Leveraging Gröbner Bases and SAT for Hardware Verification

Can Gröbner bases help SAT solving?

Priyank Kalla

Associate Professor
Electrical and Computer Engineering, University of Utah
kalla@ece.utah.edu
http://www.ece.utah.edu/~kalla

Can GB help CNF-SAT solving?

My views:

- In theory, yes!
- In practise, probably not...
- GB can help improve solving decision problems before they are encoded into SAT
- Probably beneficial for SMT
- Beneficial for some-specific hardware/software verification problems
- But, there is hardly any work on the practical side, so there is hope!

Some Background...

- My interests – Design Automation and Hardware Verification
  - Formal Verification of arithmetic datapaths (RTL)
  - Word-level (bit-vector level) abstractions of hardware
  - Verification of finite-precision arithmetic
- Applications: Equivalence Check, Simulation Vector Generation
  - Arithmetic RTLs: functions over k-bit-vectors
  - k-bit-vector $\mapsto$ integers $\pmod{2^k} = \mathbb{Z}_{2^k}$
  - k-bit-vector $\mapsto$ Galois (Finite) field $\mathbb{F}_{2^k}$
- For many of these applications SAT/SMT fail miserably!
- Computer Algebra and Algebraic Geometry + SAT/SMT
  - Model: Circuits as polynomial functions $f : \mathbb{Z}_{2^k} \rightarrow \mathbb{Z}_{2^k}$, $f : \mathbb{F}_{2^k} \rightarrow \mathbb{F}_{2^k}$
  - All things Gröbner + SAT
- Objective of this talk: How to best exploit complementary features of GB & SAT?

Agenda

- Motivating applications in hardware verification
- Mini Tutorial on Gröbner bases (GB)
  - Ideals, Varieties, Buchberger's algorithm, F_4
  - Reasoning about solutions
- How I used GB & SAT for hardware verification
  - Challenge: Overcome complexity of Gröbner Basis algorithm
  - Give a simpler problem to SAT
- How others have used GB for SAT (selective review)
  - CNF Pre-processing, conflict-clause learning, quantifier elimination
- Discussion: What more can be done to leverage GB & SAT/SMT?
  - Constraint mining, UNSAT-core extraction, problem size-*vs*-hardness

Equivalence proof: Miter is infeasible

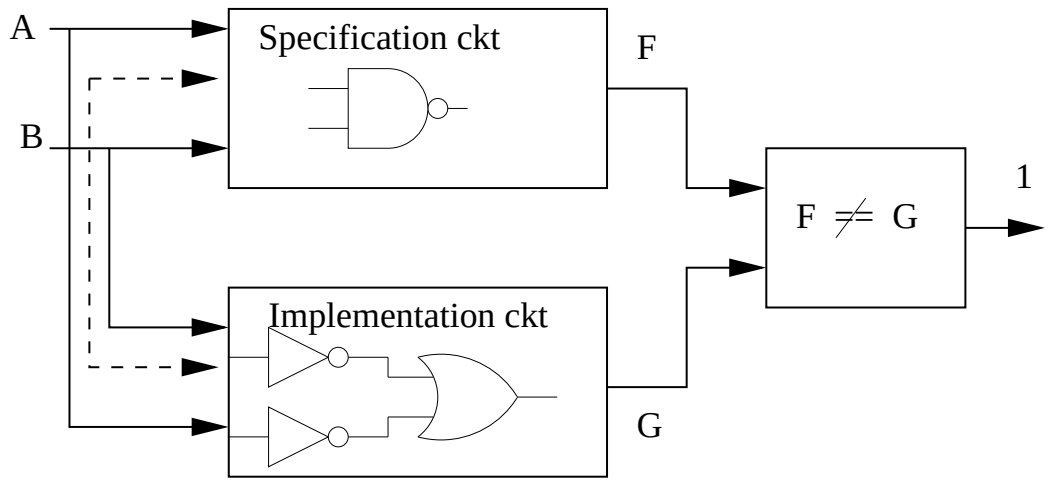


Figure: Miter the circuits F, G

- Constraints as CNF, BV-SMT, Polynomials in $\mathbb{F}_{2^k}$: Prove UNSAT, or find a counter-example
- EDA-techniques: Circuit-SAT, AIG-reductions, constraint-learning
- Limitations: No internal structural equivalences

