Exploring Algebraic Interpolants for Rectification of Finite Field Arithmetic Circuits with Gröbner Bases

Utkarsh Gupta[1], Priyank Kalla[1], Irina Ilioaea[2], and Florian Enescu[2]
[1]Electrical and Computer Engineering, University of Utah, Salt Lake City UT, USA
[2]Mathematics and Statistics, Georgia State University, Atlanta GA, USA

***Abstract*—When formal verification identifies the presence of a bug in a design, it is required to rectify the circuit at some net(s). Modern approaches formulate the rectification test as an unsatisfiability proof, and then use Craig interpolants (CI) in propositional logic to compute the corresponding rectification functions. Boolean SAT and CI engines are infeasible in rectification of finite field arithmetic circuits, where polynomial algebra is more suitable. Recently, it was shown that CI exist in polynomial algebra in finite fields. This paper presents a detailed theory and algorithms for CI in finite fields, and characterizes the lattice of all algebraic interpolants. Using the Gröbner basis algorithm, we present techniques to traverse the interpolant lattice. This allows to explore various interpolants for efficient synthesis of rectification functions for finite field arithmetic circuits. Experimental results are presented that demonstrate the efficacy of our approach.**

I. Introduction

Formal verification of arithmetic circuits checks whether a circuit netlist correctly implements a given specification model. When the presence of a bug is detected in the design, the task of rectification needs to be performed. It is required to identify a set of nets where the circuit can be rectified, and to compute the corresponding rectification functions. Contemporary debugging and rectification approaches [1] [2] [3] model the problem as one of *partial synthesis*, where: i) the rectification test is first formulated as a Boolean SAT problem [3]; ii) the computation/synthesis of the rectification function is formulated as a Quantified Boolean Formula (QBF) solving problem [1] [2]; and iii) the rectification function is practically synthesized using either iterative SAT solving [2] [4], or *Craig Interpolants* (CI) [5] in propositional logic [3].

The aforementioned approaches have become quite adept at debugging and rectification of erroneous circuits – as well as for synthesizing engineering change order (ECO) patches. However, they are mostly suitable for control dominated applications. These techniques are infeasible for debug and rectification of large non-linear arithmetic circuits, as the Boolean SAT/QBF/interpolation model is infeasible for arithmetic circuits. To overcome these limitations, polynomial models and algorithms from *symbolic computer algebra* have been recently proposed for rectification of buggy arithmetic circuits [6] [7] [8]. The approach of [8] addresses integer arithmetic circuits. However, their algorithm is incomplete and incorrect, as it cannot always compute a rectification function; interested readers may refer to [9] for an explanation and a counter-example to their claims.

In our recent publication [6], we presented an approach to debug and rectify arithmetic circuits that implement polynomial computations over finite fields of 2^k elements, denoted $\mathbb{F}_q$, where $q = 2^k$. Given a polynomial specification and a buggy finite field arithmetic circuit (with no assumptions on the number or the type of bugs), we presented an algebraic approach that ascertains whether the buggy circuit can be rectified at only one particular given net x_i – i.e. we addressed *single-fix rectification*. The problem was formulated using the *Weak Nullstellensatz in* $\mathbb{F}_q$ [10], and solved using the *Gröbner basis* (GB) algorithm [11]. Subsequently, we showed that CI exist in polynomial algebra over finite fields. We further showed how to compute an interpolant using the GB algorithm, and how to compute a rectification function from an algebraic interpolant.

In the above setting [6], algebraic interpolants correspond to *varieties* – subsets of the n-dimensional affine space $\mathbb{F}_q^n$ – and are represented by *polynomial ideals,* or more precisely, by a Gröbner basis of corresponding ideals. While our approach in [6] has shown promise in the rectification of large arithmetic circuits, the presented theory of algebraic CI is incomplete in many ways: Firstly, only the existence of an interpolant in finite fields was demonstrated, and a detailed proof was not provided. Moreover, there can be many (though a finite number of) interpolants in $\mathbb{F}_q$, each of which may correspond to rectification functions of various synthesis costs in terms of area and delay of the circuit. It is desirable to characterize and explore the *lattice of all algebraic interpolants*, thus exploring rectification functions of various implementation costs.

Contributions: This paper presents a complete theory of Craig Interpolants in finite fields $\mathbb{F}_q$, with applications to logic synthesis of rectification functions for buggy finite field arithmetic circuits. Based on the concept of interpolants in polynomial algebra over finite fields $\mathbb{F}_q$ presented in [6], this paper makes the following new contributions: 1) Formally prove the existence of CI in $\mathbb{F}_q$. 2) Derive the relationship of interpolants with elimination ideals, and show how to compute them using Gröbner bases. 3) Compute the *smallest* interpolant, i.e. the one contained in every other interpolant. Analogously, compute the *largest* interpolant, i.e. the one containing all other interpolants. 4) Count the total number of all possible interpolants. 5) We show how to enumerate all interpolants so as to synthesize various rectification functions with different area/delay costs of the implemented network. 6) Present experimental results on finite field circuits used in cryptography to demonstrate the efficacy of our approach.

Paper Organization: The following Section reviews the preliminary background. Section III reviews the results of [6] on rectifiability of buggy circuits. Section IV describes the complete theory of CI in $\mathbb{F}_q$. Experiments are described in Section V and Section VI concludes the paper.

This research is funded in part by the US National Science Foundation grants CCF-1619370 and CCF-1320385.

II. Preliminaries: Notation & Background

Let $\mathbb{F}_q$ denote the finite field of q elements where $q = 2^k$ and k is the operand width. Let $R = \mathbb{F}_q[x_1,\dots,x_n]$ be the polynomial ring in n variables $x_1,\dots,x_n$, with coefficients from $\mathbb{F}_q$. A monomial m_1 is a power product of variables $x_1^{e_1} \cdot x_2^{e_2} \cdots x_n^{e_n}$, where $e_i \in \mathbb{Z}_{\geq 0}, i \in \{1,\dots,n\}$. A *polynomial* $f \in R$ is written as a finite sum of terms $f = c_1m_1 + c_2m_2 + \cdots + c_tm_t$, where $c_1,\dots,c_t$ are coefficients and $m_1,\dots,m_t$ are monomials. A monomial order $>$ (or a term order) is imposed on the ring so that the monomials of all polynomials $f = c_1m_1 + c_2m_2 + \cdots + c_tm_t$ are ordered w.r.t. $>$, such that $m_1 > m_2 > \cdots > m_t$. Subject to $>$, $lt(f) = c_1m_1,\ lm(f) = m_1,\ lc(f) = c_1$, are the *leading term*, *leading monomial* and *leading coefficient* of f, respectively. In this work, we employ lexicographic (lex) term orders (see Definition 1.4.3 in [11]).

