Deterministic identity testing paradigms for bounded top-fanin depth-4 circuits[*]

Pranjal Dutta[†] Prateek Dwivedi[‡] Nitin Saxena[†]

Abstract

Polynomial Identity Testing (PIT) is a fundamental computational problem. The famous depth-4 reduction result by Agrawal and Vinay (FOCS 2008) has made PIT for depth-4 circuits an enticing pursuit. A restricted depth-4 circuit computing a n-variate degree-d polynomial of the form $\sum_{i=1}^{k}\prod_j g_{ij}$, where $\deg g_{ij} \le \delta$ is called $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ circuit. On further restricting g_{ij} to be sum of univariates we obtain $\Sigma^{[k]}\Pi\Sigma\wedge$ circuits. The largely open, special-cases of $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ for constant k and δ, and $\Sigma^{[k]}\Pi\Sigma\wedge$ have been a source of many great ideas in the last two decades. For eg. depth-3 ideas of Dvir and Shpilka (STOC 2005), Kayal and Saxena (CCC 2006), and Saxena and Seshadhri (FOCS 2010 and STOC 2011). Further, depth-4 ideas of Beecken, Mittmann and Saxena (ICALP 2011), Saha, Saxena and Saptharishi (Comput.Compl. 2013), Forbes (FOCS 2015), and Kumar and Saraf (CCC 2016). Additionally, geometric Sylvester-Gallai ideas of Kayal and Saraf (FOCS 2009), Shpilka (STOC 2019), and Peleg and Shpilka (CCC 2020, STOC 2021). Very recently, a subexponential-time *blackbox* PIT algorithm for constant-depth circuits was obtained via lower bound breakthrough of Limaye, Srinivasan, Tavenas (FOCS 2021). We solve two of the basic underlying open problems in this work.

We give the *first* polynomial-time PIT for $\Sigma^{[k]}\Pi\Sigma\wedge$. We also give the *first* quasipolynomial time *blackbox* PIT for both $\Sigma^{[k]}\Pi\Sigma\wedge$ and $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$. A key technical ingredient in all the three algorithms is how the *logarithmic derivative*, and its power-series, modify the top Π-gate to $\wedge$.

Keywords Polynomial identity testing, hitting set, depth-4 circuits.

2012 ACM Subject Classification Theory of computation $\to$ Algebraic complexity theory.

Acknowledgement Pranjal is supported by the project "Foundation of Lattice-based Cryptography", funded by NUS-NCS Joint Laboratory for Cyber Security. Most of the work was carried out when Pranjal was visiting CSE, IIT Kanpur and supported by Google PhD Fellowship. Nitin thanks the funding support from DST (SJF/MSA-01/2013-14), SERB (CRG/2020/000045) and N.Rama Rao Chair.

[*]A preliminary version appeared in 36th Computational Complexity Conference (CCC), 2021. [DDS21a]
[†]School of Computing, NUS. Email: `duttpranjal@gmail.com`
[‡]Dept. of Computer Science & Engineering, IIT Kanpur. Email: {`pdwivedi,nitin`}`@cse.iitk.ac.in`

Contents

1 Introduction: PIT & beyond

Algebraic circuits are natural algebraic analog of boolean circuits, with the logical operations being replaced by $+$ and $\times$ operations over the underlying field. The study of algebraic circuits comprise the large study of algebraic complexity, mainly pioneered (and formalized) by Valiant [Val79]. A central problem in algebraic complexity is an algorithmic design problem, known as Polynomial Identity Testing (PIT): given an algebraic circuit $\mathcal{C}$ over a field $\mathbb{F}$ and input variables $x_1, \ldots, x_n$, determine whether $\mathcal{C}$ computes the identically zero polynomial. PIT has found numerous applications and connections to other algorithmic problems. Among the examples are algorithms for finding perfect matchings in graphs [Lov79, MVV87, FGT19], primality testing [AKS04], polynomial factoring [KSS14, DSS22], polynomial equivalence [DDOS14], reconstruction algorithms [KS06, Shp09, KS09] and the existence of algebraic natural proofs [CKR$^+$20, KRST]. Moreover, efficient design of PIT algorithms is intrinsically connected to proving strong lower bounds [HS80, Agr05, KI04, DSY10, FSV18, CKS18, DST21]. Interestingly, PIT also emerges in many fundamental results in complexity theory such as $\mathsf{IP} = \mathsf{PSPACE}$ [Sha92, LFKN92], the PCP theorem [ALM$^+$98, AS98], and the overarching Geometric Complexity Theory (GCT) program towards $\mathsf{P} \neq \mathsf{NP}$ [Mul12b, Mul12a, Gro15, JKY16].

There are broadly two settings in which the PIT question can be framed. In the *whitebox* setup, we are allowed to look inside the wirings of the circuit, while in the *blackbox* setting we can only evaluate the circuit at some points from the given domain. There is a very simple randomized algorithm for this problem - evaluate the polynomial at a random point from a large enough domain. With very high probability, a nonzero polynomial will have a nonzero evaluation; this is famously known as the Polynomial Identity Lemma [Ore22, DL78, Zip79, Sch80]. It has been a long standing open question to derandomize this algorithm.

For many years, blackbox identity tests were only known for depth-2 circuits which compute sparse polynomials [BOT88, KS01]. In a surprising result, Agrawal and Vinay [AV08] showed that a complete derandomization of blackbox identity testing for just depth-4 algebraic circuits ($\Sigma\Pi\Sigma\Pi$) already implies a near complete derandomization for the general PIT problem. More recent depth reduction results [Koi12, GKKS16], and the bootstrapping phenomenon [AGS19, KST19, GKSS22, And20] show that even PIT for very restricted classes of depth-4 circuits (*even* depth-3) would have very interesting consequences for PIT of general circuits. These results make the identity testing regime for depth-4 circuits, a very meaningful pursuit.

Three PITs in one-shot. Following the same spirit, here we solve three important (and open) PIT questions. We give the first deterministic polynomial-time whitebox PIT algorithm for the bounded sum of product of sum of univariates circuits [SSS13, Open Prob. 2]. Further, we give a quasipolynomial-time blackbox algorithm for the same class of circuits. These circuits are denoted by $\Sigma^{[k]}\Pi\Sigma\wedge$ and compute polynomials of the form $\Sigma_{i\in[k]}\Pi_j\left(g_{ij1}(x_1)+\cdots+g_{ijn}(x_n)\right)$.

> *Whitebox and Blackbox PIT for the $\Sigma^{[k]}\Pi\Sigma\wedge$ circuits is in polynomial and quasi-polynomial time respectively.*

A similar technique also gives a quasi-polynomial time blackbox PIT algorithm for the bounded sum of product of bounded degree sparse polynomials circuits. They are denoted by $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ (where k and δ can be up to $\mathsf{poly}(\log(s))$, where s is the circuit size).

> *Blackbox PIT for the $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ circuits is in quasi-polynomial time.*

$\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ circuits compute polynomials which are of the form $\Sigma_{i\in[k]}\Pi_j g_{ij}(\boldsymbol{x})$, where $\deg(g_{ij})\le\delta$. Even $\delta=2$ was a challenging open problem [KS16, Open Problem 2]. The model has gained a lot of interest in the past few years and has generated many important results [PS20, PS21, GOS22, OS22].

1.1 Main results: An analytic detour to three PITs

Though some attempts have been made to solve PIT for $\Sigma^{[k]}\Pi\Sigma\wedge$, an efficient PIT for $k\ge 3$ *even* in the whitebox settings remains open, see [SSS13, Open Prob. 2]. Our first result addresses this problem and designs a polynomial time algorithm (Ref. Algorithm 1). In our pursuit we discover an analytic and non-ideal based new technique which we refer as DiDI. Throughout the paper, we will work with $\mathbb{F}=\mathbb{Q}$, though all the results hold for field of large characteristic.

Theorem 1.1 (Whitebox $\Sigma^{[k]}\Pi\Sigma\wedge$ PIT)**.** *There is a deterministic, whitebox $s^{O(k\,7^k)}$-time PIT algorithm for $\Sigma^{[k]}\Pi\Sigma\wedge$ circuits of size s, over $\mathbb{F}[\boldsymbol{x}]$.*

Remark.

1. Case $k \le 2$ can be solved by invoking [SSS13, Theorem 5.2]; but $k \ge 3$ was open.
2. Our technique *necessarily* blows up the exponent exponentially in k. In particular, it would be interesting to design an efficient time algorithm when $k = \Theta(\log s)$.
3. It is not clear if the current technique gives PIT for $\Sigma^{[k]}\Pi\Sigma\mathrm{M}_2$ circuits, where $\Sigma\mathrm{M}_2$ denotes sum of *bi*variate monomials computed and fed into the top product gate.

Next, we go to the blackbox setting and address two models of interest, namely— $\Sigma^{[k]}\Pi\Sigma\wedge$ and $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$, where k, δ are constants. Our work builds on previous ideas for unbounded top fanin (1) Jacobian [ASSS16], (2) the known blackbox PIT for $\Sigma\wedge\Sigma\wedge$ and $\Sigma\wedge\Sigma\Pi^{[\delta]}$ [GKS17, For15] while maneuvering with an analytic approach *via* power-series, which unexpectedly *reduces* the top Π-gate to a $\wedge$-gate.

Theorem 1.2 (Blackbox depth-4 PIT)**.**

(a) There is a $s^{O(k \log\log s)}$ time blackbox PIT algorithm for $\Sigma^{[k]}\Pi\Sigma\wedge$ circuits of size s, over $\mathbb{F}[\boldsymbol{x}]$.

(b) There is a $s^{O(\delta^2\, k\, \log s)}$ time blackbox PIT algorithm for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ circuits of size s, over $\mathbb{F}[\boldsymbol{x}]$.

Remark.

1. Theorem 1.2 (b) has a *better* dependence on k, but *worse* on s, than Theorem 1.1. Our results are quasipoly-time even up to $k, \delta = \mathsf{poly}(\log s)$.
2. Theorem 1.2 (a) is better than Theorem 1.2 (b), because $\Sigma\wedge\Sigma\wedge$ has a faster algorithm than $\Sigma\wedge\Sigma\Pi^{[\delta]}$.
3. Even for $\Sigma^{[3]}\Pi\Sigma\wedge$ and $\Sigma^{[3]}\Pi\Sigma\Pi^{[3]}$ models, we leave the *poly*-time blackbox question open.

