Deterministic Identity Testing for Multivariate Polynomials

Richard J. Lipton [*] *Nisheeth K. Vishnoi* [†]

Abstract

In this paper we present a simple deterministic algorithm for testing whether a multivariate polynomial $f(x_1, \ldots, x_n)$ is identically zero, in time polynomial in $m, n, \log(d+1)$ and H. Here m is the number of monomials in f, d is the maximum degree of a variable in f and 2^H is the least upper bound on the magnitude of the largest coefficient in f. We assume that f has integer coefficients.

The main feature of our algorithm is its conceptual simplicity. The proof uses Linnik's Theorem which is a deep fact about distribution of primes in an arithmetic progression.

1 Introduction

The problem of testing whether a multivariate polynomial $f(x_1, \ldots, x_n)$ is identically zero has proven to be extremely useful in theoretical computer science. This has been used in designing fast randomized algorithms: most notably for program checking [5, 13] and perfect matching in graphs [14, 18, 6]. It is also heavily used in complexity theory in proving results like IP=PSPACE [21, 15, 4] and the optimal PCP characterization of NP [3]. For more details on the applications the reader is referred to the book of Motwani and Raghavan [17].

To make the problem more precise, one needs to fix the representation of the given polynomial. Clearly if the polynomial is given as a list of its coefficients, the problem is trivial. Often we are given some implicit representation of the polynomial such as the determinant of a matrix or an arithmetic circuit. It is easy to see that there *exists* a set S, with size polynomial in s and d, such that any nonzero multivariate polynomial of total degree d which can be described using s bits, evaluates to a nonzero value at at least one of the points of S. Finding such a set deterministically seems elusive and it seems likely that this will involve substantial algebraic and number theoretic ideas. This fact is illustrated in the papers of Chen and Kao [7] and Lewin and Vadhan [11].

2 Previous Work

The earliest work on zero testing can be traced to DeMillo and Lipton [8], Schwartz [20] and Zippel [22] back to late 1970's. The basic idea in their work was that a univariate polynomial of degree at most d can have at most d roots. So fixing a large enough set and evaluating the polynomial at randomly chosen points from this set gives an efficient randomized test with high probability of success.

Only recently have there been improvements in these algorithms. In a remarkable paper, Chen and Kao [7] show how to reduce randomness using irrational numbers. Contrary to the classic algorithms, the error probability is reduced not by increasing the number of random bits, but by letting the algorithm run for more time.

This technique was generalized and proved optimal under the black box model of computation by Lewin and Vadhan [11]. They give an algorithm which works over any field and uses $\sum_{i=1}^{n} \lceil \log(d_i + 1) \rceil$ random bits, where d_i is the degree of f in x_i.

Further improvements and generalizations were made in the work of [2] and notably in [10]. Klivans and Spielman [10] use ideas from error correcting codes and a variant of the Isolation Lemma [18] to give a randomized polynomial time algorithm which uses $O(\log mnd)$ random bits. Here m is the number of monomials, and d is the total degree of f.

3 Our Result

3.1 Notation Let k be a field and $f(x_1, \ldots, x_n) \in k[x_1, \ldots, x_n]$. The problem is to determine whether $f \equiv 0$.

The following notation is used throughout the paper

1. Represent the polynomial as
$$f = \sum_{\alpha} c_\alpha \mathbf{x}^\alpha.$$
We assume that all the coefficients c_α are integers. Moreover $\mathbf{x}^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$.

2. n denotes the number of variables in f.

3. m denotes the number of monomials in f.

[*] College of Computing, Georgia Institute of Technology, Atlanta GA 30332. Email: rjl@cc.gatech.edu

[†] College of Computing, Georgia Institute of Technology, Atlanta GA 30332. Email: nkv@cc.gatech.edu

4. d_i, for $1 \leq i \leq n$ be the degree of x_i in f. Let $d = \max_i d_i$. d is called the **maximum degree** of f.

5. L denotes Linnik's constant (explained later).

6. H denotes the **height** of f, that is if $f = \sum_\alpha c_\alpha \mathbf{x}_\alpha$, then H is the least number such that
$$\max_\alpha |c_\alpha| \leq 2^H.$$

7. All logs in this paper are base 2 unless stated otherwise.

8. We assume that the only operation we are allowed with polynomial f is to evaluate it on inputs. We refer to this as the **Black Box** model. We further assume that the implementation of f is over a field of characteristic zero.

3.2 Main Theorem and Discussion

Now we are ready to state the result of this paper.