We model the given circuit C by a set of multivariate polynomials $f_1,\dots,f_s \in \mathbb{F}_{2^k}[x_1,\dots,x_n]$; here $x_1,\dots,x_n$ denote the nets (signals) of the circuit. Every Boolean logic gate of C is represented by a polynomial in $\mathbb{F}_2$, as $\mathbb{F}_2 \subset \mathbb{F}_{2^k}$. This is shown below. Note that in $\mathbb{F}_{2^k}$, $-1 = +1$.

$$\begin{aligned}
z &= \neg a \rightarrow z + a + 1 \pmod 2\\
z &= a \wedge b \rightarrow z + a\cdot b \pmod 2\\
z &= a \vee b \rightarrow z + a + b + a\cdot b \pmod 2\\
z &= a \oplus b \rightarrow z + a + b \pmod 2
\end{aligned} \tag{1}$$

Given a set of polynomials $F = \{f_1,\dots,f_s\}$ in R, the *ideal* $J \subseteq R$ generated by them is: $J = \langle f_1,\dots,f_s\rangle = \{\sum_{i=1}^{s} h_i \cdot f_i : h_i \in R\}$. The polynomials $f_1,\dots,f_s$ form the *generators* of J.

Let $\boldsymbol{a} = (a_1,\dots,a_n) \in \mathbb{F}_q^n$ be a point in the affine space, and f a polynomial in R. If $f(\boldsymbol{a}) = 0$, we say that f *vanishes* on $\boldsymbol{a}$. The *set of all common zeros* of the polynomials of F that lie within the field $\mathbb{F}_q$ is called the *variety*. Variety depends on the ideal generated by the polynomials. We denote it by $V_{\mathbb{F}_q}(J) = V(J) = V(f_1,\dots,f_s) = \{\boldsymbol{a} \in \mathbb{F}_q^n : \forall f \in J, f(\boldsymbol{a}) = 0\}$.

Given two ideals $J_1 = \langle f_1,\dots,f_s\rangle, J_2 = \langle h_1,\dots,h_r\rangle$, the sum $J_1 + J_2 = \langle f_1,\dots,f_s,h_1\dots,h_r\rangle$, and their product $J_1 \cdot J_2 = \langle f_i \cdot h_j : 1 \leq i \leq s, 1 \leq j \leq r\rangle$. Ideals and varieties are dual concepts: $V(J_1 + J_2) = V(J_1) \cap V(J_2)$, and $V(J_1 \cdot J_2) = V(J_1) \cup V(J_2)$. Moreover, if $J_1 \subseteq J_2$ then $V(J_1) \supseteq V(J_2)$.

Gröbner Basis of Ideals: An ideal may have many different sets of generators: $J = \langle f_1,\dots,f_s\rangle = \cdots = \langle g_1,\dots,g_t\rangle$ such that $V(J) = V(f_1,\dots,f_s) = \cdots = V(g_1,\dots,g_t)$. Given a non-zero ideal J, a *Gröbner basis* (GB) for J is a finite set of polynomials $G = \{g_1,\dots,g_t\}$ satisfying $\langle\{lm(f) \mid f \in J\}\rangle = \langle lm(g_1),\dots,lm(g_t)\rangle$. Then $J = \langle G\rangle$ holds and so $G = GB(J)$ forms a basis for J. Buchberger's algorithm (see Alg. 1.7.1 in [11]) takes as input the set of polynomials $F = \{f_1,\dots,f_s\}$ and computes the GB $G = \{g_1,\dots,g_t\}$. A GB can be *reduced* to eliminate redundant polynomials from the basis. A reduced GB is a canonical representation of the ideal.

Algebraic Miter for Equivalence Checking: Given f_{spec} as the specification polynomial and C as a given circuit implementation, we need to construct an *algebraic miter* between f_{spec} and C. For equivalence checking, we need to prove that the miter is infeasible. Fig. 1 depicts how a word-level algebraic miter is setup. Suppose that $A = \{a_0,\dots,a_{k-1}\}$ and $Z = \{z_0\dots,z_{k-1}\}$ denote the k-bit primary inputs and outputs of the finite field circuit. Then $A = \sum_{i=0}^{k-1} a_i\alpha^i, Z = \sum_{i=0}^{k-1} z_i\alpha^i$ correspond to the word-level polynomials for the inputs and outputs of the circuit. Here α is the primitive element of $\mathbb{F}_{2^k}$. Let Z_s be the word-level output for f_{spec}, which computes some polynomial function $\mathcal{F}(A)$ of A, so that $f_{spec} : Z_s + \mathcal{F}(A)$. The word-level outputs Z, Z_s are mitered to check if for all inputs, $Z \neq Z_s$ is infeasible.

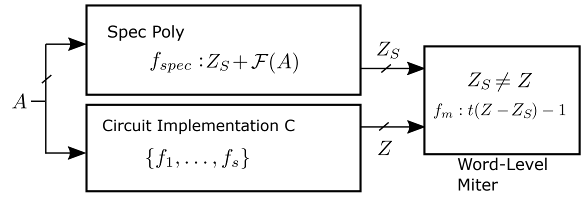

Fig. 1: Word-Level Miter

In finite fields, the disequality $Z \neq Z_s$ can be modeled as a single polynomial f_m, called the miter polynomial, where $f_m = t\cdot(Z - Z_s) - 1$, and t is introduced as a free variable. The idea behind f_m is that when $Z \neq Z_s$ for some input, then $f_m = 0$ has solutions where $t = (Z - Z_s)^{-1}$. When $Z = Z_s$, no value of t satisfies $f_m = 0$ as $t \cdot 0 \neq 1$. In this way, equivalence checking using the algebraic model is solved as follows: Construct an ideal $J = \langle f_{spec}, f_1,\dots,f_s,f_m\rangle$ for the miter as described above. Then determine if the variety $V(J) = \emptyset$? If $V(J) = \emptyset$, the miter is infeasible, and C implements f_{spec}. Otherwise, there exists a bug in the design.

