Electronic Colloquium on Computational Complexity, Report No. 202 (2015)

Building above read-once polynomials: identity testing and hardness of representation*

Meena Mahajan[1], B. V. Raghavendra Rao[2], and Karteek Sreenivasaiah[3]**

[1] The Institute of Mathematical Sciences, Chennai, India. meena@imsc.res.in
[2] Indian Institute of Technology Madras, Chennai, India. bvrr@cse.iitm.ac.in
[3] Max Planck Institute for Informatics, Saarbrücken, Germany. karteek@mpi-inf.mpg.de

Abstract. Polynomial Identity Testing (PIT) algorithms have focussed on polynomials computed either by small alternation-depth arithmetic circuits, or by read-restricted formulas. Read-once polynomials (ROPs) are computed by read-once formulas (ROFs) and are the simplest of read-restricted polynomials. Building structures above these, we show the following:

1. A deterministic polynomial-time non-black-box PIT algorithm for $\sum^{(2)} \cdot \prod \cdot \mathsf{ROF}$.
2. Weak hardness of representation theorems for sums of powers of constant-free ROPs and for ROFs of the form $\sum \cdot \prod \cdot \sum$.
3. A partial characterization of multilinear monotone constant-free ROPs.

1 Introduction

A polynomial is said to be identically zero if the coefficients of all monomials are zero. The Polynomial Identity Testing (PIT) problem is the most fundamental computational question that can be asked about polynomials: is the polynomial given by some implicit representation identically zero? The implicit representations of the polynomials can be arithmetic circuits, branching programs etc., or the polynomial could be presented as a black-box, where the black-box takes a query in the form of an assignment to the variables and outputs the evaluation of the polynomial on the assignment. PIT has a randomized polynomial time algorithm on almost all input representations, independently discovered by Schwartz [Sch80], by Zippel [Zip79] and by Demillo and Lipton in [DL78]. However, obtaining deterministic polynomial time algorithms for PIT remained open since then. In 2004, Kabanets and Impagliazzo [KI04] showed that a deterministic polynomial time algorithm for PIT implies lower bounds (either $\mathsf{NEXP} \not\subset \mathsf{P/poly}$ or permanent does not have polynomial size arithmetic circuits), thus making it one of the central problems in algebraic complexity. Following [KI04], intense efforts over the last decade have been directed towards de-randomizing PIT (see for instance [SY10,Sax14]). The attempts fall into two categories: considering special cases ([Sax14]), and optimizing the random bits used in the Schwartz-Zippel-Demillo-Lipton test [BHS08,BE11].

The recent progress on PIT mainly focusses on special cases where the polynomials are computed by restricted forms of arithmetic circuits. They can be seen as following one of the two main lines of restrictions: 1. Shallow circuits based on alternation depth of circuits

* Partially supported by the Indo-German Max Planck Center for Computer Science (IMPECS).
** Much of this work was done while the author was working in The Institute of Mathematical Sciences, Chennai, India.

ISSN 1433-8092

computing the polynomial. 2. Restriction on the number of times a variable is read by formulas (circuits with fanout 1) computing the polynomial.

The study of PIT on shallow circuits began with depth two circuits, where deterministic polynomial time algorithms are known even when the polynomial is given as a black-box [BOT88,KS01]. Further, there were several interesting approaches that led to deterministic PIT algorithms on depth three circuits with bounded top fan-in [DS07,KS07]. However, progressing from bounded top fan-in depth three circuits seemed to be a big challenge. In 2008, Agrawal and Vinay [AV08] explained this difficulty, showing that deterministic polynomial time algorithms for PIT on depth four circuits imply sub-exponential time deterministic algorithms for general circuits. There has been a lot of work towards obtaining black-box algorithms for PIT on restricted classes of depth three and four circuits, see [Sax14,SY10] for further details. Recently, Gupta, Kamath, Kayal and Saptharishi [GKKS13] showed that, over infinite fields, deterministic polynomial time algorithms for PIT on depth three circuits would also imply lower bounds for the permanent. Thus it is natural to look for other restricted models where deterministic polynomial time algorithms for PIT may be designed using current techniques.

A formula computing a polynomial that depends on all of its variables must read each variable at least once (count each leaf labeled x as reading the variable x). The simplest such formulas read each variable exactly once; these are Read-Once Formulas ROFs, and the polynomials computed by such formulas are known as read-once polynomials (ROP). In the case of an ROP f presented by a read-once formula computing it, a simple reachability algorithm on formulas can be applied to test if $f \equiv 0$. Shpilka and Volkovich [SV08] gave a deterministic polynomial time algorithm for PIT on ROPs in the black-box model. Generalizing this to formulas that read a variable more than once, they obtained a deterministic polynomial time algorithm for polynomials explicitly presented as a sum of $O(1)$ ROFs. Anderson et. al [AvMV11] showed that if a read-k formula, with $k \in O(1)$, is additionally restricted to compute multilinear polynomials at every gate, then PIT on such formulas can be done in deterministic polynomial time. The result by [AvMV11] subsumes the result in [SV08] since a k-sum of read-once formulas is read-k and computes multilinear polynomials at every gate. However, both [SV08] and [AvMV11] crucially exploit the multilinearity property of the polynomials computed under the respective models. In [MRS14], the authors explored eliminating the multilinear-at-each-gate restriction, and gave a non-blackbox deterministic polynomial time algorithm for read-3 formulas. However for the case of Read-k formulas for $k \geq 4$, even the non-blackbox version of the problem is open. Note that multilinearity checking itself is equivalent to PIT on general circuits [FMM12].

Our results: In this paper, we explore further structural properties of ROPs and polynomials that can be expressed as polynomial functions of a small number of ROPs. Our structural observations lead to efficient algorithms on special classes of bounded-read formulas.

We attempt to extend the class considered in [SV08] (namely, formulas of the form $\sum_i f_i$ where each f_i is an ROF) to the class of polynomials of the form $\sum_{i=1}^{k} f_i g_i$ where the f_is and g_is are presented as ROFs and k is some constant. These are read-$2k$ polynomials, not neces-

sarily multilinear. When $k = 2$, this class can be seen as a special case of read-4 polynomials. It turns out that we can in fact do much better. We describe an efficient deterministic non-blackbox PIT algorithm even when the number of polynomials in the product is unbounded. It solves PIT for polynomials of the form $f_1 f_2 f_3 \cdots f_m + g_1 g_2 \cdots g_s$ where f_is and g_is are presented as ROFs, but m, s can be unbounded; that is, the class $\sum^{(2)} \cdot \prod \cdot \mathsf{ROF}$. Note that this class of polynomials includes non-multilinear polynomials and also polynomials with no bound on the number of times variables are read. Thus it is incomparable with the classes considered in [SV08], [AvMV11] and [MRS14]. This result is presented in Section 3. First, we describe the algorithm over the ring of integers and the field of rationals; Theorem 1. In this case, even the bit-complexity is polynomial. Our algorithm exploits the structural decomposition properties of ROPs and combines this with an algorithm that extracts greatest common divisors of the coefficients in an ROP. We then describe how to modify it to work over any field with polynomially many field operations; Theorem 2. This modification was suggested to us by Amir Shpilka at Dagstuhl seminar 14121.

Central to the PIT algorithm in [SV08] is a "hardness of representation" lemma showing that the polynomial $\mathcal{M}_n = x_1 x_2 \cdots x_n$, consisting of just a single monomial, cannot be represented as a sum of less than $n/3$ ROPs of a particular form (0-justified). More recently, a similar hardness of representation result appeared in [Kay12]: if $\mathcal{M}_n$ is represented as a sum of powers of low-degree (at most d) polynomials, then the number of summands is $\exp(\Omega(n/d))$. As is implicit in [Kay12], such a hardness of representation statement can be used to give a PIT algorithm. We analyze this connection explicitly, and show that the results in [Kay12] lead to a deterministic sub-exponential time algorithm for black-box PIT for sums of powers of polynomials with appropriate size and degree (Section 4, Theorem 3).

