An (ϵ,δ)-Approximation Algorithm of the Number of Zeros of a Multilinear Polynomial over GF[q]

Marek Karpinski*
Dept. of Computer Science
University of Bonn
5300 Bonn 1
and
International Computer Science Institute
Berkeley, California

Barbara Lhotzky
Dept. of Computer Science
University of Bonn
5300 Bonn 1

Abstract

We construct a polynomial time (ϵ, δ)-approximation algorithm for estimating the number of zeros of an arbitrary multilinear polynomial $f(x_1, \ldots, x_n)$ over GF[q]. This extends the recent result of Karpinski/Luby ([**?**]) on approximating the number of zeros of polynomials over the field GF[2].

*Supported in part by the Leibniz Center for Research in Computer Science, by the DFG Grant KA 673/4-1 and by the SERC Grant GR-E 68297

1 Introduction

There are only few cases of algebraic counting problems known having polynomial time solutions (cf. [?, ?, ?, ?]) despite their paramount importance in algebra and various applications in coding theory and in the design of algebraic circuits. The general problems of counting exactly the number of points on algebraic varieties over arbitrary (and fixed) finite fields GF[q], even if restricted to cubic multilinear polynomials, was proven recently to be #P-complete ([?]). The first polynomial (ϵ, δ)-approximation algorithm for the number of zeros of a polynomial $f(x_1, \ldots, x_n)$ over the field GF[2] was designed recently by Karpinski and Luby ([?]).

In this paper we construct the first approximation algorithm for estimating the number of zeros of arbitrary multilinear polynomials over GF[q], extending the results of [?]. The algorithm takes as input a multilinear polynomial $f(x_1, \ldots, x_n)$, a constant c and two parameters $\epsilon > 0$ and $\delta > 0$. It computes an estimate Y which is between $(1 - \epsilon)$ and $(1 + \epsilon)$ times the number of solutions of the equation $f(x_1, \ldots, x_n) = c$ with probability at least $1 - \delta$. Let m denote the number of terms of f and $Q(q) = \log q \log\log q \log\log\log q$ the multiplication time of two elements in GF[q]. Then the running time of the approximation algorithms is $O(nm^3 q \log q Q(q) \ln(1/\delta)/\epsilon^2)$.

Besides the direct algebraic and algebraic geometric interest, our algorithm provides also the first known estimation method for the number of assignmnents of $(+, *)$-arithmetic circuits of depth 2 over arbitrary finite fields GF[q]. For the constant free circuits we can approximate the number of 0-assignments directly from the black box even without interpolating explicitely a polynomial from the black box (by extending the ground field GF[q], cf. [?]).

2 Notations and Definitions

Let GF[q] be an arbitrary ground field of order q.

$f(x_1, \ldots, x_n) \in \mathrm{GF}[q][x_1, \ldots, x_n]$ denotes a polynomial in n variables over the field GF[q]. In the following we assume that f is represented by the sum of its nonzero terms t_i:

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{m} t_i(x_1, \ldots, x_n) = \sum_{i=1}^{m} c_i \prod_{j=1}^{n} x_j^{e_{i,j}}$$

with $c_i \in \mathrm{GF}[q] \setminus \{0\}$, and all monomials $\prod_{j=1}^{n} x_j^{e_{i,j}}$ being different from each other.

We further assume f to be multilinear meaning that all exponents $e_{i,j}$ are either 0 or 1.

Let n denote the number of variables, m the number of terms and $Var(f)$ the set of variables of f.

In the following we consider the equation $f(x_1, \ldots, x_n) = c$ with a constant $c \in \mathrm{GF}[q]$. We are interested in the number of solutions of this equation, i.e. the number of assignments $s \in \mathrm{GF}[q]^n$ to the variables of f such that the equation $f(x_1, \ldots, x_n) = c$ is satisfied. Therefore we can assume w.l.o.g. that f has no constant term.

Let $\mathcal{S}_c(f) = \{s \in \mathrm{GF}[q]^n | f(s) = c\}$ denote the set of solutions of the equation $f(x_1, \ldots, x_n) = c$ and $\#_c f = |\mathcal{S}_c(f)|$ the number of solutions.

