

# Cyclotomic Identity Testing and Applications

Nikhil Balaji<sup>1</sup>, Sylvain Perifel<sup>2</sup>, Mahsa Shirmohammadi<sup>2</sup>, and James Worrell<sup>1</sup>

<sup>1</sup>Department of Computer Science, University of Oxford, UK

<sup>2</sup>Université de Paris, CNRS, IRIF, F-75013 Paris, France

## Abstract

We consider the cyclotomic identity testing problem: given a polynomial  $f(x_1, \dots, x_k)$ , decide whether  $f(\zeta_n^{e_1}, \dots, \zeta_n^{e_k})$  is zero, for  $\zeta_n = e^{2\pi i/n}$  a primitive complex  $n$ -th root of unity and integers  $e_1, \dots, e_k$ . We assume that  $n$  and  $e_1, \dots, e_k$  are represented in binary and consider several versions of the problem, according to the representation of  $f$ . For the case that  $f$  is given by an algebraic circuit we give a randomized polynomial-time algorithm with two-sided errors, showing that the problem lies in **BPP**. In case  $f$  is given by a circuit of polynomially bounded syntactic degree, we give a randomized algorithm with two-sided errors that runs in poly-logarithmic parallel time, showing that the problem lies in **BPNC**. In case  $f$  is given by a depth-2  $\Sigma\Pi$  circuit (or, equivalently, as a list of monomials), we show that the cyclotomic identity testing problem lies in **NC**. Under the generalised Riemann hypothesis, we are able to extend this approach to obtain a polynomial-time algorithm also for a very simple subclass of depth-3  $\Sigma\Pi\Sigma$  circuits. We complement this last result by showing that for a more general class of depth-3  $\Sigma\Pi\Sigma$  circuits, a polynomial-time algorithm for the cyclotomic identity testing problem would yield a sub-exponential-time algorithm for polynomial identity testing. Finally, we use cyclotomic identity testing to give a new proof that equality of compressed strings, i.e., strings presented using context-free grammars, can be decided in **coRNC**: randomized **NC** with one-sided errors.

# 1 Introduction

Identity testing is a fundamental problem in algorithmic algebra. In particular, identity testing in number fields has been much studied in relation to solving systems of polynomial equations [Ge93, Koi96], polynomial identity testing [CK00], and decision problems on matrix groups and semigroups [CLZ00, BBC<sup>+</sup>96], among many other problems. Among number fields, cyclotomic fields, i.e., those generated by roots of unity, play a particularly important role. The aim of this paper is a comprehensive study of the complexity of identity testing in cyclotomic fields.

We consider *cyclotomic identity testing problems*, where the input consists of a polynomial  $f(x_1, \dots, x_k)$  with integer coefficients together with integers  $n, e_1, \dots, e_k$ , and the task is to decide whether  $f(\zeta_n^{e_1}, \dots, \zeta_n^{e_k})$  is zero for  $\zeta_n = e^{2\pi i/n}$ . We consider four variants of this problem according to the representation of  $f$ : (i)  $f$  is given as an algebraic circuit; (ii)  $f$  is given by a circuit of polynomially bounded syntactic degree; (iii)  $f$  is given as a depth-2  $\Sigma\Pi$  circuit; (iv)  $f$  is given as a *diamond-shaped* depth-3  $\Sigma\Pi\Sigma$  circuit, that is, where the two  $+$ -layers each contain a single gate. Although  $f$  is a multivariate polynomial, since it is evaluated on powers of a common primitive  $n$ -th root of unity, in formalising the above four problems we use circuits whose input gates are labelled by powers of a single variable  $x$ .

Formally, for our purposes an algebraic circuit  $C$  is a directed, acyclic graph with labelled vertices and edges. Vertices of in-degree zero are labelled in the set of monomials  $\{x^e : e \in \mathbb{N}\}$  and the remaining vertices have labels in  $\{+, \times\}$ . Moreover the incoming edges to  $+$ -vertices have labels in  $\mathbb{Z}$ , that is, the  $+$ -gates compute integer-weighted sums. There is a unique vertex of out-degree zero which determines the output of the circuit, a univariate polynomial, in an obvious manner. We assume that all integer constants appearing in  $C$  are given in binary. The syntactic degree of  $C$  is defined inductively as follows: input gates have degree 0, the degree of an addition gate is the maximum of the degrees of its inputs, the degree of a multiplication gate is the sum of the degrees of its inputs, and the degree of  $C$  is the degree of the output gate. Note that the syntactic degree of  $C$  is *not* an upper bound on the degree of the computed polynomial since we allow monomials as inputs. We use notation such as  $\Sigma\Pi$  and  $\Sigma\Pi\Sigma$  to denote classes of circuits in which the internal gates are arranged into alternating layers of  $+$  and  $\times$  gates, with edges only between successive layers. Observe that this notation elides the variable powering at input gates in our formalism for univariate circuits.

The four main variants of the cyclotomic identity testing problem are as follows:

- In the *Cyclotomic Identity Testing (CIT)* problem the input is an algebraic circuit  $C$  representing a polynomial  $f(x)$ , together with an integer  $n$ , given in binary, and the task is to determine whether  $f(\zeta_n) = 0$ , where  $\zeta_n = e^{2\pi i/n}$  is a primitive complex  $n$ -th root of unity.
- The *Bounded-CIT* problem is defined exactly as the CIT problem, except that the input also includes an upper bound on the syntactic degree of the circuit  $C$  that is given in unary. Thus in Bounded-CIT the degree of the circuit is at most the length of the input.
- In the *Sparse-CIT* problem the polynomial  $f = \sum_{i=1}^s a_i x^{k_i}$  is given as a list of pairs of integers  $(a_1, k_1), \dots, (a_s, k_s)$  in binary. This is equivalent to restricting the CIT to  $\Sigma\Pi$  circuits.
- Finally we consider the restriction to CIT to *Diamond-shaped*  $\Sigma\Pi\Sigma$  circuits (where each  $+$ -layer has a single gate) which can be seen as a mild generalisation of the Sparse-CIT Problem.

The representation of polynomials in the CIT problem can be exponentially more succinct than in the Bounded-CIT problem, since the syntactic degree can be exponential in the size of the circuit. Likewise the representation in the Bounded-CIT problem can be exponentially more succinct than in the Sparse-CIT problem, since the former allows the number of monomials to be exponential in the circuit size. Diamond-shaped  $\Sigma\Pi\Sigma$  circuits are essentially the simplest non-trivial extension of the class of  $\Sigma\Pi$  circuits. Here, again, the number of monomials can be exponential in the circuit size.

The problem Sparse-CIT was first studied by Plaisted [Pla84], who gave a randomised polynomial-time algorithm. Subsequently, deterministic polynomial-time algorithms were given by Cheng et al. [CTV10] (See also [Che07]). A natural approach to decide zeroness of  $f(\zeta_n)$  is to compute an approximation of sufficient precision. However, given existing separation bounds for algebraic numbers, the precision required to distinguish between zero and a non-zero value precludes a polynomial time bound, and none of the existing polynomial-time procedures follows this naive route.

The conclusion of [CTV10] raises the question of the complexity of CIT. The authors note that this problem lies in the counting hierarchy<sup>1</sup>, based on results of [ABKPM09]. Our first main result is that CIT can be placed in BPP by computing modulo a suitable prime ideal in the ring of integers of the number field  $\mathbb{Q}(\zeta_n)$ . Effectively this amounts to working in a finite field  $\mathbb{Z}_p$  that contains a primitive  $n$ -th root of unity.

**Theorem 1.** *The CIT problem is in BPP.*

Observe that the CIT problem is at least as hard as the Polynomial Identity Testing problem for circuits of unbounded degree, which is a well-known P-hard problem [Mit13, Theorem 2.4.6, Theorem 2.6.3] (See also Proposition 7).

Next we pass to the Bounded-CIT problem, in which the syntactic degree of the circuit is polynomially bounded, and give a randomized procedure with two-sided errors that runs in polylogarithmic parallel time. Here we forsake the approach via finite arithmetic because computing powers in a finite field is not known to be in NC. Instead, we follow the identity testing method of Chen and Kao [CK00]: we pick a Galois conjugate of  $f(\zeta_n)$  uniformly at random and determine the zeroness of the conjugate by numerical computation. The reason that we have two-sided errors is that our procedure for generating conjugates fails with a small probability. Thus we have:

**Theorem 2.** *The Bounded-CIT problem is in BPNC.*

Moving to the problem Sparse-CIT, we revisit the approach of [CTV10] to giving a polynomial-time decision procedure. Here we give a simpler reformulation of their method and, as a by-product, we observe that the problem can be solved in NC.

**Theorem 3.** *The Sparse-CIT problem is in NC.*

We further build on Theorem 3 to give a polynomial-time algorithm for what is essentially the simplest non-sparse case of CIT, subject to the Generalised Riemann Hypothesis (GRH).

