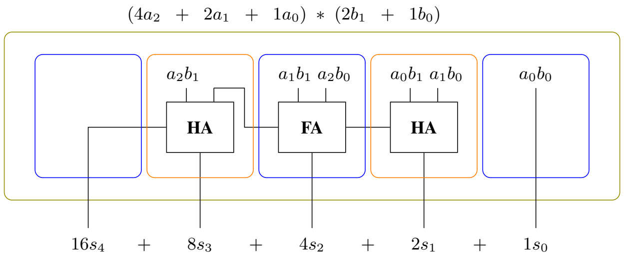


Fig. 1. Column-wise slicing for a 3×2-bit multiplier.

II. Algebra

In the verification approach using computer algebra the circuit and the word-level specification are modeled as multivariate polynomials. For each circuit gate a pseudo-boolean polynomial relating the gate output to its inputs is defined. To gain correctness it is checked if the circuit specification is implied by the gate polynomials [11], [12], [13], [14], [15], [16]. In this section we summarize the theory behind this approach following [3], [4], [7], [13], [14]:

- The ring $\mathbb{Q}[X] = \mathbb{Q}[x_1, \ldots, x_n]$ contains all polynomials p in variables $X = \{x_1, \ldots, x_n\}$ with coefficients in $\mathbb{Q}$.
- A term is a polynomial $x_1^{u_1} \cdots x_n^{u_n}$ over the ring variables X with non-negative exponents $u_i \in \mathbb{N}$. By $[X]$ we denote the set of all terms in $\mathbb{Q}[X]$.
- A monomial $ax_1^{u_1} \cdots x_n^{u_n}$ is a constant multiple of a term with a coefficient $a \in \mathbb{Q} \setminus \{0\}$.
- A polynomial is a finite sum of monomials.
- The set of terms is sorted according to an order $\leq$ compatible under multiplication such that for all terms τ, σ_1, σ_2 it holds that $1 \leq \tau$ and $\sigma_1 \leq \sigma_2 \Rightarrow \tau\sigma_1 \leq \tau\sigma_2$.
- An order $\leq$ is called a *lexicographic term order* if for all terms $\sigma_1 = x_1^{u_1} \cdots x_n^{u_n}$, $\sigma_2 = x_1^{v_1} \cdots x_n^{v_n}$ it holds that $\sigma_1 < \sigma_2$ if and only if there exists an index i such that $u_j = v_j$ for all indices $j < i$, and $u_i < v_i$.
- The largest monomial of a polynomial $p \in \mathbb{Q}[X] \setminus \{0\}$ (w.r.t. $\leq$) is called leading monomial $lm(p)$. The corresponding term is the leading term $lt(p)$. The coefficient of this monomial is called leading coefficient $lc(p)$. We define $tl(p) = p - lm(p)$ as the *tail* of p.
- A *remainder* of a polynomial $f \in \mathbb{Q}[X]$ with respect to a set $P = \{p_1, \ldots, p_m\} \subseteq \mathbb{Q}[X]$ is a polynomial $r \in \mathbb{Q}[X]$ such that no term in r is a multiple of some $\mathrm{lt}(p_i)$ and there exist polynomials $q_1, \ldots, q_m \in \mathbb{Q}[X]$ with $f = q_1p_1 + \ldots + q_mp_m + r$.
- A set of polynomials $I \subseteq \mathbb{Q}[X]$ is called an ideal if for all $f, g, h \in \mathbb{Q}[X]$ (i) $0 \in I$, (ii) if $f, g \in I$, also their sum $f + g \in I$, (iii) if $f \in I, h \in \mathbb{Q}[X]$, then $hf \in I$.
- If $I \subseteq \mathbb{Q}[X]$ and $J \subseteq \mathbb{Q}[X]$ are ideals, then their sum is the set $I + J = \{f + g \mid f \in I, g \in J\}$, which is also an ideal in $\mathbb{Q}[X]$.
- A basis of an ideal $I \subseteq \mathbb{Q}[X]$ is a non-empty set $P = \{p_1, \ldots, p_m\} \subseteq \mathbb{Q}[X]$ such that $I = \{q_1p_1 + \cdots + q_mp_m \mid q_1, \ldots, q_m \in \mathbb{Q}[X]\}$. We say I is *generated by* P and denote this by $I = \langle p_1, \ldots, p_m \rangle = \langle P \rangle$.
- A Gröbner basis $G = \{g_1, \ldots, g_m\}$ of an ideal $I \subseteq \mathbb{Q}[X]$ is a basis (w.r.t. $\leq$) with the property $\langle lt(g_1), \ldots, lt(g_m) \rangle = \langle lt(I) \rangle$.

Lemma 1. Every non-empty ideal $I \subseteq \mathbb{Q}[X]$ has a Gröbner basis w.r.t. a fixed term order.

