Craig Interpolants in Finite Fields using Algebraic Geometry: Theory and Application

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

***Abstract*—This paper considers Craig interpolation for a mutually inconsistent pair of polynomial constraints over finite fields $\mathbb{F}_q$, for q any prime power. Using techniques from algebraic geometry, we show that Nullstellensatz over finite fields admits Craig interpolation. The constraints are represented as polynomial ideals with inconsistent varieties, and it is shown how various interpolants, including the smallest and the largest one, can be computed using the Gröbner basis (GB) algorithm. The number of all possible interpolants can also be easily identified. As an application, we demonstrate how the theory of interpolants in finite fields can be applied to circuit rectification for arithmetic circuits. We perform rectification by applying single-fix rectification where we modify the functionality of exactly one gate output net. We also show how the correction for that faulty output can be constructed using interpolants.**

I. Introduction

Craig interpolation is a method to construct and refine abstractions of functions. It finds application in formal verification of hardware designs and software programs, in logic synthesis of Boolean functions, and also as a tool in proof complexity theory. It is a logical tool to extract concise explanations for the infeasibility of a mutually inconsistent set of statements. Craig [1] showed that for a valid implication $A \implies B$, where A,B are first order formulae containing no free variables, there is a formula I such that $A \implies I$, $I \implies B$ and the non-logical symbols of I appear in both A and B. The formula I is called the *Craig interpolant*, or interpolant for short. As propositional logic also admits Craig interpolation, the formal verification community has extensively investigated interpolants and their computation from resolution proofs of CNF-SAT problems. In the propositional logic domain, the concept is stated with a slight modification.

Definition I.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 **interpolant** of (A,B).

Given the pair (A,B) and their refutation proof, a procedure called the *interpolation system* constructs the interpolant in linear time and space in the size of the proof [2]. As the abilities of SAT solvers for proof refutation have improved, interpolants have been exploited as abstractions in various problems that can be formulated as unsatisfiable instances, e.g. model checking [2], logic synthesis [3], etc. Their use as abstractions have also been replicated in other (combinations of) theories [4] [5] [6] [7], etc.

In this paper, we introduce the notion of *Craig interpolants in polynomial algebra over finite fields* ($\mathbb{F}_q$) of q elements, where $q = p^k$ is a prime power. Given a mutually inconsistent pair of sets of polynomials with coefficients from $\mathbb{F}_q$ that have no common zeros, we show that Nullstellensatz over finite fields admits interpolation. We represent the sets A,B (from Def. I.1) as varieties of corresponding ideals, and prove the existence of an interpolant for the pair (A,B). In this setting, *interpolants correspond to varieties* – subsets of the n-dimensional affine space $\mathbb{F}_q^n$ – and are represented by polynomial ideals, more precisely, by a *Gröbner basis of corresponding ideals.* We demonstrate the application of the interpolants for circuit rectification.

Intuitively, it should be apparent that polynomial algebra over finite fields would admit Craig interpolation (a first order theory over $\mathbb{F}_q$ admits quantifier elimination [8]). However, our literature search for interpolants and their computation with polynomials in arbitrary finite fields did not reveal much prior work in this area. Recent years have witnessed investigations in formal verification, abstraction and synthesis of datapath circuits with k-bit operands, where the problems have been modeled over finite fields ($\mathbb{F}_{2^k}$) [9] [10] or over finite integer rings ($\mathbb{Z}_{2^k}$) [11]. Interpolants can be exploited as abstractions of functions ($f : \mathbb{F}_{2^k} \rightarrow \mathbb{F}_{2^k}$) in this domain, and can make these approaches practical. Motivated by the above needs, this paper presents the theory of Craig interpolation in finite fields, and a preliminary application on circuit rectification.

Contributions: Using the extensive machinery of algebraic geometry in finite fields, this paper makes the following contributions: 1) Formally define the notion of interpolants in polynomial algebra over finite fields $\mathbb{F}_q$, and prove their existence in this domain. 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) Application of interpolants in identifying a correction function in the case of an incorrect circuit implementation after the circuit has been deemed single fix rectifiable using Thm. V.1.

Paper Organization: The following section briefly reviews

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

prior work in Craig interpolation in various theories, and contrasts it against the concepts presented in this paper. It also provides a brief background on the previous work in circuit rectification. Section III describes the preliminary concepts of algebraic geometry and Gröbner bases in finite fields. Section IV describes the theory of interpolation in finite fields and shows how they can be computed using the Gröbner basis algorithm. Section V describes the problem of circuit rectification and how interpolants can be used to find a correction. Some preliminary experiments to verify that a faulty circuit is single fix rectifiable have also been presented. Section VI concludes the paper.

II. Review of Previous Work

In the past decade, there has been an explosion in the study, classification and application of interpolants. In abstraction-based model checking, interpolants are used as over-approximate image operators [2]. In Boolean function decomposition, given a function $F(A,B,C)$ with support variables partitioned into disjoint subsets A,B,C, it is required to decompose $F = G(A,C) \odot H(B,C)$, where $\odot$ denotes the Boolean $\vee, \wedge, \oplus$ operations. The existence of such a decomposition with the given variable partition is formulated as a unsatisfiability checking problem. Craig interpolants can then be used to compute G,H [3] [12]. In proof complexity, interpolants have been used as a tool to derive lower bounds; *e.g.* by reasoning that if $A \implies B$ does not have a simple interpolant, then it cannot have a simple proof [13]. The authors in [14] present an interpolation theorem for Nullstellensatz refutations and the polynomial calculus [15] which can then be used for proving lower bounds.

