Algorithms for Modular Counting of Roots of Multivariate Polynomials

Parikshit Gopalan*
Georgia Tech.
Atlanta, GA 30332-0280.
parik@cc.gatech.edu

Venkatesan Guruswami†
University of Washington,
Seattle, WA 98195-2350.
venkat@cs.washington.edu

Richard J. Lipton*
Georgia Tech,
Atlanta, GA 30332-0280.
rjl@cc.gatech.edu

April 28, 2006

Abstract

Given a multivariate polynomial $P(X_1, \cdots, X_n)$ over a finite field $\mathbb{F}_q$, let $N(P)$ denote the number of roots over $\mathbb{F}_q^n$. The modular root counting problem is given a modulus r, to determine $N_r(P) = N(P) \bmod r$. We study the complexity of computing $N_r(P)$, when the polynomial is given as a sum of monomials. We give an efficient algorithm to compute $N_r(P)$ when the modulus r is a power of the characteristic of the field. We show that for all other moduli, the problem of computing $N_r(P)$ is NP-hard. We present some hardness results which imply that that our algorithm is essentially optimal for prime fields. We show an equivalence between maximum-likelihood decoding for Reed-Solomon codes and a root-finding problem for symmetric polynomials.

1 Introduction

Given a polynomial $P(X_1, \cdots, X_n)$ of degree d in n variables over a field $\mathbb{F}_q$ of characteristic p in sparse representation, i.e. written as a sum of m monomials, let $N(P)$ denote the number of solutions to $P(X_1, \cdots, X_n) = 0$ over $\mathbb{F}_q$. The problem of computing $N(P)$ exactly is known to be #P-complete. In this paper we study the complexity of the modular counting problem, which is given a modulus r, compute $N_r(P) = N(P) \bmod r$. We also study the related problem of deciding whether $N(P) > 0$ i.e. if the equation $P = 0$ is feasible over $\mathbb{F}_q$.

1.1 Problem History and Motivation

The problem of counting roots of a polynomial over a finite field is a fundamental and well studied problem in algebra with applications to several areas including coding theory and cryptography [13]. Ehrenfeucht and Karpinski showed that computing $N(P)$ is #P complete even when we restrict the degree to be three [4]. Hence one has to look for approximation algorithms, or algorithms that work for some special class of polynomials.

*Supported by NSF grant CCR-3606B64.
†Supported in part by NSF grant CCF-0343672 and a Sloan Research Fellowship.

Randomized algorithms for computing $N(P)$ approximately were given by Karpinski and Luby for $\mathbb{F}_2$ [9] and Grigoriev and Karpinksi for $\mathbb{F}_q$ [5]. A more randomness efficient algorithm for $\mathbb{F}_2$ was given by Luby, Velikovic and Wigderson [14]. The problem has been extensively studied for equations in few variables. Schoof gives an exact algorithm to count the number of points on an elliptic curve over $\mathbb{F}_q$ [19]. The counting problem for plane curves has been well studied [18, 1, 7]. Von zur Gathen *et.al* show that the counting problem for sparsely represented curves is #P-complete [21]. Huang and Wong give probabilistic algorithms for both the feasibility and approximate counting problems [8]. Their algorithm is polynomial in the degree of P but exponential in the number of variables n.

The related problem of computing the Zeta-function of an algebraic variety is well studied. Lauder and Wan give a polynomial time algorithm for this problem when the characteristic p is small and the number of variables is fixed [11]. There has been considerable work on this topic in computational number theory, see [10, 12, 23] and the references therein for more details.

The problem of computing $N_r(P)$ has been studied in the literature in many different contexts. A famous theorem due to Chevalley and Warning states that if P is a polynomial over a field $\mathbb{F}_q$ of characteristic p and $\deg(P) < n$, then $N_p(P) \equiv 0$ [13]. This was considerably strengthened by Ax who shows that if $k = \lceil \frac{n-d}{d} \rceil$ then $N_{q^k}(P) = 0$ (see [22]). This was extended to systems with many equations by Katz. Wan gives a simpler proof of the Ax-Katz theorem over $\mathbb{F}_p$ [22]. Moreno and Moreno observed that by reducing a system of equations over $\mathbb{F}_q$ to a system over $\mathbb{F}_p$ and then applying the Ax-Katz bound for prime fields, one can get a bound that often beats the Ax-Katz bound over $\mathbb{F}_q$. They introduced the notion of p-weight degree $w_p(P)$ of a polynomial which is upper bounded by $\deg(P)$ (see Section 3). They showed that if $q = p^h$, and if $k = \lceil h\frac{n-w_p(P)}{w_p(P)} \rceil$ then $N_{p^k}(P) = 0$. Schoof's algorithm for counting the number of points on an elliptic curve proceeds by first computing $N_r(P)$ for several small primes r and using Chinese Remaindering to recover $N(P)$ [19]. Wan describes methods to compute the reduction of the zeta-function of a curve modulo p^k [23]. Thus all these results are related to the problem of computing $N_r(P)$ for various moduli r. In this work, we address the computational complexity of computing $N_r(P)$.

1.2 Our Results

We give a simple algorithm for computing $N_{p^k}(P)$ given $P(X_1, \cdots, X_n)$ in sparse representation over a field $\mathbb{F}_q$ where $q = p^h$. The running time of our algorithm is $O(nm^{2qk})$ where m is the sparsity of the polynomial i.e. the number of monomials with non-zero coefficients. The algorithm proceeds in two steps. There is a lifting step, where we define an indicator polynomial for the zeroes of the polynomial over $\mathbb{F}_q^n$, and *lift* it to an indicator polynomial modulo p over a ring of characteristic 0. We then amplify this polynomial to get an indicator modulo p^k and sum each monomial modulo p^k over the lift of $\mathbb{F}_q^n$. This high level structure is similar to the proof of the Chevalley-Warning theorem [13] and Wan's proof of the Ax-Katz theorem over prime fields [22]. For a prime field, we lift the problem from $\mathbb{F}_p$ the integers. For non-prime fields, the lifting is from $\mathbb{F}_q$ to an appropriate ring of algebraic integers.