**Theorem 4.** *Assuming GRH, CIT can be solved in polynomial time on the class of diamond-shaped  $\Sigma\Pi\Sigma$  circuits.*

---

<sup>1</sup>The Counting Hierarchy [Wag86] is defined inductively as follows:  $\text{CH}_0 = \text{PP}$ ,  $\text{CH}_{k+1} = \text{PP}^{\text{CH}_k}$  and  $\text{CH} = \bigcup_{k \geq 0} \text{CH}_k$ . CIT is known to admit an upper bound of  $\text{P}^{\text{PP}^{\text{PP}^{\text{PP}}}}$  which is between  $\text{CH}_3$  and  $\text{CH}_4$ .

We complement Theorem 4 by exhibiting a class of  $\Sigma\Pi\Sigma$  circuits for which CIT is hard. In fact we formulate this result in terms of evaluating multivariate polynomials given by  $\Sigma\Pi$  circuits (i.e., with inputs being variables  $x_1, \dots, x_m$ ) on translations of roots of unity.

**Theorem 5.** *Given a multivariate polynomial  $f(x_1, \dots, x_m)$  as a  $\Sigma\Pi$  circuit and given integers  $a_1, \dots, a_m$  and  $e_1, \dots, e_m$  in binary, if one can test  $f(a_1 + \zeta_n^{e_1}, \dots, a_m + \zeta_n^{e_m}) = 0$  in deterministic polynomial time, then PIT for circuits of size  $s$  and degree  $d \leq s$  can be solved in  $s^{O(\sqrt{d})}$  time.*

## 1.1 Testing equality of compressed strings

In terms of applications, we observe that cyclotomic identity testing can be used to obtain a new coRNC algorithm to decide equality of compressed strings, that is, strings presented by acyclic context-free grammars. König and Lohrey [KL15] show that the problem admits a coRNC algorithm by reduction to the identity testing problem for univariate polynomials given as so-called powerful skew circuits. The main contribution of [KL15] was to give a randomised NC algorithm for the latter problem. Following the identity testing algorithm of Agrawal and Biswas [AB03], their algorithm works, by computing the value of the circuit modulo a randomly chosen polynomial  $p(x)$ . In order to perform this computation in NC they rely on the result of Fich and Tompa [FT88] that computing  $x^m \bmod p(x)$  for large powers  $m$  can be done in NC (assuming  $p$  is given in dense representation). By contrast, we observe that the same identity testing problem can be solved by numerically evaluating a polynomial at a randomly chosen conjugate of a root of unity  $\zeta_n$  of sufficiently high order. To obtain an NC bound we rely on the fact that it is straightforward to compute powers of  $\zeta_n$ . We also observe that our technique yields a randomised sequential algorithm that runs in  $\tilde{O}(n^2)$  time in the standard Turing machine model.

## 1.2 Discussion

Theorems 1, 2, and 3 all take different approaches to the CIT problem: respectively using finite arithmetic, numerical approximation, and multilinear algebra. However it is interesting to note that all three approaches involve computing a partial prime factorisation of the order of the root of unity (or some multiple thereof).

As discussed in more detail in [CTV10], cyclotomic identity testing is related to the so-called *torsion-point problem*, which asks whether a given multivariate polynomial has a zero in which all components are roots of unity [Roj07]. The univariate version of this problem is known to be NP-hard [Pla84]. Identity testing for expressions involving real roots of rational numbers is considered in [Blö98].

There has been extensive work on the problem of testing equality of compressed strings starting with the works of Hirschfeld et al. [HJM94] ( $O(n^4)$  time) and Melhorn et al. [MSU97] ( $O(n^3)$  time) who independently gave the first deterministic polynomial time algorithms for the problem. The state of the art for deterministic sequential algorithms for this problem is by Jéz [Jez12] where in he uses *recompression* to give an algorithm that runs in  $O(n^2)$  time. Note that the quadratic running time here is in a RAM model where the uncompressed string (which could be  $2^n$  letters long) fits into a single machine word. There have simpler randomized algorithms starting with the work of Gasieniec et al. [GKPR96] ( $\tilde{O}(n^2)$  time) and Schmidt-Schau and Schnitger ( $O(n^2)$  time in the RAM model). However neither of them is known to be parallelisable. [GKPR96] raised the question of whether testing compressed string equality is P-complete.

## 2 Preliminaries

We give a useful lemma on finite arithmetic and then recall some basic definitions and facts about cyclotomic fields.

**Lemma 6.** *Fix  $m \in \mathbb{N}$  and consider drawing an element  $k$  uniformly at random from the set  $\{1, \dots, m-1\}$ . Let  $A$  be the event that  $k$  and  $m$  are coprime and let  $B$  be the event that  $k$  and  $m$  share no common prime divisor  $p < 10 \log m$ . Then  $\Pr(A \mid B) > \frac{9}{10}$  for  $m$  sufficiently large.*

*Proof.* Write  $m = p_1^{e_1} \cdots p_r^{e_r}$ , where  $p_1, \dots, p_r$  are distinct primes and  $e_1, \dots, e_r \geq 1$ . For  $i = 1, \dots, r$ , let  $E_i$  be the event that  $p_i$  does not divide the sampled number  $k$ . Then the collection of events  $E_i$  is mutually independent,  $A = \bigcap_{i=1}^r E_i$ , and  $B = \bigcap_{i: p_i < 10 \log m} E_i$ . Thus

$$\begin{aligned} \Pr(A \mid B) &= \frac{\Pr(A)}{\Pr(B)} \\ &= \prod_{i: p_i \geq 10 \log m} \Pr(E_i) \\ &= \prod_{i: p_i \geq 10 \log m} \left(1 - \frac{1}{p_i}\right) \\ &\geq \left(1 - \frac{1}{10 \log m}\right)^{\log m}. \end{aligned}$$

Since the expression above converges to  $e^{-0.1} > 0.9$  as  $m$  tends to infinity, for sufficiently large  $m$  we have  $\Pr(A \mid B) > \frac{9}{10}$ .  $\square$

Fix  $n \in \mathbb{N}$  and write  $\mathbb{Q}(\zeta_n)$  for the field generated over  $\mathbb{Q}$  by a primitive complex  $n$ -th root of unity  $\zeta_n = e^{\frac{2\pi i}{n}}$ . The minimum polynomial of  $\zeta_n$  is denoted  $\Phi_n(x)$  and has degree  $\varphi(n)$ , where  $\varphi$  is the Euler totient function. We recall the lower bound [HW<sup>+</sup>79, Theorem 328]

$$\varphi(n) \geq \frac{cn}{\log \log n}, \tag{1}$$

where  $c$  is an effectively computable constant. An easy consequence is the following

**Proposition 7.** *For any primitive  $n$ -th root of unity  $\zeta_n$  and univariate polynomial  $p(x) \in \mathbb{Z}[x]$  of degree strictly smaller than  $\varphi(n)$ ,  $p(\zeta_n) \neq 0$ .*

It is well known that  $\alpha \in \mathbb{Q}(\zeta_n)$  is an algebraic integer just in case  $\alpha = \sum_{j=0}^{n-1} a_j \zeta_n^j$  for some  $a_0, \dots, a_{n-1} \in \mathbb{Z}$ . We call such a number a *cyclotomic integer* and write  $\mathbb{Z}[\zeta_n]$  for the subring of  $\mathbb{Q}(\zeta_n)$  comprised of cyclotomic integers.

Let  $\text{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q})$  denote the group of automorphisms of  $\mathbb{Q}(\zeta_n)$ . Then  $\text{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q})$  is isomorphic to the multiplicative group  $\mathbb{Z}_n^*$  of integers mod  $n$ . For each  $k \in \mathbb{Z}_n^*$ , the corresponding automorphism in  $\text{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q})$  sends  $\zeta_n$  to  $\zeta_n^k$ .

## 3 A Randomised Polynomial-time Algorithm for CIT

In this section we give a randomised polynomial-time algorithm, with two-sided errors, for the CIT problem. The idea is to work in a finite field, obtained by quotienting the ring of cyclotomic integers by a suitable rational prime.

Recall that the *norm* of  $\alpha \in \mathbb{Q}(\zeta_n)$  is defined by

$$N_{\mathbb{Q}(\zeta_n)/\mathbb{Q}}(\alpha) := \prod_{\sigma \in \text{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q})} \sigma(\alpha).$$

For short, we will write  $N(\alpha)$  for  $N_{\mathbb{Q}(\zeta_n)/\mathbb{Q}}(\alpha)$ , i.e., the underlying field will be understood from the context. Recall that the norm of a cyclotomic integer lies in  $\mathbb{Z}$ .

If a polynomial  $f \in \mathbb{Z}[x]$  is computed by a circuit  $C$  and if  $s \in \mathbb{N}$  is the sum of size of  $C$  and bit-length of  $n$ , then we say that the cyclotomic integer  $f(\zeta_n)$  is *computed by a circuit of size  $s$* .