The use of interpolants as abstractions has also been replicated in other combinations of theories. For example, the theory of linear inequality [4], data-type theories [5], linear arithmetic and difference logic [6], bit-vector SMT theories [7], etc., are just a few of the many instances of the usage of interpolation in various domains outside of purely propositional logic. The aforementioned works derive interpolants from resolution proofs obtained from SAT/SMT-solvers [6], or generate them by solving constraints in the theories of linear arithmetic with uninterpreted functions ([16]), or exploit their connection to quantifier elimination ([5]), etc. As an alternative to interpolation, [17] suggests the use of local proofs and symbol eliminating inferences for invariant generation. However, the problem has been insufficiently investigated over polynomial ideals in finite fields from an algebraic geometry perspective.

The works that come closest to ours are by Gao *et al.* [8] and [18]. While they do not address the interpolation problem per se, they do describe important results of Nullstellensatz, projections of varieties and quantifier elimination over finite fields that we extensively utilize in this paper.

The work of [19] classifies (orders) the interpolants according to their logical strength for model checking. They present a labeled interpolation system built on the resolution proof where each vertex of the proof is annotated with partially ordered labels. Interpolants generated from different sets of labels have the same order of strength as the order of the labels. This way a (sub-)lattice of interpolants is generated with the smallest interpolant being the same as obtained from the McMillan's system (L_M) [2] and the largest being the complement of inverse of L_M. The labeled interpolation system of [19] is generalized to support propositional hyper-resolution proofs [20]. More recently, [21] presents the notion of interpolation abstraction, and describes a semantic framework for exploring interpolant lattices. In contrast to these works that qualitatively order the interpolants w.r.t. a given application (e.g. model checking), we describe a method to explore interpolants based on the cardinality of the zero-sets of polynomial ideals, which in turn corresponds to the size of the abstraction.

Automated diagnosis and rectification of digital circuits has been addressed in [22], [23]. The paper [24] presents algorithms for synthesizing Engineering Change Order (ECO) patches. The use of interpolation for ECO has been presented in [25], [26], [27]. The single-fix rectification function approach in [26], [27] has been extended in [25] to generate multiple partial-fix functions while guaranteeing that the number of different minterms becomes smaller in each step. The application of interpolants in finite fields, presented in Section V, is motivated by the formulation of circuit rectification problem and the use of interpolation in [25].

III. Notation and Preliminary Concepts

Let $\mathbb{F}_q$ denote the finite field of q elements where $q = p^k$ is a prime power, $\overline{\mathbb{F}}_q$ be its algebraic closure, and $R = \mathbb{F}_q[x_1,\ldots,x_n]$ the polynomial ring in n variables $x_1,\ldots,x_n$, with coefficients from $\mathbb{F}_q$. A monomial is a power product of the form $X = x_1^{e_1} \cdot x_2^{e_2} \cdots x_n^{e_n}$, where $e_i \in \mathbb{Z}_{\geq 0}, 1 \leq i \leq n$. A *polynomial* $f \in R$ is written as a finite sum of terms $f = c_1X_1 + c_2X_2 + \cdots + c_tX_t$, where $c_1,\ldots,c_t$ are coefficients and $X_1,\ldots,X_t$ are monomials. Impose a monomial order $>$ (a term order) on the ring – i.e. a total order and a well-order on all the monomials of R s.t. multiplication with another monomial preserves the order. Then the monomials of all polynomials $f = c_1X_1 + c_2X_2 + \cdots + c_tX_t$ are ordered w.r.t. to $>$, such that $X_1 > X_2 > \cdots > X_t$. Subject to $>$, $lt(f) = c_1X_1$, $lm(f) = X_1$, $lc(f) = c_1$, are the *leading term*, *leading monomial* and *leading coefficient* of f, respectively. In this work, we consider lexicographic (lex) term orders (see Definition 1.4.3 in [28]).

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

Let $\boldsymbol{a} = (a_1,\ldots,a_n) \in \mathbb{F}_q^n$ be a point in the affine space, and f a polynomial in R. If $f(\boldsymbol{a}) = 0$, we say that f *vanishes* on $\boldsymbol{a}$. We have to analyze the *set of all common zeros* of the polynomials of F that lie within the field $\mathbb{F}_q$. This zero set is called the *variety*. It depends not just on the given set of polynomials but rather on the ideal generated by them. We denote it by $V_{\mathbb{F}_q}(J) = V_{\mathbb{F}_q}(f_1,\ldots,f_s)$, where:

$$V_{\mathbb{F}_q}(J) = V_{\mathbb{F}_q}(f_1,\ldots,f_s) = \{\boldsymbol{a} \in \mathbb{F}_q^n : \forall f \in J, f(\boldsymbol{a}) = 0\}.$$

Varieties can be different when restricted to the given field $\mathbb{F}_q$ or considered over its algebraic closure $\overline{\mathbb{F}}_q$. We will generally drop the subscript when considering varieties over $\mathbb{F}_q$ and denote $V(J)$ to imply $V_{\mathbb{F}_q}(J)$. The subscripts will be

used, however, to avoid any ambiguities, e.g. when comparing $V_{\mathbb{F}_q}(J)$ against the one over the closure $V_{\overline{\mathbb{F}}_q}(J)$.

Given two ideals $J_1 = \langle f_1, \ldots, f_s \rangle, J_2 = \langle h_1, \ldots, h_r \rangle$, the sum $J_1 + J_2 = \langle f_1, \ldots, f_s, h_1 \ldots, h_r \rangle$, 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 $V(J_1) \supseteq V(J_2)$.