Proof: Cor. 6 in Chap. 2 §5 of [7]. ■

Lemma 2. Let $G \subseteq \mathbb{Q}[X] \setminus \{0\}$ be a basis of an ideal $I = \langle G \rangle$. We define *S-polynomials* as

$$\mathrm{spol}(p, q) := \mathrm{lcm}(\mathrm{lt}(p), \mathrm{lt}(q)) \left(\frac{p}{\mathrm{lm}(p)} - \frac{q}{\mathrm{lm}(q)} \right)$$

for all $p, q \in \mathbb{Q}[X] \setminus \{0\}$, with lcm the least common multiple. Then G is a Gröbner basis of the ideal I if and only if the remainder of the division of $\mathrm{spol}(p, q)$ by G is zero for all pairs $(p, q) \in G \times G$.

Proof: Thm. 6 in Chap. 2 §6 of [7]. ■

Lemma 3. (Product criterion) If $p, q \in \mathbb{Q}[X] \setminus \{0\}$ are such that $\mathrm{lcm}(\mathrm{lt}(p), \mathrm{lt}(q)) = \mathrm{lt}(p)\,\mathrm{lt}(q)$ then $\mathrm{spol}(p, q)$ reduces to zero mod $\{p, q\}$.

Proof: Prop. 4 in Chap. 2 §9 of [7]. ■

To answer the question if a circuit fulfills its specification we need to check if the specification polynomial is an element of the ideal generated by the circuit polynomials. This problem is called *ideal membership problem*: Given a polynomial $f \in \mathbb{Q}[X]$ and an ideal $I = \langle p_1, \ldots, p_m \rangle = \langle P \rangle \subseteq \mathbb{Q}[X]$, determine if $f \in I$. In general, this question is not easy to answer, but if we have a Gröbner basis for the ideal I, membership can be decided via multivariate polynomial division.

Lemma 4. If $G = \{g_1, \ldots, g_m\}$ is a Gröbner basis, then every $f \in \mathbb{Q}[X]$ has a unique remainder with respect to G.

Proof: Prop. 1 in Chap. 2 §6 of [7]. ■

Given f and G, we can compute the unique remainder of f with respect to G by repeatedly subtracting from f suitable multiples of elements of G such as to eliminate all the terms that are not allowed to appear in a remainder. This process will terminate after finitely many steps.

Lemma 5. Let $G = \{g_1, \ldots, g_m\} \subseteq \mathbb{Q}[X]$ be a Gröbner basis, and let $f \in \mathbb{Q}[X]$. Then $f \in \langle G \rangle$ iff the remainder of f with respect to G is zero.

Proof: Cor. 2 in Chap. 2 §6 of [7]. ■

The theory presented so far covers the basic approach of verifying circuits using computer algebra. In [14] we showed a method where we rewrite the Gröbner basis by variable elimination. To this end we need further results of Gröbner bases theory [7].

Lemma 6. Let $I = \langle f_1, \ldots, f_r \rangle$ and $J = \langle g_1, \ldots, g_s \rangle$ be two ideals in $\mathbb{Q}[X]$. Then $I + J = \langle f_1, \ldots, f_r, g_1, \ldots, g_s \rangle$ and furthermore $\langle f_1, \ldots, f_r \rangle = \langle f_1 \rangle + \ldots + \langle f_r \rangle$.

Proof: Prop. 2 and Cor. 3 in Chap. 4 §3 of [7]. ■

In [14] we split the overall ideal of the circuit into two smaller ideals where one ideal represents a full- or half-adder and the other ideal represents the remaining circuit. In the ideal describing the full- or half-adder we want to eliminate internal adder variables. For this purpose we use the elimination theory of Gröbner bases [7].

In the simple case where all polynomials are linear, we can perform elimination by Gaussian elimination.

Example 1 (Gaussian elimination)**.** Consider the following system of linear equations in $\mathbb{Q}[x, y, z]$:

$$\begin{aligned} 3x + 2y + 3z - 7 &= 0 \\ x + \ \ y + 2z - 3 &= 0 \\ x + \ \ y + \ \ z - 2 &= 0 \end{aligned}$$

If V is the vector space consisting of all $\mathbb{Q}$-linear combinations of the polynomials on the left hand side, then every solution $(x, y, z) \in \mathbb{Q}^3$ of the system is also a root of all polynomials in V. In a sense, V contains all the linear polynomials whose zeroness follows from the zeroness of the generators. In order to find elements of V that do not contain x, we can triangularize the system using Gaussian elimination. This may lead to the equivalent system:

$$\begin{aligned} x + 3y + 2z - 3 &= 0 \\ y + 3z + 2 &= 0 \\ z + 1 &= 0 \end{aligned}$$

The polynomials on the left hand sides are obtained as linear combinations of the polynomials in the original system, therefore in particular they belong to V. More is true: the elements of V which do not contain x are precisely the linear combinations of the x-free polynomials in the triangularized system: $y + 3z + 2$ and $z + 1$.

Using Gröbner bases, the reasoning in the example above extends to systems of nonlinear equations.

Definition 1. [7] Write $X = Y \dot\cup Z$. A term order $<$ on $[X]$ is called an *elimination order* for Z if for all terms σ, τ where σ contains a variable from Z, but τ does not, we have $\tau < \sigma$. We denote this property of the elimination order by $Y < Z$.

For example, if $Z = \{x_1, \ldots, x_i\}$ and $Y = \{x_{i+1}, \ldots, x_n\}$, then the lexicographic term order is an elimination order.