We also present a more naive algorithm to compute $N_{p^k}(P)$ for a polynomial over $\mathbb{F}_q$, which works by reducing the problem to the $\mathbb{F}_p$ case. While the running time of this algorithm is exponential in the degree of the polynomial, it is only singly exponential in the extension degree h of $\mathbb{F}_q$ over $\mathbb{F}_p$, as opposed to the previous algorithm which is doubly exponential in h. This suggests that there might be an algorithm over $\mathbb{F}_q$ with running time singly exponential in h and polynomial in the degree. Such an algorithm has been found subsequently by Wan [25], see the discussion in Section 6.

The amplification step of our algorithm uses constructions of low-degree modulus amplifying polynomials from complexity theory. Such polynomials were first constructed for the proof of Toda's theorem [20]. Subsequently, better constructions were given by Yao [26] and by Beigel and Tarui [3] to prove upper bounds on a circuit class called ACC. Ours appears to be the first work to make algorithmic use of these polynomials. The construction of Beigel *et.al* gives degree $2k-1$. We show a matching lower bound on the degree of any such polynomial using Mason's theorem.

On the hardness side, we show that over any field $\mathbb{F}_q$ of characteristic p, if r is not a power of p, the problem of computing $N_r(P)$ given the polynomial P in sparse representation is NP-hard under randomized reductions. More precisely, the problem of deciding whether $N_r(P)$ belongs to a particular congruence class modulo r is NP-hard. We study the related feasibility problem for sparse polynomials, which is to decide if $N(P) > 0$. While the problem is easy for constant size fields, we show that it becomes NP-complete, when either the characteristic p or the extension degree h becomes large. As consequence of this, we show that exponential dependence on p and h in our algorithms is unavoidable, since the corresponding counting problems are hard when these parameters are large. Also, when $k = n$, then $N_{p^k}(P) = N(P)$ hence having k in the exponent is also unavoidable. Thus our algorithm for $\mathbb{F}_p$ with running time is $O(nm^{2pk})$ is asymptotically optimal.

Finally we pose the problem of feasibility for symmetric polynomials over $\mathbb{F}_q$, which are sparsely represented over the basis of elementary symmetric polynomials. Our motivation for studying this problem comes from the maximum-likelihood decoding problem for Reed-Solomon codes. Building on work of Guruswami and Vardy [6], we show that this decoding problem is equivalent to a certain root-finding problem for symmetric multilinear polynomials over $\mathbb{F}_q$.

This paper is organized as follows: in Section 2 we discuss modulus amplifying polynomials. We present our algorithmic results in Section 3 and our hardness results in Section 4. We discuss maximum-likelihood decoding of Reed-Solomon codes in Section 5. An extended abstract of this paper appears in LATIN 2006.

2 On Modulus Amplifying Polynomials

Definition 1 *A univariate integer polynomial $A_k(X)$ is k-modulus amplifying if for every integer r, the following condition holds:*

$$\begin{aligned} x \equiv 0 \bmod r \quad &\Rightarrow \quad A_k(x) \equiv 0 \bmod r^k \\ x \equiv 1 \bmod r \quad &\Rightarrow \quad A_k(x) \equiv 1 \bmod r^k \end{aligned} \tag{1}$$

We use the following Lemma by Beigel and Tarui.

Lemma 1 *[3] The polynomial $A_k(X) \in \mathbb{Z}[X]$ is k-modulus amplifying iff:*

$$A_k(X) \equiv \begin{cases} 0 & \bmod X^k \\ 1 & \bmod (X-1)^k \end{cases} \tag{2}$$

Beigel *et.al* derive the polynomial $A_k(X)$ by truncating the power series expansion of $(1-X)^{-k}$. We give an alternate derivation of their construction below.

Lemma 2 *[3] The following polynomial is k-modulus amplifying:*

$$A_k(X) = X^k \sum_{i=0}^{k-1} \binom{2k-1}{k+i} X^i (1-X)^{k-1-i}$$

PROOF: Note that $X + (1-X) = 1$. Raising both sides to power $2k-1$, we get

$$1 = (X + (1-X))^{2k-1} = \sum_{i=0}^{2k-1} \binom{2k-1}{i} X^i (1-X)^{2k-1-i}$$

We divide the summation into two parts based on $i < k$ and $i \geq k$. In these two parts the power of $(1-X)$ and X respectively is at least k.

$$\begin{aligned} 1 &= (1-X)^k \sum_{i<k} \binom{2k-1}{i} X^i (1-X)^{k-1-i} \\ &\quad + X^k \sum_{i \geq k} \binom{2k-1}{i} X^{i-k} (1-X)^{2k-1-i} \\ &= V(X)(X-1)^k + U(X)X^k \end{aligned}$$

We set

$$A_k(X) = U(X)X^k = 1 - V(X)(X-1)^k$$

It has degree $2k-1$ and satisfies Equation (2). □

Since $A_k(X)$ must be divisible by X^k, it must have degree at least k. The running time of our algorithms depends exponentially on the degree of $A_k(X)$ so even a factor 2 saving in the degree would be significant. But we will show that the degree needs to be $2k-1$. The proof uses Mason's theorem which proves the ABC-conjecture for polynomials [15]. Let $z(P)$ denote the number of *distinct* roots of a polynomial over the complex numbers.

Mason's Theorem. [15] *Given polynomials $A(X), B(X), C(X) \in \mathbb{Z}[X]$ which are relatively prime such that $A(X) + B(X) = C(X)$,*

$$\max\{\deg(A), \deg(B), \deg(C)\} \leq z(ABC) - 1$$

Here $z(ABC)$ is the number of distinct complex roots of $A(X)B(X)C(X)$.

Lemma 3 *If $A_k(X)$ is k-modulus amplifying, then* $\deg(A_k) \geq 2k-1$.

PROOF: Note that $A(X) = U(X)X^k = V(X)(X-1)^k + 1$ by Lemma 1. Hence

$$U(X)X^k - V(X)(X-1)^k = 1$$

Assume that $\deg(U) = d$. Since the leading term cancels out with the leading term of $V(X)(X-1)^k$, we have $\deg(V) = d$. We set

$$A(X) = U(X)X^k, \quad B(X) = V(X)(X-1)^k, \quad C(X) = 1$$

It is clear that these are relatively prime, so we can apply Mason's theorem. Note that the maximum degree is $d + k$. The product polynomial is

$$A(X)B(X)C(X) \;=\; U(X)V(X)X^k(X-1)^k$$

which can have at most $2 + 2d$ distinct roots over the complex numbers. Hence

$$d + k \leq 2d + 2 - 1 \quad \Rightarrow \quad k - 1 \leq d$$

This shows that the degree of $A_k(X) = U(X)X^k$ is at least $2k - 1$. □

We note that modulus amplifying polynomials work for every modulus r. In our algorithms, it suffices that the polynomial is amplifying for a specific modulus p, the characteristic of $\mathbb{F}_q$. It is interesting to ask if the same lower bound holds asymptotically for polynomials that are amplifying only for the modulus p.