**Proposition 8** (Norm upper bound for circuits). *Let  $\alpha \in \mathbb{Z}(\zeta_n)$  be a cyclotomic integer that is computed by a circuit of size  $s$ . Then  $|N(\alpha)| \leq 2^{2^{2s}}$ .*

*Proof.* Write  $\alpha = \sum_{j=0}^{n-1} a_j \zeta_n^j$ , where  $a_0, \dots, a_{n-1} \in \mathbb{Z}$  and let  $H := \sum_{0 \leq i \leq n-1} |a_i|$ . Since  $\alpha$  is computed by a circuit of size  $s$ , by an easy induction on  $s$  we have  $H \leq 2^{2^s}$ . We can give an upper bound on  $|N(\alpha)|$  as follows:

$$\begin{aligned} N(\alpha) &= N \left( \sum_{j=0}^{n-1} a_j \zeta_n^j \right) \\ &= \prod_{\sigma \in \text{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q})} \sigma \left( \sum_{j=0}^{n-1} a_j \zeta_n^j \right) \\ &= \prod_{\ell \in \mathbb{Z}_n^*} \left( \sum_{j=0}^{n-1} a_j \zeta_n^{j\ell} \right). \end{aligned}$$

Since  $\left| \sum_{j=0}^{n-1} a_j \zeta_n^{j\ell} \right| \leq \sum_{i=0}^{n-1} |a_i| = H$  for all  $\ell$ , we have

$$|N(\alpha)| \leq \prod_{\ell \in \mathbb{Z}_n^*} H \leq (2^{2^s})^n \leq (2^{2^s})^{2^s} = 2^{2^{2s}}.$$

□

**Theorem 9.** *Let  $p \in \mathbb{Z}$  be a prime such that the field  $\mathbb{Z}_p$  contains a primitive  $n$ -th root of unity  $\omega_n$ . Given  $g(x) \in \mathbb{Z}[x]$ , we have that*

1. *if  $g(\zeta_n) = 0$  then  $g(\omega_n) = 0$ , and*
2. *if  $g(\omega_n) = 0$  then  $p \mid N(g(\zeta_n))$ .*

*Proof.* Define a ring homomorphism  $\text{ev} : \mathbb{Z}[x] \rightarrow \mathbb{Z}_p$  by  $\text{ev}(g) = g(\omega_n) \bmod p$ . For  $d < n$ , since  $\Phi_d \mid x^d - 1$  and  $\text{ev}(x^d - 1) \neq 0$ , we have  $\text{ev}(\Phi_d) \neq 0$ . Since also  $x^n - 1 = \prod_{d \mid n} \Phi_d$ , we have  $\text{ev}(\Phi_n) = 0$ . It follows that  $\text{ev}$  factors through  $\mathbb{Z}(\zeta_n)$  via a homomorphism  $\text{ev}' : \mathbb{Z}(\zeta_n) \rightarrow \mathbb{Z}_p$  given by  $\text{ev}'(g(\zeta_n)) = g(\omega_n) \bmod p$  for  $g \in \mathbb{Z}[x]$ .

For Item 1, we have that if  $g(\zeta_n) = 0$  then  $g(\omega_n) = \text{ev}'(g(\zeta_n)) = 0$ .

For Item 2, observe that the kernel of  $\text{ev}'$  is a prime ideal  $\mathfrak{p}$  in  $\mathbb{Z}(\zeta_n)$  satisfying  $\mathfrak{p} \cap \mathbb{Z} = p\mathbb{Z}$ . Hence if  $g(\omega_n) = 0$  then  $g(\zeta_n) \in \mathfrak{p}$  and so  $p \mid N(g(\zeta_n))$ . □

**Proposition 10.** *Let  $\alpha \in \mathbb{Z}(\zeta_n)$  be a non-zero cyclotomic integer that is computed by a circuit of size  $s$ . If  $p$  is chosen uniformly at random among primes in  $\mathbb{N}$  of magnitude at most  $2^{3s}$ , then with probability at least  $1 - \frac{3s}{2^s}$  we have that  $p \nmid N(g(\zeta_n))$ .*

*Proof.* By Proposition 8, the norm of  $\alpha$  has absolute value at most  $2^{2s}$ . It follows that  $N(\alpha)$  has at most  $2^{2s}$  distinct prime factors. There are at least  $\frac{2^{3s}}{3s}$  primes in the range  $[2, 2^{3s}]$ . Thus the probability that a prime  $p$  chosen uniformly at random does not divide the norm of  $\alpha$  is at least  $1 - \frac{3s}{2^s}$ .  $\square$

Proposition 10 suggests a natural test for CIT: evaluate the circuit in a finite field  $\mathbb{Z}_p$  that contains a primitive  $n$ -th root of unity. Since the multiplicative group  $\mathbb{Z}_p^*$  is cyclic, it is clear that  $\mathbb{Z}_p$  contains a primitive  $n$ -th root of unity just in case  $n \mid (p-1)$ , i.e.,  $p \equiv 1 \pmod n$ . We will use the following estimate on the density of primes that are congruent to 1 modulo  $n$ .

**Theorem 11** (Primes in arithmetic progressions). *Given  $a \in \mathbb{Z}_n^*$ , write  $\pi_{n,a}(x)$  for the number of primes less than  $x$  that are congruent to  $a$  modulo  $n$ . Then*

$$\pi_{n,a}(x) \geq \frac{x}{|\mathbb{Z}_n^*| \log x} - \frac{x}{2^{\sqrt{\log x}}} \quad (2)$$

**Proposition 12.** *Let  $\alpha \in \mathbb{Z}(\zeta_n)$  be computed by a circuit  $C$  and let  $s$  be an upper bound on the size of  $C$  and bit-length of  $n$ . If  $\alpha$  is non-zero and prime  $p \in \mathbb{Z}$  is chosen uniformly at random among those primes less than  $2^{s^4}$  that are congruent to 1 modulo  $n$ , then the probability that  $p$  divides  $N(\alpha)$  is at most  $2^{5s-s^4}$ .*

*Proof.* By Proposition 8, the norm of  $\alpha$  has absolute value at most  $2^{2s}$ . It follows that  $N(\alpha)$  has at most  $2^{2s}$  distinct prime factors. By Theorem 11, the number of primes less than  $2^{s^4}$  that are congruent to 1 modulo  $n$  is

$$\pi_{n,1}(2^{s^4}) \geq \frac{2^{s^4}}{|\mathbb{Z}_n^*| s^4} - \frac{2^{s^4}}{2^{s^2}} \geq 2^{s^4} \left( \frac{1}{2^s s^4} - \frac{1}{2^{s^2}} \right). \quad (3)$$

Assume that  $s \geq 5$ . It follows that  $\frac{1}{2^s s^4} - \frac{1}{2^{s^2}} \geq \frac{1}{2^{3s}}$ . Consequently, the probability that  $p$  divides  $N(\alpha)$  is at most  $2^{5s-s^4}$ .  $\square$

The following straightforward proposition enables us to find primitive  $n$ -th roots of unity in  $\mathbb{Z}_p$  in case  $p \equiv 1 \pmod n$

**Proposition 13.** *For a prime  $p$ , let  $h$  be chosen uniformly at random from the set*

$$\left\{ a \in \mathbb{Z}_p^* : \bigwedge_{2 < q < 10 \log(p-1)} a^{\frac{p-1}{q}} \neq 1 \right\}.$$

*Then  $h$  is a primitive root of  $\mathbb{Z}_p^*$  with probability at least 0.9.*

*Proof.* Fix a primitive root  $g \in \mathbb{Z}_p^*$ . For  $a$  distributed uniformly at random over  $\mathbb{Z}_p^*$ , we have that  $\log_g a$  is distributed uniformly at random over  $\{0, \dots, p-2\}$ . Moreover, for every prime divisor  $q$  of  $p-1$ ,  $q$  divides  $\log_g a$  if and only if  $a^{\frac{p-1}{q}} = 1 \pmod p$ . It follows that for  $h$  as in the statement of corollary,  $\log_g h$  is distributed uniformly at random among those elements in  $\{2, \dots, p-2\}$  that do not share a prime divisor less than  $10 \log(p-1)$  with  $p-1$ . Applying Lemma 6 we have that  $\log_g h$  is coprime with  $p-1$  with probability at least 0.9. But  $\log_g h$  is coprime with  $p-1$  if and only if  $h$  is itself a primitive root of  $\mathbb{Z}_p^*$ .  $\square$

<b>Algorithm for Cyclotomic Identity Testing</b>	
<b>Input:</b>	Algebraic circuit $C$ and integer $n$ , written in binary, of combined size $s$
<b>Output:</b>	Whether $f(\zeta_n) = 0$ for the polynomial $f(x)$ computed by $C$ .
1:	Pick $p$ u.a.r. from $\left\{q \in \mathbb{N} : q \leq 2^{s^4}, q \text{ prime, and } q \equiv 1 \pmod{n}\right\}$ .
2:	Pick $h$ u.a.r. from $\left\{a : a \in \mathbb{Z}_p^*, \bigwedge_{2 < q < \log(p-1)} a^{\frac{p-1}{q}} \neq 1\right\}$ .
3:	Set $\omega_n := h^{\frac{p-1}{n}} \in \mathbb{Z}_p^*$ .
4:	Output ‘Zero’ if $f(\omega_n) \equiv 0 \pmod{p}$ ; otherwise output ‘Non-Zero’.