THEOREM 3.1. *Given a multivariate polynomial $f(x_1, \ldots, x_n)$ with integer coefficients as a black box, there is a deterministic algorithm which decides whether f is identically zero in time polynomial in $m, n, \log(d+1)$ and H. Moreover the bit lengths of the queries to the black box are logarithmic in $m, n, \log(d+1)$ and H.*

The input bit lengths are important. This is illustrated in a later section on the number of roots of sparse polynomials.

Although we use some deep results from analytic number theory, the main idea in the paper is a clever use of the following mantra

> A positive integer s can have at most $\log s$ distinct prime factors.

We do not attempt to optimize the running time of the algorithm in this paper, instead just illustrate the basic ideas to get a deterministic test for zero testing with parameters mentioned above.

We remark here that via a different approach, Klivans and Spielman [10] obtain similar results. But their result implies a deterministic identity test which runs in time polynomial in m, n, d compared to ours, which runs in time polynomial in m, n, H and $\log(d+1)$. So for exponential degree sparse polynomials, Klivans-Spielman implies an exponential algorithm, while ours is still a polynomial time algorithm.

The main feature of our algorithm is that it is conceptually simpler than the one by Klivans and Spielman.

We also mention the basic lower bound argument for any deterministic test: Any oracle access to f gives a linear equation over the coefficients of the polynomial. So One has to make at least $\Omega(m)$ queries to decide whether f is identically zero or not. So apart from the restriction on the largest coefficient, our algorithm is tight up to a polynomial. But notice that the degree d does not play a role in the lower bound. That is where our is better than the previous algorithms.

3.3 Organization

First we give an interesting fact about roots of sparse polynomials. In the next section we introduce some number theoretic preliminaries we will be needing for the algorithm.

Then we give the algorithm and in the subsequent section its analysis.

4 Number of Positive Roots of Sparse Polynomials

In this section we address the following question

> How many real positive roots can a univariate polynomial (over the reals) with at most m monomials can have ?

The answer to this question follows from a result of Michell ([16]) and also from the well known Descartes' Rule of Signs: the number of such zeros has to be less than m. So for such a polynomial, evaluating them at $\{1, \ldots, m\}$ is good enough to decide whether it is zero or not. First notice that such a result is not possible for multivariate polynomials: consider $x - y$. Moreover the reduction of a multivariate polynomial to a univariate one may cause an exponential blow up in the degree. So the input bit lengths may become exponential in the number of variables, which is not desirable.

5 Number Theoretic Preliminaries

For integers a, d, (a, d) denotes the greatest common divisor of a and d. For $d > 0$, let $\phi(d)$ be the Euler's totient function of d, or the cardinality of the set
$$\{a | (a, d) = 1, 1 \leq a \leq d\}.$$
Notice that for a prime p, $\phi(p) = p - 1$.

LEMMA 5.1. *The number of distinct prime divisors of an integer $s \geq 1$ is at most $\log_2 s$.*

For a prime p let $GF(p)$ be the finite field on p elements.

THEOREM 5.1. *For a prime p, and integers s, t the polynomials x^s and x^t are the same over $GF(p)$ if and only if $p - 1$ divides $(s - t)$.*

5.1 Distribution of Primes in Arithmetic Progressions

The following Lemma follows easily from the Prime Number Theorem or even relaxed versions of it.

LEMMA 5.2. *[19] There is a constant $c > 0$ such that the n-th prime $p_n \leq cn \log n$.*

THEOREM 5.2. (DIRICHLET) *[19] For integers $d \geq 2$ and $a \geq 1$, with $(a, d) = 1$, there are infinitely many primes in the arithmetic progression $\{a + kd | k \geq 0\}$.*

For integers $d \geq 2$ and $a \geq 1$, with $(a, d) = 1$, let $p(d, a)$ be the smallest prime in the arithmetic progression $\{a + kd | k \geq 0\}$. Define

$$p(d) = \max_{1 \leq a < d, (a,d)=1} p(d, a).$$

Under the widely believed Generalized Riemann Hypothesis, Heath-Brown proved ([19]) that there is a constant c such that

$$p(d) \leq c(\phi(d))^2 (\log d)^2.$$

The following unconditional theorem which we will use is attributed to Linnik can be found in [12].