3 Algorithms for Counting Roots

We use the notation $\mathbf{X} = (X_1, \cdots, X_n)$ for a vector of variables and $\mathbf{x} = (x_1, \cdots, x_n)$ for a vector of constants. Given a vector $D = (d_1, \cdots, d_n)$ in $\mathbb{Z}^n$, we use $\mathbf{X}^D$ to denote the monomial $\prod_i X_i^{d_i}$.

3.1 Modular Counting over Prime Fields

We define a *lift* of $\mathbb{F}_p$ to $\mathbb{Z}$ which maps $i \in \mathbb{F}_p$ to the integer i. We use the same notation for $i \in \mathbb{F}_p$ and its lift in $\mathbb{Z}$, whether i belongs to $\mathbb{F}_p$ or $\mathbb{Z}$ will be clear from the context. We can similarly lift vectors (polynomials) over $\mathbb{F}_p$ to vectors (polynomials) over $\mathbb{Z}$.

The input to the algorithm is a polynomial $P(\mathbf{X}) = \sum_D c_D \mathbf{X}^D$ over $\mathbb{F}_p$. We first lift it to $\mathbb{Z}[\mathbf{X}]$ and then define a polynomial $Q(\mathbf{X}) \in \mathbb{Z}[\mathbf{X}]$ using:

$$Q(\mathbf{X}) = A_k(1 - P(\mathbf{X})^{p-1})$$

Let $Q(\mathbf{X}) = \sum_E c_E \mathbf{X}^E$ where the sum is over at most $2km^{(p-1)(2k-1)}$ monomials. $Q(\mathbf{X})$ satisfies the following relations for $\mathbf{x} \in \mathbb{F}_p^n$:

$$\begin{aligned} P(\mathbf{x}) = 0 \text{ over } \mathbb{F}_p \quad &\Rightarrow \quad Q(\mathbf{x}) \equiv 1 \bmod p^k \text{ over } \mathbb{Z} \\ P(\mathbf{x}) \neq 0 \text{ over } \mathbb{F}_p \quad &\Rightarrow \quad Q(\mathbf{x}) \equiv 0 \bmod p^k \text{ over } \mathbb{Z} \end{aligned}$$

$$\text{Hence} \quad N(P) \;\equiv\; \sum_{\mathbf{x} \in \mathbb{F}_p^n} Q(\mathbf{x}) \;\equiv\; \sum_{\mathbf{x} \in \mathbb{F}_p^n} \sum_E c_E \mathbf{x}^E \;\equiv\; \sum_E c_E \sum_{\mathbf{x} \in \mathbb{F}_p^n} \mathbf{x}^E \bmod p^k$$

where the sum is over the lift of $\mathbb{F}_p^n$ to $\mathbb{Z}^n$. To sum each monomial, observe that

$$\sum_{\mathbf{x} \in \mathbb{F}_p^n} \mathbf{x}^E \;\equiv\; \prod_{i=1}^{n} \left(\sum_{x_i=0}^{p-1} x_i^{e_i} \right)$$

Each e_i is at most $2pk$. Note that we cannot use the substitution $X^p = X$ since this need not hold modulo p^k. Thus the time to compute the sum for each monomial is bounded by $O(np^2k^2)$. Hence we can compute $N_{p^k}(P)$ in time $O(2km^{(p-1)(2k-1)}np^2k^2) = O(m^{2pk}n)$. We summarize the algorithm below.

Computing $N_{p^k}(P)$ over $\mathbb{F}_p$.
Input: $P(\mathbf{X}) = \sum_D c_D \mathbf{X}^D$ over $\mathbb{F}_p$.
1. Compute the integer polynomial

$$Q(\mathbf{X}) \;=\; A_k(1 - P(\mathbf{X})^{p-1}) \;=\; \sum c_E \mathbf{X}^E$$

2. Compute $c_E \sum_{\mathbb{F}_p^n} \mathbf{X}^E \bmod p^k$ for each E and output the sum.

Theorem 1 *Given a polynomial $P(\mathbf{X}) \in \mathbb{F}_p[\mathbf{X}]$ in n variables with m monomials, there is an $O(m^{2pk}n)$ algorithm to compute $N_{p^k}(P)$. For fixed p and k, $N_{p^k}(P)$ can be computed in polynomial time.*

3.2 Modular Counting over Arbitrary Fields

Let $q = p^h$ and let $\mathbb{F}_q = \mathbb{F}_p(\alpha)$ be a degree h field extension of $\mathbb{F}_p$ generated by α. Let $H(X) \in \mathbb{F}_p[X]$ be the monic irreducible polynomial of degree h so that $P(\alpha) = 0$ over $\mathbb{F}_q$. We will assume that the $H(X)$ is given as input. We lift $H(X)$ to the integers, and then define the quotient

$$\mathbb{Z}(\alpha) = \frac{\mathbb{Z}[X]}{(H(X))}$$

where α is a formal root of $H(X)$ over $\mathbb{Z}$. In fact $H(X)$ is irreducible over $\mathbb{Z}$, but we will not use this fact.

Lemma 4 *There is an isomorphism between $\mathbb{Z}[\alpha]/(p)$ and $\mathbb{F}_q$.*

PROOF: Note that

$$\frac{\mathbb{Z}[\alpha]}{(p)} \;=\; \frac{\mathbb{Z}[X]}{(H(X), p)} \;=\; \frac{\mathbb{F}_p[X]}{(H(X))}$$

where in the last expression, $H(X)$ is taken to be a polynomial over $\mathbb{F}_p$. By our choice of $H(X)$, this quotient is precisely $\mathbb{F}_q = \mathbb{F}_p(\alpha)$. It is easy to check that mapping $\alpha \in \mathbb{Z}[\alpha]/(p)$ to $\alpha \in \mathbb{F}_q$ gives an isomorphism. □