The Weak Nullstellensatz: To ascertain whether $V(J) = \emptyset$, we employ the Weak Nullstellensatz over $\mathbb{F}_q$. For all elements $\alpha \in \mathbb{F}_q, \alpha^q = \alpha$. Therefore, the polynomial $x^q - x$ vanishes everywhere in $\mathbb{F}_q$, and is called the vanishing polynomial of the field. Let $J_0 = \langle x_1^q - x_1,\dots,x_n^q - x_n\rangle$ be the ideal of all vanishing polynomials in R.

Theorem II.1 (*The Weak Nullstellensatz over finite fields (from Theorem 3.3 in [12])*)**.** *For a finite field $\mathbb{F}_q$ and the ring $R = \mathbb{F}_q[x_1,\dots,x_n]$, let $J = \langle f_1,\dots,f_s\rangle \subseteq R$, and let $J_0 = \langle x_1^q - x_1,\dots,x_n^q - x_n\rangle$ be the ideal of vanishing polynomials. Then $V(J) = \emptyset \iff 1 \in J + J_0 \iff GB(J + J_0) = \{1\}$.*

Thus, for equivalence check, we compute a reduced GB $G = GB(J + J_0)$ corresponding to the miter, and see if $G = \{1\}$. If $G \neq \{1\}$, then there exists a bug in the design, which has to be rectified. Note that when $G = GB(J + J_0) \neq \{1\}$, the variety $V_{\mathbb{F}_q}(J)$ is finite. Consequently, Gröbner bases allow to count the number of solutions $|V_{\mathbb{F}_q}(J)|$. For a GB G, let $LM(G)$ denote the set of leading monomials of all elements of G: $LM(G) = \{lm(g_1),\dots,lm(g_t)\}$.

Definition II.1 (*Standard Monomials*)**.** Let $\boldsymbol{X^e} = x_1^{e_1}\cdots x_n^{e_n}$ denote a monomial. The set of standard monomials of G is defined as $SM(G) = \{\boldsymbol{X^e} : \boldsymbol{X^e} \notin \langle LM(G)\rangle\}$.

Theorem II.2 (*Counting the number of solutions (Theorem 3.7 in [12])*)**.** *Let $G = GB(J + J_0)$, and $|SM(G)| = m$, then the ideal J vanishes on m distinct points in $\mathbb{F}_q^n$. In other words, $|V(J)| = |SM(G)|$.*

The Strong Nullstellensatz: Given an ideal $J \subset R$ and $V(J) \subseteq \mathbb{F}_q^n$, the *ideal of polynomials that vanish on $V(J)$* is $I(V(J)) = \{f \in R : \forall \boldsymbol{a} \in V(J), f(\boldsymbol{a}) = 0\}$. If $I_1 \subset I_2$ are ideals then $V(I_1) \supset V(I_2)$, and if $V_1 \subset V_2$ are varieties, then $I(V_1) \supset I(V_2)$. The Strong Nullstellensatz [13] describes $I(V(J))$.

Theorem II.3 (*The Strong Nullstellensatz over finite fields (Theorem 3.2 in [10])*)**.** For any ideal $J \subset \mathbb{F}_q[x_1,\dots,x_n]$, $I(V_{\mathbb{F}_q}(J)) = J + J_0$.

Projection of varieties and elimination ideals: Given an ideal $J = \langle f_1,\dots,f_s\rangle \subset R$ and its variety $V(J) \subset \mathbb{F}_q^n$, the l-th projection of $V(J)$ denoted as $Pr_l(V(J))$ is the mapping $Pr_l(V(J)) : \mathbb{F}_q^n \to \mathbb{F}_q^{n-l}$, $Pr_l(a_1,\dots,a_{l+1},\dots,a_n) = (a_{l+1},\dots,a_n)$, for every $\boldsymbol{a} = (a_1,\dots,a_n) \in V(J)$. Projections of varieties are related to elimination ideals, used extensively in this paper.

Definition II.2. Given an ideal $J \subset \mathbb{F}_q[x_1,\dots,x_n]$, the l-th elimination ideal J_l is an ideal in R defined as $J_l = J \cap \mathbb{F}_q[x_{l+1},\dots,x_n]$. Moreover, given an ideal $J \subset R$ and its GB G *w.r.t.* the lexicographical (lex) order on the variables where $x_1 > x_2 > \cdots > x_n$, then for every $0 \le l \le n$ we denote by G_l the GB of l-th elimination ideal of J and compute it as: $G_l = G \cap \mathbb{F}_q[x_{l+1},\dots,x_n]$.

Lemma II.1 (Lemma 3.4 in [10])**.** Given an ideal $J \subset R$ that contains the vanishing polynomials of the field (i.e. $J \supset J_0$), then $Pr_l(V(J)) = V(J_l)$, i.e. the l-th projection of the variety of ideal J is equal to the variety of its l-th elimination ideal.

III. Rectification and Craig Interpolants in $\mathbb{F}_q$

Based on the model of Fig. 1, suppose that equivalence checking detects the presence of a bug in the design. Suppose further that subsequent diagnosis reveals a subset $\mathcal{X}$ of nets of C where the circuit might be rectified. Rectification at net $x_i \in \mathcal{X}$ implies that there exists a (hitherto unknown) polynomial function $U(X_{PI})$ s.t. $x_i = U(X_{PI})$, where $X_{PI} \subset \{x_1,\dots,x_n\}$ is the set of primary inputs of C. To ascertain single-fix rectifiability at x_i, [6] presented the following result:

Represent the rectification function at target net x_i by a polynomial $f_i : x_i + U(X_{PI})$. Construct the ideal J from the miter, with the polynomials for the gates $f_1,\dots,f_s$ of the circuit where $f_i = x_i + U(X_{PI})$, the specification polynomial f_{spec}, and the miter polynomial f_m, so that $J = \langle f_{spec}, f_1,\dots,f_i : x_i + U(X_{PI}),\dots,f_s,f_m\rangle$.