THEOREM 5.3. *There is a constant $L > 1$ (called Linnik's constant) such that for every sufficiently large $d \geq q_0$,*

$$p(d) < d^L.$$

The best known value for L is 5.5 [9], while Schinzel, Sierpinski, and Kanold ([19]) have conjectured the value value to be 2. We mention here that it follows from the Prime Number Theorem that for every $\epsilon > 0$ and large enough d :

$$p(d) > (1 - \epsilon)\phi(d) \log d.$$

For a detailed discussion on these facts the reader is referred to the book by Ribenboim [19].

6 The Algorithm

First we need the following simple but useful Lemma.

LEMMA 6.1. *Given a polynomial $f(x_1, \ldots, x_n)$ over a characteristic zero field with maximum degree no more than d, the substitution $x_i \rightarrow x^{(d+1)^{i-1}}$ has the property that f is identically zero if and only if the new univariate polynomial is identically zero. Denote this polynomial by $g_f(x)$. The degree of $g_f(x)$ is at most $n(d + 1)^n$.*

Note that the above transformation preserves sparsity.

Let t be a parameter to be fixed later. Denote by q_r the r-th prime bigger than q_0. Here q_0 is the constant above which one can use Linnik's Theorem. Further let p_i be the smallest prime in the arithmetic progression $\{jq_i + 1 | j \geq 1\}$. It follows from Linnik's Theorem that $p_i < q_i^L$. Now we are ready to describe the algorithm:

> For $r = 1, \ldots, t$, compute $g_f(x)$ at all points of the field $GF(p_r)$ and output $f \equiv 0$ if and only if the value at each evaluated point is 0.

Notice that it may be the case that $p_i = p_j$ for $i \neq j$, but we do not care as long as there are enough distinct primes we run the algorithm on. We consider this issue in the analysis below. First we prove that the algorithm is correct for a suitable choice of t.

We also remark that since we assume an oracle access to f, we can implement the function g as follows: Once a prime p is fixed, and a point $x \in GF(p)$, we compute the transformation mentioned in Lemma 6.1 modulo p. Though the degree is exponential, still exponentiation can be done fast modulo a prime. Once we have the transformation, we query f at that point. Then we take the output modulo p. Since the oracle of f is over a characteristic zero field, the modular arithmetic described above is justified.

6.1 Analysis

Let $g_f(x) = \sum_i c_i x^{k_i}$. So g has at most m nonzero terms. Call a prime p from our test set **bad** if at least one of the following occurs

1. If $p - 1 | (k_i - k_j)$ for some $1 \leq i \neq j \leq m$.
2. If p divides c_i for some $1 \leq i \leq m$.

FACT 6.1. *It follows from 5.1 that if $g_f(x)$ is not identically zero, then the algorithm works correctly as long as we make sure that there is at least one prime in our set $p_1, \ldots, p_t$ which is not bad.*

To estimate the number of bad primes for the first case notice that whenever $p - 1$ divides some $d_i - d_j$ we have a prime factor q of $d_i - d_j$. Since the number of such terms is $O(m^2)$, and each of them could be at most $(d + 1)^n$. Hence we have the following:

FACT 6.2. *By Lemma 5.1 The number of bad primes for the first case is at most $m^2 n \log(d + 1)$.*

FACT 6.3. *By Lemma 5.1 the number of bad primes for the second case is at most*

$$\sum_i \log |c_i| \leq H \cdot m.$$

For the second case we have to deal with the case when for $i \neq j$, $p_i = p_j$. Or q_i, q_j have the same smallest prime in their corresponding arithmetic progressions. We do so with the following Lemma.

LEMMA 6.2. *Let $q_0 < q_1 < q_2 < \cdots q_v$ be primes such that the smallest prime in each of the arithmetic progression $\{jq_i + 1 | j \geq 1\}$ is p, and q_0 is the constant above which Linnik's Theorem applies, then*

$$v < L.$$

Proof. By hypothesis, there is an integer $k' > 1$ such that

$$p - 1 = k' q_1 q_2 = \cdots q_v.$$

But $q_1^v < p < q_1^L$. Here the last inequality follows from Linnik's Theorem. This implies the Lemma.

REMARK 6.1. *In fact one can tighten the analysis by the observation that all we need to consider is just one non-zero coefficient and its monomial in case the polynomial is not identically zero. This way we get rid of a factor of m.*

This Remark and the Lemma above helps establish the following Theorem which lower bounds the value of t.

THEOREM 6.1. *There is a constant $c > 0$ such that the algorithm works correctly if*

$$t > c \cdot (mn \log(d + 1) + H).$$

As a Corollary we get Theorem 3.1.