An important role for our algorithm plays the set $D(f) = \{s \in \mathrm{GF}[q]^n | \exists t_i \quad t_i(s) \neq 0\}$ of all those assignments to the variables such that at least one term of f evaluates to nonzero.

Let $\bigcup M_i$ denote the union and $\sum M_i$ denote the disjoint union of the sets M_i.

An (ϵ, δ)-approximation algorithm (cf. e.g. [**?**]) is an algorithm which takes as input a problem instance (in our case f, c) and two additional parameters $\epsilon > 0$, the accuracy requirement, and $\delta > 0$, the confidence level. It computes an estimate Y for the desired value $\#_c f$ which satisfies the condition

$$\Pr\{(1-\epsilon)\#_c f \leq Y \leq (1+\epsilon)\#_c f\} \geq 1 - \delta.$$

The running time has to be polynomial in the length of the problem instance, $1/\epsilon$, $1/\delta$ and q.

3 Main Theorem

The core of our approximation algorithm is a polynomial bound on the fraction of the size of the set of assignments that evaluate at least one term of f to nonzero to the size of the whole solution set. This can be viewed as a generalization of a method applied in the Karpinski/Luby paper ([**?**]) for the field GF[2]. We will prove the existence of such a bound in our Main Theorem.

Let us start with a Lemma, bounding the minimum number of solutions of a multilinear equation over GF[q].

Lemma 3.1 *Let $g \in GF[q][x_1, \ldots, x_n]$ be a multilinear polynomial (with a possible constant term), $g \not\equiv const.$ Then*

$$\forall c \in GF[q]: \qquad \#_c g \geq (q-1)^{n-1}.$$

Proof: By induction on n, the number of variables.

Induction Basis ($n = 1$):
$g(x) = ax + b$ with $a \neq 0$. Since every $a \neq 0$ is a generator of the additive group $(\mathrm{GF}[q], +)$, there exists for all c a unique solution $s \in \mathrm{GF}[q]$ with $g(s) = c$. Therefore

$$\#_c g = 1 = (q-1)^0.$$

Induction Assumption:

$$\forall g \in \mathrm{GF}[q][x_1, \ldots, x_n],\ g \not\equiv \mathrm{const},\ c \in \mathrm{GF}[q]: \qquad \#_c g \geq (q-1)^{n-1}.$$

Induction Step:

Let const $\not\equiv g \in \mathrm{GF}[q][x_1, \ldots, x_{n+1}]$ and $c \in \mathrm{GF}[q]$ be fixed. Decompose g according to the variable x_{n+1}:

$$g(x_1, \ldots, x_{n+1}) = x_{n+1} \cdot h(x_1, \ldots, x_n) + k(x_1, \ldots, x_n).$$

1st case: $h(x_1, \ldots, x_n) \equiv 0$.
Then

$$\#_c g = q \cdot \#_c k \geq q \cdot (q-1)^{n-1} > (q-1)^n.$$

2nd case: $h(x_1, \ldots, x_n) \equiv d \neq 0$ for a constant $d \in GF[q]$.
Then for all possible values a of $k(x_1, \ldots, x_n)$ there is a unique solution of $dx_{n+1} + a = c$. Therefore

$$\#_c g = 1 \cdot q^n \geq (q-1)^n.$$

3rd case: $h(x_1, \ldots, x_n) \not\equiv d$ for any constant $d \in \mathrm{GF}[q]$.
According to the induction hypothesis, there are at least $(q-1)^{n-1}$ solutions $s \in \mathrm{GF}[q]^n$ to the equation $h(x_1, \ldots, x_n) = d$ for a constant $0 \neq d \in \mathrm{GF}[q]$. For all these solutions s there exists a unique value for x_{n+1} whatever the value of $k(s)$ is such that $g(x) = c$ is satisfied. Since there are $(q-1)$ possible values of $d \neq 0$, we get

$$\#_c g = \sum_{0 \neq d \in \mathrm{GF}[q]} \#_d h \geq (q-1) \cdot (q-1)^{n-1} = (q-1)^n.$$

□

We shall use now Lemma 3.1 to prove our Main Theorem.