Theorem III.1 (Rectification Theorem, Thm III.1 in [6])**.** Construct two ideals:

- $J_L = \langle f_{spec}, f_1,\dots,f_i : x_i+1,\dots,f_s,f_m\rangle$ where $f_i : x_i + U(X_{PI})$ in J is replaced with $f_i : x_i + 1$.
- $J_H = \langle f_{spec}, f_1,\dots,f_i : x_i,\dots,f_s,f_m\rangle$ where $f_i : x_i + U(X_{PI})$ in J is replaced with $f_i : x_i + 0$.

Compute $E_L = (J_L + J_0) \cap \mathbb{F}_{2^k}[X_{PI}]$ and $E_H = (J_H + J_0) \cap \mathbb{F}_{2^k}[X_{PI}]$ to be the respective elimination ideals, where all the non-primary input variables have been eliminated. Then the circuit can be rectified with a logic function at net x_i with the polynomial function $f_i : x_i + U(X_{PI})$ to implement the specification iff $1 \in E_L + E_H$ or iff $reducedGB(E_L + E_H) = \{1\}$.

If $reducedGB(E_L + E_H) = \{1\}$, then $V(E_L) \cap V(E_H) = \emptyset$, due to the Weak Nullstellensatz (Thm. II.1), which is a polynomial analog of UNSAT checking. For UNSAT problems, the formal logic and verification communities have explored the notion of abstraction of functions by means of Craig interpolants [5]. In propositional logic, the concept is defined as follows:

Definition III.1. Let (A,B) be a pair of CNF formulae (sets of clauses) such that $A \wedge B$ is unsatisfiable. Then there exists a formula I such that: (i) $A \implies I$; (ii) $I \wedge B$ is unsatisfiable; and (iii) I refers only to the common variables of A and B, i.e. $Var(I) \subseteq Var(A) \cap Var(B)$. The formula I is called the **Craig interpolant** (CI), or interpolant for short, of (A,B).

Associating varieties of ideals $V(J_A), V(J_B)$ to Boolean functions A, B in Def. III.1, it was mentioned in [6] that there exists an ideal J_I s.t. its variety $V(J_I)$ corresponds to a CI in finite fields. Let polynomials $\{g_1,\dots,g_t\}$ represent the GB of ideal J_I. Then a polynomial function U can be computed from J_I as $U = (1+g_1)(1+g_2)\cdots(1+g_t)+1$, and $x_i = U$ serves as a corresponding rectification function. The reader may refer to Section III in [6] for full details.

In what follows, we provide a detailed theory of CI in finite fields that helps to characterize the entire interpolant lattice. This allows to explore/compute various interpolants, including the smallest and the largest ones. This, in turn, allows to synthesize various rectification functions in order to search for a low-cost implementation.

IV. Theory

We describe the setup for Craig interpolation in the ring $R = \mathbb{F}_q[x_1,\dots,x_n]$. Partition the variables $\{x_1,\dots,x_n\}$ into subsets A, B, C. We are given two ideals $J_A \subset \mathbb{F}_q[A,C]$ and $J_B \subset \mathbb{F}_q[B,C]$ such that the C-variables are common to the generators of both J_A, J_B. *From here on, we will assume that all ideals include the corresponding vanishing polynomials.* For example, generators of J_A include $\boldsymbol{A^q - A}, \boldsymbol{C^q - C}$, where $\boldsymbol{A^q - A} = \{x_i^q - x_i : x_i \in A\}$, and so on. Then these ideals become radicals and we can apply Lemma II.1. We use $V_{A,C}(J_A)$ to denote the variety of J_A over the $\mathbb{F}_q$-space spanned by A and C variables, i.e. $V_{A,C}(J_A) \subset \mathbb{F}_q^A \times \mathbb{F}_q^C$. Similarly, $V_{B,C}(J_B) \subset \mathbb{F}_q^B \times \mathbb{F}_q^C$.

Now let $J = J_A + J_B \subseteq \mathbb{F}_q[A,B,C]$, and suppose that it is found by application of the Weak Nullstellensatz (Thm. II.1) that $V_{A,B,C}(J) = \emptyset$. When we compare the varieties of J_A and J_B, then we can consider the varieties in $\mathbb{F}_q^A \times \mathbb{F}_q^B \times \mathbb{F}_q^C$, as $V_{A,B,C}(J_A) = V_{A,C}(J_A) \times \mathbb{F}_q^B \subset \mathbb{F}_q^A \times \mathbb{F}_q^B \times \mathbb{F}_q^C$. With this setup, we define the interpolants as follows.

Definition IV.1 (*Interpolants in finite fields*)**.** Given two ideals $J_A \subset \mathbb{F}_q[A,C]$ and $J_B \subset \mathbb{F}_q[B,C]$ where A, B, C denote the three sets of variables such that $V_{A,B,C}(J_A) \cap V_{A,B,C}(J_B) = \emptyset$. Then there exists an ideal J_I satisfying the following properties:

1) $V_{A,B,C}(J_I) \supseteq V_{A,B,C}(J_A)$
2) $V_{A,B,C}(J_I) \cap V_{A,B,C}(J_B) = \emptyset$
3) The generators of J_I contain only the C-variables; or $J_I \subseteq \mathbb{F}_q[C]$.

We call $V_{A,B,C}(J_I)$ the **interpolant** in finite fields for the pair $(V_{A,B,C}(J_A), V_{A,B,C}(J_B))$, and the corresponding ideal J_I the **ideal-interpolant**.