Note that this idea of first going modulo p is used to characterize primes in the ring of Gaussian integers [2]. We can lift $\mathbb{F}_q$ to $\mathbb{Z}(\alpha)$ by sending $\alpha \in \mathbb{F}_q$ to $\alpha \in \mathbb{Z}(\alpha)$ and sending $i \in \mathbb{F}_p$ to $i \in \mathbb{Z}$. We now describe the algorithm for computing $N_{p^k}(P)$ over $\mathbb{F}_q$. Given a polynomial $P(\mathbf{X}) = \sum_D c_D \mathbf{X}^D$ over $\mathbb{F}_q$, lift it to $\mathbb{Z}(\alpha)[\mathbf{X}]$ and then define a polynomial $Q(\mathbf{X}) \in \mathbb{Z}(\alpha)[\mathbf{X}]$ using:

$$Q(\mathbf{X}) = (1 - P(\mathbf{X})^{p-1})$$

Let $Q(\mathbf{X}) = \sum_E c_E \mathbf{X}^E$ where the sum is over at most $m^{(p-1)(2k-1)}$ monomials. $Q(\mathbf{X})$ satisfies the following conditions

$$\begin{aligned} P(\mathbf{x}) = 0 \text{ over } \mathbb{F}_q \quad &\Rightarrow \quad Q(\mathbf{x}) \equiv 1 \bmod p \text{ over } \mathbb{Z}(\alpha) \\ P(\mathbf{x}) \neq 0 \text{ over } \mathbb{F}_q \quad &\Rightarrow \quad Q(\mathbf{x}) \equiv 0 \bmod p \text{ over } \mathbb{Z}(\alpha) \end{aligned}$$

Finally define $R(\mathbf{X}) \in \mathbb{Z}(\alpha)[\mathbf{X}]$ as

$$R(\mathbf{X}) = A_k(Q(\mathbf{X}))$$

It is easy to see that $A_k(X)$ is modulus amplifying even for $\mathbb{Z}(\alpha)$. Hence

$$\begin{aligned} P(\mathbf{x}) = 0 \text{ over } \mathbb{F}_q &\Rightarrow R(\mathbf{x}) \equiv 1 \bmod p^k \text{ over } \mathbb{Z}(\alpha) \\ P(\mathbf{x}) \neq 0 \text{ over } \mathbb{F}_q &\Rightarrow R(\mathbf{x}) \equiv 0 \bmod p^k \text{ over } \mathbb{Z}(\alpha) \\ \text{Hence } N(P) &\equiv \sum_{\mathbf{x} \in \mathbb{F}_q^n} R(\mathbf{x}) \bmod p^k \end{aligned}$$

We can compute this sum by writing $R(\mathbf{X}) = \sum_E c_E \mathbf{x}^E$ and summing each monomial individually over the lift of $\mathbb{F}_q$. It is easy to see that $R(\mathbf{X})$ has at most $2km^{(2k-1)(q-1)}$ monomials. So the running time is bounded by $O(nm^{2qk})$.

Computing $N_{p^k}(P)$ over $\mathbb{F}_q$.
Input: $\mathbb{F}_q = \mathbb{F}_p(\alpha)$ given by the irreducible polynomial $H(X)$ of α over $\mathbb{F}_p$. $P(\mathbf{X}) = \sum_D c_D \mathbf{X}^D$ over $\mathbb{F}_q$.

1. Let α satisfy $H(\alpha) = 0$ over $\mathbb{Z}$. Lift $P(\mathbf{X})$ to $\mathbb{Z}(\alpha)[\mathbf{X}]$.
2. Compute the polynomial $R(\mathbf{X}) \in \mathbb{Z}(\alpha)$ given by

$$R(\mathbf{X}) = A_k(1 - P(\mathbf{X})^{q-1}) = \sum c_E \mathbf{X}^E$$

3. Compute $c_E \sum_{\mathbb{F}_q^n} \mathbf{X}^E \bmod p^k$ for each E and output the sum.

Here the sum is over the lift of $\mathbb{F}_q$ to $\mathbb{Z}(\alpha)$. We treat α as a formal symbol satisfying $H(\alpha) = 0$ over $\mathbb{Z}$. All arithmetic operations are performed modulo p^k.

Theorem 2 *Given a polynomial $P(\mathbf{X}) \in \mathbb{F}_q[\mathbf{X}]$ in n variables with m monomials, there is an $O(nm^{2qk})$ algorithm to compute $N_{p^k}(P)$. For fixed q, p and k, $N_{p^k}(P)$ can be computed in polynomial time.*

3.3 Reduction from $\mathbb{F}_q$ to $\mathbb{F}_p$

Let $P(\mathbf{X}) = \sum_D c_D \mathbf{X}^D$ where $c_D \in \mathbb{F}_q$ be the input. For each variable X_i we substitute

$$X_i = Y_{i,0} + Y_{i,1}\alpha \cdots Y_{i,h-1}\alpha^{h-1} \quad Y_{i,j} \in \mathbb{F}_p$$

Thus we replace the monomial $\prod_i X_i^{d_1}$ of total degree d by

$$\prod_i X_i^{d_i} = \prod_i (Y_{i,0} + Y_{i,1}\alpha \cdots Y_{i,h-1}\alpha^{h-1})^{d_i}$$

Naively, this expression has sparsity h^d. We can improve this bound using the notion of p-weight degree due to Moreno and Moreno [16].

Definition 2 *Given an integer $d = d_0 + d_1 p \cdots + d_t p^t$, define its p-weight $\sigma(d) = \sum_j d_j$. The p-weight degree of a monomial $\mathbf{X}^D = \prod_i X_i^{d_i}$ is defined as $w_p(\mathbf{X}^D) = \sum_i \sigma(d_i)$. The p-weight degree $w_p(P)$ of a polynomial $P(\mathbf{X})$ is the maximum of the p-weight degree over all monomials.*

Note that $\sigma_p(d) \leq d$, hence the p-weight degree of a monomial is bounded by its degree. Returning to the monomial $\prod_i X_i^{d_i}$, let $d_i = \sum_t d_{it} p^t$. Then,

$$\begin{aligned}
\prod_i X_i^{d_i} &= \prod_i \left(\sum_{j=0}^{h-1} Y_{i,j} \alpha^j \right)^{\sum_t d_{it} p^t} \\
&= \prod_{i,t} \left(\sum_{j=0}^{h-1} Y_{i,j} \alpha^j \right)^{d_{it} p^t} \\
&= \prod_{i,t} \left(\sum_{j=0}^{h-1} Y_{i,j}^{p^t} \alpha^{j p^t} \right)^{d_{it}} \\
&= \prod_{i,t} \left(\sum_{j=0}^{h-1} Y_{i,j} \alpha^{j p^t} \right)^{d_{it}} \quad \text{since } Y_{i,j} \in \mathbb{F}_p
\end{aligned}$$