Figure 1: Algorithm for Cyclotomic Identity Testing

We are now in a position to prove the main result of this section: namely that there is a randomized polynomial-time algorithm for CIT:

**Theorem 1.** *The CIT problem is in BPP.*

*Proof.* Figure 3 presents a Monte Carlo randomized algorithm for the CIT problem. The argument for the correctness of the algorithm is as follows. Let  $p$  be a prime such that  $p \equiv 1 \pmod{n}$ , as chosen in Line 1.

It follows from Proposition 13 that with probability at least 0.9, the element  $h \in \mathbb{Z}_p^*$  that is selected in Line 2 of the algorithm is a primitive root of  $\mathbb{Z}_p^*$ . Now let us bound the error of the algorithm under the assumption that  $h$  is indeed a primitive root of  $\mathbb{Z}_p^*$ . Note that in this case we have that  $\omega_n$ , as chosen in Line 3, is a primitive  $n$ -th root of unity in the field  $\mathbb{Z}_p$ . We consider two cases. First, suppose that  $f(\zeta_n) = 0$ ; then by Theorem 9 we have  $f(\omega_n) = 0$ , and hence the output is ‘Zero’. Second, suppose that  $f(\zeta_n) \neq 0$ . Then by Theorem 9 the output will be ‘Non-Zero’ provided that  $p$  does not divide  $N(f(\zeta_n))$ . But by Proposition 12 the probability that  $p$  does not divide  $N(f(\zeta_n))$  is at least  $1 - 2^{5s-s^4}$ . Thus, in total, the probability that the algorithm gives the wrong output is  $0.1 + 2^{5s-s^4}$ .

It is clear that the algorithm runs in polynomial time. In particular, in Line 1, since the asymptotic density of primes in the set  $\left\{q \in \mathbb{N} : q \leq 2^{s^4}, q \equiv 1 \pmod{n}\right\}$  is proportional to  $\frac{1}{s^4}$  we can find a prime in this set in polynomial time with arbitrary small constant error probability by random sampling.

□

## 4 A Randomised NC Algorithm for Bounded-CIT

We use the following well-known result (see Appendix A for a proof):

**Lemma 14.** *[Chen and Kao [CK00] and Blömer [Blö98]] Let  $\alpha$  be an algebraic integer where the absolute value of all its conjugates is at most  $B$ . For all  $b \in \mathbb{N}$ , a random conjugate  $\alpha'$  of  $\alpha$  satisfies  $|\alpha'| \leq 2^{-b}$ , with probability at most  $B/(b+B)$ .*

By above, given a polynomial  $f$  and an algebraic number  $\alpha$ , whose all conjugates have absolute value bounded by  $B$ , a straightforward randomized algorithm to decide whether  $f(\alpha) = 0$  is to randomly pick a *large conjugate*  $\alpha'$  and to approximate  $f(\alpha')$  with an error  $2^{-b}$ . The challenge in



<b>Algorithm for Bounded Cyclotomic Identity Testing</b>	
<b>Input:</b>	Algebraic circuit $C$ with a unary upper bound on its syntactic degree, and integer $n$ written in binary, of combined size $s$
<b>Output:</b>	Whether $f(\zeta_n) = 0$ for the polynomial $f(x)$ computed by $C$ .
1:	Pick uniformly at random $a \in \{1, \dots, n-1\}$ such that $a$ and $n$ have no common divisor less than $10 \log n$ .
2:	Compute $f(\tilde{\zeta}_n^a)$ , which is $f(\zeta_n^a)$ truncated up to an $O(s)$ -bit precision using Taylor expansion.
3:	Output "Zero" if $f(\tilde{\zeta}_n^a) = 0$ , otherwise output "Non-Zero".

Figure 2: Algorithm for Bounded Cyclotomic Identity Testing

such algorithms is how to pick a large conjugate with a high probability and how to bound the error of computation.

In the rest of this section, we prove Theorem 2. Given an algebraic circuit  $C$  with a unary upper bound on its syntactic degree, and an integer  $n$  written in binary, of combined size  $s$ . We decide whether  $f(\zeta_n) = 0$  for the polynomial  $f$  computed by  $C$  by the random algorithm in Figure 2. We argue that the algorithm can be implemented by a uniform family of two-sided error randomized circuits of polynomial size and polylogarithmic depth, and conclude that Bounded-CIT is in BPNC.

The random algorithm, in nutshell, approximates a random conjugates  $f(\zeta_n^a)$  of  $f(\zeta_n)$  with a precision of  $2^{-\Omega(s)}$ . The two-sided errors are due to

- picking  $a$  such that  $\zeta_n^a$  is not a conjugate of  $\zeta_n$  (note that it is not known whether checking  $\gcd(a, n) = 1$  can be done in NC);
- drawing a conjugate  $\zeta_n^a$  such that  $f(\zeta_n^a)$  is non-zero but too small to distinguish from zero within the allowed precision.

**The error bound of  $2^{-\Omega(s)}$ .** By the simple observation that the constants appearing in  $f$  and the number of its terms are at most  $2^s$ , we have that  $|f(\zeta_n)| \leq 2^{2s}$ . Using Lemma 14 with  $B = 2s$  and  $b = 4s$ , for  $f(\zeta_n) \neq 0$ , a random conjugates  $f(\zeta_n)$  have absolute value larger than  $2^{-4s}$ , with probability at least  $2/3$ .

Given a conjugate  $\zeta_n^\ell$  of  $\zeta_n$ , by the Taylor series approximation to  $\zeta_n = e^{\frac{2\pi i}{n}}$ , restricted to the first  $k$  terms, we define

$$\tilde{\zeta}_n^\ell = \sum_{j=0}^k \frac{1}{j!} \left( \frac{2\pi i \ell}{n} \right)^j.$$

Notice that the error here is

$$|\zeta_n^\ell - \tilde{\zeta}_n^\ell| \leq 2 \left( \frac{2\pi \ell}{n} \right)^{k+1} \frac{1}{(k+1)!} \leq \frac{2}{(k+1)!} \leq \frac{1}{k^{k/2}},$$

which is less than  $2^{-6s}$  if  $k \geq 12s$ .

By above, to compute  $f(\zeta_n)$  within an error  $< 2^{-4s}$ , it suffices to approximate  $e^{\frac{2\pi i \ell}{n}}$  to  $6s$  bits using the  $k$  terms of the Taylor series above. Then

$$|f(\zeta_n) - f(\widetilde{\zeta}_n)| \leq \sum_{j=0}^{2^s} a_j |\zeta_n^{\ell j} - \widetilde{\zeta}_n^{\ell j}| < \sum_{j=0}^{2^s} 2^s |2^{-6s}| \leq 2^{-4s}$$

which is the desired error.

**Probabilistic correctness.** In Line 1, the algorithm iteratively chooses random numbers from  $\{1, \dots, n-1\}$  until it finds an element  $a$  such that  $a$  and  $n$  have no common divisors less than  $10 \log n$ . By Lemma 6, we have that  $a$  is coprime with  $n$ , and hence  $\zeta_n^a$  is a conjugate of  $\zeta_n$ , with probability at least  $\frac{9}{10}$ .

Using Lemma 14 with  $B = 2s$  and  $b = 4s$ , a random conjugates of  $f(\zeta_n)$  has absolute value larger than  $2^{-4s}$ , with probability at least  $\frac{2}{3}$ .

There are two sources of errors. First,  $\zeta_n^a$  may not be a conjugate of  $\zeta_n$ . This leads to two-sided errors and happens with probability at most  $\frac{1}{10}$ . The second possibility is that  $f(\zeta_n) \neq 0$  but  $f(\zeta_n^a)$  is too small to distinguish from zero within the given precision. This happens with probability at most  $\frac{1}{3}$ . Thus the total error probability is at most  $\frac{13}{30}$ .

**Theorem 2.** *The Bounded-CIT problem is in BPNC.*

*Proof.* From the work of Valiant et al. [VSB83], given a polynomial degree arithmetic circuit of size  $s$ , one can construct an equivalent circuit of depth  $O(\log^2 s)$  and size  $O(s)$  with fan-in 2 multiplication and addition gates. Moreover, such a circuit can be constructed even in logarithmic space [AJMV98]. Since we would like to compute  $f(\zeta_n)$  to error at most  $2^{-\Omega(s)}$ , this requires maintaining  $O(s)$  bits at each gate of the circuit. Every bit of numbers produced at each gate can be computed by NC circuits of size at most  $O(s \log s)$  [RT92], and hence overall this results in an NC circuit of size  $\widetilde{O}(s^3)$ .  $\square$

## 4.1 Compressed Words and Powerful Skew Circuits

An algebraic circuit computing a univariate polynomial is said to be a *powerful skew circuit* if at least one input of every multiplication gate is a leaf. Here the word *powerful* reflects our convention that leaves can be labelled with monomials  $x^m$ , where  $m$  is given in binary. The class of powerful skew circuits was introduced by König and Lohrey [KL15], where they showed that the corresponding polynomial identity testing problem can be decided in **coRNC** by combining the classical PIT algorithm of Agrawal and Biswas [AB03] and the result of Fich and Tompa [FT88] that computing  $x^m \bmod p(x)$  for large powers  $m$  can be done in **NC**. The main motivation for studying this identity testing problem is that there is an **NC** reduction of the equivalence testing problem for compressed strings to identity testing for powerful skew circuits. Briefly, a compressed word is one that is given by an acyclic context-free grammar in which each non-terminal occurs on the left-hand side of exactly one production. Such a grammar produces a single word, whose length can be exponential in the number of non-terminals and productions. We refer to [KL15] for more details.