As the generators of J_I contain only the C-variables, the interpolant $V_{A,B,C}(J_I)$ is of the form $V_{A,B,C}(J_I) = \mathbb{F}_q^A \times \mathbb{F}_q^B \times V_C(J_I)$. Therefore, the subscripts A,B for the interpolant $V_{A,B,C}(J_I)$ may be dropped for the ease of readability.

We will use the following example of rectification of a modular multiplier to explain the various concepts we introduce for interpolants and how these concepts can be used to generate multiple rectification functions.

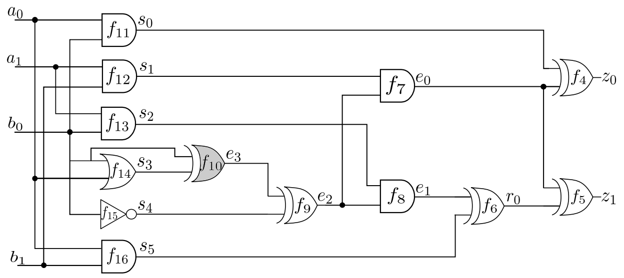

Fig. 2: A 2-bit buggy modulo multiplier implementation

Example IV.1. *Fig. 2 shows a buggy implementation of a 2-bit modulo multiplier. The gate at net e_3 should be an AND gate; instead the XOR gate generates a bug in the design. The specification polynomial is $f_{spec} : Z_S + A \cdot B \pmod{P(X)}$. We model the problem in the field $\mathbb{F}_{2^2} = \mathbb{F}_4$, as $Z = \{z_1, z_0\}$, and $A = \{a_1, a_0\}$, $B = \{b_1, b_0\}$ are the 2-bit output and input operands, respectively. $P(X) = X^2 + X + 1$ is the irreducible polynomial used to construct $\mathbb{F}_{2^2} = \mathbb{F}_2[X] \pmod{P(X)}$, with $P(\alpha) = 0$.*

The polynomials describing the circuit are given as

$$
\begin{aligned}
f_1 &: Z + z_0 + \alpha z_1; & f_9 &: e_2 + e_3 + s_4;\\
f_2 &: A + a_0 + \alpha a_1; & f_{10} &: e_3 + b_0 + s_3; \textit{[bug]}\\
f_3 &: B + b_0 + \alpha b_1; & f_{11} &: s_0 + a_0 b_0;\\
f_4 &: z_0 + s_0 + e_0; & f_{12} &: s_1 + a_1 b_1;\\
f_5 &: z_1 + e_0 + r_0; & f_{13} &: s_2 + a_1 b_0;\\
f_6 &: r_0 + e_1 + s_5; & f_{14} &: s_3 + a_0 + b_0 + a_0 b_0;\\
f_7 &: e_0 + s_1 e_2; & f_{15} &: s_4 + b_0 + 1;\\
f_8 &: e_1 + s_2 e_2; & f_{16} &: s_5 + a_0 b_1;
\end{aligned}
$$

The algebraic miter can be modeled as the polynomial $f_m = t \cdot (Z - Z_S) - 1$. We construct the ideal $J = \langle f_{spec}, f_1, \ldots, f_{16}, f_m \rangle$.

Now let's apply single-fix rectification theorem (Thm. III.1) at e_3. From J, we construct the two ideals J_L and J_H as follows,

$$
\begin{aligned}
J_L &= \langle f_{spec}, f_1, \ldots, f_9, e_3 + 1, f_{11}, \ldots, f_{16}, f_m \rangle,\\
J_H &= \langle f_{spec}, f_1, \ldots, f_9, e_3 + 0, f_{11}, \ldots, f_{16}, f_m \rangle.
\end{aligned}
$$

Next we compute the elimination ideals E_L and E_H,

$$
\begin{aligned}
E_L &= \langle b_0, b_1 + 1, a_1 + 1, a_0^2 + a_0 \rangle,\\
E_H &= \langle b_0 + 1, b_1^2 + b_1, a_1 + 1, a_0^2 + a_0 \rangle.
\end{aligned}
$$

Computing $GB(E_L + E_H)$ results in $\{1\}$ indicating that the circuit can be single-fix rectified at net e_3. As $V(E_L) \cap V(E_H) = \emptyset$, we can say that (E_L, E_H) is a possible pair for interpolation. From Def. IV.1, for $J_A = E_L$ and $J_B = E_H$, the three sets of variables A, B, C are $A = B = \emptyset, C = X_{PI}$.

Theorem IV.1. An ideal-interpolant J_I, and correspondingly the interpolant $V_{A,B,C}(J_I)$, as given in Def. IV.1, always exists.

Proof. Consider the elimination ideal $J_I = J_A \cap \mathbb{F}_q[C]$. We show J_I satisfies the three conditions for the interpolant.
Condition 1: $V_{A,B,C}(J_I) \supseteq V_{A,B,C}(J_A)$. This condition is trivially satisfied due to construction of elimination ideals. As $J_I \subseteq J_A$, $V_{A,B,C}(J_I) \supseteq V_{A,B,C}(J_A)$.
Condition 2: $V_{A,B,C}(J_I) \cap V_{A,B,C}(J_B) = \emptyset$. This condition can be equivalently stated as $V_{B,C}(J_I) \cap V_{B,C}(J_B) = \emptyset$ as neither J_I nor J_B contains any variables from the set A. We prove this condition by contradiction. Let's assume that there exists a common point $(\mathbf{b}, \mathbf{c})$ in $V_{B,C}(J_I)$ and $V_{B,C}(J_B)$. We know that the projection of the variety $Pr_A(V_{A,C}(J_A))$ is equal to the variety of the elimination ideal $V_C(J_I)$, where $J_I = J_A \cap \mathbb{F}_q[C]$, due to Lemma II.1. Therefore, the point $(\mathbf{c})$ in the variety of J_I can be extended to a point $(\mathbf{a}, \mathbf{c})$ in the variety of J_A. This implies that the ideals J_A and J_B vanish at $(\mathbf{a}, \mathbf{b}, \mathbf{c})$. This is a contradiction to our initial assumption that the intersection of the varieties of J_A and J_B is empty. Thus J_I, J_B have no common zeros.
Condition 3: The generators of J_I contain only the C-variables. This condition is trivially satisfied as J_I is the elimination ideal obtained by eliminating A-variables in J_A. □

The above theorem not only proves the existence of an interpolant, but also gives a procedure to construct its ideal: $J_I = J_A \cap \mathbb{F}_q[C]$. In other words, compute a Gröbner basis G of J_A w.r.t. the elimination order $A > B > C$ and take $G_I = G \cap \mathbb{F}_q[C]$. Then G_I gives the generators for the ideal-interpolant J_I.