Let $c_D = \sum_u c_u \alpha^u$. Then

$$c_D \prod_i X_i^{d_i} = \sum_{u=0}^{h-1} c_u \alpha^u \prod_{i,t} \left(\sum_{j=0}^{h-1} Y_{i,j} \alpha^{j p^t} \right)^{d_{it}} = \sum_E c_E \mathbf{Y}^E \alpha^{f(E)} \tag{3}$$

where $c_E \in \mathbb{Z}_p$ and $f(E)$ is some function of E. This summation involves $h^{1+\sum d_{it}} = h^{w_p(\mathbf{X}^D)+1}$ monomials. Repeating this for every monomial, we get a sum over at most mh^w monomials, where $w = w_p(P) + 1$:

$$P(\mathbf{Y}) = \sum_E c_E \mathbf{Y}^E \alpha^{f(E)} \tag{4}$$

Since $\{\alpha^0, \cdots, \alpha^{h-1}\}$ is a basis for $\mathbb{F}_q$ over $\mathbb{Z}_p$, we can each $\alpha^{f(E)}$ as a linear combination over this basis. Grouping the various powers of α gives

$$P(\mathbf{Y}) = P_0(\mathbf{Y}) + P_1(\mathbf{Y})\alpha + \cdots + P_{h-1}(\mathbf{Y})\alpha^{h-1} \tag{5}$$

Each polynomial $P_\ell(\mathbf{Y})$ has sparsity at most mh^w, since each monomial from Equation (4) contributes at most one monomial to $P_\ell(\mathbf{Y})$. Since the powers of α are linearly independent over $\mathbb{F}_p$, this sum is 0 iff for $0 \leq \ell \leq h-1$ the coefficient of α^ℓ is 0 over $\mathbb{F}_p$. This implies that for each ℓ, we must have $P_\ell(\mathbf{Y}) = 0$ over $\mathbb{F}_p$. We can combine these into a single equation $Q(\mathbf{Y}) = 0$ over $\mathbb{F}_p$ where

$$Q(\mathbf{Y}) = 1 - \prod_{\ell=0}^{h-1} \left(1 - P_\ell(\mathbf{Y})^{p-1}\right)$$

The roots of $Q(\mathbf{Y})$ over $\mathbb{F}_p$ are in one-to-one correspondence with the roots of $P(\mathbf{X})$ over $\mathbb{F}_q$, so we can use the $\mathbb{F}_p$ algorithm on $Q(\mathbf{Y})$. Since $Q(\mathbf{Y})$ only takes 0/1 values we can directly apply A_k to $1 - Q(\mathbf{Y})$. The total running time can be bounded by $O(n(mh^w)^{2hpk})$. In addition to p, k there is an exponential dependence on h and the (p-weight) degree.

Computing $N_{p^k}(P)$ over $\mathbb{F}_q$.
Input: $P(\mathbf{X}) = \sum_D c_D \mathbf{X}^D$ over $\mathbb{F}_q$, α such that $\mathbb{F}_q = \mathbb{Z}_p[\alpha]$.
1. Substitute $X_i = \sum_j Y_{i,j}\alpha^j$ with $Y_{ij} \in \mathbb{Z}_p$ and $c_D = \sum_u c_u \alpha^u$ with $c_u \in \mathbb{Z}_p$ to get

$$P(\mathbf{Y}) = \sum c_E \mathbf{Y}^E \alpha^{f(E)} \qquad c_E \in \mathbb{Z}_p$$

2. Write each $\alpha^{f(E)}$ as a $\mathbb{Z}_p$-linear combination of $\{1, \alpha, \cdots, \alpha^{h-1}\}$ to get

$$P(\mathbf{Y}) = \sum_{\ell=0}^{h-1} P_\ell(\mathbf{Y})\alpha^\ell \qquad P_\ell(\mathbf{Y}) \in \mathbb{Z}_p(\mathbf{Y})$$

3. Compute the integer polynomial

$$R(\mathbf{Y}) = A_k \left(\prod_{\ell=0}^{h-1} \left(1 - P_\ell(\mathbf{Y})^{p-1}\right) \right) = \sum_F c_F \mathbf{Y}^F$$

4. Compute $c_F \sum_{\mathbb{F}_p^{nh}} \mathbf{y}^F \bmod p^k$ for each F and output the sum.

Theorem 3 *Let $P(\mathbf{X})$ be a polynomial in n variables with m monomials over $\mathbb{F}_q$ where $q = p^h$. Let $w_p(P)$ be p-weight degree and $w = w_p(P) + 1$. There is an $O(n(mh^w)^{2hpk})$ algorithm to compute $N_{p^k}(P)$.*

4 Hardness Results for Counting

In all the results in this section, the polynomial is given in sparse representation. We refer the reader to the book by Papadimitriou [17] for the necessary complexity-theoretic definitions.

Theorem 4 *Let $\mathbb{F}_q$ be a finite field of characteristic p. Assume that r is not a power of p. Given a polynomial $P(\mathbf{X})$ over $\mathbb{F}_q$, the problem of computing $N_r(P)$ is* NP*-hard under randomized reductions.*

An instance of QE over $\mathbb{F}_q$ consists of m quadratic equations $Q_1(\mathbf{X}) = 0, \cdots, Q_m(\mathbf{X}) = 0$. It is well known that deciding if an instance of QE is feasible is NP-complete. An instance of UQE consists of a system of quadratic equation with the promise that in the Yes case, there is a unique solution. Similarly define U3SAT to be the unique version of 3SAT. Valiant and Vazirani show that U3SAT is NP-complete under randomized reductions [17]. We give a reduction from U3SAT to UQE.