In this section we provide an alternative **coRNC** algorithm for PIT on powerful skew circuits, employing the same random conjugate technique used to solve the Bounded-CIT problem. Since

the syntactic degree of a powerful skew circuit is at most the number of gates we can use our Algorithm in Figure 2 to decide PIT over the class of powerful skew circuits: we simply pick a root of unity  $\zeta_n$  with  $n$  higher than the degree of the given polynomial  $f \in \mathbb{Z}[x]$ , and approximate a random conjugate of  $f(\zeta_n)$ .

Since the algorithm is insensitive to the choice of  $n$  as long as it is larger than the degree of  $f$  (that is at most  $2^s$  where  $s$  is the size of circuit), we use this freedom and by Proposition 7, choose  $n = 2^{4s}$  ensuring that  $\zeta_n^a$  is a conjugate of  $\zeta_n^a$  for all odd numbers  $a$ ,  $1 \leq a < n$ . This prevents one-side of error in our random algorithm for the Bounded-CIT problem (error caused by picking an non-conjugate in Line 1 of Figure 2); indeed, whenever  $f(\zeta_n) = 0$  our algorithm returns “Zero” almost-surely (with probability 1). Then we conclude the following corollary noting that the approximation is efficiently computable in randomized sequential time by using Brent’s algorithm [Bre76].

**Theorem 15.** *Testing equality of two compressed words, of combined size  $s$ ,*

1. *is solvable in  $\tilde{O}(s^2)$ -time randomized sequential algorithm; and*
2. *can be implemented by  $\tilde{O}(s^3)$ -sized  $\text{NC}^2$  circuits using  $O(s)$  random bits.*

*Proof.* For the first item, having chosen the random conjugate  $\zeta_n^a$ , for each  $x^m$ , inputted to a multiplication gate, we need to compute  $f(\zeta_n^{am})$  truncated up to an  $O(s)$ -bit precision using Taylor expansion. By Brent’s algorithm [Bre76], for each  $k$ ,  $1 \leq k \leq n$ , we can compute  $e^{\frac{2\pi i k}{n}}$  within an error of  $2^{-O(s)}$  in  $O(s \log s)$  time. Since there are at most  $O(s)$  such different occurrences of  $\zeta_n^{am}$  in the powerful skew circuit, all these  $O(s)$ -bit approximations can be computed in  $O(s^2 \log s)$ -time.

We are now left with the task of evaluating a powerful-skew arithmetic circuit that has  $O(s)$  binary additions and  $O(s)$  binary multiplications on  $O(s)$ -bit numbers. Addition and multiplication of two  $O(s)$ -bit integers can be implemented in  $O(s)$  and  $O(s \log s)$  time respectively. Hence, for the whole circuit this can be implemented with an additional time complexity of  $O(s^2) + O(s^2 \log s)$ . Hence the overall time complexity is  $\tilde{O}(s^2)$ . The number of random bits used is  $O(\log n) = O(s)$  (to select a conjugate of  $\zeta_n$ ). Notice that in a RAM model where each operation is unit cost, this results in a  $O(s)$ -time algorithm, and in the log-cost model a  $O(s \log s)$ -time algorithm.

The second item is an immediate consequence of Theorem 2 and its proof.  $\square$

## 5 An NC Algorithm for Sparse-CIT

In this section we revisit the method of [CTV10] for solving Sparse-CIT in polynomial time. The main idea of [CTV10] is to give a tensor decomposition of the space of all polynomials that vanish on a given root of unity  $\zeta_n$ , based on a partial factorisation of the order  $n$ , and then to use sparsity to efficiently determine membership of this (exponential-dimension) space. Below we reformulate this idea so as to avoid working with spaces of exponential dimension, relying instead on Proposition 19—a simple proposition in multi-linear algebra. With this proposition in hand, it is straightforward to place the problem Sparse-CIT in NC.

Let  $\zeta_n$  denote a primitive  $n$ -th root of unity for a positive integer  $n$ . Given nonnegative integers  $0 \leq k_1 < \dots < k_s < n$ , we aim to compute the space of *vanishing sums*

$$V_n^{(k_1, \dots, k_s)} := \left\{ a \in \mathbb{Q}^s : \sum_{i=1}^s a_i \zeta_n^{k_i} = 0 \right\}$$

in time polynomial in the total bit length of  $n$  and  $k_1, \dots, k_s$ .

In the approach of [CTV10] the following (which is an easy consequence of the Chinese Remainder Theorem) plays a central role:

**Proposition 16.** *Suppose that  $n = n_1 n_2$  for positive integers  $n, n_1, n_2$ , with  $n_1$  and  $n_2$  coprime. Then the map  $\zeta_n \mapsto \zeta_{n_1} \otimes \zeta_{n_2}$  defines a  $\mathbb{Q}$ -algebra isomorphism between  $\mathbb{Q}(\zeta_n)$  and  $\mathbb{Q}(\zeta_{n_1}) \otimes \mathbb{Q}(\zeta_{n_2})$ .*

## 5.1 Prime Powers

We first recall how to compute the space of vanishing sums  $V_n^{(k_1, \dots, k_s)}$  for  $n$  a prime power.

**Proposition 17.** *Let  $p$  be a prime,  $e$  a positive integer, and let  $0 \leq k_1 < \dots < k_s < p^e$  be non-negative integers. Given  $a \in \mathbb{R}^s$ , we have  $\sum_{i=1}^s a_i \zeta_{p^e}^{k_i} = 0$  if and only if (i)  $a_i = a_j$  for all  $i, j$  such that  $k_i \equiv k_j \pmod{p^{e-1}}$  and (ii)  $a_i = 0$  for all  $i$  such that  $\#\{k_j : k_i \equiv k_j \pmod{p^{e-1}}\} < p$ .*

*Proof.* Recall that the minimal polynomial of  $\zeta_{p^e}$  is

$$f(x) = 1 + x^{p^{e-1}} + x^{2p^{e-1}} + \dots + x^{(p-1)p^{e-1}}.$$

For  $a \in \mathbb{Q}^s$  we have  $\sum_{i=1}^s a_i \zeta_{p^e}^{k_i} = 0$  if and only if there exists  $q \in \mathbb{Q}[x]$ ,  $\deg(q) < p^{e-1}$ , such that

$$\sum_{i=1}^s a_i x^{k_i} = q(x) f(x) = \sum_{i=0}^{p-1} q(x) x^{i(p^{e-1})}.$$

In other words, the polynomial  $\sum_{i=1}^s a_i x^{k_i}$  consists of  $p$  appropriately translated copies of  $q(x)$ . The result immediately follows.  $\square$

## 5.2 No Small Prime Divisors

Next we show how to compute the space of vanishing sums  $V_n^{(k_1, \dots, k_s)}$  in case  $n$  has no “small” prime divisors.

**Proposition 18.** *Let  $f(x) = \sum_{i=1}^s a_i x^{k_i} \in \mathbb{Q}[x]$  be a polynomial such that  $0 \leq k_1 < \dots < k_s < n$  and suppose that  $p > s$  for all prime divisors  $p$  of  $n$ . Then  $f(\zeta_n) = 0$  only if  $f$  is identically zero.*

*Proof.* Write  $n = p_1^{e_1} \dots p_m^{e_m}$  for the prime factorization of  $n$ . Write  $\ell_{ij} := k_i \bmod p_j^{e_j}$  for  $i = 1, \dots, s$  and  $j = 1, \dots, m$ . By the Chinese Remainder Theorem the  $m$ -tuples  $\ell_i = (\ell_{i1}, \dots, \ell_{im})$ ,  $i = 1, \dots, s$ , are all distinct. Now we have

$$\begin{aligned} f(\zeta_n) = 0 &\Leftrightarrow \sum_{i=1}^s a_i \zeta_n^{k_i} = 0 \\ &\Leftrightarrow \sum_{i=1}^s a_i (\zeta_{p_1^{e_1}}^{\ell_{i1}} \otimes \dots \otimes \zeta_{p_m^{e_m}}^{\ell_{im}}) = 0. \end{aligned}$$

But, by Proposition 17,  $\left\{\zeta_{p_j^{e_j}}^{\ell_{1j}}, \dots, \zeta_{p_j^{e_j}}^{\ell_{sj}}\right\}$  is a linearly independent set in  $\mathbb{Q}(\zeta_{p_j^{e_j}})$  for all  $j = 1, \dots, m$  (possibly listed with repetitions). It follows that

$$\left\{\zeta_{p_1^{e_1}}^{\ell_{i1}} \otimes \dots \otimes \zeta_{p_m^{e_m}}^{\ell_{im}} : i = 1, \dots, s\right\}$$

is a linearly independent set in  $\mathbb{Q}(\zeta_n)$ . Since the  $\ell_i$  are all distinct we conclude that  $a_1 = \dots = a_s = 0$ .  $\square$

### 5.3 Putting Things Together

Given vectors  $a, b \in \mathbb{Q}^s$ , define the *Hadamard product*  $a \odot b \in \mathbb{Q}^s$  by  $a \odot b := (a_1 b_1, \dots, a_s b_s)$ .