Example IV.2. *For the ideals $J_A = E_L$ and $J_B = E_H$ in Example IV.1, we can compute an ideal-interpolant J_I as described in the proof above: $J_I = J_A \cap \mathbb{F}_q[C] = E_L \cap \mathbb{F}_4[X_{PI}] = E_L$, because E_L contains X_{PI} variables only. As a result, the ideal E_L is itself an ideal-interpolant. From E_L, we compute $U = 1 + (1 + b_0)(1 + (b_1 + 1))(1 + (a_1 + 1)) = a_1 b_1 b_0 + a_1 b_1 + 1$ as shown in [6]. As $(+, \cdot) \pmod 2$ are XOR, AND respectively, we obtain $e_3 = \neg((a_1 \wedge b_1 \wedge b_0) \oplus (a_1 \wedge b_1))$ as a rectification function.*

Theorem IV.2. The interpolant $V_{A,B,C}(J_S)$ corresponding to the ideal $J_S = J_A \cap \mathbb{F}_q[C]$ is the smallest interpolant.

Proof. Let $J_I \subseteq \mathbb{F}_q[C]$ be any another ideal-interpolant $\neq J_S$. We show that $V_C(J_S) \subseteq V_C(J_I)$. For $V_C(J_I)$ to be an interpolant it must satisfy: $V_{A,B,C}(J_A) \subseteq V_{A,B,C}(J_I)$, which is equivalent to: $I(V_{A,B,C}(J_A)) \supseteq I(V_{A,B,C}(J_I)) \implies J_A \supseteq J_I$, due to Theorem II.3. As the generators of J_I only contain polynomials in C-variables, this relation also holds for the following

$$J_A \cap \mathbb{F}_q[C] \supseteq J_I \implies J_S \supseteq J_I \implies V_C(J_S) \subseteq V_C(J_I). \qquad \square$$

Due to this theorem, the ideal-interpolant J_I computed in Example IV.2 is the smallest interpolant. In other words, E_L for our circuit is the smallest interpolant for the pair (E_L, E_H).

Now we discuss how the largest interpolant can be computed. For this, we will make use of quotients of ideals.

Definition IV.2. (Quotient of Ideals) If J_1 and J_2 are ideals in a ring R, then $J_1 : J_2$ is the set $\{f \in R \mid f \cdot g \in J_1, \forall g \in J_2\}$ and is called the **ideal quotient** of J_1 by J_2.

We can use ideal quotients to compute the complement of a variety. Given an ideal $J' \subset R$ containing the vanishing polynomials, suppose we need to find an ideal J such that $V(J) = \mathbb{F}_q^n - V(J') = V(J_0) - V(J')$, where "$-$" corresponds to the set difference operation. Then $J = J_0 : J'$ which can also be computed using the Gröbner basis algorithm [13].

Theorem IV.3. Consider the elimination ideal $J'_L = J_B \cap \mathbb{F}_q[C]$. The complement of the variety $V_C(J'_L)$, computed as $\mathbb{F}_q^C - V_C(J'_L)$, is the largest interpolant.

Let J_L be the ideal corresponding to the largest interpolant $V_C(J_L) = \mathbb{F}_q^C - V_C(J'_L)$. This ideal-interpolant J_L can be computed as $J_L = (J_{0,C} : J'_L)$, where $J_{0,C}$ is ideal of vanishing polynomials in C-variables.

Example IV.3. *Using $J_A = E_L$ and $J_B = E_H$ from Example IV.1, we compute their largest interpolant.*

- *First compute the ideal $J'_L = J_B \cap \mathbb{F}_q[C] = E_H \cap \mathbb{F}_4[X_{PI}]$ which results in $J'_L = E_H$, as E_H comprises X_{PI} only.*
- *Then compute J_L as $J_{0,X_{PI}} : J'_L$ which results in $J_L = \langle a_0^2 + a_0, b_0^2 + b_0, b_1^2 + b_1, a_1 b_0 \rangle$.*

As $J_L = J_{0,X_{PI}} : E_H$, we will denote the largest interpolant for the circuits using E'_H.

Lemma IV.1. The total number of interpolants for the pair (J_A, J_B) is $2^{|SM(J_D)|}$, where $J_D = (J_L : J_S)$.

Proof. As $V_C(J_D) = V_C(J_L : J_S) = V_C(J_L) - V_C(J_S)$ and $|SM(J_D)| = |V_C(J_D)|$, the total number of interpolants is the size of the power set of $V_C(J_D)$. ☐

Example IV.4. *For the circuit in Example IV.1, we know (from Example IV.2 and from Example IV.3) that,*

$$J_S = E_L = \langle b_0, b_1 + 1, a_1 + 1, a_0^2 + a_0 \rangle$$
$$J_L = E'_H = \langle a_0^2 + a_0, b_0^2 + b_0, b_1^2 + b_1, a_1 b_0 \rangle$$

Computing $J_D = J_L : J_S$ gives $J_D = \langle a_0^2 + a_0, b_0^2 + b_0, b_1^2 + b_1, a_1 b_0, a_1 b_1 \rangle$. The standard monomials for J_D are $SM(J_D) = \{1, a_1, b_1, a_0, b_0, a_1 a_0, b_1 a_0, b_1 b_0, a_0 b_0, b_1 a_0 b_0\}$. Therefore, the total number of interpolants for the given pair (E_L, E_H) are $2^{|SM(J_D)|} = 2^{10} = 1024$, implying that 1024 different functions exist that can rectify the bug.

The structure of the interpolant lattice: Let $l = |SM(J_D)|$. Then, the number of levels in interpolant lattice are $l+1$, and the number of elements (interpolants) at each level i is $\binom{l}{i}$, $0 \le i \le l$.