1.2 Prior works on related models

In the last two decades, there has been a surge of results on identity testing for restricted classes of bounded depth algebraic circuits (e.g. 'locally' bounded independence, bounded read/occur, bounded variables). There have been numerous results on PIT for depth-3 circuits with bounded top fanin (known as $\Sigma^{[k]}\Pi\Sigma$-circuits). Dvir and Shpilka [DS07] gave the first quasipolynomial-time deterministic whitebox algorithm for $k = O(1)$, using rank based methods, which finally lead Karnin and Shpilka [KS11] to design algorithm of same complexity in the blackbox setting. Kayal and Saxena [KS07] gave the first polynomial-time algorithm of the same. Later, a series of works in [SS11, SS12, SS13, ASSS16] generalized the model and gave $n^{O(k)}$-time algorithm when the algebraic rank of the product polynomials are bounded. Note that in the white-box setting, our algorithm gives a poly(s) time PIT algorithm for bounded top-fanin depth-3 circuit. Moreover, the dependence on the top-fan is exponential. In the blackbox setting, our algorithm solves PIT for bounded top-fanin depth-3 circuit in quasi-poly(s) time, hence it does not offer any speedup compared to known polynomial time algorithms. However, our algorithm does give a PIT idea that is different from the known ones.

There has also been some progress on PIT for restricted classes of depth-4 circuits. A quasipolynomial-time blackbox PIT algorithm for *multilinear* $\Sigma^{[k]}\Pi\Sigma\Pi$-circuits was designed in [KMSV13], which was further improved to a $n^{O(k^2)}$-time deterministic algorithm in [SV18]. A quasipolynomial blackbox PIT was given in [BMS13, KS16] when algebraic rank of the irreducible factors in each multiplication gate as well as the bottom fanin are bounded. Further interesting restrictions like sum of product of fewer variables, and more structural restrictions have been exploited, see [FS13, ASS13, For15, Muk16, KS17]. Some progress has also been made for bounded top-fanin and bottom-fanin depth-4 circuits via incidence geometry [Gup14, Shp19, PS20]. In fact, very recently, [PS21] gave a polynomial-time blackbox PIT for $\Sigma^{[3]}\Pi\Sigma\Pi^{[2]}$-circuits.

Model	Time	Ref.
$\Sigma^{[k]}\Pi^{[d]}\Sigma$	$\mathsf{poly}(n, d^k)$	[SS12]
Multilinear $\Sigma^{[k]}\Pi\Sigma\Pi$	$\mathsf{poly}(n^{O(k^2)})$	[SV18, ASSS16]
$\Sigma\Pi\Sigma\Pi$ of bounded trdeg	$\mathsf{poly}(s^{\mathsf{trdeg}})$	[BMS13]
$\Sigma^{(k)}\Pi\Sigma\Pi^{[d]}$ of bounded *local* trdeg	$\mathsf{QP}(n)$	[KS17]
$\Sigma^{[3]}\Pi\Sigma\Pi^{[2]}$	$\mathsf{poly}(n, d)$	[PS21]
$\overline{\Sigma^{[k]}\Pi\Sigma\wedge}$	$s^{O(k\cdot 7^k\cdot\log\log s)}$	[DDS21b]
$\overline{\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}}$	$s^{O(\delta^2\cdot k\cdot 7^k\cdot\log s)}$	[DDS21b]
$\Sigma\Pi\Sigma\Pi$	SUBEXP(n)	[LST21]
Whitebox $\Sigma^{[k]}\Pi\Sigma\wedge$	$s^{O(k 7^k)}$	This work.
$\Sigma^{[k]}\Pi\Sigma\wedge$	$s^{O(k\log\log s)}$	This work.
$\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$	$s^{O(\delta^2\, k\, \log s)}$	This work.

Table 1: Time complexity comparision of PIT algorithms related to $\Sigma\Pi\Sigma\Pi$ circuits

The authors recently generalised their novel DiDI-technique to solve 'border PIT' of depth-4 circuits [DDS21b]. Specifically, they give a $s^{O(k\cdot 7^k\cdot\log\log s)}$ time and $s^{O(\delta^2\cdot k\cdot 7^k\cdot\log s)}$ time blackbox PIT algorithm for $\overline{\Sigma^{[k]}\Pi\Sigma\wedge}$ and $\overline{\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}}$ respectively. By definition, border classes capture exact complexity classes, hence border PIT results seemingly subsumes the results we present in this paper. However, the whitebox PIT algorithm here is much more efficient than their quasi-poly time blackbox algorithm. Further, the time complexity of blackbox PIT algorithms has a better dependence on k and δ compared to their exponential dependence. Lastly, the proofs in this paper are simpler as we don't have to deal with an infinitesimally close approximation of polynomials in border complexity classes. Very recently, Dutta and Saxena [DS22] showed an exponential-gap fanin-hierarchy theorem for bounded depth-3 circuits which is also based on a *finer* generalization of the DiDI-technique.

In a breakthrought result by Limaye, Srinivasan and Tavenas [LST21] the *first* superpolynomial lower bound for constant depth circuits was obtained. Their lower bound result, together with the 'hardness vs randomness' tradeoff result of [CKS18] gives the *first* deterministic blackbox PIT algorithm for general depth-4 circuits which runs in $s^{O(n^{\epsilon})}$ for all real $\epsilon > 0$. Their result is the first *sub*exponential time PIT algorithm for depth-4 circuits. Moreover, compared to their algorithm, our quasipoly time blackbox and polynomial time whitebox algorithms are significantly faster.

Limitations of known techniques. People have studied depth-4 PIT only with extra restrictions, mostly due to the limited applicability of the existing techniques as they were tailor-made for the specific models and do not generalize. E.g. the previous methods handle $\delta = 1$ (i.e. linear polynomials at the bottom) or $k = 2$ (via *factoring*, [SSS13]). While $k = 2$ to 3, or $\delta = 1$ to 2 (i.e. 'linear' to 'quadratic') already demands a qualitatively different approach.

Whitebox $\Sigma^{[k]}\Pi\Sigma\wedge$ model generalizes the famous bounded top fanin depth-3 circuits $\Sigma^{[k]}\Pi\Sigma$ of [KS07]; but their Chinese Remaindering (CR) method, loses applicability and thus fails to solve even a slightly more general model. The blackbox setting involved similar 'certifying path' ideas in [SS12] which could be thought of as general CR. It comes up with an ideal I such that $f \neq 0 \bmod I$ and finally preserves it under a constant-variate linear map. The preservation gets harder (for both $\Sigma^{[k]}\Pi\Sigma\wedge$ and $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$) due to the increased non-linearity of the ideal I generators. Intuitively, larger δ via ideal-based routes, brings us to the Gröbner basis method (which is doubly-exponential-time in n) [Vas04]. We know that ideals even with 3-generators (analogously $k = 4$) already capture the whole ideal-membership problem [Sap19b].

The algebraic-geometric approach to tackle $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ has been explored in [BMS13, Gup14, Muk16, Guo21]. The families which satisfy a certain Sylvester–Gallai configuration (called SG-circuits) is the harder case which is conjectured to have constant transcendence degree [Gup14, Conj. 1]. Non-SG circuits is the case where the nonzeroness-certifying-path question reduces to radical-ideal non-membership questions [GS20]. This is really a variety question where one could use algebraic-geometry tools to design a poly-time blackbox PIT. In fact, very recently, Guo [Guo21] gave a s^{δ^k}-time PIT by constructing explicit variety evasive subspace families. Unfortunately, this is not the case in the ideal non-membership; this scenario makes it much harder to solve $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$. From this viewpoint, radical-ideal-membership explains well why the intuitive $\Sigma^{[k]}\Pi\Sigma$ methods do not extend to $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$.

Interestingly, Forbes [For15] found a quasipolynomial-time PIT for $\Sigma\wedge\Sigma\Pi^{[\delta]}$ using shifted-partial derivative techniques; but it naively fails when one replaces the $\wedge$-gate by Π (because the 'measure' becomes too large). The duality trick of [Sax08] completely solves whitebox PIT for $\Sigma\wedge\Sigma\wedge$, by transforming it to a read-once oblivious ABP (ROABP); but it is inapplicable to our models with the top Π-gate (due to large waring rank and ROABP-width). A priori, our models are incomparable to ROABP, and thus the famous PIT algorithms for ROABP [FS13, FSS14, GKS17] are not expected to help either.

Similarly, a naive application of the *Jacobian* and *certifying path* technique from [ASSS16] fails for our models because it is difficult to come up with a *faithful* map for constant-variate reduction. Kumar and Saraf [KS16] crucially used that the computed polynomial has low individual degree (such that [DSY10] can be invoked), while in [KS17] they exploits the low algebraic rank of the polynomials computed below the top Π-gate. Neither of them hold in general for our models. Very recently, Peleg and Shpilka [PS21] gave a poly-time blackbox PIT for $\Sigma^{[3]}\Pi\Sigma\Pi^{[2]}$, via incidence geometry (e.g. Edelstein-Kelly theorem involving 'quadratic' polynomials), by solving [Gup14, Conj. 1] for $k = 3, \delta = 2$. The method seems very strenuous to generalize even to 'cubic' polynomials ($\delta = 3 = k$).

PIT for other models. Blackbox PIT algorithms for many restricted models are known. Egs. ROABP related models [RS05, JQS10, AGKS15, GKS17, GKST17, FSS14, AFS+18], log-variate circuits [FGS18, BS21], and non-commutative models [GGOW16, LMP19].

1.3 Techniques and motivation

Both the proofs are analytic as they use *logarithmic derivative*, and its power-series expansion which greatly transform the respective models. Where the nature of the first proof is inductive, the second is a more direct *one-shot* proof. In both the cases, we essentially reduce to the well-understood *wedge* models, that have unbounded top fanin, yet for which PITs are known. This reduction is unforeseeable and quite 'power'ful.

The analytic tool that we use, appears in algebra and complexity theory through the *formal power series* ring $\mathsf{R}[[x_1, \dots, x_n]]$ (in short $\mathsf{R}[[\boldsymbol{x}]]$), see [Niv69, Sin19, DSS22]. The advantages of the ring $\mathsf{R}[[\boldsymbol{x}]]$ are many and they usually emerge because of the inverse identity: $(1 - x_1)^{-1} = \sum_{i \geq 0} x_1^i$, which does not make sense in $\mathsf{R}[x]$, but is valid in $\mathsf{R}[[\boldsymbol{x}]]$. Other analytic tools used are inspired from Wronskian (linear dependence) [KPT15, Theorem 7] [KKPS15], Jacobian (algebraic dependence) [BMS13, ASSS16, PSS18], and logarithmic derivative operator $\mathsf{dlog}_{z_1}(f) = (\partial_{z_1} f)/f$.

We will be work with the division operator (e.g. $\mathsf{R}(z_1)$, over a certain ring R). However, the divisions do not come for free as they require invertibility with respect to z_1 throughout (again landing us in $\mathsf{R}[[z_1]]$. For circuit classes C, D we define class

$$\mathcal{C}/\mathcal{D} := \{f/g \mid f \in \mathcal{C}, \mathcal{D} \ni g \neq 0\}.$$

Similarly $\mathcal{C} \cdot \mathcal{D}$ to denotes the class taking respective products.

