Highly resilient correctors for polynomials

Peter Gemmell [*] Madhu Sudan [†]

Abstract

We consider the problem of correcting programs that compute multivariate polynomials over large finite fields and give an efficient procedure to transform any program that computes a multivariate polynomial f correctly on $1/2 + \delta$ fraction of its input ($\delta > 0$) into a randomized program that computes f correctly on every input with high probability. This shows that program computing polynomials are "resilient" to a high fraction of errors. The resilience shown in this paper is better than those of the previously known correction procedures [1, 4], and is close to the information theoretic optimum. The running time of the correction procedure is polynomial in the degree of f, the number of variables, and $\frac{1}{\delta}$, where calls to the incorrect program are assessed a unit cost per call. An important consequence of this result is that the $n \times n$ permanent is resilient to errors of up to $1/2 - p(n)$ for any polynomial $p(n)$.

Keywords: Theory of computation, Program correction, Multivariate polynomials, Permanent.

1 Introduction

The study of self-correcting programs was independently started by [2] and [5] as a mechanism to transform programs that are usually correct into randomized programs which are always correct, with high probability. In this paper, we consider the problem of self-correcting programs which compute multivariate polynomials. Formally, the problem we consider here is the following:

Problem 1
`Given:`
1. *A program P that computes, on all but an ϵ fraction of inputs from the domain F^n, some multivariate polynomial f of degree at most d on its input (where F is some finite field).*
2. $b_1, b_2, \cdots, b_n \in F$

`Question:` *What is $f(b_1, \cdots, b_n)$?*

The main question investigated in this paper is how large ϵ can be so that the problem can still be solved efficiently (for large enough finite fields F). In the terminology of Gemmell et.al. [4] this is the resilience of the class of multivariate polynomials of degree d. Gemmell et. al. show that multivariate polynomials are resilient to an error close to $1/4$. The running time of their algorithm appears sensitive to the structure of the finite field; in particular, the time bound that they prove depends on the order of certain primitive roots of unity in the field. This paper improves upon their

*University of California at Berkeley. Research supported by NSF Grant CCR 88-13632.
†University of California at Berkeley. Research supported by NSF PYI Grant CCR 8896202.

result in two ways: our self-corrector can tolerate an error arbitrarily close to 1/2 and its running time does not depend in any way upon the structure of the finite field over which the computation is performed. We do require though that the finite field be large enough i.e., its size must be at least .... It may be noted that this resilience is almost optimal, in the sense that if the parameter ϵ becomes equal to half then there might be two polynomials which agree with the program at half the inputs and hence the answer is not uniquely specified.

The resilience is achieved by first reducing the problem of correcting multivariate polynomials to that of correcting univariate polynomials. This reduction is similar to the reduction of Beaver and Feigenbaum [1] in that both restrict their attention to the program's computation over a smaller subdomain, where the function can be expressed as a univariate polynomial in some input. The main improvement in our result is achieved by ensuring that on the smaller domain the program is erroneous on approximately the same fraction of the inputs as on the entire input space. Section 3 describes the details of the reduction. The second part of the correction procedure, correcting univariate polynomials when the error of the program computing the polynomial is close to half, uses an elegant technique of Berlekamp and Welch [3], which they present as part of a new mechanism for correcting Reed-Solomon codes. The main advantage of the Berlekamp-Welch mechanism is that it removes, from the running time analysis, any dependence on the structure of the finite field. We include a description of this procedure in the Appendix.

2 Definitions

Notation : We use $x \in_R \mathcal{D}$ to denote that x is chosen uniformly at random from $\mathcal{D}$.

Definition 2.1 *A Program P ϵ-**computes** f on the domain $\mathcal{D}$ if*

$$\Pr_{x \in_R \mathcal{D}} [P(x) \neq f(x)] < \epsilon$$

Definition 2.2 *An ϵ-**self-corrector** for f on a domain $\mathcal{D}$ is a randomized algorithm C that uses P as a black box, such that if P ϵ-computes f on $\mathcal{D}$, then*

$$\forall x \in \mathcal{D} \qquad \Pr\left[C^P(x) = f(x)\right] \geq 2/3$$

(Here the probability is estimated over the random coin flips of C.)