A minor drawback of both these statements is that they consider a model that cannot even individually compute all monomials. One would expect any reasonable model of representing polynomials to be able to compute $\mathcal{M}_n$. In Section 5, we consider the restriction of read-once formulas to *constant-free* formulas that are only allowed leaf labels ax, where x is a variable and a is a field element. This model can compute any single monomial. We show (Theorem 4) that the elementary symmetric polynomial $\mathsf{Sym}_{n,d}$ of degree d cannot be written as a sum of powers of such formulas unless the number of summands is $\Omega(\log(n/d))$. This appears weak compared to the $n/3$ bound from [SV08], but this is to be expected since unlike in [SV08] where the ROPs could only be added, we allow sums of powers. We also consider 0-justified read-once formulas of the form $\sum \cdot \prod \cdot \sum$, and obtain a similar hardness-of-representation result for the polynomial $\mathcal{M}_n$ against sums of powers of polynomials computed by such formulas, showing that $n^{\frac{1}{2}-\epsilon}$ summands are needed (Theorem 5). Again, this appears weak compared to the $\exp(\Omega(n/d))$ bound from [Kay12], but unlike in [Kay12] where the degree of the inner functions is a parameter, our inner ROPs could have arbitrarily high degree.

Finally we return to the question of characterizing which polynomials are ROPs. This question has been recently answered by Volkovich [Vol14]. We focus on the fields of rationals or reals, and consider *monotone* polynomials (no negative coefficients). For specific multilinear monotone polynomials, several tight lower bounds on the sizes of monotone arithmetic circuits computing them are known, from [Val79] upto [RY11]. In the context of restricted-

read circuits, our study explores the question of not size but expressibility: when is a monotone polynomial computable at all by a constant-free ROF? (We show that any such ROF will also have to be monotone.) Using the characterization of Boolean read once formulas from [KLN+93], we answer this question completely when the coefficients are all 0 or 1. This result, Theorem 6, is described in Section 6.

2 Preliminaries

An arithmetic formula on n variables $X = \{x_1, \ldots, x_n\}$ is a rooted binary tree with leaves labeled from $\mathbb{F} \cup X$ and internal nodes labeled by $\circ \in \{+, \times\}$. Each node computes a polynomial in the obvious way, and the formula computes the polynomial computed at the root gate. An arithmetic formula is said to be read-once (ROF) if each $x \in X$ appears at most once at a leaf. Polynomials computed by ROFs are called read-once polynomials ROPs.

It is more convenient for us to allow leaves to be labeled by forms $ax + b$ for some $x \in X$ and some $a, b \in \mathbb{F}$. This does not change the class of polynomials computed, even when restricted to ROFs. Henceforth we assume that ROFs are of this form.

The alternation depth of the formula is the maximum number of maximal blocks of $+$ and $\times$ gates on any root-to-leaf path in the formula.

We say that an ROF is constant-free (denoted CF-ROF) if the labels at the leaves are of the form ax for $x \in X$ and $a \in \mathbb{F} \setminus \{0\}$. We call polynomials computed by such formulas constant-free ROPs, denoted CF-ROP.

For a polynomial $f \in \mathbb{F}[x_1, x_2, \cdots, x_n]$, a set $S \subseteq [n]$ and a possibly partial assignment a that assigns values to all x_j for $j \in S$, let $f|_{S \to a_S}$ denote the polynomial on variables $\{x_i : i \notin S\}$ obtained from f by setting $x_j = a_j$ for $j \in S$. For a set of assignments $\mathcal{A} \subseteq \mathbb{F}^n$, we say $f|_{\mathcal{A}} \equiv 0$ if and only if f vanishes on all assignments in $\mathcal{A}$. For any $i \in [n]$, we say that the polynomial f depends on the variable x_i non-trivially if $\frac{\partial f}{\partial x}$ is not identically zero. Using notation from [SV08], for a polynomial f, $\mathsf{var}(f)$ denotes the set of variables that f depends on non-trivially. We say that f is 0-justified if for all $S \subseteq \mathsf{var}(f)$, $\mathsf{var}(f|_{S \to 0_S}) = \mathsf{var}(f) \setminus S$. For multilinear $f \not\equiv z$ (and hence for ROPs), this is equivalent to the condition that for each variable $x \in \mathsf{var}(f)$, the degree-1 monomial x has a non-zero coefficient in f. (Note that the identically-zero polynomial is vacuously 0-justified.)

We re-state an important result by Noga Alon here:

Proposition 1 (Combinatorial Nullstellensatz, [Alo99]). *Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial where for every $i \in [n]$, the degree of x_i is bounded by t. Let $R \subseteq \mathbb{F}$ have size at least $t + 1$, and $S = R^n$. Then $P \equiv 0 \Leftrightarrow P|_S \equiv 0$.*

3 Identity testing for $\sum^{(2)} \cdot \prod \cdot$ROPs

In this section we show that PIT can be solved efficiently for formulas presented in the form $f_1 f_2 \ldots f_m + g_1 g_2 \ldots g_s$, where each of the f_i, g_j is an ROF.

The idea is to decompose each f_i, g_j into their irreducible factors, obtaining ROFs for these factors, and and then match them up using the PIT algorithm from [SV08]. However, the

decomposition is unique only up to scalar multiples, and this presents some difficulties. We first describe how to circumvent this difficulty when the field is rationals. Then we describe how to do the same over arbitrary fields.

Theorem 1. *Given Read-Once Formulas computing each of the polynomials $f_1, f_2, \cdots, f_r$, $g_1, g_2, \ldots, g_s \in \mathbb{Q}[x_1, \ldots, x_n]$, checking if $f_1 \cdot f_2 \cdots f_r \equiv g_1 \cdot g_2 \cdots g_s$ can be done in deterministic polynomial time.*

A crucial ingredient in our proof is the following structural decomposition property from [RS11,RS13] and its constructive version; this is a direct consequence of the properties of ROPs given in [SV08].

Lemma 1 ([RS13]). *Let f be an ROP. Then exactly one of the following holds:*

1. *$k \geq 1$, there exist ROPs $f_1, \ldots, f_k$, with $\mathsf{var}(f_i) \cap \mathsf{var}(f_j) = \emptyset$ for all distinct $i, j \in [k]$, such that $f = a + f_1 + \cdots + f_k$, for some $a \in \mathbb{F}$, and each f_i is either uni-variate or decomposes into variable-disjoint factors.*
2. *$k \geq 2$, there exist ROPs $f_1, \ldots, f_k$, with $\mathsf{var}(f_i) \cap \mathsf{var}(f_j) = \emptyset$ for all distinct $i, j \in [k]$, such that $f = f_1 \times f_2 \times \cdots \times f_k$ for some $a \in \mathbb{F} \setminus \{0\}$, and none of the f_is can be factorised into variable-disjoint factors.*

Furthermore, ROFs computing such f_is can be constructed from an ROF computing f in polynomial time.

Given an ROF over $\mathbb{Q}$, we can clear all denominators to get an ROF over $\mathbb{Z}$, without changing the status of the PIT question. So we now assume that all the numbers a, b appearing in the ROF (recall, leaf labels are of the form $ax+b$) are integers. For a polynomial $p(X)$, let $\mathsf{content}(p(X))$ denote the greatest common divisor (gcd) of the non-zero coefficients of p. For an ROF f, we use $\mathsf{content}(f)$ to mean the content of the polynomial computed by f. The next crucial ingredient in our proof is that for an ROF f, we can efficiently compute $\mathsf{content}(f)$.

Lemma 2. *There is a polynomial-time algorithm that, given an ROF f in $\mathbb{Z}[X]$, computes $\mathsf{content}(f)$ and constructs an ROF f' in $\mathbb{Q}[X]$ such that $f = \mathsf{content}(f) \cdot f'$.*

Proof. It suffices to show how to compute $\mathsf{content}(f)$; then the ROF f' is just $\frac{1}{\mathsf{content}(f)} \times f$. We prove this by induction on the structure of f.

For a polynomial $p \in \mathbb{Z}[X]$, let $\hat{p} = p - p(0)$, where $p(0) = p(0, \ldots, 0)$, and let $\hat{p}'$ be the polynomial such that $\hat{p} = \mathsf{content}(\hat{p})\hat{p}'$. We proceed bottom-up, computing $\mathsf{content}(p)$ and $\mathsf{content}(\hat{p})$ for the polynomials p computed at each node of the ROF f.