1.3.1 The DiDI-technique

In Theorem 1.1 we introduce a novel technique for designing PIT algorithms which comprises of inductively applying two fundamental operations on the input circuits to reduce it to a more tractable model. Suppose we want to test $\sum_{i\in[k]} T_i \overset{?}{=} 0$ where each T_i is computable by $\Pi\Sigma\wedge$. The idea is to *DI*vide it by T_k to obtain $1 + \sum_{i\in[k-1]} T_i/T_k$ and then *D*erivative to reduce the fanin to $k-1$ and obtain $\sum_{i\in[k-1]} \mathcal{T}_i$. Naturally, these operations pushes us to work with the fractional ring (e.g. $\mathsf{R}(z_1)$, over a certain ring R), further it also distorts the model as $\mathcal{T}_i$'s are no longer computable by simple $\Pi\Sigma\wedge$ circuits. However, with careful analytically analysis we establish that the non-zeroness is preserved in the reduced model. The process is then repeated until we reach $k = 1$, while maintaining the invariants which help us in preserving the non-zeroness till the end. We finish the proof by showing that the identity testing of reduced model can be done using known PIT algorithms.

1.3.2 Jacobian hits again

In Theorem 1.2 we exploit the prowess of the Jacobian polynomial first introduced in [BMS13] and later explored in [ASSS16] to unify known PIT algorithms and design new ones. Suppose we want to test $\sum_{i\in[k]} T_i \overset{?}{=} 0$, where $T_i \in \Pi\Sigma\Pi^{[\delta]}$ (respec. $\Pi\Sigma\wedge$). We associate the Jacobian $J(T_1,\ldots,T_r)$ to captures the algebraic independence of $T_1,\ldots,T_r$ assuming this to be a transcendence basis of the T_i's. We design a variable reducing linear map Φ which preserves the algebraic independedece of $T_1,\ldots,T_r$ and show that for any C: $C(T_1,\ldots,T_k) = 0 \iff C(\Phi(T_1),\ldots,\Phi(T_k)) = 0$. Such a map is called 'faithful' [ASSS16]. The map Φ ultimately provides a hitting set for $T_1 + \ldots + T_k$, as we reduce to a PIT of a polynomial over 'few' (roughly equal to k) variables, yielding a QP-time algorithm.

2 Preliminaries

Before proving the results, we describe some of the assumptions and notations used throughout the paper. $\mathbf{x}$ denotes $(x_1,\ldots,x_n)$. $[n]$ denotes $\{1,\ldots,n\}$.

2.1 Notations and Definitions

- **Logarithmic derivative.** Over a ring R and a variable y, the logarithmic derivative $\mathsf{dlog}_y : \mathsf{R}(y) \to \mathsf{R}(y)$ is defined as $\mathsf{dlog}_y(f) := \partial_y f/f$; here ∂_y denotes the partial derivative with respect to variable y. One important property of dlog is that it is additive over a product as
$$\mathsf{dlog}_y(f\cdot g) = \frac{\partial_y(f\cdot g)}{f\cdot g} = \frac{(f\cdot\partial_y g + g\cdot\partial_y f)}{f\cdot g} = \mathsf{dlog}_y(f) + \mathsf{dlog}_y(g).$$
We refer this effect as *linearization* of product.

- **Circuit size.** Sparsity $\mathsf{sp}(\cdot)$ refers to the number of nonzero monomials. In this paper, it is a parameter of the circuit size. In particular, $\mathsf{size}(g_1 \cdots g_s) = \sum_{i\in[s]} (\mathsf{sp}(g_i) + \deg(g_i))$, for $g_i \in \Sigma\wedge$ (respectively $\Sigma\Pi^{[\delta]}$). In whitebox settings, we also include the *bit-complexity* of the circuit (i.e. bit complexity of the constants used in the wires) in the size parameter. Some of the complexity parameters of a circuit are *depth* (number of layers), *syntactic degree* (the maximum degree polynomial computed by any node), *fanin* (maximum number of inputs to a node).

- **Hitting set.** A set of points $\mathcal{H} \subseteq \mathbb{F}^n$ is called a *hitting-set* for a class $\mathcal{C}$ of n-variate polynomials if for any nonzero polynomial $f \in \mathcal{C}$, there exists a point in $\mathcal{H}$ where f evaluates to a nonzero value. A $T(n)$-time hitting-set would mean that the hitting-set can be generated in time $T(n)$, for input size n.

- **Valuation.** Valuation is a map $\mathsf{val}_y : \mathsf{R}[y] \to \mathbb{Z}_{\geq 0}$, over a ring R, such that $\mathsf{val}_y(\cdot)$ is defined to be the maximum power of y dividing the element. It can be easily extended to fraction field $\mathsf{R}(y)$, by defining $\mathsf{val}_y(p/q) := \mathsf{val}_y(p) - \mathsf{val}_y(q)$; where it can be negative.

- **Field.** We denote the underlying field as $\mathbb{F}$ and assume that it is of characteristic 0. All our results hold for other fields (eg. $\mathbb{Q}_p, \mathbb{F}_p$) of *large* characteristic (see Remarks in Section 3-4).

- **Jacobian.** The Jacobian of a set of polynomials $\mathbf{f} = \{f_1, \ldots, f_m\}$ in $\mathbb{F}[\boldsymbol{x}]$ is defined to be the matrix $\mathcal{J}_{\boldsymbol{x}}(\mathbf{f}) := \left(\partial_{x_j}(f_i)\right)_{m\times n}$. Let $S \subseteq \boldsymbol{x} = \{x_1, \ldots, x_n\}$ and $|S| = m$. Then, polynomial $J_S(\mathbf{f})$ denotes the minor (i.e. determinant of the submatrix) of $\mathcal{J}_{\boldsymbol{x}}(\mathbf{f})$, formed by the columns corresponding to the variables in S.

2.2 Basics of Algebraic Complexity Theory

For detailed discussion on the basics of Algebraic Complexity Theory we will encourage readers to refer [SY10, Sax09, Mah13, Sax14, Sap19a]. Here we will formally state a few of the PIT results and properties of circuits for the later reference.

Trivial PIT Algorithm

The simplest PIT algorithm for any circuit in general is due to Polynomial Identity Lemma [Ore22, DL78, Zip79, Sch80]. When the number of variables is small, say $O(1)$, then this algorithm is very efficient.

Lemma 2.1 (Trivial PIT)**.** *For a class of n-variate, individual degree $< d$ polynomial $f \in \mathbb{F}[\boldsymbol{x}]$ there exists a deterministic PIT algorithm which runs in time $O(d^n)$.*

Sparse Polynomial

Sparse PIT is testing the identity of polynomials with bounded number of monomials. There have been a lot of work on sparse-PIT, interested readers can refer [BOT88, KS01] and references therein. For the proof of poly-time hitting set of Sparse PIT see [Sax09, Thm. 2.1].

Theorem 2.2 (Sparse-PIT map [KS01])**.** *Let $p(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ with individual degree at most d and sparsity at most m. Then, there exists $1 \leq r \leq (mn\log d)^2$, such that*

$$p(y, y^d, \ldots, y^{d^{n-1}}) \neq 0, \bmod y^r - 1.$$

If p is computable by a size-s $\Sigma\Pi$ circuit, then there is a deterministic algorithm to test its identity which runs in time $\mathsf{poly}(s, m)$.

Indeed if identity of sparse polynomial can be tested efficiently, product of sparse polynomial can be tested efficiently. We formalise this in the following:

Lemma 2.3 ([Sap13] Lemma 2.3)**.** *For a class of n-variate, degree d polynomial $f \in \mathbb{F}[\boldsymbol{x}]$ computable by $\Pi\Sigma\Pi$ of size s, there is a deterministic PIT algorithm which runs in time $\mathsf{poly}(s, d)$.*

A set $\mathcal{H} \subseteq \mathbb{F}^n$ is called a Hitting Set for a class polynomial $\mathcal{C} \subseteq \mathbb{F}[x]$, if for all $g \in \mathcal{C}$

$$g \neq 0 \iff \exists \boldsymbol{\alpha} \in \mathcal{H} : g(\boldsymbol{\alpha}) \neq 0.$$

In literature, PIT has a close association with Hitting set as the two notions are provably equivalent (refer Lemma 3.2.9 and 3.2.10 [For14]). Note that the set $\mathcal{H}$ works for every polynomial of the class. Instead of a PIT algorithm occasionally we will use such a set.

Lemma 2.4 (Hitting Set of $\Pi\Sigma\wedge$)**.** *For a class of n-variate, degree d polynomial $f \in \mathbb{F}[\boldsymbol{x}]$ computable by $\Pi\Sigma\Pi$ of size s, there is an explicit Hitting Set of size $\mathsf{poly}(s, d)$.*

Algebraic Branching Program (ABP)

An ABP is a layered directed acyclic graph with $q+1$ many layers of vertices $V_0, \ldots, V_q$ with a source a and a sink b such that all the edges in the graph only go from a to V_0, V_{i-1} to V_i for any $i \in [q]$, and V_q to b. The edges have *uni*variate polynomials as their weights. The ABP is said to compute the polynomial

$$f(\boldsymbol{x}) \;=\; \sum_{p \in \mathsf{paths}(a,b)} \prod_{e \in p} W(e)\,,$$

where $W(e)$ is the weight of the edge e. The ABP has width-w if $|V_i| \leq w, \forall i \in \{0, \ldots, q\}$. In an equivalent definition, polynomials computed by ABP are of the form $A^T(\prod_{i\in[q]} D_i)B$, where $A, B \in \mathbb{F}^{w\times 1}[\boldsymbol{x}]$, and $D_i \in \mathbb{F}^{w\times w}[\boldsymbol{x}]$, where entries are univariate polynomials. We encourage interested readers to refer [SY10, Mah13] for more detailed discussion.

Definition 2.5 (Read-once oblivious ABP (ROABP))**.** *An ABP is called a* read-once oblivious

ABP (ROABP) *if the edge weights are univariate polynomials in* distinct *variables across layers. Formally, there is a permutation π on the set $[q]$ such that the entries in the i-th matrix D_i are univariate polynomials over the variable $x_{\pi(i)}$, i.e., they come from the polynomial ring $\mathbb{F}[x_{\pi(i)}]$.*

A polynomial $f(x)$ is said to be computed by width-w ROABPs in *any order*, if for every permutation σ of the variables, there exists a width-w ROABP in the variable order σ that computes the polynomial $f(\boldsymbol{x})$. In whitebox setting, identity testing of any-order ROABP is completely solved.

Theorem 2.6 (Theorem 2.4 [RS05]). *For n-variate polynomials computed by size-s ROABP, a hitting set of size $O(s^5 + s \cdot n^4)$ can be constructed.*

There have been quite a few results on blackbox PIT for ROABPs as well [FS13, FSS14, GKS17]. The current best known algorithm works in quasipolynomial time.