Definition 2. [7] Given an ideal $I \subset \mathbb{Q}[Y, Z]$ the *elimination ideal* J is an ideal of $\mathbb{Q}[Y]$ defined by

$$J = I \cap \mathbb{Q}[Y].$$

Theorem 1. [7] Let $I \subset \mathbb{Q}[Y, Z]$ be an ideal and let G be a Gröbner basis of I with respect to an elimination order $Y < Z$. Then the set

$$H = G \cap \mathbb{Q}[Y]$$

is a Gröbner basis of the elimination ideal $J = I \cap \mathbb{Q}[Y]$.

III. Circuit Verification

We show how the theory of Section II is applied to verification of multiplier circuits based on an example. We present how the Gröbner basis is derived and how reduction works.

Figure 2 depicts the multiplier of Fig. 1 which takes a bitvector A of size 3 and a bitvector B of size 2 as input and computes the product $S = A * B$. On the left side of the figure the circuit representation is shown. The column in the middle shows the gate representation of the circuit and the right column shows for each gate the corresponding polynomial representation. The polynomials in the ring $\mathbb{Q}[X]$ are chosen in such a way that the roots of the polynomials are the solutions of the corresponding gate constraints and vice versa. Since the polynomials are elements of the ring $\mathbb{Q}[X]$ this does not hold in general. It only holds because we restrict the input variables to the boolean domain by adding for each input variable a the "field polynomial" $a(1 - a)$. Restricting only the inputs to the boolean domain is enough because the boolean property is propagated by gate polynomials.

Example 2 (Gate polynomials)**.** The possible boolean solutions for the gate constraint $s_0 = a_0 \wedge b_0$ represented as (s_0, a_0, b_0) are $(1, 1, 1), (0, 1, 0), (0, 0, 1), (0, 0, 0)$ which are all solutions of the polynomial $-s_0 + a_0 b_0 = 0$, when a_0, b_0 are restricted to the boolean domain.

From now on we denote by G the set of all gate polynomials and field polynomials for the circuit depicted in Fig. 2. The set G is represented by the right column in Fig. 2. The circuit is a correct multiplier if and only if

$$(16s_4+8s_3+4s_2+2s_1+s_0)-(4a_2+2a_1+a_0)(2b_1+b_0) \in \langle G \rangle.$$

To check this efficiently we need to find a Gröbner basis for the ideal $\langle G \rangle$. Then we can reduce the specification by the Gröbner basis using multivariate division with remainder.

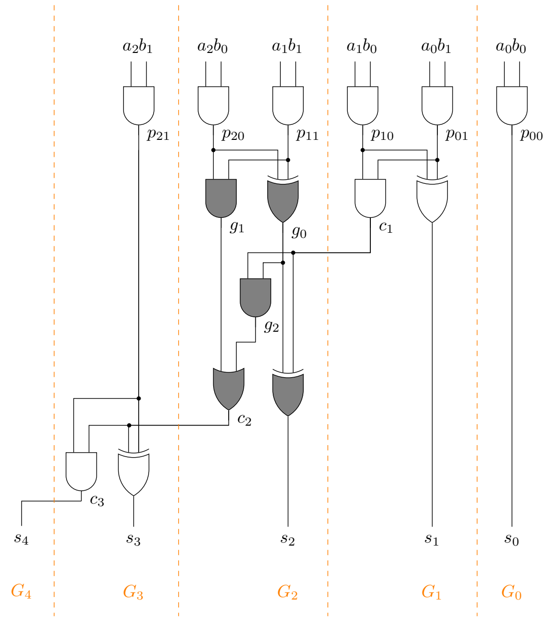
Consequently we fix a lexicographic ordering on the variables. We choose a reverse topological ordering of the gate variables, meaning that the output of a gate is always greater than the inputs of the gate.

The polynomials in Fig. 2 follow such an ordering $<_G$, actually even a column-wise ordering as used in [13]:

$$b_0 < b_1 < a_0 < a_1 < a_2 < p_{00} < s_0 < p_{01} < p_{10} < s_1 < c_1 <$$
$$p_{11} < p_{20} < g_0 < g_1 < g_2 < s_2 < c_2 < p_{21} < s_3 < c_3 < s_4$$

For a reverse topological term ordering the leading term of gate polynomials will always be the gate output itself. The leading term of a field polynomial will always be the square of an input variable. Thus all leading terms are coprime and we can apply the product criterion to all possible pairs, meaning that G is a Gröbner basis. To solve the ideal membership problem we compute a remainder of the specification polynomial with respect to the Gröbner basis G.