Algorithm 1 Buchberger's Algorithm

```
1:  procedure compute_gb(F = {f_1, ..., f_s})
2:      G := F;
3:      repeat
4:          G' := G
5:          for each pair {f_i, f_j}, i ≠ j in G' do
6:              Spoly(f_i, f_j) --G'-->_+ h
7:              if h ≠ 0 then
8:                  G := G ∪ {h}
9:      until G = G'
10:     return G
```

Gröbner Basis: An ideal may have many different sets of generators: $J = \langle f_1, \ldots, f_s \rangle$ $= \cdots = \langle g_1, \ldots, g_t \rangle$. Given a non-zero ideal J, a *Gröbner basis* (GB) for J is a finite set of polynomials $G = \{g_1, \ldots, g_t\}$ satisfying $\langle \{lm(f) \mid f \in J\} \rangle = \langle lm(g_1), \ldots, lm(g_t) \rangle$. Then $J = \langle G \rangle$ holds and so $G = GB(J)$ forms a basis for J. A GB G possesses important properties that allow to solve many polynomial computation and decision problems. The famous Buchberger's algorithm ([28]) takes as input the set of polynomials $F = \{f_1, \ldots, f_s\}$ and computes the GB $G = \{g_1, \ldots, g_t\}$. The pseudo-code is shown in Algorithm 1. The polynomial $Spoly(f_i, f_j)$ in the pseudo-code is computed as $\frac{L}{lt(f_i)} \cdot f_i - \frac{L}{lt(f_j)} \cdot f_j$, where $L = LCM(lt(f_i), lt(f_j))$. The symbol $f \xrightarrow{G}_+$ denotes reduction (multivariate division) of f by the polynomials in the set G. A GB can be *reduced* to eliminate redundant polynomials from the basis. A reduced GB is a canonical representation of the ideal. In this work, the set G will denote a reduced GB, and any reference to computation of an ideal can be construed as computing its GB.

Varieties over finite fields and the structure of Gröbner bases: When the variety of an ideal is finite, then the ideal is said to be *zero-dimensional*. As $V_{\mathbb{F}_q}(J)$ is a finite set, J is zero-dimensional. A GB for a zero dimensional ideal exhibits a special structure that we exploit in this work.

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, sometimes also referred to as the field polynomial. Denote by $J_0 = \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$ the ideal of all vanishing polynomials in the ring R. Then $V_{\mathbb{F}_q}(J_0) = V_{\overline{\mathbb{F}}_q}(J_0) = \mathbb{F}_q^n$. Therefore, given any ideal J, $V_{\mathbb{F}_q}(J) = V_{\overline{\mathbb{F}}_q}(J) \cap \mathbb{F}_q^n = V_{\overline{\mathbb{F}}_q}(J) \cap V_{\overline{\mathbb{F}}_q}(J_0) = V_{\overline{\mathbb{F}}_q}(J + J_0) = V_{\mathbb{F}_q}(J + J_0)$.

Theorem III.1 (*The Weak Nullstellensatz over finite fields (from Theorem 3.3 in [18])*). *For a finite field $\mathbb{F}_q$ and the ring $R = \mathbb{F}_q[x_1, \ldots, x_n]$, let $J = \langle f_1, \ldots, f_s \rangle \subseteq R$, and let $J_0 = \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$ be the ideal of vanishing polynomials. Then $V_{\mathbb{F}_q}(J) = \emptyset \iff 1 \in J + J_0 \iff G = reducedGB(J + J_0) = \{1\}$.*

To find whether a set of polynomials $f_1, \ldots, f_s$ have no common zeros in $\mathbb{F}_q$, we can compute the reduced GB G of $\{f_1, \ldots, f_s, x_1^q - x_1, \ldots, x_n^q - x_n\}$ and see if $G = \{1\}$. If $G \neq \{1\}$, then $f_1, \ldots, f_s$ do have common zeros in $\mathbb{F}_q$, and G consists of the finite set of polynomials $\{g_1, \ldots, g_t\}$ with the following properties.

Theorem III.2 (*Gröbner bases in finite fields (application of Theorem 2.2.7 from [28] over $\mathbb{F}_q$)*). *For $G = GB(J + J_0) = \{g_1, \ldots, g_t\}$, the following statements are equivalent:*

1) *The variety $V_{\mathbb{F}_q}(J)$ is finite.*
2) *For each $i = 1, \ldots, n$, there exists some $j \in \{1, \ldots, t\}$ such that $lm(g_j) = x_i^l$ for some $l \in \mathbb{N}$.*
3) *The quotient ring $\frac{\mathbb{F}_q[x_1, \ldots, x_n]}{\langle G \rangle}$ forms a finite dimensional vector space.*

For a GB G, let $LM(G)$ denote the set of leading monomials of all elements of G: $LM(G) = \{lm(g_1), \ldots, lm(g_t)\}$.

Definition III.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 III.3 (*Counting the number of solutions (Theorem 3.7 in [18])*). *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, $|V(J)| = |SM(G)|$.*

Definition III.2. 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 similarly if $V_1 \subset V_2$ are varieties, then $I(V_1) \supset I(V_2)$.

Definition III.3. For any ideal $J \subset R$, the **radical** of J is defined as $\sqrt{J} = \{f \in R : \exists m \in \mathbb{N} \; s.t. f^m \in J\}$.

When $J = \sqrt{J}$, J is called a radical ideal. Over algebraically closed fields, the *Strong Nullstellensatz* establishes the correspondence between radical ideals and varieties. Over finite fields, it has a special form.

Lemma III.1. (From [8]) For an arbitrary ideal $J \subset \mathbb{F}_q[x_1, \ldots, x_n]$, and $J_0 = \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$, the ideal $J + J_0$ is radical; i.e. $\sqrt{J + J_0} = J + J_0$.

Theorem III.4 (*The Strong Nullstellensatz over finite fields (Theorem 3.2 in [8])*). For any ideal $J \subset \mathbb{F}_q[x_1, \ldots, x_n]$, $I(V_{\mathbb{F}_q}(J)) = J + J_0$.

Definition III.4. Given an ideal $J = \langle f_1, \ldots, f_s \rangle \subset R$ 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

$$Pr_l(V(J)) : \mathbb{F}_q^n \to \mathbb{F}_q^{n-l}, \; Pr_l(a_1, \ldots, a_n) = (a_{l+1}, \ldots, a_n)$$

for every $\boldsymbol{a} = (a_1, \ldots, a_n) \in V(J)$.