If f is a single leaf node, then computing $\mathsf{content}(f)$ and $\mathsf{content}(\hat{f})$ is trivial. Otherwise, say $f = g \circ h$. Since f is an ROF, $\mathsf{var}(g) \cap \mathsf{var}(h) = \emptyset$.

Case $f = g + h$: Then $\hat{f} = \hat{g} + \hat{h}$, and $f(0) = g(0) + h(0)$. So

$$\mathsf{content}(f) := \gcd(\mathsf{content}(\hat{g}), \mathsf{content}(\hat{h}), g(0) + h(0)),$$
$$\mathsf{content}(\hat{f}) := \gcd(\mathsf{content}(\hat{g}), \mathsf{content}(\hat{h})).$$

Case $f = g \times h$: Then $\hat{f} = \hat{g}\hat{h} + h(0)\hat{g} + g(0)\hat{h}$, and $f(0) = g(0)h(0)$.

Claim (Folklore). For any two variable-disjoint polynomials $p, q \in \mathbb{Z}[X]$, $\mathsf{content}(pq) = \mathsf{content}(p) \cdot \mathsf{content}(q)$.

Proof. This is known as Gauss's lemma. We include a proof for completeness. Let $p = \mathsf{content}(p)(a_1M_1 + a_2M_2 + \cdots + a_kM_k)$ and $q = \mathsf{content}(q)(b_1N_1 + b_2N_2 + \cdots + b_\ell N_\ell)$, where M_i, N_j are monomials. We have $\gcd(a_1, \ldots, a_k) = \gcd(b_1, \ldots, b_\ell) = 1$ by the definition of content. Since p and q are variable-disjoint, every monomial of the form $\mathsf{content}(p) \cdot \mathsf{content}(q) \cdot (a_ib_jM_iN_j)$ appears in the polynomial $p \times q$, and there are no other monomials, and hence $\mathsf{content}(p) \cdot \mathsf{content}(q) | \mathsf{content}(p \times q)$. For the converse, it suffices to show that $\gcd(S) = 1$, where $S = \{a_ib_j \mid i \in [k], j \in [\ell]\}$. Suppose not. Let c be the largest prime that divides all numbers in S. Then, $\forall i \in [k]$,

$$\begin{aligned} & c|a_ib_1 \text{ and } c|a_ib_2 \text{ and } \ldots \text{ and } c|a_ib_k. \\ \text{Hence } & c|a_i \text{ or } (c|b_1, c|b_2, \cdots, c|b_\ell)\,. \\ \text{Hence } & c|a_i \text{ or } c = 1, \text{ since } \gcd(b_1, \ldots, b_\ell) = 1. \end{aligned}$$

Thus we conclude that c divides $\gcd(a_1, \ldots, a_k) = 1$, a contradiction. □

Using this claim, we see that

$$\begin{aligned} &\mathsf{content}(f) := \mathsf{content}(g) \times \mathsf{content}(h), \\ &\mathsf{content}(\hat{f}) := \gcd(\mathsf{content}(\hat{g})\mathsf{content}(\hat{h}), h(0)\mathsf{content}(\hat{g}), g(0)\mathsf{content}(\hat{h})). \end{aligned}$$

□

Now we have all the ingredients for proving Theorem 1.

Proof (of Theorem 1). Let $f = f_1 \cdot f_2 \cdots f_r$ and $g = g_1 \cdot g_2 \cdots g_s$. As discussed above, without loss of generality, each f_i, g_i is in $\mathbb{Z}[X]$. Using Lemma 1 and 2, we can compute the irreducible variable-disjoint factors of each f_i and each g_i, and also pull out the content for each factor. That is, we express each f_i as $\alpha_i f_{i,1} \cdots f_{i,k_i}$, and each g_i as $\beta_i g_{i,1} \cdots g_{i,\ell_i}$ where the $f_{i,j}$s, $g_{i,j}$s are irreducible and have $\mathsf{content}$ 1. Due to the $\mathsf{content} = 1$ condition, this decomposition is unique. We obtain ROFs in $\mathbb{Q}[X]$ for each of the $f_{i,j}$s and $g_{i,j}$s. (The ROFs are in $\mathbb{Q}[X]$, but the polynomials $f_{i,j}, g_{i,j}$ they compute are in $\mathbb{Z}[X]$.)

Note that if $\sum_i k_i \neq \sum_j \ell_j$, then there cannot be a component-wise matching between the factors of f and g, and hence we conclude $f \not\equiv g$. Otherwise, $\sum_i k_i = \sum_j \ell_j$. We now form multisets of the factors of f and of g, and we knock off equivalent factors one by one. (See Algorithm 1.) Detecting equivalent factors (the condition in Step 6) requires an identity test $p \equiv q?$, or $p - q \equiv 0?$, for ROFs in $\mathbb{Q}[X]$. Since we have explicit ROFs computing p and q, this can be done using [SV08].

Consider the bit complexity of the above procedure. For moving over to $\mathbb{Z}[X]$, polynomial time suffices. Before factorising the ROPs computed by the ROFs, we can convert each ROF to an unbounded-fanin ROF where nodes strictly alternate between $+$ and $\times$, ensure that no node has more than one input that is a scalar, and prune out all zeroes. Applying Lemma 1

Algorithm 1 Test if $\prod_{i=1}^{r} \alpha_i \prod_{j=1}^{k_i} f_{i,j} \equiv \prod_{i=1}^{s} \beta_i \prod_{j=1}^{\ell_i} g_{i,j}$

$S \leftarrow \{f_{1,1}, \cdots, f_{1,k_1}, f_{2,1}, \cdots, f_{2,k_2}, \ldots, f_{r,1}, \cdots, f_{r,k_r}\}$
$T \leftarrow \{g_{1,1}, \cdots, g_{1,\ell_1}, g_{2,1}, \cdots, g_{2,\ell_2}, \ldots, g_{s,1}, \cdots, g_{s,\ell_s}\}$
(Both S and T are multisets; repeated factors are retained with multiplicity.)
for $p \in S$ **do**
    for $q \in T$ **do**
        if $p \equiv q$ **then**
            if S and T have unequal number of copies of p and q **then**
                Return No
            else
                $S \leftarrow S \setminus \{p\}$. (Remove all copies).
                $T \leftarrow T \setminus \{q\}$. (Remove all copies).
            end if
        end if
    end for
end for
if $(\alpha_1\alpha_2\cdots\alpha_r = \beta_1\beta_2\cdots\beta_s) \wedge (S = T = \emptyset)$ **then**
    Return Yes
else
    Return No
end if

merely involves graph-theoretic navigation on the ROFs computing each f_i, g_j, moving down from the root until a + gate is encountered. Applying Lemma 2 involves applying a gcd algorithm polynomially many times on numbers obtained from the coefficients in the formula. Since we have formulas, not circuits, all computed numbers have polynomial bit complexity. To detect equivalent factors, we use the PIT test from [SV08] on $p-q$, where p, q are computed by explicitly given ROFs. The test amounts to setting $W = \{0, 1\}$, finding a common 0-1 "justifying assignment" $\hat{a}$ by identifying the variables that p, or q (or both) depend on, and evaluating $p - q$ at $O(n^7)$ assignments. (For each $\hat{w} \in W^n$ with Hamming-weight at most 6, evaluate $p - q$ at $\hat{w} - \hat{a}$.) Clearly, the test has polynomial bit-complexity as well. □

Extension to arbitrary fields

Recently, Amir Shpilka pointed out to us (at Dagstuhl Seminar 14121) that the proof of Theorem 1 can be modified to work for polynomials over any field $\mathbb{F}$. Now time refers to the number of field operations. We sketch the proof specifically for the part that is different from proof of Theorem 1:

Theorem 2 (Amir Shpilka). *Given Read-Once Formulas computing each of the polynomials $f_1, f_2, \cdots, f_r, g_1, g_2, \ldots, g_s \in \mathbb{F}[x_1, \ldots, x_n]$, checking if $f_1 \cdot f_2 \cdots f_r \equiv g_1 \cdot g_2 \cdots g_s$ can be done in deterministic polynomial time.*

Proof. We use Lemma 1 to obtain a product of irreducible factors for each f_i and g_i. That is, we express each f_i as $f_{i,1} \cdots f_{i,k_i}$, and each g_i as $g_{i,1} \cdots g_{i,\ell_i}$ where the $f_{i,j}$s and $g_{i,j}$s are irreducible. Since these polynomials are over an arbitrary field $\mathbb{F}$, the notion of content does not exist. The factorization is now unique only up to scalar multiples. We show how to handle this.