Custom Optimized Arithmetic Circuits

- Specification circuit: $F = A \cdot B \pmod{P}$ over $\mathbb{F}_{2^k}$

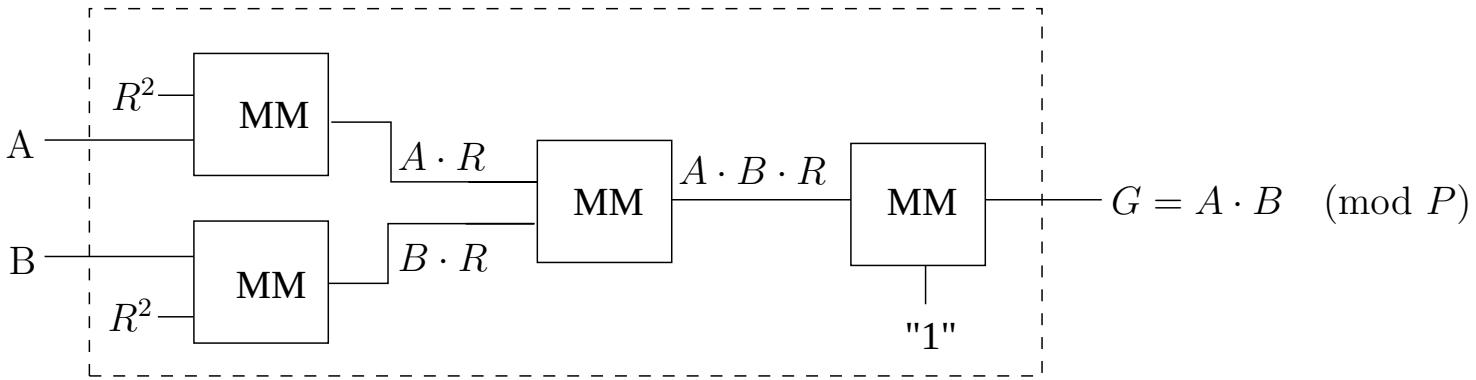


Figure: Montgomery multiplier over $\mathbb{F}_{2^k}$

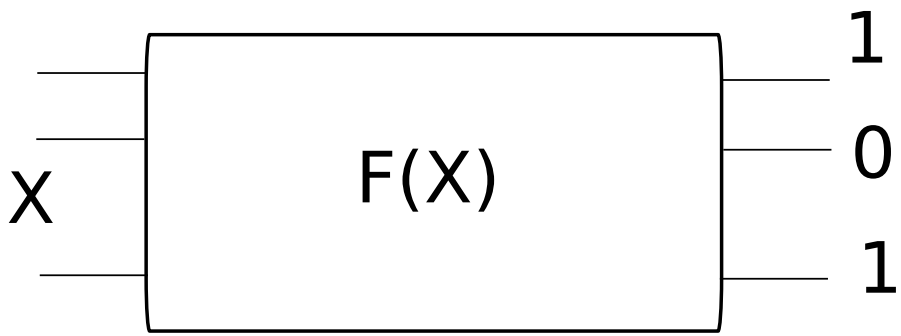
- $MM(X, Y) = X \cdot Y \cdot R^{-1}$
- Used in cryptography, error-correction, $k = 163, 233, 283, 471, \ldots$
- Same function, but completely different circuits

Equivalence Check: $\mathbb{F}_{2^k}$ Multipliers in Cryptography

Table: Modern SAT/SMT-solvers fair poorly; TO = 10hrs

	Word size of the operands k-bits		
Solver	8	12	16
MiniSAT	22.55	*TO*	*TO*
CryptoMiniSAT	7.17	16082.40	*TO*
PrecoSAT	7.94	*TO*	*TO*
PicoSAT	14.85	*TO*	*TO*
Yices	10.48	*TO*	*TO*
Beaver	6.31	*TO*	*TO*
CVC	*TO*	*TO*	*TO*
Z3	85.46	*TO*	*TO*
Boolector	5.03	*TO*	*TO*
Sonolar	46.73	*TO*	*TO*
SimplifyingSTP	14.66	*TO*	*TO*
ABC	242.78	*TO*	*TO*
BDD	0.10	14.14	1899.69

Simulation vector generation



- Find all $X = X_0$ s.t. $F(X_0) = \{101\}$
- Suppose $\{101\}$ is a rare occurrence (Hardware Trojan)
- Assignments exist? How many? (algebraic)
- A solver needed to obtain the solution (SAT/SMT)

Simplify the problems using GB and then use SAT

A Mini Tutorial on Gröbner Bases

Notation:

- $\mathbb{F}$ denotes a field, $\overline{\mathbb{F}}$ its algebraic closure
- $R = \mathbb{F}[x_1, \ldots, x_n]$ multivariate polynomial ring
  - $\mathbb{F} : \mathbb{R}, \mathbb{Q}, \mathbb{C}, \mathbb{F}_q, q = p^k$
- Gröbner bases can be computed over rings$\mathbb{Z}, \mathbb{Z}_m$ (more involved)
- Polynomial $f = c_1 X_1 + c_2 X_2 + \cdots + c_t X_t$
  - A monomial ordering is imposed on $f : X_1 > X_2 > \cdots > X_t$
  - Leading term $lt(f) = c_1 X_1, \quad tail(f) = c_2 X_2 + \cdots + c_t X_t$
  - Leading coefficient $lt(f) = c_1$ and leading monomial $lm(f) = X_1$

Varieties

Given a set of polynomials:

- $f, f_1, f_2, \ldots, f_s \in \mathbb{F}[x_1, \ldots, x_n]$
- Find solutions to $f_1 = f_2 = \cdots = f_s = 0$
- Variety: Set of ALL solutions to a given system of polynomial equations: $V(f_1, \ldots, f_s)$
  - $V(x^2 + y^2 - 1) = \{\textit{all points on circle} : x^2 + y^2 - 1 = 0\}$
  - $V_{\mathbb{R}}(x^2 + 1) = \emptyset$;
  - $V_{\mathbb{C}}(x^2 + 1) = \{(\pm i)\}$
- Variety depends on the ideal generated by the polynomials
- Reason about the Variety by analyzing the Ideals