Definition III.5. Given an ideal $J \subset \mathbb{F}_q[x_1, \ldots, x_n]$, the l-th elimination ideal J_l is an ideal in R defined as $J_l = J \cap \mathbb{F}_q[x_{l+1}, \ldots, x_n]$.

The next theorem shows how we can obtain the generators of the l-th elimination ideal using Gröbner bases.

Theorem III.5 (*Elimination Theorem [29]*)**.** 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 $0 \leq l \leq n$ we denote by G_l the GB of l-th elimination ideal of J and compute it as:

$$G_l = G \cap \mathbb{F}_q[x_{l+1}, \ldots, x_n]$$

In a general setting, the projection of a variety is a subset of the variety of an elimination ideal: $Pr_l(V(J)) \subseteq V(J_l)$. However, operating over finite fields, when the ideals contain the vanishing polynomials, then the above set inclusion turns into an equality.

Lemma III.2 (Lemma 3.4 in [8])**.** Given an ideal $J \subset R$ that contains the vanishing polynomials of the field, 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.

We will utilize all of the above concepts to derive the results in this paper.

IV. Theory

We describe the setup for Craig interpolation in the ring $R = \mathbb{F}_q[x_1, \ldots, x_n]$. Partition the variables $\{x_1, \ldots, x_n\}$ into disjoint subsets A, B, C. We are given two ideals $J_A \subset \mathbb{F}_q[A, C]$, $J_B \subset \mathbb{F}_q[B, C]$ 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 $\boldsymbol{A^q - A, C^q - C}$, 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 Lemmas III.1 and III.2. 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. III.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 disjoint 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 of 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.

Example IV.1. *Consider the ring $R = \mathbb{F}_2[a, b, c, d, e]$, partition the variables as $A = \{a\}, B = \{e\}, C = \{b, c, d\}$. Let ideals*

$$J_A = \langle ab, bd, bc + c, cd, bd + b + d + 1 \rangle + J_{0,A,C}$$
$$J_B = \langle b, d, ec + e + c + 1, ec \rangle + J_{0,B,C}$$

where $J_{0,A,C}$ and $J_{0,B,C}$ are the corresponding ideals of vanishing polynomials. Then, we have

$$V_{A,B,C}(J_A) = \mathbb{F}_q^B \times V_{A,C}(J_A) = (abcde) : \{01000, 00010, 01100, 10010, 01001, 00011, 01101, 10011\}$$
$$V_{A,B,C}(J_B) = \mathbb{F}_q^A \times V_{B,C}(J_B) = (abcde) : \{00001, 00100, 10001, 10100\}$$

The ideals J_A, J_B have no common zeros as $V_{A,B,C}(J_A) \cap V_{A,B,C}(J_B) = \emptyset$. The pair (J_A, J_B) admits a total of 8 interpolants:

1) $V(J_S) = (bcd) : \{001, 100, 110\}$
   $J_S = \langle cd, b + d + 1 \rangle$
2) $V_C(J_1) = (bcd) : \{001, 100, 110, 101\}$
   $J_1 = \langle cd, bd + b + d + 1, bc + cd + c \rangle$
3) $V_C(J_2) = (bcd) : \{001, 100, 110, 011\}$
   $J_2 = \langle b + d + 1 \rangle$
4) $V_C(J_3) = (bcd) : \{001, 100, 110, 111\}$
   $J_3 = \langle b + cd + d + 1 \rangle$
5) $V_C(J_4) = (bcd) : \{001, 100, 110, 011, 111\}$
   $J_4 = \langle bd + b + d + 1, bc + b + cd + c + d + 1 \rangle$
6) $V_C(J_5) = (bcd) : \{001, 100, 110, 101, 111\}$
   $J_5 = \langle bc + c, bd + b + d + 1 \rangle$
7) $V_C(J_6) = (bcd) : \{001, 100, 110, 101, 011\}$
   $J_6 = \langle bd + b + d + 1, bc + cd + c \rangle$
8) $V_C(J_L) = (bcd) : \{001, 011, 100, 101, 110, 111\}$
   $J_L = \langle bd + b + d + 1 \rangle$.

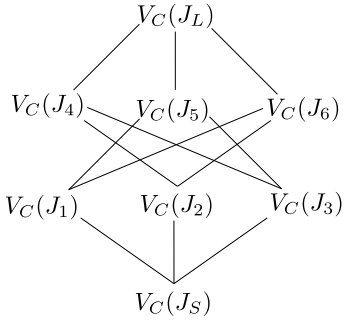

Fig. 1: Interpolant lattice

It is easy to check that all $V(J_I)$ satisfy the 3 conditions of Def. IV.1. Note also that $V(J_S)$ is the smallest interpolant, contained in every other interpolant. Likewise, $V(J_L)$ contains all other interpolants and it is the largest. The other containment relationships are shown in the corresponding interpolant lattice in Fig. 1; i.e. $V_C(J_1) \subset V_C(J_5), V_C(J_1) \subset V_C(J_6)$, and so on.

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 III.2. 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 reduced 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. *The elimination ideal J_I computed for J_A from Example IV.1 is $J_I = J_S = \langle cd, b+d+1\rangle$ with variety $V_C(J_I) = (bcd) : \{001, 100, 110\}$. This variety over the variable set A and C is $V_{A,C}(J_I) = (abcd) : \{0001, 0100, 0110, 1001, 1100, 1110\}$, and it contains $V_{A,C}(J_A)$. Moreover, $V_{A,B,C}(J_I)$ also has an empty intersection with $V_{A,B,C}(J_B)$.*

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 $V_{A,B,C}(J_S) \subseteq V_{A,B,C}(J_I)$. For $V_{A,B,C}(J_I)$ 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 3.4. As the generators of J_I only contain polynomials in C-variables, this relation also holds for the following

$$J_A \cap \mathbb{F}_q[C] \supseteq J_I$$
$$\implies J_S \supseteq J_I$$
$$\implies V_{A,B,C}(J_S) \subseteq V_{A,B,C}(J_I).$$

□

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 $J = J_0 : J'$ (see Theorem III.2 and Corollary III.1 in [10] for a proof outline). Once again, the Gröbner basis algorithm can be used to compute $J_0 : J'$ [29].