Because the leading terms of G contain only one variable, computing a remainder with respect to G has the same effect as substituting each leading term with the corresponding tail until no further substitution is possible.



$$
\begin{array}{rl}
s_4 = c_3 & -s_4 + c_3, \\
c_3 = p_{21} \wedge c_2 & -c_3 + p_{21}c_2, \\
s_3 = p_{21} \oplus c_2 & -s_3 + p_{21} + c_2 - 2p_{21}c_2, \\
p_{21} = a_2 \wedge b_1 & -p_{21} + a_2b_1, \\
c_2 = g_2 \vee g_1 & -c_2 + g_2 + g_1 - g_2g_1, \\
s_2 = g_0 \oplus c_1 & -s_2 + g_0 + c_1 - 2g_0c_1, \\
g_2 = g_0 \wedge c_1 & -g_2 + g_0c_1, \\
g_1 = p_{20} \wedge p_{11} & -g_1 + p_{20}p_{11}, \\
g_0 = p_{20} \oplus p_{11} & -g_0 + p_{20} + p_{11} - 2p_{20}p_{11}, \\
p_{20} = a_2 \wedge b_0 & -p_{20} + a_2b_0, \\
p_{11} = a_1 \wedge b_1 & -p_{11} + a_1b_1, \\
c_1 = p_{10} \wedge p_{01} & -c_1 + p_{10}p_{01}, \\
s_1 = p_{10} \oplus p_{01} & -s_1 + p_{10} + p_{01} - 2p_{10}p_{01}, \\
p_{10} = a_1 \wedge b_0 & -p_{10} + a_1b_0, \\
p_{01} = a_0 \wedge b_1 & -p_{01} + a_0b_1, \\
s_0 = p_{00} & -s_0 + p_{00}, \\
p_{00} = a_0 \wedge b_0 & -p_{00} + a_0b_0, \\
\\
a_2, a_1, a_0 \in \mathbb{B} & a_2(1-a_2),\ a_1(1-a_1),\ a_0(1-a_0), \\
b_1, b_0 \in \mathbb{B} & b_1(1-b_1),\ b_0(1-b_0)
\end{array}
$$

Fig. 2. A gate-level 3x2 multiplier circuit, gate constraints, and polynomials. Dashed lines separate column-wise slices. Colored gates represent a full adder.

Example 3 (Reduction)**.** The first two Gröbner basis reduction steps according to the lexicographic term order $<_G$ for the multiplier circuit of Fig. 2 are computed as follows:

$$
\begin{aligned}
&(16s_4 + 8s_3 + 4s_2 + 2s_1 + s_0) - \\
&\qquad (4a_2 + 2a_1 + a_0)(2b_1 + b_0) \xrightarrow{-s_4 + c_3} \\
&(16c_3 + 8s_3 + 4s_2 + 2s_1 + s_0) - \\
&\qquad (4a_2 + 2a_1 + a_0)(2b_1 + b_0) \xrightarrow{-c_3 + p_{21}c_2} \\
&(16p_{21}c_2 + 8s_3 + 4s_2 + 2s_1 + s_0) - \\
&\qquad (4a_2 + 2a_1 + a_0)(2b_1 + b_0)
\end{aligned}
$$

In [13], [14] we solved the ideal membership problem with the computer algebra systems Mathematica [17] and Singular [8]. The code for verifying the circuit of Fig. 2 in the open-source computer algebra system Singular can be seen in Fig. 3. The order in which the variables are listed in the definition of the ring $R = \mathbb{Q}[X]$ determines the lexicographic term ordering of the variables. According to the column-wise verification procedure of [13] we decompose the circuit Gröbner basis G into sliced Gröbner bases G_i. The separation of the slices is indicated by the dashed lines in Fig. 2. For each sliced Gröbner basis the corresponding polynomials are listed in the Singular encoding. Furthermore the field polynomials are introduced in the set F. It can easily be seen that these sliced Gröbner bases are again Gröbner bases, because the product criterion holds for each slice.

In the non-incremental approach we reduce the whole word-level specification by the overall Gröbner basis G. For the incremental approach we further need to define the sum of partial products for each slice. Then we incrementally reduce the column-wise specification of each slice. In both approaches Singular returns zero, i.e., the specification is contained in the ideal and thus the circuit implements a correct multiplier.

More about the theory of the general approach of circuit verification using computer algebra can be found in [13], [14], [15], [16] for integer multipliers and in [11], [12] for Galois field multipliers. For integer multipliers as the one in Fig. 2 soundness and completeness proofs of this algorithm are given in [13], as well as proofs for the correctness of the discussed column-wise incremental approach.

IV. Rewriting the Gröbner Basis by Variable Elimination

Simply reducing the specification by gate polynomials and field polynomials, as shown in Fig. 3 generally leads to a blow-up in the intermediate reduction results [9], [10], [13]. Optimizations which improve the reduction process are necessary to speed up computation. Since the (non-reduced) Gröbner basis of an ideal is not unique, we might wonder whether Gröbner bases are better than others. A natural choice among all the Gröbner bases is the unique reduced Gröbner basis [7], but it was shown empirically in [14] that the computation of this basis for multipliers is not feasible in practice.

In recent work [15] an optimization called logic reduction rewriting was introduced which partially reduces the Gröbner basis. The goal is to cancel vanishing monomials, i.e., monomials which always evaluate to zero. We adapted this optimization in [13]. We further simplified the circuit Gröbner basis G by splitting it into sliced Gröbner bases G_i. Additionally we also divide the specification polynomial into carry recurrence relations, allowing that partial products are eliminated directly in each slice.