Definition

Ideal of Polynomials: Let $f_1, f_2, \ldots, f_s \in R = \mathbb{F}[x_1, \ldots, x_n]$. Let

$$J = \langle f_1, f_2 \ldots, f_s \rangle = \{ f_1 \cdot h_1 + f_2 \cdot h_2 + \cdots + f_s \cdot h_s \ : \ h_1, \ldots, h_s \in R \}$$

$J = \langle f_1, f_2 \ldots, f_s \rangle$ is an ideal generated by $f_1, \ldots, f_s$ and the polynomials are called the generators.

Definition

Ideal Membership: Let $f, f_1, f_2, \ldots, f_s \in \mathbb{F}[x_1, \ldots, x_n]$. Let
$J = \langle f_1, f_2 \ldots, f_s \rangle$ be an ideal $\subset \mathbb{F}[x_1, \ldots, x_n]$.
If $f = f_1 h_1 + f_2 h_2 + \cdots + f_s h_s$, then $f \in J$.

If $f_1(\mathbf{a}) = \cdots = f_s(\mathbf{a}) = 0$, then $f(\mathbf{a}) = 0$

Gröbner Basis of Ideals

- Different generators can generate the same ideal
- $\langle f_1, \cdots, f_s \rangle = \cdots = \langle g_1, \cdots, g_t \rangle$ such that $V(f_1, \ldots, f_s) = \cdots = V(g_1, \ldots, g_t)$
- Some generators are a "better" representation of the ideal
- A **Gröbner basis** is one such "canonical" representation of an ideal

Definition

$$G = \{g_1, \ldots, g_t\} = GB(J) \iff \forall f \in J, \exists g_i \text{ s.t. } lm(g_i) \mid lm(f)$$

Definition

$$G = GB(J) \iff \forall f \in J, f \xrightarrow{g_1, g_2, \cdots, g_t}_{+} 0$$

Implies a "decision procedure" for ideal membership

Buchberger's Algorithm Computes a Gröbner Basis

Buchberger's Algorithm

INPUT : $F = \{f_1, \ldots, f_s\}$

OUTPUT : $G = \{g_1, \ldots, g_t\}$

$G := F$;

REPEAT

  $G' := G$

  For each pair $\{f, g\}, f \neq g$ in G' DO

    $S(f, g) \xrightarrow{G'}_{+} r$

    IF $r \neq 0$ THEN $G := G \cup \{r\}$

UNTIL $G = G'$

$$S(f, g) = \frac{L}{lt(f)} \cdot f - \frac{L}{lt(g)} \cdot g$$

$L = \text{LCM}(lm(f), lm(g))$, $lm(f)$: leading monomial of f

Gröbner Bases: Applications

- A GB can be "reduced" by eliminating redundant elements
- A reduced GB is a canonical representation of the ideal

Theorem (Weak Nullstellensatz)

Given ideal $J \subset \mathbb{F}[x_1, \ldots, x_n]$, $V_{\overline{\mathbb{F}}}(J) = \emptyset \iff reducedGB(J) = \{1\}$.

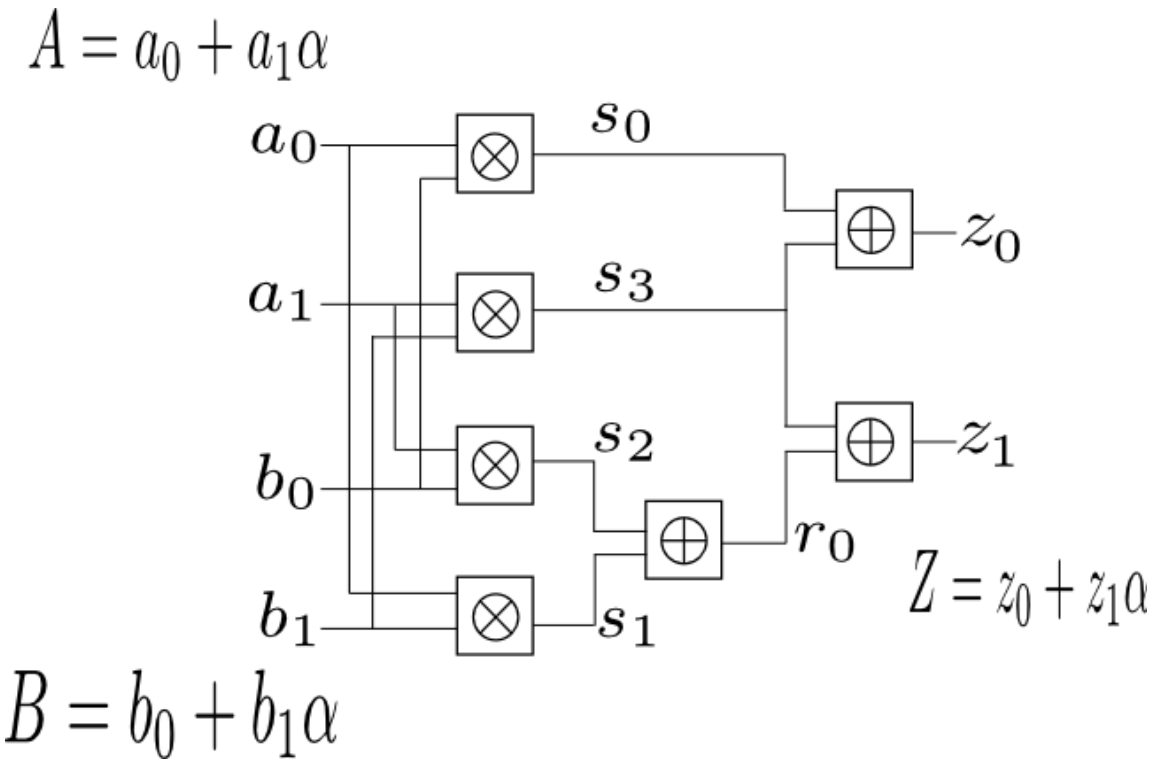
Theorem (Finite Variety)