Similar to proof of Theorem 1, we want to find a match on the right side for each irreducible component from the left side. For ease of notation, let $p = f_{i,j}$ and $q = g_{u,v}$. We want to check if q is a match for p. i.e., is $p = cq$ for some $c \in \mathbb{F}$? Since p and q are both ROPs, the individual degree of each variable is at most 1. We know that $p \not\equiv 0$ and $q \not\equiv 0$. By Proposition 1, it must be the case that there is an $\boldsymbol{a} \in \{0,1\}^n$ such that $p(a) \neq 0$. We find a using Algorithm 2. Step 3 of the algorithm can be achieved using the algorithm from [SV08].

Algorithm 2 Find $\boldsymbol{a} \in \{0,1\}^n$ such that $p(a) \neq 0$

```
1: for k = 1 to n do
2:     a[k] ← 0
3:     if p|_{x_k=0} ≡ 0 then
4:         a[k] ← 1
5:         p ← p|_{x_k=1}
6:     else
7:         p ← p|_{x_k=0}
8:     end if
9: end for
```

Once we have $\boldsymbol{a}$, we check that $q(\boldsymbol{a})$ is non-zero (if it is zero, then q and p cannot be matched). Now q is a scalar multiple of p if and only if $p \equiv cq$, where $c = p(\boldsymbol{a})/q(\boldsymbol{a})$. We then check if $p - cq \equiv 0$, again using the algorithm from [SV08]. If yes, then we knock off p and q from their respective sides and continue this process of finding a component wise matching while retaining c as a scalar multiple on the right side. If $p - cq \not\equiv 0$, then q is not a match for p and we continue trying to find a match for p exactly like in proof of Theorem 1. If no match is found, then the inputs are not identically equal. If all factors from both sides have been knocked off, then we check if $\Pi_i c_i = 1$. If yes, we conclude that the polynomial $(f_1 \cdot f_2 \cdots f_r) - (g_1 \cdot g_2 \cdots g_s)$ is identically zero. Else, the polynomial is not identically zero. □

4 PIT for sums of powers of low degree polynomials

In this section, we give a blackbox identity testing algorithm for the class of multilinear polynomials that can be expressed as a sum of powers of low-degree polynomials.

We say that a polynomial f has a sum-powers representation of degree d and size s if there are polynomials f_i each of degree at most d, and a set of positive integers e_i, such that $f = f_1^{e_1} + \ldots + f_s^{e_s}$. In [Kay12], it is shown that computing the full multilinear monomial $\mathcal{M}_n = x_1 x_2 \cdots x_n$ using sums of powers of low-degree polynomials requires exponentially many summands:

Proposition 2. *[Kay12] There is a constant c such that for the polynomial $x_1 x_2 \cdots x_n$, any sum-powers representation of degree d requires size $s \geq 2^{\frac{cn}{d}}$.*

Recall that an ROP f is said to be 0-justified if for every $S \subseteq \{x_1, \ldots, x_n\}$, $\mathsf{var}(f|_{S \to 0}) = \mathsf{var}(f) \setminus S$. Shpilka and Volkovich [SV08] proved that sum of less than $n/3$ 0-justified-ROPs

cannot equal $\mathcal{M}_n$, and used it to obtain a black-box PIT algorithm for bounded sums of ROPs. Using these ideas along with Proposition 2, we note that such a hardness of representation for sums of powers of low-degree polynomials, where the final sum is multilinear, gives sub-exponential time algorithms for black-box PIT for this class.

Let $R = \{0, 1\} \subseteq \mathbb{F}$. For any $k > 0$, define

$$W_k^n(R) \triangleq \{\boldsymbol{a} \in R^n | \ \boldsymbol{a} \text{ has at most } k \text{ non-zero coordinates}\}.$$

In Theorem 7.4 of [SV10], it is shown that for a certain kind of formula F (k-sum of degree-d 0-justified preprocessed ROP), and for any $R \subseteq \mathbb{F}$ containing 0 and of size at least $d+1$, $F \equiv 0$ if and only if $F\ |_{W_{3k}^n(R)} \equiv 0$. The proof uses Proposition 1, see also Lemma 2.13 in [SV10].

Along similar lines, using Propositions 2,1, we show that

Lemma 3. *Let $C(n, s, d)$ be the class of all n-variate multilinear polynomials that have a sum-powers representation of degree d and size s. Let c be the constant from Proposition 2. For $f \in C(n, s, d)$, $R = \{0, 1\}$, and $k = (d \log s)/c$, $f|_{W_k^n(R)} \equiv 0 \iff f \equiv 0$.*

Proof. The $\Leftarrow$ direction in the claim is trivial. To prove the $\Rightarrow$ direction, we proceed by induction on n.
Base case: $n \leq k$. Then $W_k^n(R) = R^n$. Using Proposition 1 (since f is multilinear, R is large enough), we conclude that $f \equiv 0$.
Induction Step: $n > k$. Suppose $f \not\equiv 0$. Consider any $i \in [n]$, and let $f_i' = f|_{x_i=0}$. Then $f_i' \in C(n-1, s, d)$. Since $f|_{W_k^n(R)} \equiv 0$, we have $f_i'|_{W_k^{n-1}(R)} \equiv 0$. So by the induction hypothesis, $f_i' \equiv 0$. Hence $x_i | f$. Since this holds for every $i \in [n]$, the monomial $x_1 \cdots x_n$ must divide f. Since f is multilinear, it must be that $f = x_1 \cdots x_n$. But $n > k = (d \log s)/c$, so $s < 2^{cn/d}$. This contradicts Proposition 2. Hence we conclude $f \equiv 0$. □

This gives the required black-box PIT algorithm: just query the black-box for f at every point in W_k^n. For our choice of k in the above lemma, $|W_k^n(\{0, 1\})| \in n^{O(k)} \subseteq 2^{O(d \log s \log n)}$, and this bounds the running time. Thus

Theorem 3. *Let $C(n, s, d)$ be the class of all n-variate multilinear polynomials that have a sum-powers representation of degree d and size s. There is a deterministic black-box PIT algorithm for $C(n, s, d)$ running in time $2^{O(d \log n \log s)}$.*

Remark 1. Though f is multilinear in Lemma 3 (and hence Theorem 3), the polynomials f_i in the sum-powers representation of f need not be multilinear.

5 Hardness of representation for sum of powers of CF-ROPs

The hardness of representation result from [Kay12], stated in Proposition 2, and its precursor from [SV08],[SV10], are both for $\mathcal{M}_n$, the former using low-degree polynomials and the latter using a kind of ROPs called 0-justified-ROPs. Note that ROPs, even when 0-justified, can have

high degree, so these results are incomparable. In this section, we give two different hardness results.

Our first hardness result is for elementary symmetric polynomials $\mathsf{Sym}_{n,d}$, not just for $d = n$. It works against another subclass of ROPs, CF-ROF; as is the case in [SV08,SV10], this class too can have high-degree polynomials. Recall that this class consists of polynomials computed by read-once formulas that have + and × gates, and labels ax at leaves ($a \neq 0$). Hence for any f in this class, $f(0) = 0$. (However, it is not the case that every ROP p with $p(0) = 0$ is computed by a CF-ROF. Consider for instance, $p(\hat{x}) = \prod_{i=1}^{n}(x_i + 1) - 1$. Even to write it as a sum of CF-ROFs would need many summands.) We show that powers of such polynomials cannot add up to elementary symmetric polynomials of arbitrary degree $d \leq n$ unless there are many such summands. First, we establish a useful property of this class.