In general, for a nonnegative integer  $k$  and list of vectors  $w_1, \dots, w_s \in \mathbb{Q}^k$ , write  $R(w_1, \dots, w_s)$  for the row space of the matrix with columns  $w_1, \dots, w_s$ . Recall that  $R(w_1, \dots, w_s)$  is the orthogonal complement of  $\{a \in \mathbb{Q}^s : \sum_{i=1}^s a_i w_i = 0\}$ .

**Proposition 19.** *Let  $U, V$  be finite dimensional vector spaces over  $\mathbb{Q}$  with  $u_1, \dots, u_s \in U$  and  $v_1, \dots, v_s \in V$  for some  $s \in \mathbb{N}$ . Define the following three vector subspaces of  $\mathbb{Q}^s$ :*

$$\begin{aligned} A &:= \{a \in \mathbb{Q}^s : \sum_{i=1}^s a_i u_i = 0\} \\ B &:= \{b \in \mathbb{Q}^s : \sum_{i=1}^s b_i v_i = 0\} \\ C &:= \{c \in \mathbb{Q}^s : \sum_{i=1}^s c_i (u_i \otimes v_i) = 0\} . \end{aligned}$$

Then  $C^\perp = \{a \odot b : a \in A^\perp, b \in B^\perp\}$ .

*Proof.* Without loss of generality, suppose that  $U = \mathbb{Q}^m$  and  $V = \mathbb{Q}^n$ . Then we can identify  $U \otimes V$  with  $\mathbb{Q}^{mn}$  by taking  $u \otimes v$  to be the Kronecker product of  $u \in U$  and  $v \in V$ . Now we have

$$\begin{aligned} A^\perp &= R(u_1, \dots, u_s) \\ B^\perp &= R(v_1, \dots, v_s) \\ C^\perp &= R(u_1 \otimes v_1, \dots, u_s \otimes v_s) . \end{aligned} \tag{4}$$

But it clearly also holds that

$$R(u_1 \otimes v_1, \dots, u_s \otimes v_s) = \text{span}(\{a \odot b : a \in R(u_1, \dots, u_s), b \in R(v_1, \dots, v_s)\}) . \tag{5}$$

The result follows immediately from Equations (4) and (5).  $\square$

**Theorem 3.** *The Sparse-CIT problem is in NC.*

*Proof.* Given  $f(x) = \sum_{i=0}^s a_i x^{k_i}$  and  $n \in \mathbb{N}$ , we wish to determine whether  $f(\zeta_n) = 0$ . We may assume without loss of generality that  $\deg(f) < n$ : otherwise take the remainder on division of  $f$  by  $x^n - 1$  (which is easy to do in NC).

Since integer division is in NC, given  $n \in \mathbb{N}$  one can compute in NC a factorisation  $n = p_1^{e_1} \dots p_\ell^{e_\ell} m$  such that  $p_1, \dots, p_s \leq s$  are prime and all prime factors of  $m$  are strictly greater than  $s$ .

Propositions 17 and 18 give respective characterisations of the vanishing spaces  $V_{p_i^{e_i}}^{(k_1, \dots, k_s)}$  for  $i = 1, \dots, \ell$  and  $V_m^{(k_1, \dots, k_s)}$  as sets of solutions of linear equations. This directly yields descriptions

of the respective orthogonal complements. Moreover, since only integer division required, the given characterisations can be computed in NC.

Finally, one uses Proposition 19 to combine the orthogonal complements of the individual vanishing spaces  $V_{p_i}^{(k_1, \dots, k_s)}$  and  $V_m^{(k_1, \dots, k_s)}$  to obtain the orthogonal complement of  $V_n^{(k_1, \dots, k_s)}$ . With the latter in hand we can directly test whether  $f(\zeta_n) = 0$ .

In terms of complexity, we remark that given sets of vectors  $a_1, \dots, a_m$  and  $b_1, \dots, b_n$  in  $\mathbb{Q}^s$ , one can compute in NC a maximal linearly independent subset of  $\{a_i \odot b_j : 1 \leq i \leq m, 1 \leq j \leq n\}$ . Thus we can combine a pair of vanishing spaces in NC and hence we can combine all vanishing spaces in NC by a straightforward divide-and-conquer approach.  $\square$

## 6 Diamond-Shaped $\Sigma\Pi\Sigma$ Circuits

The results of the previous section show that for the class of polynomials computed by  $\Sigma\Pi$  circuits, the Cyclotomic Identity Testing Problem is decidable in NC. In this section we move to a slightly more general setting: we give an algorithm for essentially the simplest non-trivial class of depth-3 circuits, namely  $\Sigma\Pi\Sigma$  circuits with a single gate in each  $+$ -layer. For obvious reasons, we call these circuits *diamond-shaped*. Such circuits compute polynomials  $g(x)$  of the form

$$g(x) := \sum_{i=1}^s b_i (a_1 x^{e_1} + \dots + a_m x^{e_m})^i,$$

for integer coefficients  $a_1, \dots, a_m$  and  $b_1, \dots, b_s$  and natural-number exponents  $e_1, \dots, e_m$ . We give an algorithm that solves the CIT for this class of circuits in polynomial time, assuming the Generalized Riemann Hypothesis (GRH).

**Theorem 4.** *Assuming GRH, CIT can be solved in polynomial time on the class of diamond-shaped  $\Sigma\Pi\Sigma$  circuits.*

*Proof.* The algorithm is given in Figure 6. It involves an integer parameter  $G(n)$  and a rational parameter  $\varepsilon(g)$  that are both functions of the input. We will say more about both parameters shortly, suffice to say for now that  $G(n)$  is chosen such that  $\{k \in \mathbb{Z}_n^* : 1 \leq k \leq G(n)\}$  generates  $\mathbb{Z}_n^*$ .

Line 1 refers to the action of the group  $\mathbb{Z}_n^*$  on field  $\mathbb{Q}(\zeta_n)$ , obtained by associating with  $\ell \in \mathbb{Z}_n^*$  the automorphism of  $\mathbb{Q}(\zeta_n)$  that maps  $\zeta_n$  to  $\zeta_n^\ell$ . Observe that if the algorithm halts in Line 2 then the output is correct: if  $f(\zeta_n)$  has more than  $s$  distinct conjugates then we cannot possibly have  $g(\zeta_n) = \sum_{i=1}^s b_i f(\zeta_n)^i = 0$ .

Now suppose that  $|\text{Orb}(f(\zeta_n))| \leq s$  in Line 2. We will use this assumption to bound the degree and height of  $g(\zeta_n)$ . (Recall that the degree and height of an algebraic integer are, respectively, the degree and height of its minimal polynomial.) By the assumption that  $\{k \in \mathbb{Z}_n^* : 1 \leq k \leq G(n)\}$  generates  $\mathbb{Z}_n^*$ , we have that  $\text{Orb}(f(\zeta_n))$  consists of all Galois conjugates of  $f(\zeta_n)$ . Since  $|\text{Orb}(f(\zeta_n))| \leq s$  it follows that  $f(\zeta_n)$ , and hence also  $g(\zeta_n)$ , have degree at most  $s$ . Furthermore, for every  $\ell \in \mathbb{Z}_n^*$  we have  $|g(\zeta_n^\ell)| \leq smM$ , where  $M$  is the maximum of  $|a_i b_j|$  for  $i \in \{1, \dots, m\}$  and  $j \in \{1, \dots, s\}$ . By writing the coefficients of the minimal polynomial of  $g(\zeta_n)$  in terms of the Galois conjugates of  $g(\zeta_n)$ , we have that  $g(\zeta_n)$  has height at most  $2^s(smM)^s$ .

Now, a non-zero algebraic number of degree  $d$  and height  $H$  has magnitude at least  $\frac{2}{d^{d+1}H^d}$ . We choose the value of  $\varepsilon(g)$  by substituting  $d := s$  and  $H := 2^s(smM)^s$  into this bound, that is, we

Algorithm for Diamond-Shaped $\Sigma\Pi\Sigma$ Circuits	
<b>Input:</b>	Polynomial $g(x) = \sum_{i=1}^s b_i(a_1x^{e_1} + \dots + a_mx^{e_m})^i$ .
<b>Output:</b>	Whether $g(\zeta_n) = 0$ .
1:	Let $f(x) := \sum_{i=1}^s a_ix^{e_i}$ and compute the orbit $\text{Orb}(f(\zeta_n))$ of $f(\zeta_n)$ w.r.t. the set $\{k \in \mathbb{Z}_n^* : k \leq G(n)\}$ .
2:	If $ \text{Orb}(f(\zeta_n))  > s$ then return "not zero".
3:	If $ \text{Orb}(f(\zeta_n))  \leq s$ then compute rational number $\alpha$ such that $ \alpha - g(\zeta_n)  < \frac{\varepsilon(g)}{3}$ and return 'zero' if $\alpha < \frac{\varepsilon(g)}{3}$ and return 'not zero' otherwise.