Definition 2.3 *A function f is ϵ-**resilient**, over a domain $\mathcal{D}$, if there exists an ϵ-self-corrector for f on domain $\mathcal{D}$.*

3 Self-Correcting Multivariate Polynomials

For $\delta > 0$ and d a positive integer, let F be a finite field of size $\Omega((\frac{1}{\delta} + d)^2)$. Let $f : F^n \mapsto F$ be a multivariate polynomial of degree at most d. We restate the problem to be solved here:

Problem 2
`Given :` *P that $(1/2 - \delta)$-computes f and $b_1, b_2, \cdots, b_n \in F$.*

`Output :` $f(b_1, b_2, \cdots, b_n)$.

In this section we describe a randomized reduction from Problem 2 to a problem of univariate self-correction.

We construct (by careful sampling from F^n) a domain $D \subset F^n$ parametrized by a single variable x (i.e., the points in the domain D are given by $\{D(x) | x \in F\}$), such that D satisfies the following properties:

1. The function $f'(x) \equiv f(D(x))$, is a polynomial of degree at most $2d$ in x.
2. $\hat{b} \equiv < b_1, \cdots, b_2 > \in D$; In fact we will ensure $\hat{b} = D(0)$.
3. With high probability, P computes f on approximately the same fraction of inputs from the domain D as from the domain F^n.

The three properties listed above help us as follows: The first property ensures that we are looking at univariate polynomials over the domain D, while the second property makes sure that this helps us find $f(b_1, \cdots, b_n)$. The last property ensures that we do not lose too much information about f during the process of the reduction.

The properties 1 and 3 are contrasting in nature. Property 1 requires the domain D to be nicely *structured* while Property 3 is what would be expected if D were a *random sample* of F^n. In order to get both properties we pick D to be a pairwise independent sample of F^n. In particular the following sampling scheme works nicely.

$$\text{Pick } \hat{\alpha} \text{ and } \hat{\beta} \text{ uniformly and randomly from } F^n$$

$$D_{\hat{\alpha},\hat{\beta}}(x) \equiv \hat{\alpha} * x^2 + \hat{\beta} * x + \hat{b}$$

$$D_{\hat{\alpha},\hat{\beta}} \equiv \{D_{\hat{\alpha},\hat{\beta}}(x) | x \in F\}$$

It is clear that $D_{\hat{\alpha},\hat{\beta}}$ as picked above satisfies properties 1 and 2 listed above. (Each coordinate of $D_{\hat{\alpha},\hat{\beta}}(x)$ is a polynomial of degree 2 in x; hence f' is a polynomial of degree at most $2d$ in x. Also $D_{\hat{\alpha},\hat{\beta}}(0) = \hat{b}$.) The following claim establishes that $D_{\hat{\alpha},\hat{\beta}}$ forms a pairwise independent sample of F^n.

Claim 1 *For a finite field F, $\hat{a_1}, \hat{a_2} \in F^n$, and for distinct $x_1, x_2 \in F \setminus \{0\}$,*

$$\Pr_{\hat{\alpha},\hat{\beta}}[D_{\hat{\alpha},\hat{\beta}}(x_1) = \hat{a_1} \textit{ and } D_{\hat{\alpha},\hat{\beta}}(x_2) = \hat{a_2}] = \frac{1}{|F|^{2n}}$$

Proof: For each coordinate $i \in [n]$, there exists exactly one degree 2 polynomial p_i in x, such that $p_i(0) = b_i$, $p_i(x_1) = (\hat{a_1})_i$ and $p_i(x_2) = (\hat{a_2})_i$. Thus when we pick a random polynomial p_i such that $p_i(0) = b_i$ for the ith coordinate, the probability that $p_i(x_1) = (\hat{a_1})_i$ and $p_i(x_2) = (\hat{a_2})_i$, is $\frac{1}{|F|^2}$. Since the events are independent for each coordinate, we have

$$\Pr_{\hat{\alpha},\hat{\beta}}[D_{\hat{\alpha},\hat{\beta}}(x_1) = \hat{a_1} \text{ and } D_{\hat{\alpha},\hat{\beta}}(x_2) = \hat{a_2}] = \frac{1}{|F|^{2n}}$$

□

The above claim establishes that any set S of the form $S \subset \{D_{\hat{\alpha},\hat{\beta}}(x) | x \in F \setminus \{0\}\}$ is a pairwise independent sample of F^n. The following lemma gives a well-known property of pairwise independent spaces; for completeness we include its proof in the appendix.