Theorem 3.1 (Main Theorem) *Let $f \in GF[q][x_1, \ldots, x_n]$ be a multilinear polynomial and $c \in GF[q]$. Let m be the number of terms of f. $D(f) = \{s \in GF[q]^n | \exists t_i \quad t_i(s) \neq 0\}$ and let $\mathcal{S}_c(f) = \{s \in GF[q]^n | f(s) = c\}$. If f is constant free or the constant term of f is c, then*

$$\frac{|D(f)|}{|\mathcal{S}_c(f)|} \leq (q-1) \cdot m.$$

Proof:

The idea of the proof is to define a partition of $D(f)$ into sets $D_{i,j}(f)$ and to define an equal number of sets $R_{i,j}(f)$ which cover the set $\mathcal{S}_c(f)$ in such a way that the ratio between the sizes of two corresponding sets $D_{i,j}$ and $R_{i,j}$ is bounded by $(q-1)$. Therefore the Theorem follows.

At first, we divide the set $D(f)$ into sets $D_i(f)$ of roughly those assignments by which the i-th term t_i evaluates to nonzero:

$$D_i(f) = \{s \in \mathrm{GF}[q]^n | t_i(s) \neq 0 \quad \text{and} \quad \forall j \neq i,\ t_j(s) \neq 0 : \quad Var(t_i) \not\subset Var(t_j)\}$$

for $i = 1, \ldots, m$.
(The sets $D_i(f)$ are not necessarily disjoint.)

So $D_i(f)$ is the set of the assignments for which the term t_i is variable maximal out of all those which evaluate to nonzero by this assignment. We will need this technical detail in the following.

Now define a partition of the sets $D_i(f)$ into $q^{n-\deg t_i}$ many disjoint sets $D_{i,j}(f)$ of those assignments in $D_i(f)$ that are identical on the variables not in $Var(t_i)$. Obviously

$$D(f) = \bigcup_{i=1}^{m} D_i(f) = \bigcup_{i=1}^{m} \sum_{j=1}^{q^{n-\deg t_i}} D_{i,j}(f).$$

The size of each set $D_{i,j}(f)$ is bounded by the number of nonzero assignments to the variables of the term t_i since the values of all other variables are fixed:

$$|D_{i,j}(f)| \leq (q-1)^{\deg t_i}.$$

In order to get the sets $R_{i,j}(f)$ for $i = 1, \ldots, m$ and $j = 1, \ldots, q^{n-\deg t_i}$ covering $\mathcal{S}_c(f)$, we consider the partial assignment of the fixed values of $D_{i,j}(f)$ to all variables not in $Var(t_i)$. Then define $R_{i,j}(f)$ to be the set of those assignments consisting of the fixed values for the variables not in $Var(t_i)$ and those values for the variables of t_i such that the equation $f(x_1, \ldots, x_n) = c$ is satisfied.

Since the second condition in the definition of the sets $D_i(f)$ makes sure that f does not become constant under the partial assignment to all variables except to those of one variable maximal term t_i, we get a lower bound on the sizes of the sets $R_{i,j}(f)$ by Lemma 3.1:

$$|R_{i,j}(f)| \geq (q-1)^{\deg t_i - 1} \qquad \text{for all } i,j.$$

The sets $R_{i,j}(f)$ are not necessarily disjoint, but they cover $\mathcal{S}_c(f)$:

$$\bigcup_{i,j} R_{i,j}(f) = \mathcal{S}_c(f).$$

Consequently we have

$$|\mathcal{S}_c(f)| \leq \sum_{i,j} |R_{i,j}(f)|.$$

Since the elements in $R_{i,j}(f)$ and $R_{i,k}(f)$ for $j \neq k$ define different assignments to the variables not in t_i, they are disjoint. So every element $s \in \mathcal{S}_c(f)$ may appear in at most m different sets $R_{i,j}(f)$.