6.2 A Number Theoretic Remark It is a result of Adelman, Pomerance and Rumely that there are infinitely many numbers m such that, the number of divisors of it of the form $p - 1$, where p is some prime is of the order $\exp(c \log m \log \log m)$. This means that the least common multiples of these divisors coming from these primes is very small.

A direct Corollary of Linnik's Theorem which seems to lie at the heart of our algorithm is the following number theoretic fact:

COROLLARY 6.1. *For any $t \geq 1$, there is a constant q and a set of primes $\{p_1, \ldots, p_t\}$, such that the least common multiple of the numbers $\{p_1 - 1, \ldots, p_t - 1\} \geq q^t$.*

It will be interesting to find other applications of this Corollary.

7 Acknowledgments

The authors would like to thank Salil Vadhan for bringing to their attention the paper by Klivans and Spielman [10] and Igor Shparlinski for referring us to [1].

References

[1] L. Adleman, C. Pomerance, R. Rumely *On distinguishing prime numbers from composite numbers*, Ann. Math., 117, 173–206, 1983.

[2] M. Agarwal, S. Biswas. *Primality and Identity Testing via Chinese Remaindering.* IEEE conference on Foundations of Computer Science, 1999.

[3] S. Arora, C. Lund, R. Motwani, M. Sudan, M.Szegedy, *Probabilistic checking of proofs: a new characterization of NP*, Journal of the ACM, Vol. 45 No. 1, 70–122, 1998.

[4] L. Babai, L. Fortnow, C. Lund, *Non-deterministic exponential time has twoprover interactive protocols.* Computational Complexity, 1(1):3-40, 1991.

[5] M. Blum, S. Khanna, *Designing Programs that Check their Work.* ACM Symposium on the Theory of Computing, 86-97, 1989.

[6] S. Chari, P. Rohatgi, A. Srinivasan, *Randomness-optimal unique element isolation with applications to perfect matching and related problems*, SIAM J. Comput. 24(5):1036-1050, 1995.

[7] Z. Chen, M. Kao, *Reducing randomness via irrational numbers*, ACM Symposium on Theory of Computing, 200-209, 1997.

[8] R. DeMillo, R. Lipton, *A probabilistic remark on algebraic program testing.* Information Processing Letters 7, 4 (1978), 193-195.

[9] D. R. Heath-Brown, *Zero-free regions for Dirichlet L-functions and the least prime in an arithmetic progression* , Proc. Lond. Math. Soc., 64, 265–338, 1992.

[10] A. Klivans, D. Spielman, *Randomness efficient identity testing of multivariate polynomials*, ACM Symposium on Theory of Computing, 216–223, 2001.

[11] D. Lewin, S. Vadhan, *Checking polynomial identities over any field: towards a derandomization?*,ACM Symposium on Theory of Computing, 438–447, 1998.

[12] Yu. V. Linnik, *On the least prime in an arithmetic progression, I. The basic theorem; II. The Deuring-Heilbronn's phenomenon*, Rec. Math. (Mat. Sbornik) N.S. 15(57), (1944). 139–178 and 347–368.

[13] R. Lipton, *New Directions in Testing*, Distributed Computing and Cryptography, DIMACS Series on Discrete Mathematics and Theoretical Computer Science 2 (1991), American Mathematical Society, 191-202.

[14] L. Lovasz, *On determinants, matchings and random algorithms*, In L.Budach, editor, Fundamentals of Computing Theory, Akademia-Verlag, 1979.

[15] C. Lund, L. Fortnow, H. Karloff, N. Nisan, *Algebraic methods for interactive proof systems.* Journal of the ACM, Vol. 39, 859-868, 1992.

[16] O. H. Michell, *Note on Determinants of Powers*, American Journal of Mathematics, Vol 4, 341-344, 1881.

[17] R. Motwani, P. Raghavan, **Randomized Algorithms**, Cambridge University Press, 1995.

[18] K. Mulmuley, U. Vazirani, V. Vazirani, *Matching is as Easy as Matrix Inversion*, ACM Symposium on Theory of Computing, 345-354, 1987.

[19] P. Ribenboim, **The Book of Prime Number Records**, Springer Verlag, 1989.

[20] J. Schwartz, *Fast probabilistic algorithms for verification of polynomial identities*, Journal of the ACM, Vol. 27, 701-717, 1980.

[21] A. Shamir, *IP=PSPACE*, Journal of the ACM, Vol. 39, No. 4, 869–877, 1992.

[22] R. Zippel, **Probabilistic Algorithms for Sparse Polynomials**. Ph.D. thesis, MIT 1979.