Lemma 4. *For every* CF-ROP $f \in \mathbb{F}[x_1, \ldots, x_n]$*, there is a set* $S \subseteq [n]$ *with* $|S| \leq |\mathsf{var}(f)|/2$ *such that* $\deg(f|_{S\to 0}) \leq 1$*.*

Proof. Consider a CF-ROF F computing f. If F has a single node, then f is already linear, so $S = \emptyset$. Otherwise, $F = G_1 \circ G_2$, where G_1, G_2 are variable-disjoint CF-ROFs computing CF-ROPs g_1, g_2, respectively.
Case 1: $\circ = \times$**.** Without loss of generality, assume $|\mathsf{var}(g_1)| \leq |\mathsf{var}(f)|/2$. For $S = \{i : x_i \in \mathsf{var}(g_1)\}$, $g_1|_{S\to 0} \equiv f|_{S\to 0} \equiv 0$.
Case 2: $\circ = +$**.** Inductively, we can find sets S_i of at most half the variables of each g_i, such that $g_i|_{S_i\to 0}$ has degree at most 1. Define $S = S_1 \cup S_2$. Since G_1, G_2 are variable-disjoint, $|S| \leq |\mathsf{var}(f)|/2$, and $f|_{S\to 0}$ has degree at most 1. □

We use this to get our hardness-of-representation result for CF-ROPs, irrespective of degree.

Theorem 4. *Fix any* $d \in [n]$*. Suppose there are CF-ROPs* $f_1, \ldots, f_s$*, and positive integers* $e_1, \ldots, e_s$ *such that*

$$\sum_{i=1}^{s} f_i^{e_i} = \mathsf{Sym}_{n,d}.$$

Then $s \geq \min\{\log \frac{n}{d}, 2^{\Omega(d)}\}$.

Proof. Let $f = \mathsf{Sym}_{n,d}$.

We repeatedly apply Lemma 4 to restrictions of the f_i's to obtain a formula of degree at most 1. Let $Q_0 = T_0 = \emptyset$, and let Q_{i+1} be the set obtained by applying the Lemma to $f_{i+1}|_{T_i\to 0}$, where each $T_i = Q_1 \cup \ldots \cup Q_i$. Define $Q = T_s$. Since at least half the variables survive in f at each stage, we see that $r \triangleq |\mathsf{var}(f|_{Q\to 0})| \geq |\mathsf{var}(f)|/2^s = n/2^s$.

- If $r < d$, then $n/2^s \leq r < d$. So $s > \log(\frac{n}{d})$.
- If $r \geq d$, then $f|_{Q\to 0} = \mathsf{Sym}_{r,d} \not\equiv 0$. Add any $r - d$ surviving variables to the set Q to obtain the expression $\mathsf{Sym}_{d,d} = f|_{Q\to 0} = \sum_{i=1}^{s}(f_i|_{Q\to 0})^{e_i}$ where each f_i is either linear or identically 0. Let s' be the number of non-zero polynomials $f_i|_{S\to 0}$. By Proposition 2, $s' \in 2^{\Omega(d)}$, and $s \geq s'$.

Thus if $s \leq \log \frac{n}{d}$, then $s \in 2^{\Omega(d)}$. □

What this tells us is that there is a threshold $r \sim \log\log n$ such that any sum-powers representation of $\mathsf{Sym}_{n,d}$ using CF-ROPs needs size $2^{\Omega(d)}$ for $d \leq r$, and size $\geq \log \frac{n}{d}$ for $d \geq r$.

Our second hardness result is for $\mathcal{M}_n$, but works against a different class of ROFs. These ROFs may not be constant-free, but they have bounded alternation-depth, and are also 0-justified. Again, first we establish a useful property of the class.

Lemma 5. *Let $\mathbb{F}$ be any field of size at least 3. Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be computed by an* ROF *of the form $\sum \cdot \prod \cdot \sum$. For any degree bound $1 \leq d \leq n$, there is an $S \subseteq [n]$ of size at most $|\mathsf{var}(f)|/d$, and an assignment of values A_S to the variables x_i for $i \in S$, such that $\deg(f|_{S \to A}) \leq d$. Moreover, if f is* 0-justified*, then we can find an A_S with all non-zero values.*

Proof. Let f be computed by the $\sum \cdot \prod \cdot \sum$ ROF F, where no gate computes the 0 polynomial.

Since the top gate in F is a $+$, we can write $F = \sum_{i=1}^{r} f_i$, where each summand f_i is of the form $\prod_{j=1}^{t_i} \ell_{i,j}$ and the factors $\ell_{i,j}$'s are linear forms on disjoint variable sets. We find a partial assignment that kills all summands of degree more than d. For each such summand f_i, identify the factor with fewest variables, and assign values to the variables in it to make it 0. We assign values to at most $|\mathsf{var}(f_i)|/d$ variables, so overall no more than $|\mathsf{var}(f)|/d$ variables are set.

Further, if f is 0-justified and read-once, then each f_i is also a 0-justified ROF. Hence no factor of f_i vanishes at 0; each factor $\ell_{i,j}$ is of the form $\sum_{k=1}^{p} a_{i,j,k} x_{i,j,k} - c_{i,j}$ where $c_{i,j} \neq 0$. We can kill such a factor with an assignment avoiding 0s. For instance, over rationals or reals we can set $x_{i,j,k} = c_{i,j}/pa_{i,j,k}$. Over possibly smaller fields, the following claim suffices.

Claim. If $\mathbb{F}$ is a field of size at least 3, and $a_0, a_1, \ldots, a_k$ are non-zero elements of $\mathbb{F}$, then there is an assignment $b_1, \ldots, b_k \in (\mathbb{F} \setminus \{0\})^k$ such that $a_0 + a_1 b_1 + \ldots + a_k b_k = 0$.

Proof. Choose values b_i sequentially. Let $c_0 = a_0$, and $c_i = c_{i-1} + a_i b_i$. Ensure that for all $1 \leq i < k$, the c_i values are non-zero. This is true initially. Setting b_i to any value other than 0 or $a_i^{-1}(-c_{i-1})$ ensures that $c_i \neq 0$. Now set $b_k = -(c_{k-1} \cdot a_k^{-1})$. This gives $c_k = 0$ as required. □

□

Using this, we get a hardness of representation result for 0-justified $\sum \cdot \prod \cdot \sum$ ROFs.

Theorem 5. *Let $\mathbb{F}$ be any infinite field. For every $\epsilon \in (0, \frac{1}{2})$, there exists an $n_\epsilon \geq 0$ such that for every $n \geq n_\epsilon$, if there are* 0-justified *$\sum \prod \sum$ ROPs $f_1, \ldots, f_s$, and non-negative integers $e_1, \ldots, e_s$ such that*

$$\sum_{i=1}^{s} f_i^{e_i} = x_1 \cdots x_n$$

then $s \geq n^{\frac{1}{2}-\epsilon}$.

Proof. Let d be a parameter to be chosen later. We identify a subset of variables S and an assignment A avoiding zeroes to variables of S, such that under this partial assignment, all the f_i's are reduced to degree at most d. We show that for any $d \in [n]$, this is possible with $|S| = t \leq \frac{s^2 n}{d}$. This gives a sum-powers representation of degree d and size s for $\prod_{x_i \notin S} x_i = M_{n-t}$. Invoking Kayal's result from Proposition 2, we see that $s \geq 2^{c(n-t)/d}$, and hence $\log s + \frac{cns^2}{d^2} \geq \frac{cn}{d}$. Choose $d = 4n^{1-2\epsilon}$, then there exists an $n_\epsilon > 0$ that depends only on ϵ and c such that $s \geq n^{\frac{1}{2}-\epsilon}$ for $n \geq n_\epsilon$.

The construction of S proceeds in stages. At the kth stage, polynomials $f_1, \ldots, f_{i-1}$ have already been reduced to low-degree polynomials, and we consider f_i. We want to use Lemma 5 at each stage. This requires that each polynomial f_i, **after all the substitutions from the previous stages**, is still a 0-justified $\sum \cdot \prod \cdot \sum$ ROF. The $\sum \cdot \prod \cdot \sum$ form is obvious; it is only maintaining 0-justified that is a bit tricky. We describe the construction for stage 1; the other stages are similar.

Applying Lemma 5 to f_1 with d as the parameter, we obtain a set R_1 of variables with $|R_1| \leq n/d$ and an assignment A_{R_1} avoiding 0, such that $\deg(f_1|_{R_1 \to A_{R_1}}) \leq d$. It may be the case that for some $i > 1$, the polynomial $f_i|_{R_1 \to A_{R_1}}$ is no longer 0-justified. We fix this by augmenting R_1 as follows.