Theorem 2.7 (Theorem 4.9 [GKS17]). *For n-variate, individual-degree-d polynomials computed by width-w ROABPs in any order, a hitting set of size $(ndw)^{O(\log\log w)}$ can be constructed.*

Depth-4 Circuits

A polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ is computable by $\Sigma\wedge\Sigma\Pi^{[\delta]}$ circuits if $f(\boldsymbol{x}) = \sum_{i\in[s]} f_i(\boldsymbol{x})^{e_i}$ where $\deg f_i \le \delta$. The first nontrivial PIT algorithm for this model was designed in [For15].

Theorem 2.8 (Proposition 4.18 [For15]). *There is a $\mathsf{poly}(n, d, \delta\log s)$-explicit hitting set of size $(nd)^{O(\delta\log s)}$ for the class of n-variate, degree-($\le d$) polynomials $f(\boldsymbol{x})$, computed by $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit of size s.*

Similarly, $\Sigma\wedge\Sigma\wedge$ circuits compute polynomials of the form $f(\boldsymbol{x}) = \sum_{i\in[s]} f_i^{e_i}$ where f_i is a sum of univariate polynomials. Using duality trick [Sax08] and PIT results from [RS05, GKS17], one can design efficient PIT algorithm for $\Sigma\wedge\Sigma\wedge$ circuits.

Lemma 2.9 (PIT for $\Sigma\wedge\Sigma\wedge$-circuits). *Let $P \in \Sigma\wedge\Sigma\wedge$ of size s. Then, there exists a $\mathsf{poly}(s)$ (respectively $s^{O(\log\log s)}$) time whitebox (respectively blackbox) PIT for the same.*

Proof sketch. We show that any $g(\boldsymbol{x})^e = (g_1(x_1) + \ldots + g_n(x_n))^e$, where $\mathsf{deg}(g_i) \le s$ can be written as $\sum_j h_{j1}(x_1)\cdots h_{jn}(x_n)$, for some $h_{j\ell} \in \mathbb{F}[x_\ell]$ of degree at most es. Define, $G := (y + g_1)\cdots(y+g_n) - y^n$. In its e-th power, notice that the leading-coefficient is $\mathsf{coef}_{y^{e(n-1)}}(G^e) = g^e$. So, interpolate on $e(n-1)+1$ many points ($y = \beta_i \in \mathbb{F}$) to get

$$\mathsf{coef}_{y^{e(n-1)}}(G^e) \;=\; \sum_{i=1}^{e(n-1)+1} \alpha_i\, G^e(\beta_i)\ .$$

Now, expand $G^e(\beta_i) = ((\beta_i + g_1)\cdots(\beta_i + g_n) - \beta_i^n)^e$, by binomial expansion (without expanding the inner n-fold product). The top-fanin can be atmost $s\cdot(e+1)\cdot(e(n-1)+1) = O(se^2n)$. The individual degrees of the intermediate univariates can be at most es. Thus, it can be computed by an ROABP (of *any order*) of size at most $O(s^2e^3n)$.

Now, if $f = \sum_{j\in[s]} f_j^{e_j}$ is computed by a $\Sigma\wedge\Sigma\wedge$ circuit of size s, then clearly, f can also be computed by an ROABP (of any order) of size at most $O(s^6)$. So, the whitebox PIT follows from Theorem 2.6, while the blackbox PIT follows from Theorem Theorem 2.7. $\square$

Further, $\Sigma\wedge\Sigma\wedge$ can be shown to be closed under multiplication i.e., product of two polynomials, each computable by a $\Sigma\wedge\Sigma\wedge$ circuit, is computable by a single $\Sigma\wedge\Sigma\wedge$ circuit. To prove that we will need an efficient way to write a product of a few powers as a sum of powers, using simple interpolation. For an algebraic proof, see [CCG12, Proposition 4.3].

Lemma 2.10 (Waring Identity for a monomial)**.** *Let $M = x_1^{b_1} \cdots x_k^{b_k}$, where $1 \le b_1 \le \ldots \le b_k$, and* roots of unity $\mathcal{Z}(i) := \{z \in \mathbb{C} : z^{b_i+1} = 1\}$. *Then,*

$$M = \sum_{\varepsilon(i) \in \mathcal{Z}(i): i=2,\cdots,k} \gamma_{\varepsilon(2),\ldots,\varepsilon(k)} \cdot (x_1 + \varepsilon(2)x_2 + \ldots + \varepsilon(k)x_k)^d \ ,$$

where $d := \deg(M) = b_1 + \ldots + b_k$, and $\gamma_{\varepsilon(2),\ldots,\varepsilon(k)}$ are scalars ($\mathsf{rk}(M) := \prod_{i=2}^k (b_i+1)$ many).

Remark. We actually need not work with $\mathbb{F} = \mathbb{C}$. We can go to a small extension (at most d^k), for a monomial of degree d, to make sure that $\varepsilon(i)$ exists.

Using the above lemma we prove the closure result.

Lemma 2.11. *Let $f_i(\mathbf{x}, y) \in \mathbb{F}[y][\mathbf{x}]$, of syntactic degree $\le d_i$, be computed by a $\Sigma\wedge\Sigma\wedge$ circuit of size s_i, for $i \in [k]$ (wrt $\mathbf{x}$). Then, $f_1 \cdots f_k$ has $\Sigma\wedge\Sigma\wedge$ circuit of size $O((d_2+1)\cdots(d_k+1)\cdot s_1 \cdots s_k)$.*

Proof. Let $f_i = \sum_j f_{ij}^{e_{ij}}$; by assumption $e_{ij} \le d_i$ (by assumption). Then using Lemma 2.10, $f_{1j_1}^{e_{1j_1}} \cdots f_{kj_k}^{e_{kj_k}}$ has size at most $(d_2+1)\cdots(d_k+1)\cdot\left(\sum_{i\in[k]} \mathsf{size}(f_{ij_i})\right)$, for indices $j_1, \ldots, j_k$. Summing up for all $s_1 \cdots s_k$ many products (atmost) gives the upper bound. $\square$

3 Whitebox PIT for $\Sigma^{[k]}\Pi\Sigma\wedge$

We consider a bloated model of computation which naturally generalizes $\Sigma\Pi\Sigma\wedge$ circuits and works ideally under the DiDI-techniques.

Definition 3.1. *We call a circuit $\mathcal{C} \in \mathsf{Gen}(k,s)$, over $\mathsf{R}(\mathbf{x})$, for any ring R, with parameter k and size-s, if $\mathcal{C} \in \Sigma^{[k]}(\Pi\Sigma\wedge/\Pi\Sigma\wedge)\cdot(\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$. It computes $f \in \mathsf{R}(\mathbf{x})$, if $f = \sum_{i=1}^k T_i$, where*

- $T_i =: (U_i/V_i)\cdot(P_i/Q_i)$, *for $U_i, V_i \in \Pi\Sigma\wedge$, and $P_i, Q_i \in \Sigma\wedge\Sigma\wedge$,*
- $\mathsf{size}(T_i) = \mathsf{size}(U_i) + \mathsf{size}(V_i) + \mathsf{size}(P_i) + \mathsf{size}(Q_i)$, *and $\mathsf{size}(f) = \sum_{i\in[k]} \mathsf{size}(T_i)$.*

It is easy to see that all size-s $\Sigma^{[k]}\Pi\Sigma\wedge$ circuit are in $\mathsf{Gen}(k,s)$. We will design the *recursive* algorithm on $\mathsf{Gen}(k,s)$.

Proof of Theorem 1.1. Begin with defining $T_{i,0} := T_i$ and $f_0 := f$ where $T_{i,0} \in \Pi\Sigma\wedge$; $\sum_i T_{i,0} = f_0$, and f_0 has size $\le s$. Assume $\deg(f) < d \le s$; we keep the parameter d separately, to help optimize the complexity later. In every recursive call we work with $\mathsf{Gen}(\cdot,\cdot)$ circuits.

As the input case, define $U_{i,0} := T_{i,0}$ and $V_{i,0} := P_{i,0} := Q_{i,0} := 1$. We will use the hitting set of product of sparse polynomials (refer section 2.2) to obtain a point $\boldsymbol{\alpha} = (a_1, \ldots, a_n) \in \mathbb{F}^n$

such that $U_{i,0}|_{\boldsymbol{x}=\boldsymbol{\alpha}} \neq 0$, for all $i \in [k]$. Eventually this evaluation point will help in maintaining the invertibility of $\Pi\Sigma\wedge$. Consider

$$g \;:=\; \prod_{i\in[k]} T_{i,0} \;=\; \prod_{i\in[k]} U_{i,0} \;=\; \prod_{i\in[\ell]} \sum_{j\in[n]} f_{ij}(x_j)\,,$$

where $f_{ij}(x_j)$ are univariate polynomials of degree at most d and $\ell \leq k \cdot s$. Note that $\deg g \leq d \cdot k \cdot s$ and g is computable by a $\Pi\Sigma\wedge$ circuit of size $O(s)$. Invoke Lemma 2.4 to obtain a hitting set $\mathcal{H}$, then evaluate g on every point of $\mathcal{H}$ to find an element $\boldsymbol{\alpha} \in \mathcal{H}$ such that $g(\boldsymbol{\alpha}) \neq 0$. We emphasise that in whitebox setting all $U_{i,0}$, are readily available for evaluation. Since, the size of the set is $\mathsf{poly}(s)$ and each evaluation takes $\mathsf{poly}(s)$ time, this preliminary step will add $\mathsf{poly}(s)$ time to the overall time complexity. Moreover, we obtain the $\boldsymbol{\alpha} \in \mathbb{F}^n$ which possess the required property.

To capture the non-zeroness, consider a 1-1 homomorphism $\Phi : \mathbb{F}[\boldsymbol{x}] \longrightarrow \mathbb{F}[\boldsymbol{x}, z]$ such that $x_i \mapsto z \cdot x_i + a_i$ where a_i is the i-th coordinate of $\boldsymbol{\alpha}$, obtained earlier. Invertibility implies that $f_0 = 0 \iff \Phi(f_0) = 0$. Now we proceed with the recursive algorithm which first reduces the identity testing from top-fanin k to $k-1$. Note: $k = 1$ is trivial.