$V_{\overline{\mathbb{F}}}(J)$ *is finite* $\iff \forall x_i, \exists g_i \in GB \; s.t. g_i = x_i^{\nu} + tail(g_i)$

A bound on the number of solutions can also be found by analyzing $GB(J)$.

Application of GB to Equivalence Checking

Given: $\mathbb{F}_{2^k} \equiv \mathbb{F}_2[x] \pmod{P(x)}, P(\alpha) = 0$
Spec: $Z = A \cdot B$, or polynomial $f : Z + A \cdot B$
Prove: *Spec* $\equiv$ *Circuit*, or find counter-example



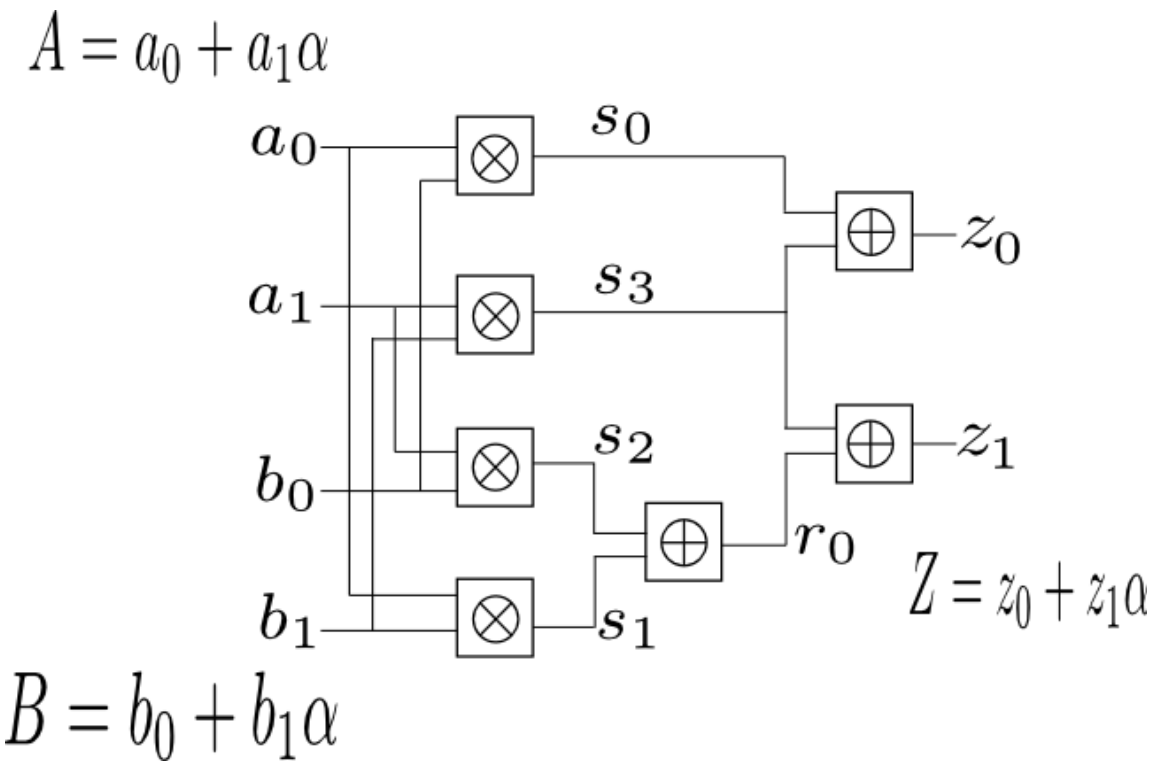
Verification: The Mathematical Problem

- Given specification polynomial: $f : Z = A \cdot B \pmod{P(x)}$ over $\mathbb{F}_{2^k}$, for given k, and given $P(x)$, s.t. $P(\alpha) = 0$
- Given circuit implementation C
  - Primary inputs: $A = \{a_0, \ldots, a_{k-1}\}, B = \{b_0, \ldots, b_{k-1}\}$
  - Primary Output $Z = \{z_0, \ldots, z_{k-1}\}$
  - $A = a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_{k-1}\alpha^{k-1}$
  - $B = b_0 + b_1\alpha + \cdots + b_{k-1}\alpha^{k-1}$, $Z = z_0 + z_1\alpha + \cdots + z_{k-1}$
- Does the circuit C correctly compute specification f?