Therefore

$$m \cdot |\mathcal{S}_c(f)| \geq \sum_{i,j} |R_{i,j}(f)|.$$

Combining the intermediate results from above, we get

$$\frac{|D(f)|}{|\mathcal{S}_c(f)|} \leq \frac{\sum_{i,j} |D_{i,j}(f)|}{1/m \cdot \sum_{i,j} |R_{i,j}(f)|} \leq \frac{\sum_{i,j} (q-1)|R_{i,j}(f)|}{1/m \cdot \sum_{i,j} |R_{i,j}(f)|} = m \cdot (q-1).$$

□

The bound given in Theorem 3.1 is sharp. Consider for instance the polynomial

$$f(x_1, \ldots, x_n) = \prod_{i=1}^{n} x_i.$$

There are $(q-1)^n$ many assignments evaluating the single term of f to nonzero and $(q-1)^{n-1}$ many assignments evaluating f to 1. Therefore

$$\frac{|D(f)|}{|\mathcal{S}_1(f)|} = \frac{(q-1)^n}{(q-1)^{n-1}} = (q-1) \cdot m.$$

We derive a new bound for the number of zeros for the special case of constant free polynomials.

Corollary 3.1 *Let $f \in GF[q][x_1, \ldots, x_n]$ be a multilinear polynomial with $f(0, \ldots, 0) = 0$ (without a constant term). Let m be the number of terms of f. Then*

$$\frac{q^n}{\#_0(f)} \leq (q-1) \cdot m + 1.$$

Proof:

Consider the function $\tilde{f}(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) + c$ for an arbitrary $c \neq 0$ out of GF[q] and define the sets $D_{i,j}(f)$, $R_{i,j}(f)$ and $D_{i,j}(\tilde{f})$, $R_{i,j}(\tilde{f})$ as shown in the proof of Theorem 3.1.

We have $D_{i,j}(\tilde{f}) = D_{i,j}(f)$ and $R_{i,j}(\tilde{f}) = R_{i,j}(f)$ for $i = 1, \ldots, m$. There are the additional sets $D_{m+1,j}(\tilde{f})$ partitioning the set $D_{m+1}(\tilde{f}) = \{s \in \mathrm{GF}[q]^n | \forall i < m : \quad t_i(s) = 0\}$ of all those assignments to the variables that evaluate all nonconstant terms of $\tilde{f}$ to zero. These assignments are obviously solutions to the equation $\tilde{f}(x_1, \ldots, x_n) = c$. Therefore $R_{m+1,j}(\tilde{f}) = D_{m+1,j}(\tilde{f})$.

Since $D(\tilde{f}) = \mathrm{GF}[q]^n$ and $\mathcal{S}_0(f) = \mathcal{S}_c(\tilde{f})$, we get the inequality

$$\begin{aligned}
\frac{q^n}{\#_0(f)} &= \frac{|D(\tilde{f})|}{|\mathcal{S}_c(\tilde{f})|} \\
&\leq \frac{\sum_{i \leq m,j} |D_{i,j}(\tilde{f})| + \sum_j |D_{m+1,j}(\tilde{f})|}{1/m \sum_{i \leq m,j} |R_{i,j}(\tilde{f})| + \sum_j |R_{m+1,j}(\tilde{f})|} \\
&\leq \frac{(q-1) \sum_{i \leq m,j} |R_{i,j}(\tilde{f})| + \sum_j |R_{m+1,j}(\tilde{f})|}{\sum_{i \leq m,j} |R_{i,j}(\tilde{f})| + \sum_j |R_{m+1,j}(\tilde{f})|} \\
&\leq (q-1) \cdot m + 1
\end{aligned}$$

□

4 The Algorithm

We are ready to formulate our approximation algorithm. The specific construction of the universe set U and the indicator function $\varphi(u)$ will be given later.

Input: $f \in \mathsf{GF}[q][x_1, \ldots, x_n],\ c \in \mathsf{GF}[q],\ \epsilon > 0,\ \delta > 0;$
Output: Y with $\Pr\{(1-\epsilon) \cdot \#f \leq Y \leq (1+\epsilon) \cdot \#f\} \geq 1 - \delta;$

1. fix a universe U;
2. $N = b \cdot 4/\epsilon^2 \cdot \ln 2/\delta$ with $b \geq |U|/\#_c f$;
3. choose independently N elements u of U according to a uniform distribution;
4. $Y = |U| \cdot \sum_u \varphi(u)/N$.