In [14] we further improved the incremental column-wise checking approach by so-called Adder Rewriting. In nearly every multiplier circuit full- and half-adders are found. The idea of Adder Rewriting is to extract these adder structures and then eliminate the internal gate polynomials of these

```
ring R  = 0, (
  s4,
  c3, s3, p21,
  c2, s2, g2, g1, g0, p20, p11,
  c1, s1, p01, p10,
  s0, p00,
  b1, b0, a2, a1, a0
), lp;

ideal G0 =
  -s0 + p00,
  -p00 + a0 * b0;

ideal G1 =
  -p10 + b0 * a1,
  -p01 + a0 * b1,
  -s1 + p10 + p01 - 2 * p10 * p01,
  -c1 + p10 * p01;

ideal G2 =
  -p11 + a1 * b1,
  -p20 + a2 * b0,
  -g0 + p20 + p11 - 2 * p20 * p11,
  -g1 + p20 * p11,
  -g2 + c1 * g0,
  -s2 + c1 + g0 - 2 * c1 * g0,
  -c2 + g1 + g2 - g1 * g2;

ideal G3 =
  -p21 + a2 * b1,
  -s3 + p21 + c2 - 2 * p21 * c2,
  -c3 + p21 * c2;

ideal G4 =
  -s4 + c3;

ideal F = -b0 + b0^2, -b1 + b1^2,
-a0 + a0^2, -a1 + a1^2, -a2 + a2^2;

/*non incremental checking*/
poly spec =
 (a0 + 2*a1 + 4*a2) * (b0 + 2*b1) -
 (s0 + 2*s1 + 4*s2 + 8*s3 + 16*s4);

ideal G = F + G0 + G1 + G2 + G3 + G4;
reduce (spec, G);

/*incremental column-wise checking*/
poly P0 = a0 * b0;
poly P1 = a1 * b0 + a0 * b1;
poly P2 = a2 * b0 + a1 * b1;
poly P3 = a2 * b1;
poly P4 = 0;

reduce(s0 + 2 *
 reduce(s1 + 2 *
  reduce(s2 + 2 *
   reduce(s3 + 2 *
    reduce(s4 + 2 * 0 - P4, G4)
   -P3, G3)
  -P2, G2)
 -P1, G1)
-P0, G0 + F);
quit;
```

Fig. 3. Singular code for verification of the circuit of Fig. 2.

adders. Thus fewer reduction steps are necessary. When G is a Gröbner basis and f is any element of the ideal $\langle G\rangle$, then $G \cup \{f\}$ is also a Gröbner basis for $\langle G\rangle$. We are therefore allowed to add the specification polynomials $2c+s-a-b-i$ for a full adder and $2c+s-a-b$ for a half adder to our basis. These polynomials are linear. Reducing by a linear polynomial helps to speed up computation and reduces the risk of a blow-up. While adding ideal polynomials to a Gröbner basis is always allowed, removing some polynomials is dangerous. We shall explain how to do this based on the example of Fig. 2.

The slice G_2 of Fig. 2 contains a full-adder, depicted by the colored gates. This full adder consists of the carry gate c_2, the sum gate s_2 and inputs p_{20}, p_{11}, c_1. The gates g_2, g_1, g_0 are only used internally in the full-adder and thus we want to eliminate them. So in this setting the elimination variables are $Z = \{g_2, g_1, g_0\}$. Furthermore we want to include the specification $2c_2 + s_2 - p_{20} - p_{11} - c_1$ in G respectively G_2.

The requirements of Thm. 1 demand that the Gröbner basis needs to be calculated w.r.t. an elimination ordering where terms containing Z are the largest elements. This is not the case for our Gröbner basis derived from the circuit using a topological ordering $<_G$. Thus we would really need to calculate a Gröbner basis for the circuit ideal $\langle G\rangle$ for a different ordering $Y < Z$, leading to the same computation issues as for computing the unique reduced Gröbner basis [7].

In [14] we overcome this issue by splitting the overall Gröbner basis G of the circuit into two smaller Gröbner bases G_A and G_B. Since all leading terms of G are coprime, also the leading terms of G_A and G_B have to be coprime and thus by the product criterion G_A and G_B are Gröbner bases. Also the circuit ideal $\langle G\rangle = \langle G_A\rangle + \langle G_B\rangle$ is split. The Gröbner basis G_B contains all polynomials in which variables of Z occur. The Gröbner basis G_A contains the remaining polynomials $G_A = G \setminus G_B$ without any variables in Z.

In our example of Fig. 2 we derive for G_A and G_B:

$$
\begin{aligned}
G_A = \quad & G_0 \cup G_1 \cup \{-p_{11} + a_1 b_1, -p_{20} + a_2 b_0\} \cup \\
& G_3 \cup G_4 \cup F \\
G_B = \; & \{-g_0 + p_{20} + p_{11} - 2p_{20}p_{11}, \\
& -g_1 + p_{20}p_{11}, \\
& -g_2 + c_1 g_0, \\
& -s_2 + c_1 + g_0 - 2c_1 g_0, \\
& -c_2 + g_1 + g_2 - g_1 g_2\}
\end{aligned}
$$

We apply variable elimination only in G_B, because G_A does not contain any element of Z. We calculate a new Gröbner basis H_B w.r.t. an elimination order $<_H$ satisfying $Y < Z$. The elimination order $<_H$ is chosen such that $<_G$ and $<_H$ restricted on Y are equal. Thus the order of terms containing only variables in Y is the same for G_A and H_B. From H_B we remove all polynomials containing variables of Z.