Figure 3: Algorithm for Diamond-Shaped  $\Sigma\Pi\Sigma$  Circuits

define

$$\varepsilon(g) := \frac{2}{s^{s+1}(2smM)^{s^2}}. \quad (6)$$

With this choice, if  $g(\zeta_n) \neq 0$  then  $|g(\zeta_n)| > \varepsilon(g)$ : hence for the number  $\alpha$  computed in Line 3 we have  $|\alpha| > \frac{2\varepsilon(g)}{3}$ . On the other hand, if  $g(\zeta_n) = 0$  then  $|\alpha| < \frac{\varepsilon(g)}{3}$ . Thus the output produced in Line 3 is correct. This completes the proof that the algorithm gives the correct output.

We turn now to the complexity. Note that we can use the procedure presented in the previous section to determine in polynomial time whether or not two conjugates  $f(\zeta_n^\ell)$  and  $f(\zeta_n^j)$  are identical. Since the computation of  $\text{Orb}(f(\zeta_n))$  terminates as soon as  $|\text{Orb}(f(\zeta_n))| > s$ , we see that Line 1 can be executed in time polynomial in the size of the input and the parameter  $G(n)$ . Now it was shown in [Mon71] that under GRH there is a function  $G(n) = O(\log^2 n)$  such that  $\{k \in \mathbb{Z}_n^* : 1 \leq k \leq G(n)\}$  generates  $\mathbb{Z}_n^*$ .<sup>2</sup> It follows that Line 1 of the procedure can be executed in polynomial time, assuming GRH. Finally, from Expression (6) we see that  $|\log(\varepsilon(g))|$  is polynomially bounded in the input size. Thus  $g(\zeta_n)$  can be computed to within precision  $\frac{\varepsilon(g)}{3}$  in polynomial time, e.g., using the approach described in Section 4.  $\square$

## 7 Lower Bounds

We now show that efficient deterministic algorithms for a mild generalisation of Sparse-CIT, entails sub-exponential time algorithms for the Polynomial Identity Testing problem, a longstanding open problem in Complexity theory [Sax09, Sax14, KI04]. More formally, we have the following:

**Theorem 5.** *Given a multivariate polynomial  $f(x_1, \dots, x_m)$  as a  $\Sigma\Pi$  circuit and given integers  $a_1, \dots, a_m$  and  $e_1, \dots, e_m$  in binary, if one can test  $f(a_1 + \zeta_n^{e_1}, \dots, a_m + \zeta_n^{e_m}) = 0$  in deterministic polynomial time, then PIT for circuits of size  $s$  and degree  $d \leq s$  can be solved in  $s^{O(\sqrt{d})}$  time.*

*Proof.* Similar to Koiran [Koi11, Proposition 1], we combine depth reduction and Kronecker substitution. We start from the following result about the expressiveness of depth-4 circuits (see for example [Sap15, Theorem 5.17] or [KKPS15, Proposition 1]):

<sup>2</sup>The paper [BH93] gives heuristic arguments and experimental data suggesting that the choice  $G(n) = (\log 2)^{-1} \log n \log \log n$  will yield a set of generators.

**Theorem 20.** Any  $m$ -variate polynomial  $P(\mathbf{x})$  of degree at most  $d = m^{O(1)}$  computed by a circuit of size  $s = m^{O(1)}$  can be expressed as:

$$P(\mathbf{x}) = \sum_{i=1}^{s^{O(\sqrt{d})}} c_i Q_i(\mathbf{x})^{d_i} \quad (7)$$

where  $c_i \in \mathbb{Q}$ ,  $d_i = O(\sqrt{d})$  and the  $Q_i(\mathbf{x})$  are multivariate polynomials of sparsity at most  $s^{O(\sqrt{d})}$  and degree  $\sqrt{d}$ . Moreover, such a representation can be computed in  $s^{O(\sqrt{d})}$  time.

It follows that a  $\text{poly}(m, s, d)$  time algorithm for identity testing depth-4 circuits of the form in Equation 7 yields an  $s^{O(\sqrt{d})}$  time algorithm for identity testing circuits of arbitrary-depth computing a low degree polynomial.

Let  $P$  be a polynomial of degree  $\leq d$  as in Equation 7. It is easy to see that the univariate polynomial  $p(x) = P(x, x^{(d+1)}, \dots, x^{(d+1)^{m-1}})$  of degree at most  $(d+1)^m$  is non-zero if and only if the multivariate polynomial  $P(\mathbf{x})$  is non-zero. Thus a  $\text{poly}(m, s, d)$  time algorithm for identity testing univariate polynomials of the form

$$\begin{aligned} p(x) &= \sum_{i=1}^s c_i q_i(x)^{d_i} \\ &= \sum_{i=1}^s c_i (a_{i1}x^{e_{i1}} + \dots + a_{is}x^{e_{is}})^{d_i}, \end{aligned} \quad (8)$$

where  $q_i(x) = Q_i(x, x^{(d+1)}, \dots, x^{(d+1)^{m-1}}) = a_{i1}x^{e_{i1}} + \dots + a_{is}x^{e_{is}}$ ,  $e_{ij} \leq (d+1)^m$  and  $d_i \leq d$ , is sufficient to get sub-exponential time algorithms for PIT.

We will now show that the univariate polynomial  $p(x)$  obtained above can be expressed as in the statement of the theorem. We will need the following version (Lemma 4.7 in [GKKS16]) of a lemma originally due to Saxena [Sax08]:

**Lemma 21.** For every  $m, d > 0$ , there exist  $\alpha_i, \beta_{ij} \in \mathbb{Q}$  ( $0 \leq i \leq md, 0 \leq j \leq d$ ) such that

$$(u_1 + \dots + u_m)^d = \sum_{i=0}^{md} \sum_{j=0}^d \beta_{ij} (u_1 + \alpha_i)^j \dots (u_m + \alpha_i)^j.$$

We provide a proof of this lemma in Appendix B for the sake of completeness. Applying Lemma 21 to the terms  $(a_{i1}x^{e_{i1}} + \dots + a_{is}x^{e_{is}})^{d_i}$  above, we get

$$\begin{aligned} (a_{i1}x^{e_{i1}} + \dots + a_{is}x^{e_{is}})^{d_i} &= \sum_{r=0}^{sd_i} \sum_{j=0}^{d_i} \beta_{irj} (a_{i1}x^{e_{i1}} + \alpha_r)^j \dots (a_{is}x^{e_{is}} + \alpha_r)^j \\ &= \sum_{r=0}^{sd_i} \sum_{j=0}^{d_i} \beta'_{irj} (x^{e_{i1}} + a_{ir1})^j \dots (x^{e_{is}} + a_{irs})^j \end{aligned}$$



where  $\beta'_{irj} = \beta_{irj}/(a_{i1}^j \dots a_{is}^j)$  and  $a_{ir1} = \alpha_r/a_{i1}, \dots, a_{irs} = \alpha_r/a_{is}$ . After plugging this into Equation 8, we get

$$p(x) = \sum_{i=1}^s c_i \sum_{r=0}^{sd_i} \sum_{j=0}^{d_i} \beta'_{irj} (x^{e_{i1}} + a_{ir1})^j \dots (x^{e_{is}} + a_{irs})^j,$$

which is a polynomial  $f$  of degree  $\leq d(d+1)^m$ , sparsity  $\leq s(d+1)(sd+1)$ , in  $\leq s^2(sd+1)$  variables and evaluated at  $((x^{e_{i1}} + a_{ir1}), \dots, (x^{e_{is}} + a_{irs}))_{i,r}$ .