Karp/Luby/Madras ([?]) derived the bound $N = |U|/\#_c f \cdot 4/\epsilon^2 \cdot \ln 2/\delta$ (Zero-One Estimator Theorem) for the number of trials necessary to obtain an estimate of the required precision using the Bernstein inequality ([?]) in the general settings of universe sets U and indicator functions φ. The following conditions have to be satisfied:

- The size $|U|$ is efficiently computable.
- Elements of U can be chosen efficiently according to a uniform distribution.
- The ratio $|U|/\#_c f$ is polynomially bounded.
- The indicator function $\varphi : U \longrightarrow I\!N$ is efficiently computable.
- The mean value $E[\varphi]$ is equal to $\#_c f/|U|$.

Now we shall distinguish between two cases.

The first case is for $c = 0$ and for f without a constant term.

Here the Corollary 3.1 is applicable. So we choose

$$U = \mathrm{GF}[q]^n$$

and

$$\varphi(s) = \begin{cases} 1 & \text{if } f(s) = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Since we need $O(n \cdot \log q)$ many random bits to write an arbitrary element of $\mathrm{GF}[q]^n$ and the evaluation of $f(s)$ takes at most $O(mnQ(q))$ time (recall that $Q(q) = \log q \log\log q \log\log\log q$ denotes the time necessary for the multiplication of two elements in $\mathrm{GF}[q]$), the approximation algorithm needs

$$((m+1)(q-1)+1)4\ln(2/\delta)/\epsilon^2$$

many trials and every trial needs $O(nmQ(q))$ bit operations.

Therefore in this case the algorithm takes

$$O(nm^2qQ(q)\ln(1/\delta)/\epsilon^2) \text{ time.}$$

In the second case $f(x_1, \ldots, x_n) = c$ with $c \neq 0$ and f has no constant term. The choice of universe $U = \mathrm{GF}[q]^n$ is not good now because there are easy examples for equations with the ratio $\mathrm{GF}[q]^n/\#_c f$ growing exponentially (for instance $\prod_{i=1}^n x_i = 1$). choose elements from $D(f)$ according to a uniform distribution. But we shall see that the following choice of U and φ satisfies the conditions:

$$U = \sum_{i=1}^{m} U_i \qquad \text{with } U_i = \{(s,i) | s \in \mathrm{GF}[q]^n \text{ and } t_i(s) \neq 0\}$$

(note that $U_i \neq D_i(f)$) and

$$\varphi(s,i) = \begin{cases} 1 & \text{if } f(s) = c \text{ and } i = \min\{j | (s,j) \in U\} \\ 0 & \text{otherwise.} \end{cases}$$

Now we show that the conditions formulated above are satisfied.

The size $|U| = \sum_{i=1}^{m} |U_i| = \sum_{i=1}^{m} q^{n-\deg t_i} \cdot (q-1)^{\deg t_i}$ is computable in $O(mn \log nQ(q))$ time. This precomputation has to be done only once.

A random element $(s,i) \in U$ can be chosen uniformly by the following two step process:

1. Randomly choose $i \in \{1,\ldots,m\}$ with probability $|U_i|/|U|$.
2. Randomly choose $(s,i) \in U_i$ such that (s,i) is chosen with probability $1/|U_i|$.

The first step can be implemented by choosing a random value r in the interval $[1,\ldots,|U|]$ and selecting that i which satisfies $\sum_{j=1}^{i-1} |U_j| < r \leq \sum_{j=1}^{i} |U_j|$ using binary search ([**?**]). For the second step, we need $O(n \log q)$ random bits. Therefore the choice of a random element from U takes at most $O(\log(mq^n) \log m + n \log q) = O(\log^2 m + n \log m \log q)$ time.

The ratio $|U|/\#_c f$ is bounded by $m^2(q-1)$:

$$\frac{|U|}{\#_c f} = \frac{|U|}{|D(f)|} \cdot \frac{|D(f)|}{\#_c f} \leq \frac{|U|}{|D(f)|} \cdot m(q-1) \leq m^2 \cdot (q-1).$$

The first inequality holds because of the Main Theorem and the second since every element of $D(f)$ evaluates at most m terms to nonzero.

The cost of the computation of $\varphi(s,i)$ is dominated by the evaluation of $f(s)$. This takes $O(mnQ(q))$ many operations.