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

Proof. The proof for this theorem is very similar to the proof of Theorem IV.2.

□

Let J_L be the radical 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. *The ideal-interpolant $J_L = \langle bd+b+d+1\rangle$ in Example IV.1 is computed as:*

- *First compute the ideal $J'_L = J_B \cap \mathbb{F}_q[C]$ which results in $J'_L = \langle b, d\rangle$.*
- *Then compute J_L as $J_L = J_{0,C} : J'_L$ which results in $J_L = \langle bd+b+d+1\rangle$*

The variety $V_C(J_L) = (bcd) : \{001, 011, 100, 101, 110, 111\}$ and it is the largest interpolant for the given pair (J_A, J_B).

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. *From Example IV.1 $J_L = \langle bd+b+d+1\rangle$ and $J_S = \langle cd, b+d+1\rangle$. Computing $J_D = J_L : J_S$ gives $J_D = \langle d+1, bc+b+c+1, c^2+c, b^2+b\rangle$, where the variety $V_C(J_D) = V_C(J_L) - V_C(J_S) = (bcd) : \{011, 101, 111\}$. The standard monomials for J_D are $SM(J_D) = \{1, b, c\}$. Therefore, the total number of interpolants for the given pair (J_A, J_B) is $2^{|\{1,b,c\}|} = 2^3 = 8$.*

The structure of the interpolant lattice: Note that our results do provide some insights into 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 $\binom{l}{i}$, $0 \le i \le l$ (Fig. 1).

V. An application: Circuit Rectification

In this section, we show an application of interpolants in finite fields. Circuit rectification is performed to find a correction in a circuit after verification against a specification has failed. The process of single fix rectification involves identifying a potential site where the fault can be corrected and then synthesizing a correction function that can be used instead of the currently implemented function/logic. We do not make assumption about the type of bug(s)/fault(s) but try to fix the incorrect implementation at exactly one gate output.

Using the weak Nullstellensatz (Theorem III.1), we formulate the test for rectifiability at a gate output x_i in the circuit, and show that a correction function (in primary inputs) for that output can be obtained using interpolants.

We will first formulate the problem statement more formally and then present the main theorem. Later in this section we present experimental results for our approach.

Let $q = 2^k$, and let ring $R = \mathbb{F}_{2^k}[x_1,\ldots,x_n]$, where $X = \{x_1,\ldots,x_n\}$. Let $f_{spec} \in R$ be a specification polynomial, and let C be a circuit that implements f_{spec}. Let $X_{PI} \subset X$ be the set of primary input variables. The gates of C can be modeled as polynomials in $\mathbb{F}_2$:

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

Let the implementation and specification models be mappings of the type $\mathbb{F}_{2^k} \rightarrow \mathbb{F}_{2^k} : Z_m + \mathcal{F}(A)$ and $\mathbb{F}_{2^k} \rightarrow \mathbb{F}_{2^k} : Z_s + \mathcal{F}(A)$, respectively, implying that they both operate k-bit input and output operands. The bit level input variables $a_0,\ldots,a_{k-1}$ and word-level input variable A can be related using the polynomial $A = \sum_{i=0}^{k-1} a_i \alpha^i$, where α is a primitive element of $\mathbb{F}_{2^k}$. Similarly the output bit level and word level variables can be related. We write polynomials for the gates of C that describe an ideal $I = \langle f_1,\ldots,f_s\rangle$.

We can perform an equivalence check between f_{spec} and the circuit C by formulating the problem using the weak Nullstellensatz over $\mathbb{F}_{2^k}$. Consider the polynomial,

$$f_m : t(Z_m - Z_s) - 1 \tag{2}$$

where t is a free variable. Note that when $Z_s = Z_m, Z_s - Z_m = 0$, $t \cdot 0 = 1$ has no solution; this implies that C is a correct implementation of f_{spec}. And when $Z_s \neq Z_m, Z_s - Z_m \neq 0$. So let $t^{-1} = Z_s - Z_m$, then $t \cdot t^{-1} = 1$ always has a solution, which actually corresponds to the bug.

For performing equivalence check between the f_{spec} and the circuit C, we construct an ideal $J = \langle f_1,\ldots,f_s,f_m,f_{spec}\rangle$ and check if $GB(J+J_0) = 1$? J_0 is the vanishing ideal containing the polynomials $x_i^2 - x_i$, for bit level variables, and polynomials $A^{2^k} - A$ for word level variables. If $GB(J+J_0) = 1$, it implies that for no input value the circuit and specification result in different output and they are equivalent (not equivalent when $GB(J+J_0) \neq 1$).

We are now ready to formulate the problem statement pertaining to circuit rectification. Let x_i denote the gate output in the circuit which is given to us as a possible location of the fault. The equivalent polynomial for the Boolean function at output x_i is $f_i : x_i + P$. The following theorem provides a procedure to check if the circuit can indeed be rectified at this location given the specification polynomial f_{spec}.

Theorem V.1 (Checking if C can be rectified at a location x_i)**.** Consider the two ideals J_L and J_H constructed as:

- $J_L = \langle f_{spec}, f_1,\ldots,f_i : x_i,\ldots,f_s,f_m\rangle$ where $f_i : x_i + P$ is replaced with $f_i : x_i$.
- $J_H = \langle f_{spec}, f_1,\ldots,f_i : x_i + 1,\ldots,f_s,f_m\rangle$ where $f_i : x_i + P$ is replaced with $f_i : x_i + 1$.