First Step: Efficient reduction from k to $k-1$

By assumption, $\sum_{i=1}^{k} T_{i,0} \;=\; f_0$ and $T_{k,0} \neq 0$. Apply Φ both sides, then divide and derive:

$$\begin{aligned}
\sum_{i\in[k]} T_{i,0} \;=\; f_0 \quad \iff& \quad \sum_{i\in[k]} \Phi(T_{i,0}) \;=\; \Phi(f_0) \\
\iff& \quad \sum_{i\in[k-1]} \frac{\Phi(T_{i,0})}{\Phi(T_{k,0})} + 1 \;=\; \frac{\Phi(f_0)}{\Phi(T_{k,0})} \\
\implies& \quad \sum_{i\in[k-1]} \partial_z \left(\frac{\Phi(T_{i,0})}{\Phi(T_{k,0})}\right) \;=\; \partial_z \left(\frac{\Phi(f_0)}{\Phi(T_{k,0})}\right) \\
\iff& \quad \sum_{i=1}^{k-1} \frac{\Phi(T_{i,0})}{\Phi(T_{k,0})} \cdot \mathsf{dlog}_z \left(\frac{\Phi(T_{i,0})}{\Phi(T_{k,0})}\right) \;=\; \partial_z \left(\frac{\Phi(f_0)}{\Phi(T_{k,0})}\right) . \qquad (3.2)
\end{aligned}$$

Here onwards we say dlog to mean dlog_z, unless stated otherwise. Define the following:

- $\mathsf{R}_1 \;:=\; \mathbb{F}[z]/\langle z^d \rangle$. Note that, Equation 3.2 holds over $\mathsf{R}_1(\boldsymbol{x})$.

- $\widetilde{T}_{i,1} := \Phi(T_{i,0})/\Phi(T_{k,0}) \cdot \mathsf{dlog}(\Phi(T_{i,0})/\Phi(T_{k,0})), \forall\, i \in [k-1]$.

- $f_1 := \partial_z(\Phi(f_0)/\Phi(T_{k,0}))$, over $\mathsf{R}_1(\boldsymbol{x})$.

Definability of $T_{i,1}$ and f_1. It is easy to see that these are well-defined terms. Here, we emphasize that we do not exactly compute/store $\widetilde{T}_{i,1}$ as a fraction where the degree in z is $< d$; instead it is computed as an element in $\mathbb{F}(z, \boldsymbol{x})$, where z is a formal variable. Formally, we compute $T_{i,1} \in \mathbb{F}(z, \boldsymbol{x})$, such that $\widetilde{T}_{i,1} = T_{i,1}$, over $\mathsf{R}_1(\boldsymbol{x})$. We keep track of the degree of z in $T_{i,1}$. Thus, $\sum_{i\in[k-1]} T_{i,1} = f_1$, over $\mathsf{R}_1(\boldsymbol{x})$.

The 'iff' condition. To show that our one step of DiDI has reduced to the identity testing of $\mathsf{Gen}(k-1, \cdot)$, we need an $\iff$ condition. So far equality in Equation 3.2 over $\mathsf{R}_1(\boldsymbol{x})$ is *one-sided*. Note that $f_1 \neq 0$ implies $\mathsf{val}_z(f_1) < d =: d_1$. By assumption, $\Phi(T_{k,0})$ is invertible over $\mathsf{R}_1(\boldsymbol{x})$. Further, $f_1 = 0$, over $\mathsf{R}_1(\boldsymbol{x})$, which implies –

1. Either, $\Phi(f_0)/\Phi(T_{k,0})$ is z-free. Then $\Phi(f_0)/\Phi(T_{k,0}) \in \mathbb{F}(\boldsymbol{x})$, which further implies it is in $\mathbb{F}$, because of the map Φ (z-free implies $\boldsymbol{x}$-free, by substituting $z = 0$). Also, note that $f_0, T_{k,0} \neq 0$ implies $\Phi(f_0)/\Phi(T_{k,0})$ is a *nonzero* element in $\mathbb{F}$. Thus, it suffices to check whether $\Phi(f_0)|_{z=0}$ is non-zero or not.

2. Or, $\partial_z(\Phi(f_0)/\Phi(T_{k,0})) = z^{d_1} \cdot p$ where $p \in \mathbb{F}(z, \boldsymbol{x})$ s.t. $\mathsf{val}_z(p) \geq 0$. By simple power series expansion, one can show that $p \in \mathbb{F}(x)[[z]]$.

   Lemma 3.3 (Valuation)**.** *Consider $f \in \mathbb{F}(\boldsymbol{x}, y)$ such that $\mathsf{val}_y(f) \geq 0$. Then, $f \in \mathbb{F}(\boldsymbol{x})[[y]] \bigcap \mathbb{F}(\boldsymbol{x}, y)$.*

   Proof Sketch. Let $f = g/h$, where $g, h \in \mathbb{F}[\boldsymbol{x}, y]$. Now, $\mathsf{val}_y(f) \geq 0$, implies $\mathsf{val}_y(g) \geq \mathsf{val}_y(h)$. Let $\mathsf{val}_y(g) = d_1$ and $\mathsf{val}_y(h) = d_2$, where $d_1 \geq d_2 \geq 0$. Write $g = y^{d_1} \cdot \tilde{g}$ and $h = y^{d_2} \cdot \tilde{h}$. Write, $\tilde{h} = h_0 + h_1\, y + h_2\, y^2 + \ldots + h_d\, y^d$, for some d. Note that $h_0 \neq 0$. Thus,

   $$\begin{aligned} f &= y^{d_1-d_2} \cdot \tilde{g}/(h_0 + h_1\, y + \ldots + h_d\, y^d) \\ &= y^{d_1-d_2} \cdot (\tilde{g}/h_0) \cdot (1 + (h_1/h_0)\, y + \ldots + (h_d/h_0)\, y^d)^{-1} \in \mathbb{F}(\boldsymbol{x})[[y]]\,. \end{aligned}$$

   The last conclusion follows by the inverse identity in the power-series ring. □

   Hence, $\Phi(f_0)/\Phi(T_{k,0}) = z^{d_1+1} \cdot q$ where $q \in F(\boldsymbol{x})[[z]]$, i.e.

   $$\Phi(f_0)/\Phi(T_{k,0}) \in \langle z^{d_1+1} \rangle_{\mathbb{F}(\boldsymbol{x})[[z]]} \implies \mathsf{val}_z(\Phi(f_0)) \geq d+1,$$

   a contradiction.

Conversely, it is obvious that $f_0 = 0$ implies $f_1 = 0$. Thus, we have proved the following

$$\sum_{i\in[k]} T_{i,0} \neq 0 \;\text{ over } \mathbb{F}[\boldsymbol{x}] \iff \sum_{i\in[k-1]} T_{i,1} \neq 0 \;\text{ over } \mathsf{R}_1(\boldsymbol{x}),\;\text{ or },\; 0 \neq \Phi(f_0)|_{z=0} \in \mathbb{F}\,.$$

Eventually, we show that $T_{i,1} \in (\Pi\Sigma\wedge/\Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$, over $\mathsf{R}_1(\boldsymbol{x})$, with polynomial blowup in size (Claim 3.6). So, the above circuit is in $\mathsf{Gen}(k-1, \cdot)$, over $\mathsf{R}_1(\boldsymbol{x})$, which we recurse on to finally give the identity testing. The subsequent steps will be a bit more tricky:

Induction step

Assume that we are in the j-th step ($j \geq 1$). Our induction hypothesis assumes –

1. $\sum_{i \in [k-j]} T_{i,j} = f_j$, over $\mathsf{R}_j(\boldsymbol{x})$, where $\mathsf{R}_j := \mathbb{F}[z]/\langle z^{d_j} \rangle$ for $d_j < d$, and $T_{i,j} \neq 0$.

2. $\mathsf{val}_z(T_{i,j}) \geq 0, \forall i \in [k-j]$.

3. Non-zero preserving iff condition

$$\begin{aligned} f \neq 0, \text{ over } \mathbb{F}[\boldsymbol{x}] \iff & f_j \neq 0, \text{ over } \mathsf{R}_j(\boldsymbol{x}), \\ & \text{or } \bigvee_{i=0}^{j-1} ((f_i/T_{k-i,i})|_{z=0} \neq 0, \text{ over } \mathbb{F}(\boldsymbol{x})) \end{aligned}$$

4. Here, $T_{i,j} =: \left(U_{i,j}/V_{i,j}\right) \cdot \left(P_{i,j}/Q_{i,j}\right)$, where $U_{i,j}, V_{i,j} \in \Pi\Sigma\wedge$, and $P_{i,j}, Q_{i,j} \in \Sigma\wedge\Sigma\wedge$, each in $\mathsf{R}_j[\boldsymbol{x}]$. Think of them being computed as $\mathbb{F}(z, \boldsymbol{x})$, with the degrees being tracked. Wlog, assume that $\mathsf{val}_z(T_{k-j,j})$ is the minimal among all $T_{i,j}$'s.

5. $U_{i,j}|_{z=0}, V_{i,j}|_{z=0} \in \mathbb{F}\backslash\{0\}$.

We follow as before without applying homomorphism any further. Note that the 'or condition' in the hypothesis 3 is similar to the $j = 0$ case except that there is no Φ: this is because $\Phi(f_0)|_{z=0} \neq 0 \iff \Phi(f_0/T_{k,0})|_{z=0} \neq 0$. This condition just separates the derivative from the constant-term.

Efficient reduction from $k - j$ **to** $k - j - 1$**.** Let $\mathsf{val}_z(T_{i,j}) =: v_{i,j}$, for all $i \in [k-j]$. Note that

$$\min_i \mathsf{val}_z(T_{i,j}) = \min_i \mathsf{val}_z(P_{i,j}/Q_{i,j}) = v_{k-j,j}$$

since $\mathsf{val}_z(U_{i,j}) = \mathsf{val}_z(V_{i,j}) = 0$ (else we reorder). We remark that $0 \leq v_{i,j} < d_j$ for all i's in j-th step; upper-bound is strict, since otherwise $T_{i,j} = 0$ over $\mathsf{R}_j(x)$.

Similar to the first step, we divide with $T_{k-j,j}$ which has min val and then derive:

$$\begin{aligned} \sum_{i \in [k-j]} T_{i,j} \;=\; f_j \iff & \sum_{i \in [k-j-1]} T_{i,j}/T_{k-j,j} + 1 \;=\; f_j/T_{k-j,j} \\ \implies & \sum_{i \in [k-j-1]} \partial_z(T_{i,j}/T_{k-j,j}) \;=\; \partial_z(f_j/T_{k-j,j}) \\ \iff & \sum_{i=1}^{k-j-1} T_{i,j}/T_{k-j,j} \cdot \mathsf{dlog}(T_{i,j}/T_{k-j,j}) \;=\; \partial_z(f_j/T_{k-j,j}) \end{aligned} \tag{3.4}$$

Define the following:

- $\mathsf{R}_{j+1} := \mathbb{F}[z]/\langle z^{d_{j+1}} \rangle$, where $d_{j+1} := d_j - v_{k-j,j} - 1$.

- $\widetilde{T}_{i,j+1} := T_{i,j}/T_{k-j,j} \cdot \mathsf{dlog}(T_{i,j}/T_{k-j,j}), \forall\, i \in [k-j-1]$.

- $f_{j+1} := \partial_z(f_j/T_{k-j,j})$, over $\mathsf{R}_{j+1}(\boldsymbol{x})$.