Testing if  $p$  is zero can now be done by deciding whether  $f((\zeta_n^{e_{i1}} + a_{ir1}, \dots, \zeta_n^{e_{is}} + a_{irs})_{i,r}) = 0$  where  $n > d(d+1)^m$ , thanks to Proposition 7. Thus, if CIT for this particular form is in deterministic polynomial time, this yields a  $\text{poly}(s, d, m)$  time PIT for  $p$  and hence for depth-4 circuits, proving the theorem.  $\square$

## References

- [AB03] Manindra Agrawal and Somenath Biswas. Primality and identity testing via chinese remaindering. *Journal of the ACM (JACM)*, 50(4):429–443, 2003.
- [ABKPM09] Eric Allender, Peter Bürgisser, Johan Kjeldgaard-Pedersen, and Peter Bro Miltersen. On the complexity of numerical analysis. *SIAM Journal on Computing*, 38(5):1987–2006, 2009.
- [AJMV98] Eric Allender, Jia Jiao, Meena Mahajan, and V Vinay. Non-commutative arithmetic circuits: depth reduction and size lower bounds. *Theoretical Computer Science*, 209(1-2):47–86, 1998.
- [BBC<sup>+</sup>96] László Babai, Robert Beals, Jin-yi Cai, Gábor Ivanyos, and Eugene M Luks. Multiplicative equations over commuting matrices. In *Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 498–507, 1996.
- [BH93] Rric Bach and Lorenz Huelsbergen. Statistical evidence for small generating sets. *Mathematics of Computation*, 61(203):69–82, 1993.
- [Blö98] Johannes Blömer. A probabilistic zero-test for expressions involving roots of rational numbers. In *European Symposium on Algorithms*, pages 151–162. Springer, 1998.
- [Bre76] Richard P Brent. Fast multiple-precision evaluation of elementary functions. *Journal of the ACM (JACM)*, 23(2):242–251, 1976.
- [Che07] Qi Cheng. Derandomization of sparse cyclotomic integer zero testing. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 74–80. IEEE, 2007.
- [CK00] Zhi-Zhong Chen and Ming-Yang Kao. Reducing randomness via irrational numbers. *SIAM Journal on Computing*, 29(4):1247–1256, 2000.
- [CLZ00] Jin-yi Cai, Richard J Lipton, and Yeheskel Zalcstein. The complexity of the abc problem. *SIAM Journal on Computing*, 29(6):1878–1888, 2000.

- [CTV10] Qi Cheng, Sergey P Tarasov, and Mikhail N Vyalyi. Efficient algorithms for sparse cyclotomic integer zero testing. *Theory of Computing Systems*, 46(1):120–142, 2010.
- [FT88] Faith E Fich and Martin Tompa. The parallel complexity of exponentiating polynomials over finite fields. *Journal of the ACM (JACM)*, 35(3):651–667, 1988.
- [Ge93] Guoqiang Ge. Testing equalities of multiplicative representations in polynomial time. In *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pages 422–426. IEEE, 1993.
- [GKKS16] Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth 3. *SIAM Journal on Computing*, 45(3):1064–1079, 2016.
- [GKPR96] Leszek Gasieniec, Marek Karpinski, Wojciech Plandowski, and Wojciech Rytter. Randomized efficient algorithms for compressed strings: The finger-print approach (extended abstract). In Daniel S. Hirschberg and Eugene W. Myers, editors, *Combinatorial Pattern Matching, 7th Annual Symposium, CPM 96, Laguna Beach, California, USA, June 10-12, 1996, Proceedings*, volume 1075 of *Lecture Notes in Computer Science*, pages 39–49. Springer, 1996.
- [HJM94] Yoram Hirshfeld, Mark Jerrum, and Faron Moller. A polynomial-time algorithm for deciding equivalence of normed context-free processes. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 623–631. IEEE Computer Society, 1994.
- [HW<sup>+</sup>79] Godfrey Harold Hardy, Edward Maitland Wright, et al. *An introduction to the theory of numbers*. Oxford university press, 1979.
- [Jez12] Artur Jez. Faster fully compressed pattern matching by recompression. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 533–544. Springer, 2012.
- [KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *computational complexity*, 13(1-2):1–46, 2004.
- [KKPS15] Neeraj Kayal, Pascal Koiran, Timothée Pecatte, and Chandan Saha. Lower bounds for sums of powers of low degree univariates. In *International Colloquium on Automata, Languages, and Programming*, pages 810–821. Springer, 2015.
- [KL15] Daniel König and Markus Lohrey. Parallel identity testing for skew circuits with big powers and applications. In *International Symposium on Mathematical Foundations of Computer Science*, pages 445–458. Springer, 2015.
- [Koi96] Pascal Koiran. Hilbert’s nullstellensatz is in the polynomial hierarchy. *Journal of complexity*, 12(4):273–286, 1996.

- [Koi11] Pascal Koiran. Shallow circuits with high-powered inputs. In Bernard Chazelle, editor, *Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 309–320. Tsinghua University Press, 2011.
- [Mit13] Johannes Mittmann. *Independence in algebraic complexity theory*. PhD thesis, Universitäts- und Landesbibliothek Bonn, 2013.
- [Mon71] H. L. Montgomery. Multiplicative number theory. 227, 1971.
- [MSU97] Kurt Mehlhorn, R. Sundar, and Christian Uhrig. Maintaining dynamic sequences under equality tests in polylogarithmic time. *Algorithmica*, 17(2):183–198, 1997.
- [Pla84] David A Plaisted. New np-hard and np-complete polynomial and integer divisibility problems. *Theoretical Computer Science*, 31(1-2):125–138, 1984.
- [Roj07] J.M. Rojas. Efficiently detecting subtori and torsion points. 448:213233, 2007.
- [RT92] John H Reif and Stephen R Tate. On threshold circuits and polynomial computation. *SIAM Journal on Computing*, 21(5):896–908, 1992.
- [Sap15] Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. *Github survey*, 2015.
- [Sax08] Nitin Saxena. Diagonal circuit identity testing and lower bounds. In *International Colloquium on Automata, Languages, and Programming*, pages 60–71. Springer, 2008.
- [Sax09] Nitin Saxena. Progress on polynomial identity testing. *Bulletin of the EATCS no.*, 99:49–79, 2009.
- [Sax14] Nitin Saxena. Progress on polynomial identity testing-ii. In *Perspectives in Computational Complexity*, pages 131–146. Springer, 2014.
- [VSB83] Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12(4):641–644, 1983.
- [Wag86] Klaus W Wagner. The complexity of combinatorial problems with succinct input representation. *Acta informatica*, 23(3):325–356, 1986.

## A Proofs from Section 4

**Lemma 22.** *[Chen and Kao [CK00] and Blömer [Blö98]] Let  $\alpha$  be an algebraic integer where the absolute value of all its conjugates is at most  $B$ . For all  $b \in \mathbb{N}$ , a random conjugate  $\alpha'$  of  $\alpha$  satisfies  $|\alpha'| \leq 2^{-b}$ , with probability at most  $B/(b+B)$ .*

*Proof.* For the algebraic integer  $\alpha$ , let  $\alpha_0 = \alpha, \alpha_1, \dots, \alpha_{d-1}$  be the conjugates. Let  $k$  be the number of conjugates that are at most  $2^{-b}$  in absolute value. Recall that  $|\prod_{i=0}^d \alpha_i|$  is the absolute value of the constant term of the minimal polynomial of  $\alpha$ . Since the minimal polynomial by definition is integral, the product  $|\prod_{i=0}^d \alpha_i|$  is at least 1. Together with the upper bound  $2^B$  on  $|\alpha_i|$  we get

$$1 \leq \left| \prod_{i=0}^d \alpha_i \right| \leq (2^B)^{d-k} (2^{-b})^k$$

This implies that

$$\begin{aligned} (2^B)^{d-k} (2^{-b})^k &\geq 1 \\ 2^{dB} 2^{-k(B+b)} &\geq 1 \\ dB - k(B+b) &\geq 0 \\ \frac{k}{d} &\leq \frac{B}{B+b} \end{aligned}$$

□

## B Proofs from Section 7

**Lemma 23.** *For every  $m, d > 0$ , there exist  $\alpha_i, \beta_{ij} \in \mathbb{Q}$  ( $0 \leq i \leq md, 0 \leq j \leq d$ ) such that*

$$(u_1 + \dots + u_m)^d = \sum_{i=0}^{md} \sum_{j=0}^d \beta_{ij} (u_1 + \alpha_i)^j \dots (u_m + \alpha_i)^j.$$

We provide a proof due to Gupta et al. [GKKS16]

*Proof.* Consider

$$\begin{aligned} P_u(z) &= (z + u_1) \dots (z + u_m) - z^m \\ &= z^{m-1}(u_1 + \dots + u_m) + \text{lower order terms} \\ \implies P_u(z)^d &= z^{(m-1)d}(u_1 + \dots + u_m)^d + \text{lower order terms} \end{aligned}$$

Hence we can compute  $(u_1 + \dots + u_m)^d$  as a coefficient of  $z^{(m-1)d}$  via interpolation by evaluating  $P_u(z)^d$  on  $(m-1)d$  points. That is, for every distinct  $\alpha_0, \dots, \alpha_{md} \in \mathbb{Q}$ , there exist  $\beta'_0 \dots \beta'_{md}$  such

that

$$\begin{aligned}
(u_1 + \dots + u_m)^d &= \sum_{i=0}^{md} \beta'_i P_u(z)^d \\
&= \sum_{i=0}^{md} \beta'_i ((u_1 + \alpha_i) \dots (u_m + \alpha_i) - \alpha_i^m)^d \\
&= \sum_{i=0}^{md} \beta'_i \sum_{j=0}^d \binom{d}{j} (-\alpha_i^m)^{(d-j)} ((u_1 + \alpha_i) \dots (u_m + \alpha_i))^j \\
&= \sum_{i=0}^{md} \sum_{j=0}^d \beta_{ij} ((u_1 + \alpha_i) \dots (u_m + \alpha_i))^j
\end{aligned}$$

where  $\beta_{ij} = \beta'_i \binom{d}{j} (-\alpha_i)^{m(d-j)}$ .

□