Compute $E_L = J_L \cap \mathbb{F}_{2^k}[X_{PI}]$ and $E_H = J_H \cap \mathbb{F}_{2^k}[X_{PI}]$ where E_L and E_H are elimination ideals containing only X_{PI} variables. Then the circuit can be rectified at f_i *w.r.t.* the specification if and only if $1 \in E_L + E_H + J_0$.

Corollary V.1. Let $J_A = E_L + J_0, J_B = E_H + J_0$ and $1 \in E_L + E_H + J_0$, then compute an ideal-interpolant J_I for the pair (J_A, J_B), where X_{PI} is the set of common variables. Then there exists a polynomial $U(X_{PI})$ s.t. the variety $V_{\mathbb{F}_{2^k}}(J_I) = V_{\mathbb{F}_{2^k}}(U(X_{PI}))$, and $f_i : x_i + U(X_{PI}) + 1$ can be used as the rectification.

The proofs of the Theorem V.1 and Corollary V.1 are based on the approach presented in [25] and are omitted here.

Example V.1. Let us consider a rectification problem over $R = \mathbb{F}_2[x_1,\ldots,x_n]$. Let $f_{spec} : z_s + ac + a + b + bc + c$ in R be a given polynomial specification. Fig. 2 shows a circuit that is supposed to implement f_{spec} However, the gate f_4 is faulty and should have been a AND gate.

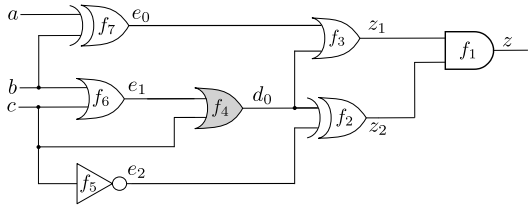

Fig. 2: An incorrect Implementation of the Spec.: $z_s + ac + a + b + bc + c$

The ideal I_{ckt} containing the polynomials for the gates of the circuit is,

$$\begin{aligned} I_{ckt} = \langle & f_1 : z + z_1 z_2, & & f_2 : z_2 + d_0 + e_2, \\ & f_3 : z_1 + e_0 d_0 + e_0 + d_0, & & f_4 : d_0 + e_1 c + e_1 + c, \\ & f_5 : e_2 + c + 1, & & f_6 : e_1 + bc + b + c, \\ & f_7 : e_0 + a + b \rangle \end{aligned}$$

We are given that f_4 with output d_0 is a potential location where the circuit can be rectified. First we must perform the check that a correction at f_4 can make the circuit a correct implementation of f_{spec}. For that, we use Theorem V.1, *i.e.* to see if $1 \in E_L + E_H + J_0$. Note that if we are given multiple such locations, we must apply Theorem V.1 to all such locations.

To perform the check, we construct two ideals $J_L = \langle f_1, f_2, f_3, f_4' : d_0, f_5, f_6, f_7, f_{spec}, f_m\rangle$ and $J_H = \langle f_1, f_2, f_3, f_4'' : d_0 + 1, f_5, f_6, f_7, f_{spec}, f_m\rangle$, where the polynomial $f_m : t(Z_m - Z_s) + 1$ ($Z_m = z$ from circuit and $Z_S = z_s$ from the specification; t is a free variable). We compute the elimination ideals $E_L = J_L \cap \mathbb{F}_q[X_{PI}]$ and $E_H = J_H \cap \mathbb{F}_q[X_{PI}]$, where X_{PI} is the

set of primary input variables ($\{a,b,c\}$). The ideals are as follows,

$$E_L = \langle c+1, b^2+b, a^2+a \rangle$$
$$E_H = \langle c, b^2+b, a+b+1 \rangle$$

We check if the $GB(E_L+E_H)$ is 1 or not, which in this example is 1 as we know we can rectify the circuit at f_4.

After determining that the circuit can indeed be rectified at the location f_4, we need to find a correct Boolean gate (polynomial) for the output d_0. The variety $V(E_L) = (abc) : \{001, 011, 101, 111\}$ and $V(E_H) = (abc) : \{100, 010\}$. The points in the difference of variety of $V(E_H)$ and $V(E_L)$ are $(abc) : \{000, 110\}$. A possible ideal-interpolant for $E_L(= J_A)$ and $E_H(= J_B)$ must have all the points in the variety $V(E_L)$ and must not have any point from the variety $V(E_H)$.

The ideal J_D computed for E_L and E_H has 2 standard monomials. As a result, we can have four possible rectifications (or 4 ideal-interpolants) at the location f_4 corresponding to the varieties: $V(E_L)$, $V(E_L) \cup \{000\}$, $V(E_L) \cup \{110\}$ and $V(E_L) \cup \{000, 110\}$.

The respective ideal interpolants are as follows,

$J_S = J_A = \langle c+1, b^2+b, a^2+a \rangle$ (smallest interpolant)
$V(J_S) = \{001, 011, 101, 111\}$
$J_1 = \langle bc+b, ac+a \rangle$
$V(J_1) = \{001, 011, 101, 111, 000\}$
$J_2 = \langle bc+b+c+1, ac+a+c+1 \rangle$
$V(J_2) = \{001, 011, 101, 111, 110\}$
$J_L = \langle ac+a+bc+b \rangle$ (largest interpolant)
$V(J_L) = \{001, 011, 101, 111, 000, 110\}$

For an ideal interpolant J_I, we can obtain a polynomial f such that $V(J_I) = V(f)$, and use that polynomial as a solution. For example, the variety of the polynomial $c+1=0$ is the same as $V(E_L)$. Therefore, we can rectify the circuit by replacing the incorrect gate polynomial at the location f_4 by the polynomial function $f_4^{corr} : d_0 + (c+1) + 1$, which constrains d_0 to be 1 at the points in $V(c+1)$. So a Boolean function that realizes the polynomial $f_4^{corr} : d_0 + c$ is a possible solution for rectifying the circuit.