We emphasize on the fact again that we do not exactly compute $\widetilde{T}_{i,j+1} \bmod z^{d_{j+1}}$; instead it is computed as a fraction in $\mathbb{F}(z,\boldsymbol{x})$, with formal z. Formally, we compute $T_{i,j+1} \in \mathbb{F}(z,\boldsymbol{x})$, such that $\widetilde{T}_{i,j+1} = T_{i,j+1}$, over $\mathsf{R}_{j+1}(\boldsymbol{x})$. We keep track of the degree of z in $T_{i,j+1}$. Next, we will show that all the inductive hypotheses assumed hold in the j^{th} step as well.

Hypothesis (1): Definability of $T_{i,j+1}$ and f_{j+1}. By the minimal valuation assumption, it follows that $\mathsf{val}(f_j) \geq v_{k-j,j}$, and thus $\widetilde{T}_{i,j+1}$ and f_{j+1} are all well-defined over $\mathsf{R}_{j+1}(\boldsymbol{x})$. Note that, Equation 3.4 holds over $\mathsf{R}_{j+1}(\boldsymbol{x})$ as $d_{j+1} < d_j$ (because, whatever identity holds true mod z^{d_j} must hold mod $z^{d_{j+1}}$ as well). Hence, we must have $\sum_{i=1}^{k-j-1} \widetilde{T}_{i,j+1} = f_{j+1}$, over $\mathsf{R}_{j+1}(\boldsymbol{x})$ thus proving the induction hypothesis (1).

Hypothesis (2): Positivity of Valuation. Since we divide by the min val, by definition we immediately get $\mathsf{val}_z(T_{i,j+1}) \geq 0$ proving the hypothesis. Further, we claim that min val computation in DiDI is easy. For this, recall from the definition of valuation

$$\min_i \mathsf{val}_z(P_{i,j}/Q_{i,j}) = \min_i(\mathsf{val}_z(P_{i,j}) - \mathsf{val}_z(P_{i,j})).$$

Therefore, for min val we compute $\mathsf{val}_z(P_{i,j})$ and $\mathsf{val}_z(Q_{i,j})$ for all $i \in [k-j]$.

Here is an important lemma which shows that coefficient of y^e of a polynomial $f(\boldsymbol{x},y) \in \mathbb{F}[\boldsymbol{x},y]$, computed by a $\Sigma\wedge\Sigma\wedge$ circuit, can be computed by a small $\Sigma\wedge\Sigma\wedge$ circuit.

Lemma 3.5 (Coefficient extraction). *Let $f(\boldsymbol{x},y) \in \mathbb{F}[y][\boldsymbol{x}]$ be computed by a $\Sigma\wedge\Sigma\wedge$ circuit of size s and degree d. Then, $\mathsf{coef}_{y^e}(f) \in \mathbb{F}[\boldsymbol{x}]$ can be computed by a small $\Sigma\wedge\Sigma\wedge$ circuit of size $O(sd)$, over $\mathbb{F}[\boldsymbol{x}]$.*

Proof Sketch. Let, $f = \sum_i \alpha_i \cdot g_i^{e_i}$. Of course, $e_i \leq s$ and $\deg_y(f) \leq d$. Thus, write $f = \sum_{i=0}^{d} f_i \cdot y^i$, where $f_i \in \mathbb{F}[\boldsymbol{x}]$. We can interpolate on $d+1$-many distinct points $y \in \mathbb{F}$ and conclude that f_i has a $\Sigma\wedge\Sigma\wedge$ circuit of size at most $O(sd)$. □

Using Lemma 3.5 we known $\mathsf{coef}_{z^e}(P_{i,j})$ and $\mathsf{coef}_{z^e}(Q_{i,j})$ are in $\Sigma\wedge\Sigma\wedge$ over $F[\boldsymbol{x}]$. We can keep track of z degree and thus interpolate to find the minimum $e < d_j$ such that the computed coefficients are $\neq 0$, which gives the respective val.

Hypothesis (3): The 'iff' condition. The above Equation 3.4 pioneers to reduce from $k-j$-summands to $k-j-1$. But we want a $\iff$ condition to efficiently reduce the identity testing. If $f_{j+1} \neq 0$, then $\mathsf{val}_z(f_{j+1}) < d_{j+1}$. Further, $f_{j+1} = 0$, over $\mathsf{R}_{j+1}(\boldsymbol{x})$ implies–

1. Either, $f_j/T_{k-j,j}$ is z-free. This implies it is in $\mathbb{F}(\boldsymbol{x})$. Now, if indeed $f_0 \neq 0$, then the computed $T_{i,j}$ as well as f_j must be non-zero over $\mathbb{F}(z,\boldsymbol{x})$, by induction hypothesis (as they are non-zero over $\mathsf{R}_j(\boldsymbol{x})$). However,

$$\left(\frac{T_{i,j}}{T_{k-j,j}}\right)\Big|_{z=0} = \left(\frac{U_{i,j} \cdot V_{k-j,j}}{U_{k-j,j} \cdot V_{i,j}}\right)\Big|_{z=0} \cdot \left(\frac{P_{i,j} \cdot Q_{k-j,j}}{P_{k-j,j} \cdot Q_{i,j}}\right)\Big|_{z=0}$$

$$\in \mathbb{F} \cdot \left(\frac{\Sigma\wedge\Sigma\wedge}{\Sigma\wedge\Sigma\wedge} \right).$$

Thus,

$$\frac{f_j}{T_{k-j,j}} \in \sum \mathbb{F} \cdot \left(\frac{\Sigma\wedge\Sigma\wedge}{\Sigma\wedge\Sigma\wedge} \right) \in \left(\frac{\Sigma\wedge\Sigma\wedge}{\Sigma\wedge\Sigma\wedge} \right).$$

Here we crucially use that $\Sigma\wedge\Sigma\wedge$ is closed under multiplication (Lemma 2.11). Thus, this identity testing can be done in poly-time (Lemma 2.9). For, detailed time-complexity and calculations, see Claim 3.6 and its subsequent paragraph.

2. Or, $\partial_z(f_j/T_{k-j,j}) = z^{d_{j+1}} \cdot p$, where $p \in \mathbb{F}(z, \boldsymbol{x})$ s.t. $\mathsf{val}_z(p) \geq 0$. By a simple power series expansion, one concludes that $p \in \mathbb{F}(\boldsymbol{x})[[z]]$ (Lemma 3.3). Hence, one concludes that

$$\frac{f_j}{T_{k-j,j}} \in \left\langle z^{d_{j+1}+1} \right\rangle_{\mathbb{F}(\boldsymbol{x})[[z]]} \implies \mathsf{val}_z(f_j) \geq d_j,$$

i.e. $f_j = 0$, over $\mathsf{R}_j(\boldsymbol{x})$.

Conversely, $f_j = 0$, over $\mathsf{R}_j(\boldsymbol{x})$, implies

$$\mathsf{val}_z(f_j) \geq d_j \implies \mathsf{val}_z \left(\partial_z \left(\frac{f_j}{T_{k-j,j}} \right) \right) \geq d_j - v_{k-j,j} - 1$$
$$\implies f_{j+1} = 0, \text{ over } \mathsf{R}_{j+1}(\boldsymbol{x}).$$

Thus, we have proved that $\sum_{i\in[k-j]} T_{i,j} \neq 0$ over $\mathsf{R}_j(\boldsymbol{x})$ iff

$$\sum_{i\in[k-j-1]} T_{i,j+1} \neq 0 \text{ over } \mathsf{R}_{j+1}(\boldsymbol{x}) \text{ , or , } 0 \neq \left(\frac{f_j}{T_{k-j,j}} \right) \Big|_{z=0} \in \mathbb{F}(\boldsymbol{x}) \,.$$

Therefore induction hypothesis (3) holds.

Hypothesis (4): Size analysis. We will show that $T_{i,j+1} \in (\Pi\Sigma\wedge / \Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge / \Sigma\wedge\Sigma\wedge)$, over $\mathsf{R}_{j+1}(\boldsymbol{x})$, with only polynomial blowup in size. Let $\mathsf{size}(T_{i,j}) \leq s_j$, for $i \in [k-j]$, and $j \in [k]$. Note that, by assumption, $s_0 \leq s$.

Claim 3.6 (Final size). *$T_{1,k-1} \in (\Pi\Sigma\wedge / \Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge / \Sigma\wedge\Sigma\wedge)$ of size $s^{O(k7^k)}$, over $\mathsf{R}_{k-1}(\boldsymbol{x})$.*

Proof. Steps $j = 0$ and $j > 0$ are slightly different because of the Φ. However the main idea of using power-series is the same which eventually shows that $\mathsf{dlog}(\Sigma\wedge) \in \Sigma\wedge\Sigma\wedge$.

We first deal with $j = 0$. Let $A - z \cdot B = \Phi(g) \in \Sigma\wedge$, for some $A \in \mathbb{F}$ and $B \in \mathsf{R}_1[\boldsymbol{x}]$. Note that $A \neq 0$ because of the map Ψ. Further, $\mathsf{size}(B) \leq O(d \cdot \mathsf{size}(g))$, as a single monomial of the form x^e can produce $d + 1$-many monomials. Over $\mathsf{R}_1(\boldsymbol{x})$,

$$\mathsf{dlog}(\Phi(g)) = -\frac{\partial_z(B \cdot z)}{A(1 - \frac{B}{A} \cdot z)} = -\frac{\partial_z(B \cdot z)}{A} \cdot \sum_{i=0}^{d_1 - 1} \left(\frac{B}{A} \right)^i \cdot z^i \,. \tag{3.7}$$

B^i has a trivial $\wedge\Sigma\wedge$-circuit of size $O(d\cdot\mathsf{size}(g))$. Also, $\partial_z(B\cdot z)$ has a $\Sigma\wedge$-circuit of size at most $O(d\cdot\mathsf{size}(g))$. Using waring identity (Lemma 2.10), we get that each $\partial_z(B\cdot z)\cdot(B/A)^i\cdot z^i$ has size $O(i\cdot d\cdot\mathsf{size}(g))$, over $\mathsf{R}_1(\boldsymbol{x})$. Summing over $i\in[d_1-1]$, the overall size is at most $O(d_1^2\cdot d\cdot\mathsf{size}(g))=O(d^3\cdot\mathsf{size}(g))$, as $d_0=d_1=d$.

For the j-th step, we emphasize that the degree could be larger than d. Assume that syntactic degree of denominator and numerator of $T_{i,j}$ (each in $\mathbb{F}[\boldsymbol{x},z]$) are bounded by D_j (it is *not* d_j as seen above; this is to save on the trouble of mod-computation at each step). Of course, $D_0<d\le s$.

For $j>0$, the above summation in Equation 3.7 is over $\mathsf{R}_j(\boldsymbol{x})$. However the degree could be D_j (possibly more than d_j) of the corresponding A and B. Thus, the overall size after the power-series expansion would be $O(D_j^2\cdot d\cdot\mathsf{size}(g))$.