As discussed in the proof of Lemma 5, each f_i has the form $\sum \prod \ell_{j,k}$ where each $\ell_{j,k}$ is a linear form. If $f_i|_{R_1 \to A_{R_1}}$ is not 0-justified, then some of the linear forms in it are homogeneous linear (no constant term). We identify such linear forms in each f_i, $i \geq 2$. Call this set L_1. That is,

$$L_1 = \left\{ \ell \mid \begin{array}{c} \ell \text{ is a linear form at level-2 of some} \\ f_i;\ \ell|_{R_1 \to A_{R_1}} \text{ is homogeneous lin-} \\ \text{ear but not identically 0.} \end{array} \right\}$$

Since each f_i is a ROF, it contributes at most $|R_1|$ linear forms to L_1. Hence $|L_1| \leq (s-1)|R_1|$. Now pick a minimal set T_1 of variables from $X \setminus R_1$ that intersects each of the linear forms in L_1. By minimality, $|T_1| \leq |L_1| \leq (s-1)|R_1|$. We want to assign non-zero values A_{T_1} to variables in T_1 in such a way that for all $i \geq 2$, the $f_i|_{R_1 \to A_{R_1}; T_1 \to A_{T_1}}$ are 0-justified. We must ensure that the linear forms in L_1 become homogeneous (or vanish altogether), and we must also ensure that previously non-homogeneous forms do not become homogeneous. To achieve this, consider

$$L_2 = \left\{ \ell \mid \begin{array}{c} \ell \text{ is a linear form at level-2 of some } f_i; \\ \ell|_{R_1 \to A_{R_1}} \not\equiv 0;\ \ell|_{R_1 \to A_{R_1}} \text{ contains a variable} \\ \text{from } T_1. \end{array} \right\}$$

Clearly, $L_1 \subseteq L_2$. It suffices to find an assignment A_{T_1} to variables in T_1, avoiding zeroes, such that under the partial assignment $R_1 \to A_{R_1}; T_1 \to A_{T_1}$, every linear form in L_2 becomes either zero or non-homogeneous. That is,

$$\forall \ell \in L_2,\ \text{either } \ell|_{R_1 \to A_{R_1}; T_1 \to A_{T_1}} \equiv 0 \text{ or } \ell|_{R_1 \to A_{R_1}; T_1 \to A_{T_1}}(0) \neq 0 \tag{1}$$

This can be done in a sequential greedy fashion as follows. Choose any variable $x \in T_1$. There is a finite number of values b for which setting $x = b$ can potentially violate (1). As

$\mathbb{F}$ is infinite, there is a value a such that assigning $x = a$ is safe and will not immediately violate (1). Set $x = a$ in A_{T_1}. Continuing the process with the remaining variables in T_1, we get the required assignment A_{T_1} that satisfies (1).

Now we set $S_1 = R_1 \cup T_1$, and $A_1 = A_{R_1} \cup A_{T_1}$. We have ensured the following:

1. $\deg(f_1|_{S_1 \to A_1}) \leq d$; and
2. for $i \geq 2$, $f_i|_{S_1 \to A_1}$ is $\mathsf{0\text{-}justified}$.

Furthermore, $|S_1| = |R_1| + |T_1| \leq |R_1|(1 + (s-1)) \leq sn/d$.

Other stages are identical, working on the polynomials restricted by the already-chosen assignments. Finally, $S = S_1 \cup \ldots \cup S_s$, and so $|S| \leq s^2 n/d$, as required. □

Our result may well be far from optimal. For instance, an old idea of Ben-Or (see [SV08] for more details) yields depth-3 ROFs for $\mathsf{Sym}_{n,d}$ for any $d \leq n$. We repeat the argument here for completeness, and note that the summands are in fact $\mathsf{0\text{-}justified}$. However, Theorem 5 only says that we need at least $n^{1/2-\epsilon}$ such summands.

Proposition 3. *Over any infinite field $\mathbb{F}$, for any $d \leq n$, $\mathsf{Sym}_{n,d}$ can be represented as a sum of $n+1$ $\Pi\Sigma$ $\mathsf{0\text{-}justified\text{-}ROP}$s.*

Proof. View the polynomial $\prod_{i=1}^{n}(x_i + t)$, where t is an indeterminate, as a univariate polynomial $p(t)$ with coefficients from $\mathbb{F}[X]$. Then $\mathsf{Sym}_{n,d}(X)$ is the coefficient of t^{n-d} in $p(t)$, and all coefficients can be computed by interpolation through $n+1$ distinct points. That is, for any $n+1$ distinct non-zero elements $\alpha_0, \ldots, \alpha_n$ in $\mathbb{F}$, $\mathsf{Sym}_{n,d}(X)$ is an $\mathbb{F}$-linear combination of $p(\alpha_0), \ldots, p(\alpha_n)$. Each $p(\alpha_j) = \prod_{i=1}^{n}(x_i + \alpha_j)$, $0 \leq j \leq n$, is a $\Pi\Sigma$ ROP in the variables $X = \{x_1, \ldots, x_n\}$. Choosing α_j values avoiding zero ensures that each $p(\alpha_j)$ is $\mathsf{0\text{-}justified}$.

6 Characterizing monotone CF-ROPs

Every ROP is multilinear. But the converse is not true. So we can ask:

Question 1. When is a multilinear polynomial $p(x_1, \ldots, x_n)$ an ROP?

Volkovich gave an answer to this:

Proposition 4 ([Vol14]). *For sufficiently large fields, $p(\bar{x})$ is an ROP if and only if for every assignment $\bar{a}$ in the field, p is $\bar{a}$-three-locally read-once.*

Without getting into the details of what the terms in this characterization mean, let us examine a variant of this question. We consider *monotone* polynomials and monotone formulas / circuits. To keep things simple, we fix the field $\mathbb{F}$ to be reals or rationals, so that we can meaningfully talk about negative and positive values. Then a polynomial with real coefficients is *monotone* if it has no negative coefficients, and a formula (or a circuit) is monotone if it has no negative constants. (Note that a more general notion of monotonicity, applicable to more fields, is defined in [Val79]. Over rationals and reals, it specialises to these definitions, which are also quite standard. See for instance [RY11].)

A monotone read-once formula computes a polynomial that is multilinear, an ROP, and monotone. But the converse is not true. For instance, $p(x, y) = x + y + xy$ is a multilinear monotone polynomial computed by the ROF $(x+1)(y+1)-1$. But we can show that it has no monotone ROF. Suppose it does; let F be that formula, with root r, a leaf labeled x, another leaf labeled y, and all other leaves labeled by positive constants. Let ℓ be the node that is the least common ancestor of the leaves labeled x and y. Then ℓ computes a polynomial p_ℓ, and the polynomial p_r computed at the root is of the form $Ap_\ell + B$. Further, p_ℓ is of the form $(Cx + D) \circ (Ey + F)$. And the constants A, B, C, D, E, F are all non-negative. Since $p_r = x+y+xy$, $\circ$ must be $\times$ (to generate the monomial xy), so $p_r = A(Cx+D)(Ey+F)+B$. Equating the coefficients, we get $B + ADF = 0$. Since there are no negative values, we must have $B = ADF = 0$. If $A = 0$, then $p_r = 0$, a contradiction. So $DF = 0$. Say $D = 0$. Then $p_r = ACx(Ey + F)$, and the monomial y is not generated. So there is no monotone ROF for p.

So now we can ask:

Question 2. When is a multilinear monotone read-once polynomial $p(x_1, \ldots, x_n)$ computable by a monotone ROF?

The above example $x + y + xy$ motivates studying the restriction of formulas with no additive constants. These are formulas where each leaf is labeled ax_i for some non-zero $a \in \mathbb{F}$. We have called such formulas constant-free, though technically they do use multiplicative constants. A constant-free read-once formula (CF-ROF) with n leaves depends on n variables and computes a multilinear polynomial $p(\bar{x})$ satisfying $p(\bar{0}) = 0$. A monotone CF-ROF computes a polynomial $p(\bar{x})$ that is monotone, multilinear, an ROP, and satisfies $p(\bar{0}) = 0$. Is the converse true? The polynomial $x + y + xy$ shows that it is not. So we can ask:

Question 3. When is a multilinear monotone read-once polynomial $p(x_1, \ldots, x_n)$ with $p(\bar{0}) = 0$ computable by a monotone CF-ROF?