Lemma 5 UQE *is* NP*-complete under randomized reductions over any field.*

PROOF: We give a reduction from 3SAT to QE. The reduction itself is folklore, we just need to verify that it preserves the number of solutions. Assume that we have a 3SAT formula $\varphi(\mathbf{X})$ with clauses $C_1, \cdots, C_m$. We add auxiliary variables $Y_1, \cdots, Y_m$ and add the constraints

$$X_i^2 = X_i, \;\; Y_i^2 = Y_i$$

This ensures that all the variables need to be $0/1$. Assume that $C_1 = X_1 \vee X_2 \vee X_3$. We replace this by

$$Y_1 = X_1 \vee X_2, \;\; Y_1 \vee X_3 = 1$$

This is done by the equations

$$Y_1 = X_1 + X_2 - X_1X_2, \;\; Y_1 + X_3 - Y_1X_3 = 1$$

We perform a similar substitution for every clause. It is clear that this instance of QE is feasible iff $\varphi(\mathbf{X})$ is feasible. Further, this reduction preserves the number of solutions since the values of the auxiliary variables $Y_1, \cdots, Y_m$ are uniquely determined from the values assigned to $X_1, \cdots, X_n$.

Thus starting with an instance of U3SAT, we get an instance of UQE. Since U3SAT is NP-complete under randomized reductions [17], we infer the hardness of UQE for any $\mathbb{F}_q$. □

We now prove Theorem 4. The reduction used is the same reduction used by Ehrenfeucht and Karpinski to show the #P-completeness of computing $N(P)$ [4], except that the uniqueness of the solution in the Yes case is crucial.

PROOF: Given an instance $X_1, \cdots, X_n, Q_1 = 0, \cdots, Q_m = 0$ of UQE, we add new variables $Z_1, \cdots, Z_m$ and let $P(X_1, \cdots, X_n, Z_1, \cdots, Z_m)$ be the equation

$$\sum_{i=1,\cdots,m} Z_i Q_i(X_1, \cdots, X_n) = 1$$

Assume that $x_1, \cdots, x_n$ is a solution to the system of quadratic equations. Then the above equation reduced to $0 = 1$, so there is no solution. On the other hand if some equation say Q_m is unsatisfied, then we are left with a linear equation

$$\sum_{i=1,\cdots,m} c_i Z_i = 1, \qquad c_m \neq 0$$

Since $c_m \not\equiv 0$, we can pick values for $Z_1, \cdots, Z_{m-1}$ arbitrarily, and then pick Z_m so that the above equation is satisfied. Thus when the instance of UQE is satisfiable,

$$N(P) \;=\; (q^n - 1)q^{m-1}$$

whereas when it is unsatisfiable

$$N(P) \;=\; q^{n+m-1}$$

Since $q = p^h$, if r is not a power of p,

$$(q^n - 1)q^{m-1} \;\not\equiv\; q^{n+m-1} \bmod r$$

Hence an algorithm to compute $N_r(P)$ can be used to solve UQE. More precisely, deciding whether $N(P)$ lies in a particular congruence class modulo r is NP-hard. □

In fact, our reduction shows that if r is not a power of p, computing $N_r(P)$ is as hard as counting 3SAT solutions modulo r.

4.1 Hardness Results for Feasibility

The feasibility problem is, given a polynomial $P(\mathbf{X})$ over $\mathbb{F}_q$ does it have a root? When the field size q is constant, there is a simple algorithm for feasibility [5]. Compute $P(\mathbf{X})^{q-1}$. Reduce the degree in each variable to $q-1$ using $X_i^q = X_i$. If we are left with 1, then $P(\mathbf{X})$ has no roots. Else it has a root. Correctness follows from the fact that every function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ is computed by a unique

polynomial with degree in each variable at most $q-1$. If $P(\mathbf{X})$ has no zeros, then $P(\mathbf{X})^{q-1}$ computes the constant function 1. This algorithm runs in time $O(nm^q)$.

Daqing Wan has observed that for $q = p^h$, the running time can in fact be bounded by $O(nm^{ph})$ [24]. Observe that

$$\begin{aligned} q-1 &= (p-1)(1+p\cdots+p^{h-1}) \\ \text{Hence} \quad P(X)^{q-1} &= \prod_{i=1}^{h-1} P(X)^{(p-1)p^i} \\ &= \prod_{i=1}^{h-1} P(X^{p^i})^{p-1} \end{aligned}$$

We compute this product and then apply the substitution $X_i^q = X_i$. This product has $O(m^{ph})$ terms, rather than $O(m^q)$ giving the improved time bound.

On the other hand, we show that the problem becomes NP-complete, when either the characteristic p or the extension degree h (more precisely the product ph) becomes large. A consequence of this is that exponential dependence on p and h in our algorithms is unavoidable, since the corresponding counting problems are also hard. To precisely quantify how large the field size needs to be, we parameterize an instance by the number of variables n.

Theorem 5 *The problem of deciding whether a polynomial $P(\mathbf{X})$ over $\mathbb{Z}_p$ has a root is* NP*-complete for $p \geq 2n$.*

PROOF: By reduction from 3SAT. Consider an instance $\varphi(\mathbf{X})$ of 3SAT with t clauses $(C_1, \cdots, C_t)$. Let $p > t$. Arithmetize each clause over $\mathbb{Z}_p$ such that for $X_i \in \{0,1\}$, $C_i(\mathbf{X}) = 0$ if clause C_i is satisfied and 1 otherwise. This can be done with a multilinear polynomial of sparsity at most 8. Replace each X_i with X_i^{p-1} and output the polynomial

$$P(\mathbf{X}) = \sum_i C_i(X_1^{p-1}, \cdots, X_n^{p-1})$$

The substitution $X_i \to X_i^{p-1}$ maps $\mathbb{Z}_p \to \{0,1\}$. Since we have chosen $p > t$, the polynomial $P(\mathbf{X})$ counts the number of unsatisfied clauses. Hence $P(\mathbf{X})$ has a root over $\mathbb{Z}_p$ iff $\varphi(\mathbf{X})$ is satisfiable. If we begin with a 3SAT instance where every variable occurs in at most 5 clauses, the above reduction shows that deciding if $N(P) > 0$ when p and m are both linear in n is hard. □

Corollary 6 *The problem of computing $N_p(P)$ given $P(\mathbf{X})$ over $\mathbb{Z}_p$ is* NP*-hard under randomized reductions for $p \geq 2n$.*

PROOF: We repeat the above reduction starting with an instance of U3SAT. In the Yes case, assume that there is a solution $\mathbf{x} \in \{0,1\}^n$ to the 3SAT instance with k ones. Then any vector $\mathbf{y} \in \mathbb{Z}_p^n$ where $y_i^{p-1} = x_i$ is a solution to P. Hence $N(P) = (p-1)^k \not\equiv 0 \bmod p$. On the other hand, for a No instance of 3SAT, $N(P) = 0$. In particular, deciding if the number of roots is $0 \bmod p$ is hard. □