So the complete approximation algorithm for estimating the number of solutions of $f(x_1,\ldots,x_n) = c \neq 0$ demands

$$m^2(q-1)4\ln(2/\delta)/\epsilon^2$$

many trials, where each trial costs $O(nmQ(q))$.

Consequently the time complexity of the algorithm is

$$O(nm^3 q \log qQ(q) \ln(1/\delta)/\epsilon^2).$$

We summarize now our main results of this section.

Theorem 4.1 *There exists an (ϵ,δ)-approximation algorithm for the number of zeros of an arbitrary multilinear polynomial over GF[q] with m terms working in time* $O(nm^3 q \log qQ(q) \ln(1/\delta)/\epsilon^2)$.

If additionally the polynomial does not contain constant terms, there exists an (ϵ,δ)-approximation algorithm for the number of zeros working in time $O(nm^2qQ(q)\ln(1/\delta)/\epsilon^2)$.

5 Parallel Implementation of the Algorithm

The parallel arithmetic in GF[q] can be done in boolean parallel time $O(\log q)$ with $O(Q(q))$ processors (cf. [**?**, **?**]) and the evaluation of a polynomial f over GF[q] in $O(\log(mn) + \log q)$ boolean parallel time with $O(nmQ(q))$ processors.

The Monte-Carlo part of the algorithm is parallisable in $O(\log(mq\ln(1/\delta)/\epsilon^2))$ depth. Therefore for a fixed field GF[q] and fixed ϵ,δ, we have:

Corollary 5.1 *There exists for a fixed field GF[q] and fixed numbers $\epsilon, \delta > 0$, a randomized parallel (ϵ,δ)-approximation algorithm (RNC^1) for approximating the number of zeros of an arbitrary multilinear polynomial $f \in GF[q][x_1, \ldots, x_n]$ with m terms. The algorithm works in $O(\log(nm))$ parallel boolean time with $O(nm^3)$ processors.*

6 Black Box Counting Interpolation

We apply now Corollary 3.1 for the black box (for the formal definition see [**?**]) counting problem of GF[q]. The polynomial $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ with m terms is given by a black box over GF[q]; the counting problem is the problem of estimating number of zeros of f over GF[q].

We have the following Corollary:

Corollary 6.1 *Given a black box for a multilinear polynomial $f \in GF[q][x_1, \ldots, x_n]$ with m terms and no constant terms. There exists an (ϵ, δ)-approximation algorithm for estimating the number of zeros of f over GF[q]. The algorithm works is* $O(nm^2qQ(q)\ln(1/\delta)/\epsilon^2)$ *time.*

Proof:

We construct a universe $U = \mathrm{GF}[q]^n$ and pick up elements x of U uniformly. To perform our approximation algorithm (first case) we need only evaluations of the black box at x's. □

The result above is interesting in view of the computational difficulty of exact identification of f from the black box without using proper field extension (cf. [?, ?]).

Intuitively our algorithm does not depend on the exact identification of the polynomial f given by the black box.

7 Conclusion and Open Problems

Our approximation method is based on the special property of multilinear polynomials. The bounds stated in Theorem 3.1 are not valid in general. Consider for instance the following function:

$$f(x_1, \ldots, x_n) = \prod_{i=1}^{n} ({x_i}^{q-1} + 1) - 1.$$

f has a unique zero $(0, \ldots, 0)$, but it has $m = 2^n - 1$ many terms. f does not have a constant term, and the ratio is

$$\frac{q^n}{\#_0 f} = q^n = 2^{(\log q)n} = m^{\log q}.$$

Counting the number of nonzeros is even worse, because there are polynomial equations without any solution, for instance

$$\prod_{i=1}^{n} {x_i}^{q-1} = 2.$$

An important open question remains whether there is an (ϵ, δ)-approximation algorithm for approximating number of zeros of arbitrary polynomials over arbitrary finite fields GF[q].

Another important question is whether it is possible to design a deterministic $(\epsilon, 0)$-approximation algorithm for the multilinear counting problem.

8 Acknowledgements

We are indebted to Hendrik Lenstra, Dick Karp, Mike Luby and Andrew Odlyzko for the number of interesting discussions.