In our setting we define $<_G$ to be a lexicographic term ordering. Thus the elimination order $<_H$ of Z is also a lexicographic term ordering, where the variables of Z are reordered to become the biggest elements. Computing the Gröbner basis H_B w.r.t. $<_H$:

$$b_0 < b_1 < a_0 < a_1 < a_2 < p_{00} < s_0 < p_{01} < p_{10} < s_1 < c_1 < p_{11} < p_{20} < s_2 < c_2 < p_{21} < s_3 < c_3 < s_4 < g_0 < g_1 < g_2$$

leads for instance to the following Gröbner basis

$$\begin{aligned} H_B = \{&g_0 + 2p_{20}p_{11} - p_{20} - p_{11}, \\ &g_1 - p_{20}p_{11}, \\ &g_2 + 2p_{20}p_{11}c_1 - p_{20}c_1 - p_{11}c_1, \\ &s_2 - 4p_{20}p_{11}c_1 + 2p_{20}p_{11} + 2p_{20}c_1 - p_{20} + \\ &\quad 2p_{11}c_1 - p_{11} - c_1, \\ &2c_2 + s_2 - p_{20} - p_{11} - c_1\}. \end{aligned}$$

The first three colored polynomials contain variables of Z and are eliminated. We denote the remaining set $H_Y = H_B \cap \mathbb{Q}[Y]$.

In [14] we give an intuition why we can replace a part of the Gröbner basis G of the circuit by another set of polynomials. We will now show correctness of this approach more formally. The theorem shows that in order to compute a basis of the elimination ideal $\langle G\rangle \cap \mathbb{Q}[Y]$ it suffices to compute a basis of the elimination ideal $\langle G_B\rangle \cap \mathbb{Q}[Y]$.

Theorem 2. Let $G \subseteq \mathbb{Q}[X] = \mathbb{Q}[Y, Z]$ be a Gröbner basis with respect to some term order $<_G$. Let $G_A = G \cap \mathbb{Q}[Y]$ and $G_B = G \setminus G_A$. Let $<_H$ be an elimination order for Z which agrees with $<_G$ for all terms that are free of Z, i.e., terms free of Z are equally ordered in $<_G$ and $<_H$. Suppose that $\langle G_B\rangle$ has a Gröbner basis H_B with respect to $<_H$ which is such that every leading term in H_B is free of Z or free of Y. Then $(\langle G_A\rangle + \langle G_B\rangle) \cap \mathbb{Q}[Y] = \langle G_A\rangle + (\langle G_B\rangle \cap \mathbb{Q}[Y])$.

Proof: The single steps of the elimination procedure of this proof are depicted in Fig. 4. We split the Gröbner basis H_B into two disjoint subsets $H_B = H_Y \cup H_Z$. The set H_Z contains all the polynomials with leading terms in Z and the set $H_Y = H_B \setminus H_Z$ contains the remaining polynomials of H_B with leading terms in Y.

Note that the polynomials contained in H_Y cannot contain any variable of Z, because by definition of the ordering $<_H$ it holds that $Y < Z$. Furthermore G_A does not contain any variable which is an element of Z. Thus H_Z is the only set containing polynomials which include Z.

Using Lemma 6 we derive

$$\begin{aligned} \langle G_A\rangle + \langle G_B\rangle &= \langle G_A\rangle + \langle H_B\rangle \\ &= \langle G_A\rangle + \langle H_Y\rangle + \langle H_Z\rangle \\ &= \langle G_A \cup H_Y\rangle + \langle H_Z\rangle. \end{aligned}$$

Computing a Gröbner basis of an ideal basis does not change the ideal. Using $GB(S, o)$ to denote an arbitrary Gröbner basis

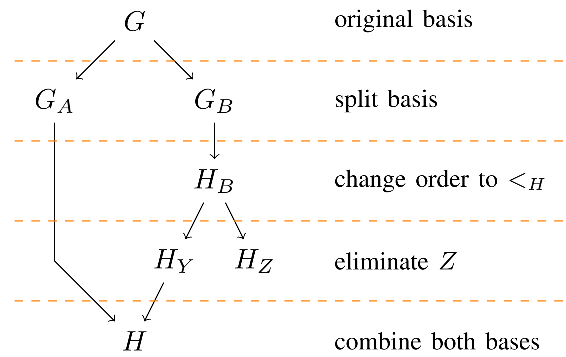


Fig. 4. Gröbner basis transformation

for the set S w.r.t. the ordering o, we thus get

$$\begin{aligned} \langle G_A \cup H_Y\rangle + \langle H_Z\rangle &= \langle GB(G_A \cup H_Y, <_H)\rangle + \langle H_Z\rangle \\ &= \langle GB(G_A \cup H_Y, <_H) \cup H_Z\rangle. \end{aligned}$$

It does not matter if we choose the ordering $<_G$ or $<_H$ to compute a Gröbner basis of $G_A \cup H_Y$, because $G_A \cup H_Y$ is a subset of $\mathbb{Q}[Y]$ and we required the orderings $<_G$ and $<_H$ to be the same for terms only involving variables from Y.