Using Lemma 3.8, we can show that $\mathsf{dlog}(P_{i,j})\in\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$ (similarly for $Q_{i,j}$), of size $O(D_j^2\cdot s_j)$. Also $\mathsf{dlog}(U_{i,j}\cdot V_{k-j,j})\in\sum\mathsf{dlog}(\Sigma\wedge)$, i.e. sum of action of dlog on $\Sigma\wedge$ (since dlog linearizes product); and it can be computed by the above formulation. Thus, $\mathsf{dlog}(T_{i,j}/T_{k-j,j})$ is a sum of 4-many $\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$ of size at most $O(D_j^2\,s_j)$ and 1-many $\Sigma\wedge\Sigma\wedge$ of size $O(D_j^2 d_j s_j)$ (from the above power-series computation) [Note: we summed up the $\Sigma\wedge\Sigma\wedge$-expressions from $\mathsf{dlog}(\Sigma\wedge)$ together]. Additionally the syntactic degree of each denominator and numerator (of the $\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$) is $O(D_j)$. We rewrite the 4 expressions (each of $\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$) and express it as a single $\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$ using waring identity (Lemma 2.11), with the size blowup of $O(D_j^{12}\,s_j^4)$; here the syntatic degree blowsup to $O(D_j)$. Finally we add the remaining $\Sigma\wedge\Sigma\wedge$ circuit (of size $O(D_j^3 s_j)$ and degree $O(dD_j)$) to get $O(s_j^5 D_j^{16} d)$. To bound this, we need to understand the degree bound D_j.

Finally we need to multiply $T_{i,j}/T_{k-j,j}\in(\Pi\Sigma\wedge/\Pi\Sigma\wedge)\cdot(\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$ where each $\Sigma\wedge\Sigma\wedge$ is a product of two $\Sigma\wedge\Sigma\wedge$ expression of size s_j and syntactic degree D_j; clubbed together owing a blowup of $O(D_j\cdot s_j^2)$. Hence multiplying it with $\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$ expression obtained from dlog computation above gives size blowup of $s_{j+1}=s^7\cdot D_j^{O(1)}\cdot d$.

Computing $T_{i,j}/T_{k-j,j}$ increases the syntactic degree 'slowly'; which is much less than the size blowup. As mentioned before, the deg-blowup in dlog-computation is $O(dD_j)$ and in the clearing of four expressions, it is just $O(D_j)$. Thus, $D_{j+1}=O(dD_j)\implies D_j=d^{O(j)}$.

The recursion on the size is $s_{j+1}=s_j^7\cdot d^{O(j)}$. Using $d\le s$ we deduce, $s_j=(sd)^{O(j\cdot 7^j)}$. In particular, s_{k-1}, size after $k-1$ steps is $s^{O(k\cdot 7^k)}$. This computation quantitatively establishes induction hypothesis (4). □

Hypothesis (5): Invertibility of $\Pi\Sigma\wedge$-circuits. For invertibility, we want to emphasise that the dlog compuation plays a crucial role here. In the following lemma we claim that the action $\mathsf{dlog}(\Sigma\wedge\Sigma\wedge)\in\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge$, is of poly-size.

Lemma 3.8 (Differentiation). *Let $f(\boldsymbol{x},y)\in\mathbb{F}[y][\boldsymbol{x}]$ be computed by a $\Sigma\wedge\Sigma\wedge$ circuit of size s and degree d. Then, $\partial_y(f)$ can be computed by a small $\Sigma\wedge\Sigma\wedge$ circuit of size $O(sd^2)$, over $\mathbb{F}[y][\boldsymbol{x}]$.*

Proof Sketch. Lemma 3.5 shows that each f_e has $O(sd)$ size circuit where $f=\sum_e f_e\,y^e$. Doing this for each $e \in [0, d]$ gives a blowup of $O(sd^2)$. $\square$

Similarly consider the action on $\Pi\Sigma\wedge$. We know dlog distributes the product additively, so it suffices to work with $\mathrm{dlog}(\Sigma\wedge)$; and earlier in Claim 3.6 we saw that $\mathrm{dlog}(\Sigma\wedge) \in \Sigma\wedge\Sigma\wedge$ of poly-size. Assuming these, we simplify

$$\frac{T_{i,j}}{T_{k-j,j}} = \frac{U_{i,j} \cdot V_{k-j,j}}{V_{i,j} \cdot U_{k-j,j}} \cdot \frac{P_{i,j} \cdot Q_{k-j,j}}{Q_{i,j} \cdot P_{k-j,j}},$$

and its dlog. Thus, using Equation 3.4, $U_{i,(j+1)}$ grows to $U_{i,j} \cdot V_{k-j,j}$ (and similarly $V_{i,(j+1)}$). This also means: $U_{i,(j+1)}|_{z=0} \in \mathbb{F} \setminus \{0\}$ and thereby proving the hypothesis.

Final time complexity

The above proof actually shows that $T_{1,k-1}$ is in $\mathsf{Gen}(1, s^{O(k \cdot 7^k)})$ over $\mathsf{R}_{k-1}(\boldsymbol{x})$; and that the degree bound on z (over $\mathbb{F}[z, \boldsymbol{x}]$, keeping denominator and numerator 'in place') is $D_{k-1} = d^{O(k)}$. We cannot directly use the identity testing algorithms of the constituent simpler models due to $\mathsf{R}_{k-1}(\boldsymbol{x})$. Moreover, using hypothesis (2) and Lemma 3.3 we know that $T_{1,k-1} \in \mathbb{F}(\boldsymbol{x})[[z]]$ and it suffices to do identity testing on the first term of the powerseries: $T_{1,k-1}|_{z=0}$ over $\mathbb{F}(\boldsymbol{x})$. Note that, hypothesis (5) guarantees that $\Pi\Sigma\wedge$ part remains non-zero on $z = 0$ evaluation, however, $\Sigma\wedge\Sigma\wedge\ /\Sigma\wedge\Sigma\wedge$ may be undefined. For this, we keep track of z degree of numerator and denominator, which will be polynomially bounded as seen in the discussion above. We can easily interpolate and cancel the z power to make it work. Basically this shows that to test $T_{1,k-1}$ we need to test $z^e \cdot \Sigma\wedge\Sigma\wedge$ over $\mathbb{F}[\boldsymbol{x}]$ where $e \geq 0$ due to positive valuation. Whitebox PIT of $\Sigma\wedge\Sigma\wedge$ is in poly-time using Lemma 2.9, and testing z^e is possible using Lemma 2.1 with appropriate degree bound. The proof above is constructive: we calculate $U_{i,j+1}$ (and other terms) from $U_{i,j}$ explicitly. Gluing everything together we conclude this part can be done in $s^{O(k7^k)}$ time.

What remains is to test the $z = 0$-part of induction hypothesis (3); it could *short-circuit* the recursion much before $j = k-1$. As we mentioned before, in this case, we need to do a PIT on $\Sigma\wedge\Sigma\wedge$ only. At the j-th step, when we substitute $z = 0$, the size of each $T_{i,j}$ can be at most s_j (by definition). We need to do PIT on a simpler model: $\sum^{[k-j]} \mathbb{F} \cdot (\Sigma\wedge\Sigma\wedge\ /\Sigma\wedge\Sigma\wedge\ )$. We can clear out and express this as a single $\Sigma\wedge\Sigma\wedge\ /\Sigma\wedge\Sigma\wedge$ expression; with a size blowup of $s_j^{O(k-j)} \leq (sd)^{O(j(k-j)7^j)}$. Since this case could short-circuit the recursion, to bound the final time complexity, we need to consider the j which maximizes the exponent.

Lemma 3.9. *Let $k \in \mathbb{N}$, and $h(x) := x(k-x)7^x$. Then, $\max_{i \in [k-1]} h(i) = h(k-1)$.*

Proof Sketch. Differentiate to get $h'(x) = (k-x)7^x - x7^x + x(k-x)(\log 7)7^x = 7^x \cdot [x^2(-\log 7) + x(k \log 7 - 2) + k]$. It vanishes at

$$x = \left(\frac{k}{2} - \frac{1}{\log 7}\right) + \sqrt{\left(\frac{k}{2} - \frac{1}{\log 7}\right)^2 - \frac{k}{\log 7}}\,.$$

Thus, h is maximized at the integer $x = k - 1$. $\square$

Therefore, $\max_{j \in [k-1]} j(k-j)7^j = (k-1)7^{k-1}$. Finally, use Lemma 2.9 for the base-case whitebox PIT. Thus, the final time complexity is $s^{O(k \cdot 7^k)}$.

Here we also remark that in $z = 0$ substitution $\Sigma\wedge\Sigma\wedge / \Sigma\wedge\Sigma\wedge$ may be undefined. However, we keep track of z degree of numerator and denominator, which will be polynomially bounded as seen in the discussion above. We can easily interpolate and cancel the z power to make it work.

Bit complexity. It is routine to show that the bit-complexity is really what we claim. Initially, the given circuit has bit-complexity s. The main blowup happens due to the dlog-computation which is a poly-size blowup. We also remark that while using Lemma 2.11 (using Lemma 2.10), we *may* need to go to a field extension of at most $s^{O(k)}$ (because of the $\varepsilon(i)$ and correspondingly the constants $\gamma_{\varepsilon(2),\ldots,\varepsilon(k)}$, but they still are $s^{O(k)}$-bits). Also, Theorem 2.2 and Lemma 2.9 computations blowup bit-complexity polynomially. This concludes the proof. $\square$

Remark.
1. The above method does *not* give whitebox PIT (in poly-time) for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$, as we donot know poly-time whitebox PIT for $\Sigma\wedge\Sigma\Pi^{[\delta]}$. However, the above methods do show that whitebox-PIT for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ polynomially *reduces* to whitebox-PIT for $\Sigma\wedge\Sigma\Pi^{[\delta]}$.
2. DiDI-technique can be used to give whitebox PIT for the general bloated model $\mathsf{Gen}(k, s)$.
3. The above proof works when the characteristic is $\geq d$. This is because the nonzeroness remains *preserved* after derivation wrt z.

3.1 Algorithm

The whitebox PIT for Theorem 1.1, that is discussed in section 3, appears (below) as Algorithm 1.

Words of caution: Throughout the algorithm there are intermediate expressions to be stored compactly. Think of them as 'special' circuits in $\mathbf{x}$, but over the *function-field* $\mathbb{F}(z)$. Keep track of their degrees wrt z; and that of the sizes of their fractions represented in 'bloated' circuit form.

4 Blacbox PIT for Depth-4 Circuits

We will give the proof of Theorem 1.2 in this section. Before the details, we will state a few important definitions and lemmas from [ASSS16] to be referenced later.

Definition 4.1 (Transcendence Degree). *Polynomials* $T_1, \ldots, T_m$ *are called* algebraically dependent *if there exists a nonzero* annihilator A *s.t.* $A(T_1, \ldots, T_m) = 0$. Transcendence degree *is the size of the largest subset* $S \subseteq \{T_1, \ldots, T_m\}$ *that is algebraically independent. Then* S *is called a* transcendence basis.