Lemma 2 *If S is a pairwise independent sample of elements from some domain $\mathcal{D}$ and I maps elements of $\mathcal{D}$ to the range $\{0, 1\}$. Then for positive c*

$$\Pr\left[|E_{x \in_R S}[I(x)] - E_{x \in_R \mathcal{D}}[I(x)]| \geq c/\sqrt{|S|}\right] \leq 1/c^2$$

Corollary 3 *If $S \subset F^n$ is a pairwise independent sample of t elements from F^n, and P $(1/2-\delta)$-computes f over the domain F^n, then the probability that P computes f on at least $t(1/2+\delta) - c\sqrt{t}$ points from S is at least $1 - \frac{1}{c^2}$.*

Proof: Follows from Lemma 2 by setting I as follows

$$I(x) = \begin{cases} 1 & \text{if } P(x) = f(x) \\ 0 & \text{otherwise} \end{cases}$$

□

Let k denote the number of points from S where P does not compute f. Then, if $t > (\frac{c}{\delta} + d)^2$, with probability at least $(1 - 1/c^2)$, we have that $k < (t+d)/2$. Thus with high probability, we can reduce the problem of self-correcting f at $\hat{b}$ to the problem of finding f' (recall that $f'(x) = f(D(x))$) and evaluating f' at 0. Formally, we wish to solve the following problem:

Problem 3

`Input:` *t pairs (x_i, y_i) such that for all but $k < (t+d)/2$ values of i, $y_i = f'(x_i)$, for some univariate polynomial f' of degree at most $2d$.*

`Output:` f'

This problem arises in various ways in coding theory and can be solved efficiently. If the x_i are of the form ω^i, such that $\omega^t = 1$, then the problem becomes one of correcting generalized BCH codes. If the x_i's are all the elements of F, then the above is the error correction problem for Reed-Solomon codes. In the general form as it is stated above (with no constraints on the forms of the x_i's), the problem can still be solved efficiently and directly due to an elegant method of Berlekamp and Welch, a description of which is included in the appendix. Combining the solution of Berlekamp and Welch with the reduction given above, we get the following theorem.

Theorem 4 *For $\delta > 0$, if F is finite field of size $\Omega((\frac{1}{\delta} + d)^2)$, and if f is a multivariate polynomial in n variables over F, then f is $(1/2-\delta)$-resilient. The time taken by the self-corrector that achieves this resilience is polynomial in n, $\frac{1}{\delta}$ and the degree f.*

By using Lipton's [5] observation that the permanent of an $n \times n$ matrix is a multivariate polynomial in n^2 variables of degree n, we get the following corollary :

Corollary 5 *The $n \times n$ permanent is resilient (in time polynomial in n) to errors of $1/2 - 1/p(n)$ for any polynomially growing function $p(n)$.*

4 Acknowledgments

We would like to thank Elwyn Berlekamp for his interest and the time he spent with us describing his error-correcting techniques and for his permission to include details of the technique here. We also wish to thank Manuel Blum and Umesh Vazirani for their interest which motivated most of the work here and for their time and patience in listening to our ideas. We are also thankful to Elwyn Berlekamp, Diane Hernek and Umesh Vazirani for their valuable comments on this writeup.

References

[1] D. Beaver and J. Feigenbaum. Hiding Instance in Multioracle Queries. In *Proc. Symposium on Theoretical Aspects of Computer Science*, 1990.

[2] M. Blum, M. Luby and R. Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. In *Proc. 22th ACM Symposium on Theory of Computing*, 1990.

[3] E. Berlekamp and L. Welch. Error Correction of Algebraic Block Codes. *US Patent Number 4,633,470.*

[4] P. Gemmell, R. Lipton, R. Rubinfeld, M. Sudan, A. Wigderson. Self-Testing/Correcting for Polynomials and for Approximate Functions. In *Proc. 23rd ACM Symposium on Theory of Computing*, 1991.

[5] R. Lipton. New directions in testing. In *Proc. DIMACS Workshop on Distributed Computing and Cryptography*, 1989.

A The Berlekamp-Welch Decoder

This section presents the solution to the following problem first introduced by Berlekamp and Welch as part of a novel method for decoding Reed-Solomon codes.