We now describe a procedure for enumerating all the interpolants for a given pair (J_A, J_B). This is made possible by exploiting the relationship between the interpolants and $SM(J_D)$. This procedure can only be applied while operating over the field $\mathbb{F}_2$.

Theorem IV.4. Given the interpolant setup over $\mathbb{F}_2[A, B, C]$, let $SM(J_D) = \{m_1, \dots, m_l\}$. Construct a polynomial f_i using any linear combination of $\{m_1, \dots, m_l\}$ as,

$$f_i = \lambda_1 \cdot m_1 + \lambda_2 \cdot m_2 + \cdots + \lambda_l \cdot m_l \tag{2}$$

where each $\lambda_j \in \mathbb{F}_2 = \{0, 1\}$. Then all the ideal-interpolants J_I can be obtained as,

$$J_I = J_S \cdot (J_D + \langle f_i \rangle). \tag{3}$$

Proof Outline: Because $\lambda_i \in \{0, 1\}$, there can be 2^l such f_i (Eqn. (2)), and as $|SM(J_D)| = l$, the number of interpolants is also 2^l. Therefore, each f_i in Eqn. (3) will result in a distinct interpolant.

As a result, we can use Theorem IV.4 to compute all the interpolants (i.e. all admissible rectification functions) for the pair (E_L, E_H).

Example IV.5. *In Example IV.4, we computed $SM(J_D) = \{1, a_1, b_1, a_0, b_0, a_1 a_0, b_1 a_0, b_1 b_0, a_0 b_0, b_1 a_0 b_0\}$. Let's construct a polynomial f_1 as in Eqn. (2) with $\{\lambda_1, \dots, \lambda_{10}\} = \{1, 1, 0, 0, 1, 0, 0, 0, 0, 1\}$. Therefore, $f_1 = 1 + a_1 + b_0 + b_1 a_0 b_0$. Using Eqn. (3), we get an ideal-interpolant $J_I = \langle b_0^2 + b_0, b_1^2 + b_1, a_1 + b_0 + 1, a_0 b_1 b_0, a_0^2 + a_0 \rangle$. Then, rectification polynomial $U = b_1 b_0 a_1 a_0 + b_1 b_0 a_0 + b_0 + a_1 + 1$, and rectification Boolean function $e_3 = \neg((b_1 \wedge b_0 \wedge a_1 \wedge a_0) \oplus (b_1 \wedge b_0 \wedge a_0) \oplus b_0 \oplus a_1)$*

As another example, if we construct a polynomial f_2 as in Eqn. (2) with $\{\lambda_1, \dots, \lambda_{10}\} = \{0, 0, 0, 0, 1, 0, 0, 0, 0, 0\}$, then $f_2 = b_0$. In this case, the ideal-interpolant $J_I = \langle a_1^2 + a_1, b_1^2 + b_1, a_0^2 + a_0, b_0 \rangle$. The rectification polynomial is $U = b_0$ corresponding to a lower cost rectification function $e_3 = b_0$.

V. Experimental Results

Based on the aforementioned theory, we have performed rectification experiments on finite field arithmetic circuit benchmarks that are used in cryptography. We have implemented the procedures to perform the rectification check, to compute ideal J_D, and to compute subsequent ideal-interpolants J_I using the SINGULAR symbolic algebra computation system [ver. 4-1-0] [14]. Synthesis of the rectification polynomial function $x_i = U(X_{PI})$ into a mapped gate-level netlist is performed using the *abc* tool [15]. The experiments are conducted on a desktop computer with a 3.5GHz Intel Core™ i7-4770K Quad-core CPU, 32 GB RAM, running 64-bit Linux OS.

We have performed experiments with two different sets of finite field benchmark circuits. The first set contains Mastrovito multipliers, with various operand widths k, that perform modular multiplication $Z = A \times B \pmod{P(x)}$, where $P(x)$ is a given irreducible polynomial in $\mathbb{F}_{2^k}$ and A, B, Z are k-bit operands. The second set comprises circuits that perform point addition over elliptic curves in $\mathbb{F}_{2^k}$ – an elliptic curve cryptography (ECC) primitive.

The experiments are setup in a similar fashion as in [6]. A bug is introduced in the implementation circuit, and based on Thm. III.1, the respective elimination ideals E_L & E_H are computed. Single-fix rectifiability is ascertained for a net x_i by checking if $1 \in E_L + E_H$ or $V(E_L) \cap V(E_H) = \emptyset$. Correspondingly, treating the ideals E_L and E_H as J_A and J_B, respectively, we compute the ideals corresponding to the smallest interpolant as $J_S = GB(E_L)$ and the largest interpolant as $J_L = GB(E'_H)$, where $E'_H = J_0 : E_H$. We then compute the ideal $J_D = J_L : J_S$, through which the standard monomials $SM(J_D)$ are obtained. Then the number of interpolants (i.e. the number of admissible rectification functions) is given by $|SM(J_D)|$. Subsequently, we use Thm. IV.4 to traverse the interpolant lattice. Starting from the smallest interpolant $V(J_S)$, we produce successively larger ones $V(J_I)$'s, terminating at the largest interpolant $V(J_L)$.