As E_L is the smallest interpolant for the pair E_L and E_H, it is always an ideal-interpolant itself.

A. Experimental Results

We have performed some preliminary experiments on finite field arithmetic circuits (used in cryptography) where the implementation is different from the specification due to exactly one gate. We implement the procedure described in the previous section (Thm. V.1) using the SINGULAR symbolic algebra computation system [ver. 4-1-0][30]. The experiments were conducted on a desktop computer with a 3.5GHz Intel Core^{TM} i7-4770K Quad-core CPU, 16 GB RAM, running 64-bit Linux OS.

Modular multiplication is an important computation used in cryptography. A Mastrovito multiplier architecture can be employed for performing this computation over the finite field of 2^k elements, i.e. $\mathbb{F}_{2^k}$. Mastrovito multipliers compute $Z = A \times B \pmod{P(x)}$ where $P(x)$ is a given primitive polynomial for the datapath size k. The product $A \times B$ is computed using an array multiplier architecture, and then the result is reduced modulo $P(x)$.

The word-level output Z can be written as a polynomial in bit-level output variables as $Z = z_0 + \alpha z_1 + \alpha^2 z_2 + \cdots + \alpha^{k-1} z_{k-1}$, where α is the root of the primitive polynomial $P(x)$ and k is the datapath size. The value k also characterizes the field size as $\mathbb{F}_{2^k}$. Similarly, the word-level input variables can also be written as polynomials in respective bit-level variables. Another architecture for modular multiplication which is preferred for exponentiation operations (often required in cryptosystems). is Montgomery multiplier [31] [32] [33]. Montgomery multipliers are considered more efficient than Mastrovito multipliers as they do not require explicit reduction modulo P after each step.

Table I compares the execution time for SAT based approach [25] and our approach (Theorem V.1) for checking whether a Mastrovito multiplier can be rectified at a certain location in the circuit against a Montgomery multiplier specification. We have implemented the SAT procedure in *abc* tool [34]. We execute the command *inter* on the ON set and OFF set as described in [25]. The SAT based procedure is unable to perform the necessary unsatisfiability check beyond 9 bits. The last column in the table is the memory usage of approach.

TABLE I: Mastrovito multiplier implementation against Montgomery multiplier circuit as specification. Time-out = 5400s; MB = MegaByte(s); GB = GigaByte(s)

Field Size (k)	# of Gates		SAT	Thm. V.1	Mem
	Mas	Mont			
4	48	96	0.09	0.03	8.16 MB
8	292	319	158.34	0.41	20.36 MB
9	237	396	4,507	0.47	18.95 MB
10	285	480	TO	0.84	28.2 MB
16	1,836	1,152	TO	73.63	0.32 GB
32	5,482	4,352	TO	3621	2.4 GB

We can also perform the rectification when a polynomial specification is given instead of a specification circuit. Table II shows the result of checking whether the incorrect Mastrovito implementation can be rectified at a particular location against the word level specification polynomial $Z = A \times B$.

TABLE II: Mastrovito multiplier implementation against polynomial specification $Z = AB$. Time-out = 5400s; MB = MegaByte(s); GB = GigaByte(s)

Field Size (k)	# of Gates	Thm. V.1	Mem
4	48	0.01	7.24 MB
8	292	0.08	14.95 MB
16	1,836	4.83	0.2 GB
32	5,482	100.52	1.42 GB
64	21,813	4,989	12.25 GB

Point addition is another important operation required for the task of encryption, decryption and authentication in Elliptic Curve Cryptography (ECC). Modern approaches represent the points in projective coordinate systems, *e.g.*, the L*ó*pez-Dahab (LD) projective coordinate [35], due to which the point addition operation can be implemented as polynomials in the

field. Each polynomial can be implemented as a circuit. Table III shows the result for one of these blocks.

TABLE III: Point Addition circuit against polynomial specification $D = B^2 \cdot (C + aZ_1^2)$. Time-out = 5400s; MB = MegaByte(s); GB = GigaByte(s)

Field Size (k)	# of Gates	Thm. V.1	Mem
8	243	0.05	9.73 MB
16	1,277	3.48	88.78 MB
32	3,918	86.75	0.47 GB
64	1,5305	4,923	7.13 GB

As apparent from these experiments, the SINGULAR tool needs a large amount of memory even for 64-bit benchmarks due to its list based data-structure. We are trying alternate implementations for polynomial operations (e.g. ZBDDs based reductions in [36]) to obtain better results.

VI. Conclusion

This paper has presented a detailed theory 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 by counting the number of points in the variety of (set) difference of the largest and the smallest interpolants. An application based on circuit verification has been presented to demonstrate the use of interpolants in finite fields. With the theory of interpolants in $\mathbb{F}_q$ in place, we are now pursuing application of interpolants for circuits in a word-level setting.

References

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

[2] K. L. McMillan, "Interpolation and SAT-Based Model Checking," in *Computer Aided Verification*, July 2003, pp. 1–13.

[3] R.-R. Lee, J.-H. R. Jiang, and W.-L. Hung, "Bi-Decomposing Large Boolean Functions via Interpolation and Satisfiability Solving," in *Proc. Design Automation Conference (DAC)*, 2008, pp. 636–641.

[4] K. McMillan, "An Interpolating Theorem Prover," *Theoretical Computer Science*, vol. 345, no. 1, pp. 101 – 121, 2005, Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2004).

[5] D. Kapur, R. Majumdar, and G. Zarba, "Interpolation for Data-structures," in *Proceedings of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2006, pp. 105–116.