It further holds that

$$\begin{aligned} &GB(GB(G_A \cup H_Y, <_H) \cup H_Z, <_H) \\ &= GB(G_A \cup H_Y, <_H) \cup H_Z, \end{aligned}$$

since all S-polynomials of pairs of polynomials $\{p, q\}$ contained in the set $GB(G_A \cup H_Y, <_H) \cup H_Z$ reduce to zero:

1) $p, q \in GB(G_A \cup H_Y, <_H)$: $GB(G_A \cup H_Y, <_H)$ is a Gröbner basis and thus $\mathrm{spol}(p, q)$ reduces to zero.
2) $p \in GB(G_A \cup H_Y, <_H), q \in H_Z$: H_Z contains only polynomials with leading terms in Z, whereas the leading terms of polynomials in $G_A \cup H_Y$ are elements of Y. Thus $\mathrm{spol}(p, q)$ reduces to zero by the product criterion.
3) $p, q \in H_Z$: The set $H_B = H_Y \cup H_Z$ is a Gröbner basis. Thus $\mathrm{spol}(p, q)$ reduces to zero w.r.t. H_B. Then it also reduces to zero w.r.t. $G_A \cup H_B = G_A \cup H_Y \cup H_Z$. Hence it also reduces to zero w.r.t. $GB(G_A \cup H_Y, <_H) \cup H_Z$, because every leading term of $G_A \cup H_Y$ is a multiple of a leading term in $GB(G_A \cup H_Y, <_H)$.

Altogether it follows that $GB(G_A \cup H_Y, <_H) \cup H_Z$ is a Gröbner basis for the ideal $\langle G_A\rangle + \langle G_B\rangle = \langle G\rangle$.

By Thm. 1 we derive

$$\begin{aligned} &(\langle G_A\rangle + \langle G_B\rangle) \cap \mathbb{Q}[Y] \\ &= \langle GB(G_A \cup H_Y, <_H) \cup H_Z\rangle \cap \mathbb{Q}[Y] \\ &= \langle GB(G_A \cup H_Y, <_H)\rangle. \end{aligned}$$

Computing a Gröbner basis does not change the ideal, hence

$$\langle GB(G_A \cup H_Y, <_H)\rangle = \langle G_A \cup H_Y\rangle = \langle G_A\rangle + \langle H_Y\rangle.$$

Since $H_Y = H_B \setminus H_Z$, i.e. H_Y does not contain any variable of Z, we conclude:

$$\langle H_Y\rangle = \langle H_B\rangle \cap \mathbb{Q}[Y] = \langle G_B\rangle \cap \mathbb{Q}[Y]$$

Composing the results we finally obtain

$$(\langle G_A\rangle + \langle G_B\rangle) \cap \mathbb{Q}[Y] = \langle G_A\rangle + (\langle G_B\rangle \cap \mathbb{Q}[Y]).$$

■

In the proof we used $\langle G_A \cup H_Y\rangle = \langle GB(G_A \cup H_Y, <_H)\rangle$. In fact we do not compute a Gröbner basis, because it would be practically infeasible. By choosing $<_H$ as in Thm. 2, the set $G_A \cup H_Y$ is a Gröbner basis.

Theorem 3. Let $G, G_A, G_B, H_B, H_Y, H_Z, <_H, <_G$ be as in Thm. 2 resp. the proof of Thm. 2. Then $H = G_A \cup H_Y$ is a Gröbner basis w.r.t. the ordering $<_H$.

Proof: We need to show that for every term $\tau \in [Y]$ which is a leading term of an element in $\langle G\rangle$ it holds that there is a $g \in G_A \cup H_Y$ with $lt(g) \mid \tau$. Let τ be such a term.

Since G is a Gröbner basis it holds that there exists an element $g \in G$ with $lt(g) \mid \tau$. Since $G = G_A \cup G_B$ it holds that either $g \in G_A$ or $g \in G_B$:

1) $g \in G_A$: Then $g \in G_A \cup H_Y$.
2) $g \in G_B$: Because $\langle G_B\rangle = \langle H_B\rangle$, there exists an element $h \in H_B$ with $lt(h) \mid lt(g)$ and consequently $lt(h) \mid \tau$. Since $\tau \in [Y]$ it holds that $lt(h) \in [Y]$. Thus $h \in H_Y$ and further $h \in G_A \cup H_Y$.

So in each case $g \in G_A$ or $g \in G_B$ we find an element in $G_A \cup H_Y$ whose leading term divides τ. ■

Theorem 3 allows us that we simply add the Gröbner basis H_Y of the elimination ideal $\langle H_Y\rangle = \langle H_B\rangle \cap \mathbb{Q}[Y]$ to the Gröbner basis G_A and get again a Gröbner basis. This means that in our elimination process, we only have to really compute one "small" Gröbner basis locally, namely H_B.