Mathematically:

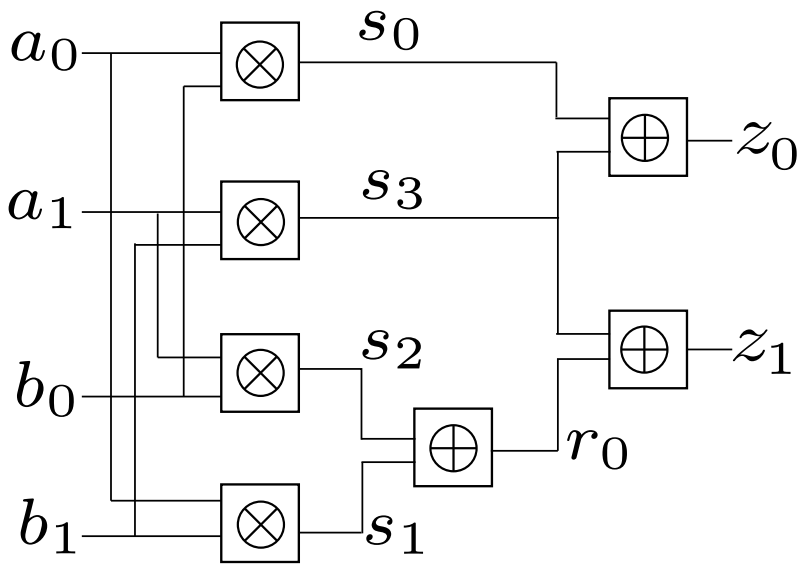
- Model the circuit (gates) as polynomials $\{f_1, \ldots, f_s\} \in \mathbb{F}_{2^k}[x_1, \ldots, x_d]$
- Do solutions to $f = 0$ (spec) agree with solutions to $f_1 = f_2 = \cdots = f_s = 0$ (implementation)?
- Does f vanish on the Variety $V(f_1, \ldots, f_s)$?

Problem Formulation



- Model logic gates as polynomials $f_1, \ldots, f_s$; ideal $J = \langle f_1, \ldots, f_s \rangle$; Spec: $f : Z = A \times B$
- Spec f "agrees with" all solutions to $f_1 = \cdots = f_s = 0$
- In other words, f vanishes on $V_{\mathbb{F}_{2^k}}(J)$

Problem Formulation



Model circuit as polynomials in $\mathbb{F}_2 \subset \mathbb{F}_{2^k}$:

$$z_0 = s_0 + s_3 \quad \Longrightarrow \quad f_1 : z_0 + s_0 + s_3$$
$$s_0 = a_0 \cdot b_0 \quad \Longrightarrow \quad f_2 : s_0 + a_0 \cdot b_0$$
$$\vdots$$
$$A + a_0 + a_1\alpha, \quad B + b_0 + b_1\alpha, \quad Z + z_0 + z_1\alpha$$

Include Vanishing Polynomials

- Over Galois fields $\mathbb{F}_q$, the polynomial $x^q - x = 0$
- $V(x^q - x) = \mathbb{F}_q$
- Generate the ideal of vanishing polynomials
  $J_0 = \langle x_1^q - x_1, \ldots, x_n^q - x_1 \rangle$
- f vanishes on $V(J)$, then $f \in I(V(J)) = J + J_0$ over $\mathbb{F}_q$
- Ideal membership test: Is $f \in (J + J_0)$?
  - Compute $G = GB(J + J_0) = \{g_1, \ldots, g_t\}$
  - Reduce $f \xrightarrow{g_1,\ldots,g_t}_+ r$ and check if $r = 0$
- Bottleneck: Complexity of computing $GB(J + J_0)$ is $q^{O(n)}$
  - For 40+ bit circuits, $GB(J + J_0)$ is infeasibly large.

How to overcome this complexity?

Complexity of Gröbner Basis and Term Orderings

- GB computation very sensitive to **term ordering**
- A term order has to be imposed for systematic polynomial manipulation

Let $f = 2x^2yz + 3xy^3 - 2x^3$

- LEX $x > y > z$: $f = -\mathbf{2x^3} + 2x^2yz + 3xy^3$
- DEGLEX $x > y > z$: $f = \mathbf{2x^2yz} + 3xy^3 - 2x^3$
- DEGREVLEX $x > y > z$: $f = \mathbf{3xy^3} + 2x^2yz - 2x^3$

Recall, S-polynomial depends on term ordering:

$$S(f,g) = \frac{L}{lt(f)} \cdot f - \frac{L}{lt(g)} \cdot g; \qquad L = \text{LCM}(lm(f), lm(g))$$

Effect of Term Orderings

Lemma

Product Criteria: *If* $lm(f) \cdot lm(g) = LCM(lm(f), lm(g))$, *then* $S(f,g) \xrightarrow{G'}_+ 0$.