Problem 4

`Given` : *m pairs of points $(x_i, s_i) \in F \times F$ such that there exists a polynomial K of degree at most d such that for all but k values of i, $s_i = K(x_i)$, where $2k + d < m$.*

`Question` : *Find K*

Consider the following set of equations:

$$\exists W, K \quad \deg(W) \leq k, \deg(K) \leq d, W \neq 0, \text{ and } \forall i \quad W(x_i) * s_i = W(x_i) * K(x_i) \tag{1}$$

Any solution W, K to the above system gives a solution to Problem 4. (Notice that we can cancel W from both sides of the equation to get $s_i = f(x_i)$, except when $W(x_i) = 0$, but this can happen at most k times.) Conversely, any soluion K to Problem 4 also gives a solution to the system of equations1. (Let $B = \{x_i | s_i \neq f(x_i)\}$. Let $W(z)$ be the polynomial $\prod_{x \in B}(z - x)$. W, K form a solution to the system 1.) Thus the problem can be reduced to the problem of finding polynomials K and W which satisfy (1). Now consider the following related set of constraints

$$\exists W, N \quad \deg(W) \leq k, \deg(N) \leq k + d, W \neq 0, \text{ and } \forall i \quad W(x_i) * s_i = N(x_i) \tag{2}$$

If a solution pair N, W to (2) can be found which has the additional property that W divides N, then this would yield K and W which satisfy (1). Berlekamp and Welch show that all solutions to the system (2) have the same N/W ratio (as rational functions) and hence if equation (2) has a solution where W divides N, then any solution to the system (2) would yield a solution to the system (1). The following lemma establishes this invariant.

Lemma 6 *Let N, W and L, U be two sets of solutions to (2). Then $N/W = L/U$.*

Proof: For i, $1 \leq i \leq m$, we have

$$L(x_i) = s_i * U(x_i) \quad \text{and} \quad N(x_i) = s_i * W(x_i)$$
$$\Rightarrow L(x_i) * W(x_i) * s_i = N(x_i) * U(x_i) * s_i$$
$$\Rightarrow L(x_i) * W(x_i) = N(x_i) * U(x_i) \quad \text{(by cancellation)}$$

(Cancellation applies even when $s_i = 0$ since that implies $N(x_i) = L(x_i) = 0$.) But both $L * W$ and $N * U$ are polynomials of degree at most $2k + d$ and hence if they agree on $m > 2k + d$ points they must be identical. Thus $L * W = N * U \Rightarrow L/U = N/W$ □

All that remains to be shown is how one obtains a pair of polynomials W and N that satisfy (2). To obtain this, we substitute unknowns for the coefficients of the polynomials i.e., let $W(z) = \sum_{j=0}^{k} W_j z^j$ and let $N(z) = \sum_{j=0}^{k+d} N_j z^j$. To incorporate the constraint $W \neq 0$ we set $W_k = 1$. Each constraint of the form $N(x_i) = s_i * W(x_i)$, $i = 1 \cdots, m$ becomes a linear constraint in the $2k+d+1$ unknowns and a solution to this system can now be found by matrix inversion.

It may be noted that the algorithm presented here for finding W and N is not the most efficient known. Berlekamp and Welch [3] present an $O(m^2)$ algorithm for finding N and W, but proving the correctness of the algorithm is harder task. The interested reader is referred to [3] for a description of the more efficient algorithm.

B Pairwise Independent Sampling

For a real valued random variable Z, let $E[Z]$ denote its expected value and let $V[Z] \equiv E[(Z - E[Z])^2]$ denote its variance.

Lemma 7 *If S is a pairwise independent sample of elements from some domain $\mathcal{D}$ and I maps elements of $\mathcal{D}$ to the range $\{0,1\}$. Then for positive c*

$$\Pr\left[|\frac{\sum_{x\in S} I(x)}{|S|} - E[I(x)]| \geq c/\sqrt{|S|} \right] \leq 1/c^2$$

Proof: Since S is a pairwise independent collection of elements from $\mathcal{D}$ we have

$$E\left[\frac{\sum_{x\in S} I(x)}{|S|}\right] = E\left[I(x)\right] \text{ and } V\left[\frac{\sum_{x\in S} I(x)}{|S|}\right] = \frac{V\left[I(x)\right]}{|S|}$$

By Chebychev's inequality, we have for any random variable Y

$$\Pr\left[|Y - E(Y)| > c\sqrt{V(Y)} \right] \leq 1/c^2$$

Applying this inequality to the variable $\frac{\sum_{x\in S} I(x)}{|S|}$, we get

$$\Pr\left[|\frac{\sum_{x\in S} I(x)}{|S|} - E[I(x)]| \geq c/\sqrt{|S|} \right] \leq 1/c^2$$

$\square$