What if we relax monotonicity but insist on no additive constants? That is, we ask:

Question 4. When is a multilinear monotone read-once polynomial $p(x_1, \ldots, x_n)$ with $p(\bar{0}) = 0$ computable by a CF-ROF?

That does not help; monotonocity is still forced, as shown below.

Lemma 6. *If $p(\bar{x})$ is monotone and is computed by a CF-ROF, then it is computed by a monotone CF-ROF.*

Proof. Induction on n, the number of variables that p depends on. Note that the CF-ROF F computing p will have exactly n leaves.

Base case $n = 1$; $p = a_1x_1$ for some $a_1 > 0$; trivially true.

Induction: Let F be of the form $G \circ H$, for $\circ \in \{+, \times\}$. Then G, H are CF-ROFs on disjoint sets of variables, computing polynomials g and h such that $p \equiv g \circ h$. Since p is monotone and g and h are variable-disjoint, either g and h are both monotone, or $\circ = \times$ and both $-g$ and $-h$ are monotone. In the first case, by induction, G and H are monotone CF-ROFs, and hence so is F. In the second case, note that $(-1) \times G$ is a ROF computing $-g$.

We can move the -1 down to the leaves to get a CF-ROF G' computing $-g$. (If $f = f_1 \times f_2$, then $(-1) \times f = [(-1) \times f_1] \times [f_2]$. If $f = f_1 + f_2$, then $(-1) \times f = [(-1) \times f_1] + [(-1) \times f_2]$.) Similarly, $-h$ is computed by a CF-ROF H'. By induction, $-g$ and $-h$ are computed by monotone CF-ROFs G'' and H'' respectively. Now the monotone CF-ROF $F' = G'' \times H''$ computes p. □

Polynomials computed by CF-ROFs have another interesting property: their monomials are incomparable with respect to divisibility. More precisely, for a multilinear polynomial p in n variables, define

$$M(p) = \left\{ T \subseteq [n] \mid m_T \triangleq \prod_{i \in T} x_i \text{ is a monomial of } p \right\}.$$

Lemma 7. *Let $p(x_1, \ldots, x_n)$ be computed by a CF-ROF. In the poset $(\mathcal{P}([n]), \subseteq)$, the sets in $M(p)$ form an antichain.*

Proof. Suppose not. Then the poset $(M(p), \subseteq)$ has at least one chain of length 2 or more. Let S, T be the lowest two elements of this chain with $S \subseteq T$; m_S, m_T are monomials of $p(.)$.

Let F be the CF-ROF computing $p(.)$. Setting variables outside T to 0 in F and removing subformulas evaluating to 0 should give a CF-ROF F' computing $p'(x) = p(x) \mid_{\bar{T} \to 0}$. Since $p'(.)$ has the monomials m_T and m_S, F' must compute both these monomials. To compute m_T, F' must have a leaf $a_i x_i$ for each $i \in T$ (it has no other leaves anyway), and must multiply all the leaves. So F' computes the single monomial m_T and cannot compute m_S, a contradiction. □

We now give a partial answer to Question 4 (or 3).

We approach one direction by associating with any monotone CF-ROF F a monotone Boolean function f. Recall that a Boolean function f is monotone if and only if for all $\bar{a}, \bar{b} \in \{0,1\}^n$, whenever $\bar{a} \preceq \bar{b}$ (in the pointwise ordering), then also $f(\bar{a}) \leq f(\bar{b})$. A minterm (maxterm, respectively) of a monotone Boolean function is a minimal cardinality subset S of variables such that assigning all variables in S to 1 (0, resp.) forces the function to evaluate to 1 (0 resp.) irrespective of the assignment to the remaining variables. For a monotone Boolean function f, let $\mathsf{minterm}(f)$ denote the set of all minterms of f and $\mathsf{maxterm}(f)$ the set of all maxterms of f. It is easy to see that $\forall S \in \mathsf{maxterm}(f), \forall T \in \mathsf{minterm}(f)$, $S \cap T \neq \emptyset$. The following result characterises monotone read-once Boolean functions with respect to its minterms and maxterms:

Proposition 5 ([KLN+93]). *A monotone Boolean function f is read-once if and only if*

$$\forall (S,T), \quad S \in \mathsf{maxterm}(f) \text{ and } T \in \mathsf{minterm}(f) \implies |S \cap T| = 1.$$

In the above, note that when f is read-once, the read-once formula computing it is also monotone.

Let p be a multilinear monotone polynomial computed by a CF-ROF. By Lemma 6, it is computed by a monotone CF-ROF F. Construct the Boolean formula F' by doing the following replacements

1. Change leaf label $a_i x_i$ (where $a_i > 0$) to x_i.
2. Change all $\times$ to $\wedge$.
3. Change all $+$ to $\vee$.

The formula F' is monotone and read-once, so the Boolean function f computed by it is monotone and read-once. Hence by Proposition 5, if S is a maxterm of f and T is a minterm of f, then $|S \cap T| = 1$. However, by construction, the minterms of f are precisely the monomials m of p. Similarly, the maxterms of f are precisely the multilinear monomials that "hit" every monomial of p (they share a variable with every monomial of p). Call the minimal such monomials the hitting monomials of p, and denote the collection of these monomials as $M^*(p)$.

$$\begin{aligned} \text{Hitting}(M(p)) &= \{S \subseteq [n] \mid \forall T \in M(p), S \cap T \neq \emptyset\}. \\ M^*(p) &= \{S \in \text{Hitting}(M(p)) \mid \forall T \subseteq S, T \neq S \Rightarrow T \notin \text{Hitting}(M(p))\}. \end{aligned}$$

Remark 2. It should be noted that in the literature of combinatorial commutative algebra, the square-free ideal generated by $M^*(p)$ is known as the *Alexander Dual* ideal of the square-free monomial ideal generated by $M(p)$. See [MS05] for more details.

Lemma 8. *Let p be a monotone polynomial computed by* CF-ROF *F. Let f denote the Boolean function computed by the Boolean formula F' obtained from F as described above. Then,*

1. *The monomials in $M(p)$ are in bijective correspondence with the minterms of f.*
2. *The monomials in $M^*(p)$ are in bijective correspondence with the maxterms of f.*

Proof. We prove the first statement; the second one follows immediately since the maxterms of a Boolean function are precisely the minimal hitting sets of the set of all minterms.

By Lemma 6, we can assume that the CF-ROF F computing p is monotone. We proceed by induction on the number of leaves in F. The base case $n = 1$ is trivially true. For $n > 1$, for some $\circ \in \{+, \times\}$ (type of the root gate of F), $F = F_1 \circ F_2$, and each F_i computes polynomial p_i. The variables of F_1 and F_2 are disjoint. By induction, $M(p_i)$ is in bijection with $\mathsf{minterm}(f_i)$, for associated Boolean function f_i. Consider the two cases.

- $\circ = +$; then $p = p_1 + p_2$, and $f = f_1 \vee f_2$. So $M(p) = M(p_1) \cup M(p_2)$. $\mathsf{minterm}(f) = \mathsf{minterm}(f_1) \cup \mathsf{minterm}(f_2)$, and the statement follows.
- $\circ = \times$; then $p = p_1 \times p_2$, and $f = f_1 \wedge f_2$. Then

$$\begin{aligned} M(p) &= \{m_1 \times m_2 \mid m_1 \in M(p_1), m_2 \in M(p_2)\} \\ \mathsf{minterm}(f) &= \{m_1 \wedge m_2 \mid m_1 \in \mathsf{minterm}(f_1), m_2 \in \mathsf{minterm}(f_2)\} \end{aligned}$$

  Hence the statement follows.