LEX: $x_0 > x_1 > x_2 > x_3$

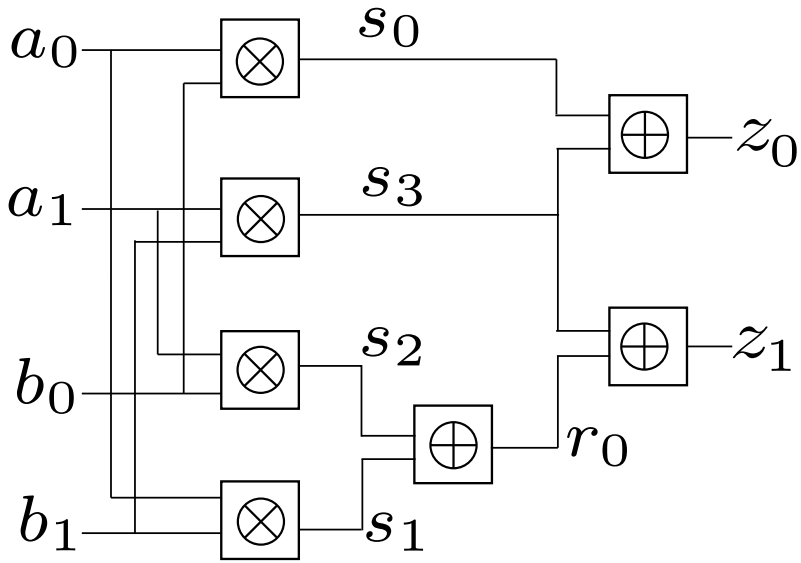
- $f = x_0x_1 + x_2$, $g = x_1x_2 + x_3$
- $lm(f) = x_0x_1$; $lm(g) = x_1x_2$; $S(f,g) \xrightarrow{G'}_+ x_0x_3 + x_2^2$

LEX: $x_3 > x_2 > x_1 > x_0$

- $f = x_2 + x_0x_1$, $g = x_3 + x_1x_2$
- $lm(f) = x_2$; $lm(g) = x_3$, $S(f,g) \xrightarrow{G'}_+ 0$

Trick: Find a term order that makes ALL $\{lm(f), lm(g)\}$ relatively prime. Then ALL $S(f,g) \xrightarrow{G}_+ 0$, so $G = GB$. already a GB!

For circuits, such an order can be derived



$$f_1 : s_0 + a_0 \cdot b_0, \ lm = s_0; \quad f_2 : s_1 + a_0 \cdot b_1, \ lm = s_1$$

$$f_3 : s_2 + a_1 \cdot b_0, \ lm = s_2; \quad f_4 : s_3 + a_1 \cdot b_1, \ lm = s_3$$

$$f_5 : r_0 + s_1 + s_2, \ lm = r_0; \quad f_6 : z_0 + s_0 + s_3, \ lm = z_0$$

$$f_7 : z_1 + r_0 + s_3, \ lm = z_1$$

- Reverse Topological Traversal of the Circuit
- $\{z_0 > z_1\} > \{r_0 > s_0 > s_3\} > \{s_1 > s_2\} > \{a_0 > a_1 > b_0 > b_1\}$
- Make every gate output a leading term

Our Discovery: Gröbner Basis of $J + J_0$

Using Our Topological Term Order:

- $F = \{f_1, \ldots, f_s\}$ is a Gröbner Basis of $J = \langle f_1, \ldots, f_s \rangle$
- $F_0 = \{x_1^q - x_1, \ldots, x_n^q - x_n\}$ is also a Gröbner basis of J_0
- But we have to compute a Gröbner Basis of $J + J_0 = \langle f_1, f_2 \ldots, f_s, \quad x_1^q - x_1, \ldots, x_n^q - x_n \rangle$
- We show that $\{f_1, f_2 \ldots, f_s, \quad x_1^q - x_1, \ldots, x_n^q - x_n\}$ is a Gröbner basis!!
- Every polynomial derived from the circuit $f_i = x_i + tail(f_i) = x_i + P_i$
- The only polynomial pairs to consider: $(f_i = x_i + P_i, \quad x_i^q - x_i)$

$$S(f_i, \quad x_i^q - x_i) = P_i x_i^{q-1} + x_i \xrightarrow{J}_+ P^q - P \xrightarrow{J_0}_+ 0$$

Conclusion: Our term order makes $\{f_1, \ldots, f_s, x_1^q - x_1, \ldots, x_n^q - x_n\}$ a Gröbner Basis [Lv TCAD'13]

The Overall Approach

- Given the circuit, perform reverse topological traversal
  - LEX: Primary outputs $>$ internal variables $>$ primary inputs
- Derive a term order to represent the polynomials for every gate
- The set: $\{F, F_0\} = \{f_1, \ldots, f_s, \quad x_1^q - x_1, \ldots, x_n^q - x_n\}$ is a Gröbner Basis
- Obtain: $f \xrightarrow{F, F_0}_+ r$
- If $r = 0$, the circuit is correct
- If $r \neq 0$, then r contains only the primary input variables
- Any SAT assignment to $r \neq 0$ generates a counter-example
- Counter-example found in no time as r is simplified by Gröbner basis reduction