[6] A. Cimatti, A. Griggio, and R. Sebastiani, "Efficient Interpolant Generation in Satisfiability Modulo Theories," in *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2008, pp. 397–412.

[7] A. Griggio, "Effective Word-level Interpolation for Software Verificaion," in *Formal Methods in Computer-Aided Design (FMCAD)*, 2011, pp. 28–36.

[8] S. Gao, A. Platzer, and E. Clarke, "Quantifier Elimination over Finite Fields with Gröbner Bases," in *Algebraic Informatics: 4th International Conference, CAI*, 2011, pp. 140–157.

[9] T. Pruss, P. Kalla, and F. Enescu, "Efficient Symbolic Computation for Word-Level Abstraction from Combinational Circuits for Verification over Finite Fields," *IEEE Trans. on CAD*, vol. 35, no. 7, pp. 1206–1218, July 2016.

[10] X. Sun, P. Kalla, and F. Enescu, "Word-level Traversal of Finite State Machines using Algebraic Geometry," in *Proc. High-Level Design Validation and Test*, 2016.

[11] S. Gopalakrishnan and P. Kalla, "Optimization of Polynomial Datapaths using Finite Ring Algebra," *ACM Trans. on Design Automation of Electronic Systems, ACM-TODAES*, vol. 7, 2007, article 49.

[12] R.-R. Lee, J.-H. R. Jiang, and W.-L. Hung, "To SAT or not to SAT: Ashenhurst Decomposition in a Large Scale," in *Proc. Intl. Conf. on CAD (ICCAD)*, 2008, pp. 32–37.

[13] P. Pudlák, "Lower bounds for resolution and cutting plane proofs and monotone computations," *J. Symbolic Logic*, vol. 62, no. 2, pp. 981–998, 1997.

[14] P. Pudlák and J. Sgall, "Algebraic Models of Computation and Interpolation for Algebraic Proof Systems," in *Proof Complexity and Feasible Arithmetics, American Mathematical Society*, 1998, pp. 279–296.

[15] M. Clegg, J. Edmonds, and R. Impagliazzo, "Using the Gröbner Basis Algorithm to Find Proofs of Unsatisfiability," in *ACM Symposium on Theory of Computing*, 1996, pp. 174–183.

[16] A. Rybalchenko and V. Sofronie-Stokkermans, "Constraint Solving for Interpolation," in *Proc. Verification, Model Checking and Abstract Interpretation (VMCAI)*, 2007, pp. 346–362.

[17] L. Kovács and A. Voronkov, "Interpolation and Symbol Elimination," in *Proc. Intl. Conf. on Automated Deduction (CADE)*, 2009, pp. 199–213.

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

[19] V. D'Silva, D. Kroening, M. Purandare, and G. Weissenbacher, "Interpolant Strength," in *Verification, Model Checking and Abstract Interpretation*, 2010, pp. 129–145.

[20] G. Weissenbacher, "Interpolant Strength Revisited," in *Proc. Intl. Conf. Theory and Applications of Satisfiability Testing*, 2012, pp. 312–326.

[21] P. Rümmer and P. Subotić, "Exploring Interpolants," in *Formal Methods in Computer-Aided Design (FMCAD)*, 2013, pp. 69–76.

[22] J. C. Madre, O. Coudert, and J. P. Billon, "Automating the Diagnosis and the Rectification of Design Errors with PRIAM," in *Proc. ICCAD*, 1989, pp. 30–33.

[23] H. T. Liaw, J. H. Tsaih, and C. S. Lin, "Efficient Automatic Diagnosis of Digital Circuits," in *Proc. ICCAD*, 1990, pp. 464–467.

[24] C. C. Lin, K. C. Chen, S. C. Chang, and M. Marek-Sadowska, "Logic Synthesis for Engineering Change," in *Proc. Design Automation Conf. (DAC)*, 1995, pp. 647–652.

[25] K. F. Tang, C. A. Wu, P. K. Huang, and C. Y. Huang, "Interpolation-Based Incremental ECO Synthesis for Multi-Error Logic Rectification," in *Proc. Design Automation Conf. (DAC)*, 2011, pp. 146–151.

[26] B. H. Wu, C. J. Yang, C. Y. Huang, and J. H. R. Jiang, "A Robust Functional ECO Engine by SAT Proof Minimization and Interpolation Techniques," in *International Conference on Computer-Aided Design (ICCAD)*, 2010, pp. 729–734.

[27] A. C. Ling, S. D. Brown, S. Safarpour, and J. Zhu, "Toward Automated ECOs in FPGAs," *IEEE Trans. CAD*, vol. 30, no. 1, pp. 18–30, 2011.

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

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

[30] 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.

[31] C. Koc and T. Acar, "Montgomery Multiplication in GF(2^k)," *Designs, Codes and Cryptography*, vol. 14, no. 1, pp. 57–69, Apr. 1998.

[32] H. Wu, "Montgomery Multiplier and Squarer for a Class of Finite Fields," *IEEE Transactions On Computers*, vol. 51, no. 5, May 2002.

[33] M. Knežević, K. Sakiyama, J. Fan, and I. Verbauwhede, "Modular Reduction in GF(2^n) Without Pre-Computational Phase," in *Proceedings of the International Workshop on Arithmetic of Finite Fields*, 2008, pp. 77–87.

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

[35] J. López and R. Dahab, "Improved Algorithms for Elliptic Curve Arithmetic in GF(2^n)," in *Proceedings of the Selected Areas in Cryptography*. London, UK, UK: Springer-Verlag, 1999, pp. 201–212. [Online]. Available: http://dl.acm.org/citation.cfm?id=646554.694442

[36] U. Gupta, P. Kalla, and V. Rao, "Boolean Gröbner Basis Reductions on Datapath Circuits Using the Unate Cube Set Algebra," *IEEE Trans. CAD (to appear)*, 2018.