Theorem 6 *The problem of deciding whether a polynomial $P(\mathbf{X})$ over $\mathbb{F}_{2^t}$ has a root is* NP*-complete for $t \geq 2n$.*

PROOF: By reduction from 3SAT. Given an instance $\varphi(\mathbf{X})$ with clauses $C_1, \cdots, C_t$, we arithmetize each clause such that for $X_i \in \{0,1\}$, $C_i(X_1, \cdots, X_n) = 0$ if clause C_i is satisfied and 1 otherwise. Let $q = 2^t$. Take α to be an irreducible element of degree t. Output the polynomial

$$P(\mathbf{X}) = \sum_i C_i(X_1^{q-1}, \cdots, X_n^{q-1})\alpha^{i-1}$$

Note that for every clause C_i and every $(x_1, \cdots, x_n) \in \mathbb{F}_q^n$, $C_i(x_1^{q-1}, \cdots, x_n^{q-1}) \in \mathbb{F}_2$. Since the powers of α are linearly independent over $\mathbb{F}_2$, any solution to P must satisfy $C_i(x_1^{q-1}, \cdots, x_n^{q-1}) = 0$. Hence, given a root $(x_1, \cdots, x_n) \in \mathbb{F}_q^n$ of the above equation, $(x_1^{q-1}, \cdots, x_n^{q-1})$ gives a solution to $\varphi(\mathbf{X})$. □

Repeating this reduction starting with U3SAT gives the following corollary.

Corollary 7 *The problem of computing $N_2(P)$ given $P(\mathbf{X})$ over $\mathbb{F}_{2^t}$ is* NP*-hard under randomized reductions for $t \geq 2n$.*

One can modify the above reduction to prove that the problem is complete for ⊕P. One can also combine the reductions in Theorems 5 and 6 to show that the feasibility problem is NP-complete for $ph > 2n$.

5 Maximum-Likelihood Reed-Solomon decoding

Let $S_k(\mathbf{X})$ denote the k^{th} elementary symmetric polynomial in $X_1, \cdots, X_n$. The polynomials $S_k(\mathbf{X})$ for $1 \leq k \leq n$ generate all symmetric polynomials [2]. If a symmetric polynomial is written as a sum of monomials in this basis, we say that it is sparsely represented. A natural question is what is the complexity of the feasibility problem for symmetric polynomials in the sparse representation. We show that maximum-likelihood decoding of Reed-Solomon codes is related to a variant of this problem.

An $[n,k]_q$ Reed Solomon codes consists of all univariate polynomials of degree at most k over $\mathbb{F}_q$ evaluated at a set of points $D = \{x_1, \cdots, x_n\} \subseteq \mathbb{F}_q$. The maximum likelihood decoding problem MLD-RS asks for the closest codeword to a vector $\mathbf{r} \in \mathbb{F}_q^n$. We will work with a different formulation of MLD-RS due to Guruswami and Vardy [6]. Given $D = \{x_1, \cdots, x_n\}$, define the matrix

$$H = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ x_1^w & x_2^w & \cdots & x_n^w \end{pmatrix}$$

We define the code $\mathbb{C} = \{\mathbf{z} \in \mathbb{F}_q^n \mid H\mathbf{z}^t = 0\}$, which is in fact a generalized Reed Solomon code. The maximum-likelihood decoding problem MLD-RS is: *Given H and a syndrome $\mathbf{e} = (e_0, \cdots, e_w) \in \mathbb{F}_q^{w+1}$, is there a vector $\mathbf{z} \in \mathbb{F}_q^n$ with $wt(\mathbf{z}) \leq w$ satisfying $H\mathbf{z} = \mathbf{e}$?*

Note that any $w+1$ columns of H are linearly independent, so we can always find a vector $\mathbf{z}$ of weight $w+1$ so that $H\mathbf{z} = \mathbf{e}$.

Theorem 7 *There exists a vector $\mathbf{z} \in \mathbb{F}_q^n$ with $wt(\mathbf{z}) \leq w$ so that $H\mathbf{z} = \mathbf{e}$ iff*

$$P(X_1, \cdots, X_w) = \sum_{i \leq w} (-1)^i e_{w-i} S_i(X_1, \cdots, X_w) = 0$$

has a root in D^w where $x_i \neq x_j$ for $i \neq j$.

PROOF: We first prove the following identity:

$$\begin{vmatrix} 1 & \cdots & 1 & e_0 \\ x_1 & \cdots & x_w & e_1 \\ x_1^2 & \cdots & x_w^2 & e_2 \\ \cdots & \cdots & \cdots & \cdots \\ x_1^w & \cdots & x_w^w & e_w \end{vmatrix} = \prod_{i \neq j} (x_i - x_j) \sum_{i=0}^{w} (-1)^i e_{w-i} S_i(x_1, \cdots, x_w) \tag{6}$$

We evaluate the LHS by comparing it to the Vandermonde determinant. Let E denote a formal variable. Then

$$\begin{vmatrix} 1 & \cdots & 1 & E^0 \\ x_1 & \cdots & x_w & E^1 \\ x_1^2 & \cdots & x_w^2 & E^2 \\ \cdots & \cdots & \cdots & \cdots \\ x_1^w & \cdots & x_w^w & E^w \end{vmatrix} = \prod_{i \neq j} (x_i - x_j) \prod_{i \leq w} (E - x_i)$$

$$= \prod_{i \neq j} (x_i - x_j) \sum_{i \leq w} (-1)^i E^{w-i} S_i(x_1, \cdots, x_w)$$

Note that by expanding the determinant along the last column, we could derive the same formula without using fact that the various column entries are powers of E. They can be treated as formal symbols. Hence we deduce Equation (6).

Suppose that there exists $\mathbf{z}$ of weight w so that $H\mathbf{z}^t = \mathbf{e}$. Assume wlog that the first w co-ordinates of $\mathbf{z}$ are non-zero. Then $\mathbf{e}$ lies in the span of the first w columns of H, hence the LHS of Equation 6 vanishes. Since $x_i \neq x_j$, this implies that $P(x_1, \cdots, x_w) = 0$.