Example 4 (Reduced slice G_2)**.** All these simplifications lead to the following representation of G_2, given in Singular code:

```
ideal G2 =
 -p11  + a1 * b1,
 -p20  + a2 * b0,
 -s2   + 4*p20 * p11 * c1 - 2*p20 * p11 -
     2*p20 * c1 + p20 - 2*p11 * c1 + p11 + c1,
 -2*c2 - s2 + p20 + p11 + c1;
```

In the slices G_1 and G_3 half-adders occur for which we also want to use linear adder specifications. Variable elimination as for full-adders is not necessary, because a half-adder does not include internal gates. For instance in G_3 we simply can exchange the polynomial $f_1 := -c_3+p_{21}c_2$ with the half adder specification $f_2 := -2*c_3+s_3+p_{21}+c_2$. The polynomial f_2 can be derived by a linear combination of polynomials of G_3, hence we are allowed to add it to G_3. We can now remove the basis polynomial f_1 and G_3 remains a Gröbner basis by the product criterion.

Example 5 (Reduced slice G_3)**.** Polynomial replacing leads to the following polynomial representation of the slice G_3, again given in Singular code.

```
ideal G3 =
  -p21  + a2 * b1,
  -s3   + p21 + c2 - 2*p21 * c2,
  -2*c3 + s3 + p21 + c2;
```

V. Conclusion

In this paper we gave a summary on the theory of arithmetic circuit verification using computer algebra. We summarized two recent papers on this work and illustrated the results by examples. We demonstrated the general approach of circuit verification using computer algebra and extended the examples by the optimization of Adder Rewriting [14]. This optimization adds linear adder specifications to the Gröbner basis, speeding up computation time [14]. A novel contribution of this paper is a technical theorem which is crucial for Adder Rewriting. It allows that the Gröbner basis is only locally simplified in such a way that the result is again a Gröbner basis.

VI. Acknowledgements

This paper extends and provides supplementary material for an invited talk at SYNASC'17 of the first author based on [13], [14] and was supported by Austrian Science Fund (FWF), NFN S11408-N23 (RiSE), Y464-N18, SFB F5004.

References

[1] A. Biere. Collection of combinational arithmetic miters submitted to the SAT Competition 2016. In *SAT Competition 2016*, volume B-2016-1 of *Department of Computer Science Series of Publications B*, pages 65–66. Univ. Helsinki, 2016.

[2] R. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Computers*, 35(8):677–691, 1986.

[3] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, University of Innsbruck, 1965.

[4] B. Buchberger and M. Kauers. Gröbner basis. *Scholarpedia*, 5(10):7763, 2010. http://www.scholarpedia.org/article/Groebner_basis.

[5] Y. Chen and R. Bryant. Verification of arithmetic circuits with binary moment diagrams. In *DAC*, pages 535–541, 1995.

[6] M. Ciesielski, C. Yu, W. Brown, D. Liu, and A. Rossi. Verification of gate-level arithmetic circuits by function extraction. In *DAC*, pages 52:1–52:6. ACM, 2015.

[7] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms*. Springer-Verlag New York, 1997.

[8] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 4-1-0. http://www.singular.uni-kl.de, 2016.

[9] A. Kandri-Rody and D. Kapur. Computing a Gröbner basis of a polynomial ideal over a Euclidean domain. *Journal of Symbolic Computation*, 6(1):37–57, 1988.

[10] A. Kandri-Rody, D. Kapur, and P. Narendran. An ideal-theoretic approach to work problems and unification problems over finitely presented commutative algebras. In *RTA*, volume 202 of *LNCS*, pages 345–364. Springer, 1985.

[11] J. Lv, P. Kalla, and F. Enescu. Efficient Gröbner basis reductions for formal verification of Galois field arithmetic circuits. *IEEE TCAD*, 32(9):1409–1420, 2013.

[12] T. Pruss, P. Kalla, and F. Enescu. Equivalence verification of large Galois field arithmetic circuits using word-level abstraction via Gröbner bases. In *DAC*, pages 152:1–152:6. ACM, 2014.

[13] D. Ritirc, A. Biere, and M. Kauers. Column-wise verification of multipliers using computer algebra. In D. Stewart and G. Weissenbacher, editors, *FMCAD*, pages 23–30. IEEE, 2017.

[14] D. Ritirc, A. Biere, and M. Kauers. Improving and extending the algebraic approach for verifying bit-level multipliers. In *DATE*, 2018, in press.

[15] A. Sayed-Ahmed, D. Große, U. Kühne, M. Soeken, and R. Drechsler. Formal verification of integer multipliers by combining Gröbner basis with logic reduction. In *DATE*, pages 1048–1053. IEEE, 2016.

[16] A. Sayed-Ahmed, D. Große, M. Soeken, and R. Drechsler. Equivalence checking using Gröbner bases. In *FMCAD*, pages 169–176. IEEE, 2016.

[17] Wolfram Research, Inc. Mathematica, 2016. Version 10.4.

[18] C. Yu, W. Brown, D. Liu, A. Rossi, and M. Ciesielski. Formal verification of arithmetic circuits by function extraction. *IEEE TCAD*, 35(12):2131–2142, 2016.