□

The above discussion (including Lemma 6, Lemma 7, Proposition 5, Lemma 8) amounts to the following:

Lemma 9. *Let $p(\bar{x})$ be a multilinear monotone polynomial $p(\bar{x})$ computable by a* CF-ROF. *Then*

1. *$(M(p), \subseteq)$ is an antichain, and*
2. *for every monomial $m \in M(p)$, and every hitting monomial $m^* \in M^*(p)$, m and m^* share exactly one variable. That is, $|m \cap m^*| = 1$.*

If we could show that the converse is also true, then we would have a characterisation, answering Question 4. Unfortunately, the converse is not true. For instance, consider the polynomials $p = x_1y_1 + 4x_2y_1 + 2x_1y_2 + 5x_2y_2$ and $q = x_1y_1 + x_2y_1 + x_1y_2 + x_2y_2 = (x_1 + x_2)(y_1+y_2)$. Since q is computed by a monotone CF-ROF, its monomials satisfy the properties in Lemma 9. Since q and p have the same set of monomials, the monomials of p also satisfy these properties. But no CF-ROF, let alone monotone CF-ROF, can compute p; this follows from Proposition 4 (consider the restriction $y_2 = 1$). This is not surprising because the coefficients play no role in the properties in Lemma 9 but are crucial for determining whether a polyomial is an ROP.

However, we can establish a weaker version: for polynomials with 0-1 coefficients, the converse is indeed true.

Lemma 10. *Let $p(\bar{x})$ be a multilinear monotone polynomial $p(\bar{x})$ with 0-1 coefficients satisfying*

1. *$(M(p), \subseteq)$ is an antichain, and*
2. *for every monomial $m \in M(p)$, and every hitting monomial $m^* \in M^*(p)$, $|m \cap m^*| = 1$.*

Then p is computable by a CF-ROF.

Proof. Consider the monotone Boolean function f defined as

$$f(x) = \bigvee_{T \in M(p)} \wedge_{i \in T} x_i.$$

Since $M(p)$ is an antichain, every monomial in $M(p)$ is a minterm of f. By construction, f has no other minterms; $\mathsf{minterm}(f) = M(p)$. Hence $\mathsf{maxterm}(f) = M^*(p)$. Along with the last property of p, we can hence invoke Proposition 5 to conclude that f is computed by a monotone read-once Boolean formula F. Now construct arithmetic formula F' by replacing all $\vee$ gates in F by $+$ gates and $\wedge$ gates by $\times$ gates. Then F' is the desired CF-ROF: the read-once property of F ensures every minterm of f has exactly one parse tree[4], and hence F' correctly computes p. □

Lemmas 9 and 10 give us the theorem:

Theorem 6. *Let $p(\bar{x})$ be any multilinear monotone polynomial with 0-1 coefficients. Then $p(\bar{x})$ is computable by a* CF-ROF *if and only if*

1. *$(M(p), \subseteq)$ is an antichain, and*
2. *for every monomial $m \in M(p)$, and every hitting monomial $m^* \in M^*(p)$, $|m \cap m^*| = 1$.*

[4] A parse tree is a sub-formula of F (i) containing the output gate, (ii) including, for every included $\vee$ gate, exactly one child, and (iii) including, for every included $\wedge$ gate, both its children.

7 Further Questions

- Can the results of [SV08] be extended to the case $\sum_{i=1}^{k} f_i^{r_i}$, where f_i's are ROFs?
- Do the results of [AvMV11] extend to read-k-multilinear branching programs?
- Can a hardness of representation for $\mathsf{Sym}_{n,d}$ be transformed into a polynomial identity test for a related model?
- Can the bound given by Theorem 5 be improved? As noted in Proposition 3, $n+1$ 0-justified depth-two ROPs are sufficient to represent any of the elementary symmetric polynomials. We conjecture:

  Conjecture 1. There is a constant $\epsilon > 0$ such that if there are 0-justified depth-three ROPs $f_1, \ldots, f_k$, and integers $e_1, \ldots e_k \geq 0$ satisfying

  $$\sum_{i=1}^{k} f_i^{e_i} = \mathsf{Sym}_{n,n/2},$$

  then $k = \Omega(n^{\epsilon})$.

- Similarly, how tight is the lower bound from Theorem 4? The question of obtaining even a polynomial, let alone linear, upper bound on the number of CF-ROPs required to represent the $\mathsf{Sym}_{n,n/2}$ is wide open.
- Does the class of CF-ROPs have a deterministic polynomial-time blackbox PIT algorithm?
- Can we completely characterize polynomials computed by monotone CF-ROFs?

Acknowledgements

The authors gratefully acknowledge Amir Shpilka's pointer regarding Theorem 2, when he and the first author were at the Dagstuhl Seminar 14121 on Computational Complexity of Discrete Problems. The authors are grateful to anonymous reviewers for their careful reading of the manuscript, several comments to improve readability, and for pointing out why the converse of Lemma 9 fails.

References

[Alo99] Noga Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8, 1999.

[AV08] Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Foundations of Computer Science (FOCS)*, pages 67–75, 2008.

[AvMV11] Matthew Anderson, Dieter van Melkebeek, and Ilya Volkovich. Derandomizing polynomial identity testing for multilinear constant-read formulae. In *CCC*, pages 273–282, 2011.

[BE11] Markus Bläser and Christian Engels. Randomness efficient testing of sparse black box identities of unbounded degree over the reals. In *Symposium on Theoretical Aspects of Computing (STACS)*, pages 555–566, 2011.

[BHS08] Markus Bläser, Moritz Hardt, and David Steurer. Asymptotically optimal hitting sets against polynomials. In *International Colloquium on Automata, Languages and Programming (ICALP) (1)*, pages 345–356, 2008.

[BOT88] Michael Ben-Or and Prasoon Tiwari. A deterministic algorithm for sparse multivariate polynominal interpolation (extended abstract). In *Symposium on Theory of Computing (STOC)*, pages 301–309, 1988.

[DL78] Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 1978.

[DS07] Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM Journal on Computing*, 36(5):1404–1434, 2007.

[FMM12] Hervé Fournier, Guillaume Malod, and Stefan Mengel. Monomials in arithmetic circuits: Complete problems in the counting hierarchy. In *Symposium on Theoretical Aspects of Computing (STACS)*, pages 362–373, 2012.

[GKKS13] Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth three. In *Foundations of Computer Science (FOCS)*, pages 578–587, 2013.

[Kay12] Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *ECCC*, 19(TR12-081):81, 2012.

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

[KLN+93] Mauricio Karchmer, Nathan Linial, Ilan Newman, Michael E. Saks, and Avi Wigderson. Combinatorial characterization of read-once formulae. *Discrete Mathematics*, 114(1-3):275–282, 1993.

[KS01] Adam Klivans and Daniel A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Symposium on Theory of Computing (STOC)*, pages 216–223, 2001.

[KS07] Neeraj Kayal and Nitin Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 16(2):115–138, 2007.

[MRS14] Meena Mahajan, B. V. Raghavendra Rao, and Karteek Sreenivasaiah. Monomials, multilinearity and identity testing in simple read-restricted circuits. *Theoretical Computer Science*, 524:90–102, 2014. preliminary version in MFCS 2012.

[MS05] Ezra Miller and Bernd Sturmfels. *Combinatorial Commutative Algebra*. Springer, 2005.

[RS11] B. V. Raghavendra Rao and Jayalal M. N. Sarma. Isomorphism testing of read-once functions and polynomials. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 115–126, 2011.

[RS13] B. V. Raghavendra Rao and Jayalal M. N. Sarma. Isomorphism testing of read-once functions and polynomials. submitted manuscript, 2013.

[RY11] Ran Raz and Amir Yehudayoff. Multilinear formulas, maximal-partition discrepancy and mixed-sources extractors. *Journal of Computer and System Sciences*, 77(1):167–190, 2011.

[Sax14] Nitin Saxena. Progress on polynomial identity testing - ii. *CoRR*, abs/1401.0976, 2014.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[SV08] Amir Shpilka and Ilya Volkovich. Read-once polynomial identity testing. In *Symposium on Theory of Computing (STOC)*, pages 507–516, 2008. See also ECCC TR-2010-011.

[SV10] Amir Shpilka and Ilya Volkovich. Read-once polynomial identity testing. Technical Report 011, ECCC, 2010. Preliminary version in STOC 2008.

[SY10] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3):207–388, 2010.

[Val79] Leslie G. Valiant. Negation can be exponentially powerful. In *Symposium on Theory of Computing (STOC)*, pages 189–196, 1979.

[Vol14] Ilya Volkovich. Characterizing arithmetic read-once formulae. *ArXiv*, arXiv:1408.1995, 2014.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM*, pages 216–226, 1979.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de