Conversely, given $(x_1, \cdots, x_w) \in D^w$ where $x_i \neq x_j$ but $P(x_1, \cdots, x_w) = 0$, the determinant on the LHS of Equation (6) vanishes. Hence a non-trivial linear combination of its columns is 0. Since $x_i \neq x_j$, the columns corresponding to various x_is are linearly independent, so the column corresponding to $\mathbf{e}$ occurs in this combination with a non-zero multiplier. Hence we can write $\mathbf{e}$ as a linear combination of the other columns, which gives a solution to $H\mathbf{z}^t = \mathbf{e}$ of weight at most w. □

If we set $e_w = \gamma$, $e_{w-1} = 1$ and $e_i = 0$ for $i \leq w - 2$, the problem reduces to finding $(x_1, \cdots, x_w) \in D^w$ so that $\sum x_i = \gamma$. Guruswami and Vardy show this is NP-complete when the field size is exponential in n, which implies NP-completeness of MLD-RS over large fields [6]. However it is possible that the above feasibility problem and hence MLD-RS are intractable over $\mathbb{F}_q$ when q is polynomial in n, and when $D = \mathbb{F}_q$.

6 Discussion and Open Problems

The main open problem left open by this work was: *Is there algorithm to compute $N_{p^k}(P)$ over $\mathbb{F}_q$ with $q = p^h$ that is singly exponential in p and h?*

This question was settled affirmatively by Daqing Wan [25], who gives an algorithm for this problem with running time $O(nm^{(h+k)p})$. The techniques used differ significantly from this paper: he first uses a formula for $N(P)$ in terms of Gauss sums, and then applies the Gross-Koblitz formula relating Gauss sums to the p-adic Γ-function.

We conclude with some open problems raised by our work:

- Is the feasibility problem NP-complete for polynomials of low degree?
- Is it possible to construct a family of modulus amplifying polynomials for a specific modulus p that have degree less than $2k-1$?
- Is the feasibility problem hard for sparse symmetric polynomials when q is polynomial in n?

Acknowledgments:

The first author would like to thank Matt Baker, Saugata Basu and Henry Cohn for useful discussions on this subject. We thank an anonymous LATIN referee and Igor Shparlinksi for several useful references. We thank Daqing Wan for telling us about his results, and for pointing out the improved running time for the feasibility algorithm [24].

References

[1] L. ADLEMAN AND M.-D. HUANG, *Counting rational points on curves and Abelian varieties over finite fields*, in Proceedings of the 1996 Algorithmic Number Theory Symposium, Springer-Verlag LNCS 1122, 1996, pp. 1–16.

[2] M. ARTIN, *Algebra*, Prentice-Hall, 1991.

[3] R. BEIGEL AND J. TARUI, *On ACC*, Computational Complexity, 4 (1994), pp. 350–366.

[4] A. EHRENFEUCHT AND M. KARPINSKI, *The computational complexity of (xor, and)-counting problems*, Tech. Rep. 8543-CS, ICSI, Berkeley, 1990.

[5] D. GRIGORIEV AND M. KARPINSKI, *An approximation algorithm for the number of zeroes of arbitrary polynomials over GF[q]*, in IEEE Symposium on Foundations of Computer Science, (FOCS'91), 1991, pp. 662–669.

[6] V. GURUSWAMI AND A. VARDY, *Maximum-likelihood decoding of Reed-Solomon codes is NP-hard*, in Proceedings of the ACM-SIAM symposium on Discrete Algorithms (SODA'05), 2005, pp. 470–478.

[7] M.-D. HUANG AND D. IERARDI, *Counting rational points on curves over finite fields*, in Proceedings of the 34^{th} IEEE Symposium on Foundations of Computer Science (FOCS'93), 1993, pp. 616–625.

[8] M.-D. HUANG AND Y. WONG, *Solvability of systems of polynomial congruences modulo a large prime*, Journal of Computational Complexity, 8 (1999), pp. 227–257.

[9] M. KARPINSKI AND M. LUBY, *Approximating the number of zeroes of a GF[2] polynomial*, Journal of Algorithms, 14 (1993), pp. 280–287.

[10] K. KEDLAYA, *Computing zeta functions via p-adic cohomology*.

[11] A. LAUDER AND D. WAN, *Counting points on varieties over finite fields of small characteristic*, to appear in Algorithmic Number Theory, J.B. Buhler and P. Stevehagen (eds), Cambridge University Press.

[12] ——, *Computing Zeta functions of Artin-Schreier curves over finite fields ii*, J. Complexity, 20(2-3): 331-349 (2004).

[13] R. LIDL AND H. NIEDERREITER, *Finite Fields, Encylopedia of Mathematics and Its Applications*, Cambridge University Press, 1997.

[14] M. LUBY, B. VELICKOVIĆ, AND A. WIGDERSON, *Deterministic approximate counting of depth-2 circuits*, in Israel Symposium on Theory of Computing Systems, 1993, pp. 18–24.

[15] R. C. MASON, *Diophantine Equations Over Function Fields*, Cambridge University Press, 1984.

[16] O. MORENO AND C. J. MORENO, *Improvements of the Chevalley-Warning and the Ax-Katz theorems*, American Jounral of Mathematics, 117(1) (1995), pp. 241–244.

[17] C. PAPADIMITRIOU, *Computational Complexity*, Addison-Wesley, 1994.

[18] J. PILA, *Frobenius maps of Abelian varieties and finding roots of unity in finite fields*, Mathematics of Computation, 55 (1990), pp. 745–763.

[19] R. SCHOOF, *Counting points on elliptic curves over finite fields*, J. Th'eor. Nombres Bordeaux, 7 (1995), pp. 219–254.

[20] S. TODA, *PP is as hard as the polynomial-time hierarchy*, SIAM Journal on Computing, 20(5) (1991), pp. 865–877.

[21] J. VON ZUR GATHEN, M. KARPINSKI, AND I. SHPARLINSKI, *Counting curves and their projections*, Computational Complexity, 6 (1996), pp. 64–99.

[22] D. WAN, *A Chevalley-Warning approach to p-adic estimate of character sums*, Proceedings of the American Mathematical Society, 123 (1995), pp. 45–54.

[23] D. WAN, *Computing Zeta functions over finite fields*, Contemporary Mathematics, 225 (1999), pp. 135–141.

[24] D. WAN, *Personal communication*, April 2006.

[25] ——, *Modular counting of rational points on sparse equations over finite fields*, Manuscript, April 2006.

[26] A. C. YAO, *On ACC and threshold circuits*, in 31^{st} IEEE Symposium on Foundations of Computer Science (FOCS'90), 1990, pp. 619–627.