TABLE I: Mastrovito Multipliers. (n): # of gates, (t): time in seconds

Op. Width k	# of Gates	Rect. Check(t)	# of Interpolants	Comp. Time(t)	Syn. Time(t)	Area Cost(n)		Cycle-time: Gate delay	
						Lowest	Highest	Smallest	Largest
4	45	0.01	8	0.03	0.53	3*	5	3*	5
8	172	0.03	16	0.04	1.02	5*	18	3*	11
16	808	0.37	32	0.08	2.05	14*	80	7*	39
32	2,858	15.70	32	0.24	1.96	17*	78	7*	39
64	11,200	741.75	64	676.57	4.48	57*	318	21*	149
96	24,523	5,985.2	64	3,170.63	4.5	59*	318	21*	149

TABLE II: Point Addition. (n): # of gates, (t): time in seconds

Op. Width k	# of Gates	Rect. Check(t)	# of Interpolants	Comp. Time(t)	Syn. Time(t)	Area Cost(n)		Cycle Time: Gate Delay	
						Lowest	Highest	Smallest	Largest
4	58	0.02	4	0.03	0.27	7*	12	4*	7
8	206	0.07	36	0.04	2.45	71*	146	32*	70
16	1,285	3.62	64	0.80	10.67	733*	1081	330*	520
32	3,926	81.45	48	2.85	3.81	188*	405	80*	200
64	15,313	4,751.9	48	98.92	3.83	200	404	82	197

Table I shows the results for Mastrovito multipliers. Column 3 shows the time to perform the single-fix rectification check. Columns 4 and 5 show the total number of interpolants generated using our approach and the time to compute all of them, respectively. For each ideal-interpolant, we compute a rectification polynomial $U(X_{PI})$. These polynomials are converted to Boolean AND-XOR logic circuits and written in BLIF format. We use the command *mfs2* in *abc* to optimize the logic and perform technology mapping on to AND, XOR gates and inverters. The total time for synthesizing circuits from all U-polynomials is shown in column 6. For the resulting mapped circuits, we depict the lowest and highest area-cost implementations (number of gates) for the respective rectification functions, shown in columns 7 and 8. The shortest and longest gate-delays (cycle-time) of the synthesized functions are also depicted in columns 9 and 10, respectively. A '*' in columns 7 and 9 implies that the lowest area-cost implementation of $U(X_{PI})$ also incurs the shortest delay.

Point addition operation in $\mathbb{F}_{2^k}$ is also implemented as a set of multivariate polynomials, which are synthesized into a circuit. Table II shows the corresponding result for one of the blocks with specification polynomial $D = B^2 \cdot (C + aZ_1^2)$. Here B, C, D, Z_1 are k-bit operands and $a \in \mathbb{F}_{2^k}$. *As seen from the results in both tables, exploring the interpolant lattice is beneficial as it allows us to choose interpolants with low cost in a reasonable amount of time.* For example, for a 96-bit Matsrovito multiplier, we enumerate 64 different interpolants in 3170 seconds. All of these can be synthesized in 4.5 seconds, where the lowest area cost implementation consists of 59 AND/XOR/INV gates, whereas the highest cost interpolant implements 318 gates. The most computationally expensive parts of the experiments are performing the rectification check and computing the ideal J_D. We are currently working on term-order based heuristics to avoid multiple expensive GB computations and make the approach more practical.

VI. CONCLUSION

This paper has presented a detailed theory and algorithm describing the notion of Craig interpolants for a pair of polynomial ideals in finite fields with no common zeros. The approach utilizes concepts from computational algebraic geometry. Interpolants always exist in this setting, and they correspond to the variety of an elimination ideal. In addition to defining the smallest and the largest interpolants, techniques are described to compute them using Gröbner basis concepts. The total number of interpolants is also determined, and a technique is presented to enumerate all possible interpolants. These algebraic interpolants are used as rectification functions, and experiments are performed that demonstrate the benefit of exploring the interpolant lattice to search for low-cost implementations.

REFERENCES

[1] K. Gitina, S. Reimer, M. Sauer, R. Wimmer, C. Scholl, and B. Becker, "Equivalence Checking of Partial Designs Using Dependency Quantified Boolean Formulae," in *Proc. Intl. Conf. Comp. Des. (ICCD)*, 2013.

[2] M. Fujita, "Toward Unification of Synthesis and Verification in Topologically Constrained Logic Design," *Proceedings of the IEEE*, 2015.

[3] K.-F. Tang, C.-A. Wu, P.-K. Huang, and C.-Y. R. Huang, "Interpolation based incremental ECO Synthesis for Multi-Error Logic Rectification," in *Proc. Design Automation Conf.*, 2011, pp. 146–151.

[4] A. Dao, N.-Z. Lee, L.-C. Chen, M. Lin, J.-H. Jiang, A. Mishchenko, and R. Brayton, "Efficient Computations of ECO Patch Functions," in *Proc. Design Auto. Conf.*, 2018.

[5] W. Craig, "Linear reasoning: A new form of the Herbrand-Gentzen theorem," *Journal of Symbolic Logic*, vol. 22, no. 3, pp. 250–268, 1957.

[6] U. Gupta, I. Ilioaea, V. Rao, A. Srinath, P. Kalla, and F. Enescu, "On the rectifiability of arithmetic circuits using craig interpolants in finite fields," in *IFIP/IEEE Intl. Conf. VLSI (VLSI-SoC)*, 2018, pp. 1–6.

[7] V. Rao, U. Gupta, I. Ilioaea, A. Srinath, P. Kalla, and F. Enescu, "Post-Verification Debugging and Rectification of Finite-Field Arithmetic Circuits using Computer Algebra Techniques," in *Proc. Formal Methods in CAD*, 2018, pp. 121–129.

[8] F. Farahmandi and P. Mishra, "Automated Debugging of Arithmetic Circuits Using Incremental Gröbner Basis Reduction," in *IEEE International Conference on Computer Design (ICCD)*, 2017.

[9] V. Rao and P. Kalla, "Discussions on recent automatic debugging approaches," 2018, available at http://eng.utah.edu/ utkarshg/cex.pdf.

[10] S. Gao, A. Platzer, and E. Clarke, "Quantifier Elimination over Finite Fields with Gröbner Bases," in *Intl. Conf. Algebraic Informatics*, 2011.

[11] W. W. Adams and P. Loustaunau, *An Introduction to Gröbner Bases*. American Mathematical Society, 1994.

[12] S. Gao, "Counting Zeros over Finite Fields with Gröbner Bases," Master's thesis, Carnegie Mellon University, 2009.

[13] D. Cox, J. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2007.

[14] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann, "SINGULAR 4-1-0 — A computer algebra system for polynomial computations," http://www.singular.uni-kl.de, 2016.

[15] R. Brayton and A. Mishchenko, "ABC: An Academic Industrial-Strength Verification Tool," in *Computer Aided Verification*, vol. 6174. Springer, 2010, pp. 24–40.