Improve GB-reduction: F_4-style reduction

New algorithm to compute a Gröbner basis by J.C. Faugère: F_4

- Buchberger's algorithm $S(f,g) \xrightarrow{G}_+ r$
- Instead, F_4 computes a "set" of $S(f,g)$ in one-go
- Reduces them "simultaneously"
- Significant speed-up in computing a Gröbner basis
- Models the problem using sparse linear algebra
- Gaussian elimination on a matrix representation of the problem

We already have a Gröbner basis. We only need F_4-style reduction: $f \xrightarrow{F,F_0}_+ r$

F_4-style reduction

- Spec: $f : Z + A \cdot B$, compute $f \xrightarrow{f_1,\ldots,f_s}_+ r$
- Find a polynomial f_i that divides f, or "cancels" $LT(f)$
- Construct a matrix: rows = polynomials, columns = monomials, entries = coefficient of monomial present in the polynomial

$$
\begin{array}{c}
 \\
f \\
f_3 \\
Bf_1 \\
a_0 f_2 \\
a_1 f_2 \\
f_5 \\
f_6 \\
f_4
\end{array}
\begin{array}{c}
\begin{array}{ccccccccccc}
Z & AB & Ba_0 & Ba_1 & z_0 & z_1 & r_0 & a_0 b_0 & a_0 b_1 & a_1 b_0 & a_1 b_1
\end{array} \\
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & \alpha & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & \alpha & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & \alpha \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

F_4-style reduction

- Construct the Matrix for polynomial reduction
- Apply Gaussian elimination on the matrix
- Last row = result of reduction $= \alpha^2 + \alpha + 1 = 0$

$$
\begin{array}{c}
\begin{array}{ccccccccccc}
Z & AB & Ba_0 & Ba_1 & z_0 & z_1 & r_0 & a_0b_0 & a_0b_1 & a_1b_0 & a_1b_1
\end{array} \\
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & \alpha & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \alpha & 1 & \alpha & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \alpha & 1 & \alpha & 0 & 1 & \alpha & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & \alpha & 0 & 1 & \alpha & \alpha & \alpha^2 \\
0 & 0 & 0 & 0 & 0 & \alpha & 0 & 0 & \alpha & \alpha & \alpha^2 + 1 \\
0 & 0 & 0 & 0 & 0 & 0 & \alpha & 0 & \alpha & \alpha & \alpha^2 + \alpha + 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha^2 + \alpha + 1
\end{pmatrix}
\end{array}
$$

Experimental Results

Table: Runtime for verifying bug-free and buggy Montgomery multipliers.

Operand size k	32	48	64	96	128	163
#variables	1194	2280	4395	6562	14122	91246
#polynomials	1130	2184	4267	6370	13866	89917
#terms	10741	18199	40021	55512	134887	484738
Bug-free (F_4)	0.86	4.47	10.11	700.59	4539	18374
Bugs (F_4)	0.88	4.49	10.12	709.03	4564	17803

- Time to prove equality $\approx$ Time to catch bugs
- Bit-level circuits $\Longrightarrow$ polynomials of low degree $\Longrightarrow$ exploit sparsity
- Move the complexity from computing a Gröbner basis to polynomial division

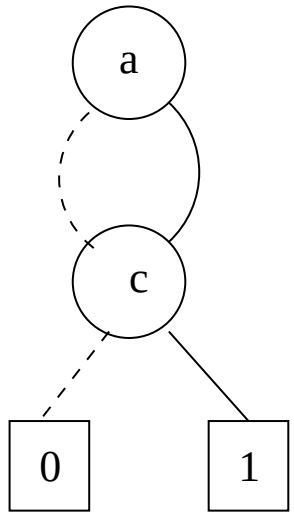
Other Related Work

- Refs. [Wienand CAV'08, Wedler DATE'11]: Similar approach for verification of integer modulo-multipliers in an SMT-framework (STABLE)
  - Works over rings $\mathbb{Z}_{2^k}$ or $\mathbb{Z}_{2^k}[X]/\langle x^2 - x : x \in X\rangle$
  - Property p, circuit with data-path (arithmetic) and control logic
  - Boolean variable assignments create proof-goals for the arithmetic sub-circuits: $f \xrightarrow{J}_{+} r$
  - Term ordering to represent ideal J as a Gröbner basis
- Ref. [Lvov FMCAD'12]: BLUEVERI tool from IBM for verification of Error Correcting Circuits
  - Galois field circuits designed over $GF(2^k) = GF((2^m)^n)$
  - For $k = 512$, circuit is a 16-word interconnection of 32-bit blocks $GF((2^{32})^{16})$
  - Polynomials have high degrees, compute a Gröbner basis
  - GB: $J = \{x + tail\}$, the reduction $x^n \pmod J \sim tail^n$

Boolean Gröbner Basis

PolyBori [Brickenstein JSC'09]: Compute Gröbner basis over Boolean rings $\mathbb{Z}_2[x_1, \ldots, x_n]$

- Use decision diagrams (ZDDs) to represent Boolean polynomials
- $f = ac + c$, paths in a ZDD for sets: $\{a, c\}$ and $\{c\}$
- Addition and multiplication implemented as DAG compositions



Gröbner Bases and CNF-SAT

Origins in [CEI STOC'96], but also in [Kapur SIGSOFT'85]: term rewriting view of GB for proving theorems in first-order predicate calculus
Solving SAT using Gröbner bases:

- Model clauses as polynomials $J = \langle f_1, \ldots, f_s \rangle$ in $\mathbb{F}_2$, and apply Weak Nullstellensatz
- If reduced $GB(\langle J + J_0 \rangle) = \{1\}$, UNSAT
- Otherwise, there are finite solutions (#SAT), and GB is infeasibly large
- Other approaches: constraint-mining, preprocessing [Condrat TACAS'07], Clause Learning [Zengler CASC'10], Quantifier Elimination [Gao ICAI'11]
- In Buchberger's $S(f, g) \xrightarrow{G}_+ r$, $r =$ new constraints
- How helpful/useful in SAT solving are these new constraints? Can we learn conflict clauses?

Gröbner Bases can learn conflict clauses

- CNF: $(y \vee f)(f' \vee g)(f' \vee g' \vee h')(u' \vee m' \vee h)(u' \vee f' \vee m)$
- Learned clauses from GB: $(y \vee u')(y \vee g)(y \vee h')(u' \vee f')(f' \vee h')$

Other issues: Term ordering and variable activity:

- High-activity variables should be lower in the order for smaller GB
- $(a' \vee b' \vee c)(a \vee b' \vee c')(a' \vee d' \vee e)(a \vee d' \vee e')$
- With LEX $a > e > c > d > b$, GB: $ab + cb, ad + ed, edb + cdb$
- With LEX $a < e < c < d < b$, GB: $de + da, bc + ba$

Hard to make strong conclusions, because GB cannot solve most benchmarks. Around 2006-2007, industrial benchmark suite: solved no more than 5 Satisfiable instances!

GB + SAT: Is there any hope?

- Interesting from a complexity theory perspective, not from a practical standpoint
- CDCL-Solvers can handle very large problems, Gröbner engines cannot
- $S(f, g)$ selection strategies need to be targeted for CNF-SAT
- Problems of academic interest:
  - Invariant mining from bit-level circuits for verification: Compute GB with LEX internal vars $>$ outputs $>$ inputs
  - Minimal UNSAT core extraction from reduced GB

Conclusions: GB + SAT/SMT

- Leverage Gröbner Bases and SAT/SMT-solvers for design verification
- Use of domain-specific solvers can be fruitful, problem encoding is important
- Yesterday, Jakob said:
  - Promise of performance [on polynomial calculus] failed to deliver
  - Full Gröbner basis does too much work, some short-cut is needed
- For circuits/RTL, circumvent the need for GB computation (term ordering)
- Practical only for low-degree, sparse constraints
- For CNF-solving, GB more suited for UNSAT problems

References I

[Lv TCAD'13] J. Lv, P. Kalla, F. Enescu, "Efficient Gröbner Basis Reductions for Formal Verification of Galois Field Arithmetic Circuits", IEEE Trans. on CAD, Sept 2013

[Wienand CAV'08] O. Wienand, M. Wedler, D. Stoffel, W. Kunz, G. Gruel, "An algebraic approach to proving data correctness in arithmetic data-paths", Computer-Aided Verification, CAV 2008.

[Wedler DATE'11] E. Pavlenko, M. Wedler, D. Stoffel, W. Kunz, "STABLE: A new QF-BV SMT Solver for hard Verication Problems combining Boolean Reasoning with Computer Algebra", Design Auto. & Test in Europe, 2011.

[Lvov FMCAD'12] A. Lvov, L. Lastras-Montano, V. Paruthi, R. Shadowen, "Formal Verification of Error Correcting Circuits using Computation Algebraic Geometry", FMCAD 2012.

References II

[Brickenstein JSC'09] M. Brickenstein, Alexander Dryer, "PolyBoRi: A framework for Gröbner-basis computations with Boolean polynomials" Journal of Symbolic Computation.

[CEI STOC'96] M. Clegg, J. Edmunds, R. Impagliazzo, "Using Gröbner basis algorithm to find proofs of unsatisfiability", Symp. Theory of Computing 1996.

[Kapur SIGSOFT'85] D. Kapur, P. Narendran, "An equational approach to theorem proving in first-order predicate calculus", ACM SIGSOFT Notes 10(4), 1985.

[Condrat TACAS07] C. Condrat, P. Kalla, "A Gröbner Basis Approach to CNF Formulae Preprocessing", TACAS 2007.

References III

[Zengler CASC'10] C. Zengler and W. Küchlin, "Extending Clause Learning of SAT Solvers with Boolean Grobner Bases", Computer Algebra in Scientific Computation, 2010.

[Gao ICAI'11] S. Gao, A. Platzer, E. Clarke, "Quantifier Elimination over Finite Fields with Gröbner Bases", Intl. Conf. on Algebraic Informatics 2011.