Algorithm 1 Whitebox PIT Algorithm for $\Sigma^{[k]}\Pi\Sigma\wedge$-circuits

```
INPUT: $f = T_1 + \ldots + T_k \in \Sigma^{[k]}\Pi\Sigma\wedge$, a whitebox circuit of size $s$ over $\mathbb{F}[\boldsymbol{x}]$
OUTPUT: $0$, if $f \equiv 0$, and $1$, if non-zero.

1: Let $\Psi : \mathbb{F}[\boldsymbol{x}] \longrightarrow \mathbb{F}[z]$, be a sparse-PIT map, using [KS01] (Theorem 2.2). Apply it on $f$ and check whether $\Psi(f) \overset{?}{=} 0$. If non-zero, output 1
2: Obtain a point $\boldsymbol{\alpha} = (a_1, \ldots, a_n) \in \mathbb{F}^n$ from Hitting Set $\mathcal{H}$ of $\Pi\Sigma\wedge$ such that $T_i|_{\boldsymbol{x}=\boldsymbol{\alpha}} \neq 0$, for all $i \in [k]$. And define $\Phi : x_i \mapsto z \cdot x_i + a_i$. Check $\sum_{i\in[k-1]} \partial_z(\Phi(T_i)/\Phi(T_k)) \overset{?}{=} 0 \bmod z^{d_1}$ $(d_1 := s)$ as follows:
3: Consider each $T_{i,1} := \partial_z(\Phi(T_i)/\Phi(T_k))$ over $R_1(\boldsymbol{x})$, where $R_1 := \mathbb{F}[z]/\langle z^{d_1}\rangle$. Use dlog computation (Claim 3.6), to write each $T_{i,1}$ in a 'bloated' form as $(\Pi\Sigma\wedge/\Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$.
4: for $j \leftarrow 1$ to $k-1$ do
5:     Reduce the top-fanin at each step using 'Divide & Derive' technique. Assume that at $j$-th step, we have to check the identity: $\sum_{i\in[k-j]} T_{i,j} \overset{?}{=} 0$ over $R_j(\boldsymbol{x})$, where $R_j := \mathbb{F}[z]/\langle z^{d_j}\rangle$ , each $T_{i,j}$ has a $(\Pi\Sigma\wedge/\Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$ representation and therein each $\Pi\Sigma\wedge|_{z=0} \in \mathbb{F} \setminus \{0\}$.
6:     Compute $v_{k-j,j} := \min_i \mathrm{val}_z(T_{i,j})$; by reordering it is for $i = k-j$. To compute $v_{k-j,j}$, use coefficient extraction (Lemma 3.5) and $\Sigma\wedge\Sigma\wedge$-circuit PIT (Lemma 2.9).
7:     'Divide' by $T_{k-j,j}$ and check whether $\left.\left(\sum_{i\in[k-j-1]} (T_{i,j}/T_{k-j,j}) + 1\right)\right|_{z=0} \overset{?}{=} 0$. Note: this expression is in $(\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$. Use— (1) $\Pi\Sigma\wedge|_{z=0} \in \mathbb{F}$, and (2) *closure* of $\Sigma\wedge\Sigma\wedge$ under multiplication. Finally, do PIT on this by Lemma 2.9.
8:     If it is non-zero, output 1, otherwise 'Derive' wrt $z$ and 'Induct' on $\left(\sum_{i\in[k-j-1]} \partial_z(T_{i,j}/T_{k-j,j})\right) \overset{?}{=} 0$, over $R_{j+1}(\boldsymbol{x})$ where $R_{j+1} := \mathbb{F}[z]/\langle z^{d_j - v_{k-j,j} - 1}\rangle$.
9:     Again using dlog (Claim 3.6), show that $T_{i,j+1} := \partial_z(T_{i,j}/T_{k-j,j})$ has small $(\Pi\Sigma\wedge/\Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$-circuit over $R_{j+1}(\boldsymbol{x})$. So call the algorithm on $\sum_{i\in[k-j-1]} T_{i,j+1} \overset{?}{=} 0$.
10:    $j \leftarrow j+1$.
11: end for
12: At the end, $j = k-1$. Do PIT (Lemma 2.9) on the single $(\Pi\Sigma\wedge/\Pi\Sigma\wedge) \cdot (\Sigma\wedge\Sigma\wedge/\Sigma\wedge\Sigma\wedge)$ circuit, over $R_{k-1}(\boldsymbol{x})$. If it is zero, output 0 otherwise output 1.
```

Definition 4.2 (Faithful homomorphism). *A homomorphism $\Phi : \mathbb{F}[\boldsymbol{x}] \to \mathbb{F}[\boldsymbol{y}]$ is faithful for $\boldsymbol{T}$ if* $\mathsf{trdeg}_{\mathbb{F}}(\boldsymbol{T}) = \mathsf{trdeg}_{\mathbb{F}}(\Phi(\boldsymbol{T}))$.

The reason for interest in faithful maps is due its usefulness in preserving the identity as shown in the following fact.

Fact 4.3 (Theorem 2.4 [ASSS16]). *For any $C \in \mathbb{F}[y_1, \ldots, y_m]$, $C(\boldsymbol{T}) = 0 \iff C(\Phi(\boldsymbol{T})) = 0$.*

Here is an important criterion about the jacobian matrix which basically shows that it *preserves* algabraic independence.

Fact 4.4 (Jacobian criterion). *Let $\mathbf{f} \subset \mathbb{F}[\boldsymbol{x}]$ be a finite set of polynomials of degree at most d, and $\mathsf{trdeg}_{\mathbb{F}}(\mathbf{f}) \le r$. If $char(\mathbb{F}) = 0$, or $char(\mathbb{F}) > d^r$, then $\mathsf{trdeg}_{\mathbb{F}}(\mathbf{f}) = \mathsf{rk}_{\mathbb{F}(\boldsymbol{x})}\mathcal{J}_{\boldsymbol{x}}(\mathbf{f})$.*

Jacobian criterion together with faithful maps give a recipe to design a map which drastically reduces number of variables, if trdeg is small.

Lemma 4.5 (Lemma 2.7 [ASSS16]). *Let $\boldsymbol{T} \in \mathbb{F}[\boldsymbol{x}]$ be a finite set of polynomials of degree at most d and $\mathsf{trdeg}_{\mathbb{F}}(\boldsymbol{T}) \le r$, and char(F)=0 or $> d^r$. Let $\Psi' : \mathbb{F}[\boldsymbol{x}] \longrightarrow \mathbb{F}[z]$ such that $\mathsf{rk}_{\mathbb{F}(\boldsymbol{x})}\mathcal{J}_{\boldsymbol{x}}(\boldsymbol{T}) = \mathsf{rk}_{\mathbb{F}(z)}\Psi'(\mathcal{J}_{\boldsymbol{x}}(\boldsymbol{T}))$.*

Then, the map $\Phi : \mathbb{F}[\boldsymbol{x}] \longrightarrow \mathbb{F}[z, t, \boldsymbol{y}]$, such that $x_i \mapsto (\sum_{j \in [r]} y_j t^{ij}) + \Psi'(x_i)$, is a faithful homomorphism for $\boldsymbol{T}$.

In the next section we will use these tools to prove Theorem 1.2(b). The proof and calculations for Theorem 1.2(a) are very similar.

4.1 PIT for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$

We solve the PIT for a more general model than $\Sigma^{[k]}\Pi\Sigma\Pi$ by solving the following problem.

Problem 4.6. *Let $\{T_i \,|\, i \in [m]\}$ be $\Pi\Sigma\Pi^{[\delta]}$ circuits of (syntactic) degree at most d and size s. Let the transcendence degree of T_i's, $\mathsf{trdeg}_{\mathbb{F}}(T_1, \ldots, T_m) = k \ll s$. Further, $C(x_1, \ldots, x_m)$ be a circuit of $(\mathsf{size} + \deg) < s'$. Design a blackbox-PIT algorithm for $C(T_1, \ldots, T_m)$.*

Trivially, $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ is a very special case of the above setting. Let $\boldsymbol{T} := \{T_1, \ldots, T_m\}$. Let $\boldsymbol{T}_k := \{T_1, \ldots, T_k\}$ be a transcendence basis. For $T_i = \prod_j g_{ij}$, we denote the set $L(T_i) := \{g_{ij} \mid j\}$.

We want to find an explicit homomorphism $\Psi : \mathbb{F}[\boldsymbol{x}] \to \mathbb{F}[\boldsymbol{x}, z]$ s.t. $\Psi(\mathcal{J}_{\boldsymbol{x}}(\boldsymbol{T}))$ is of a 'nice' form. In the image we fix $\boldsymbol{x}$ suitably, to get a composed map $\Psi' : \mathbb{F}[\boldsymbol{x}] \longrightarrow \mathbb{F}[z]$ s.t. $\mathsf{rk}_{\mathbb{F}(\boldsymbol{x})}\mathcal{J}_{\boldsymbol{x}}(\boldsymbol{T}) = \mathsf{rk}_{\mathbb{F}(z)}\Psi'(\mathcal{J}_{\boldsymbol{x}}(\boldsymbol{T}))$. Then, we can extend this map to $\Phi : \mathbb{F}[\boldsymbol{x}] \longrightarrow \mathbb{F}[z, \boldsymbol{y}, t]$ s.t. $x_i \mapsto (\sum_{j=1}^{k} y_j t^{ij}) + \Psi'(x_i)$, which is *faithful* Lemma 4.5. We show that the map Φ can be efficiently constructed using a scaling and shifting map (Ψ) which is eventually fixed by the hitting set (H' defining Ψ') of a $\Sigma\wedge\Sigma\Pi^{[\delta]}$ circuit. Overall, $\Phi(f)$ is a $k+2$-variate polynomial for which a trivial hitting set exists.

Wlog, $\mathcal{J}_{\boldsymbol{x}}(\boldsymbol{T})$ is full rank with respect to the variable set $\boldsymbol{x}_k = (x_1, \ldots, x_k)$. Thus, by assumption, $J_{\boldsymbol{x}_k}(\boldsymbol{T}_k) \neq 0$ (for notation, see section 2). We want to construct a Ψ s.t. $\Psi(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k))$ has an 'easier' PIT. We have the following identity [ASSS16, Eqn. 3.1], from the linearity of the determinant, and the simple observation that $\partial_x(T_i) = T_i \cdot \left(\sum_j \partial_x(g_{ij})/g_{ij}\right)$, where $T_i = \prod_j g_{ij}$:

$$J_{\boldsymbol{x}_k}(\boldsymbol{T}_k) \;=\; \sum_{g_1 \in L(T_1),\ldots,g_k \in L(T_k)} \left(\frac{T_1 \ldots T_k}{g_1 \cdots g_k}\right) \cdot J_{\boldsymbol{x}_k}(g_1, \ldots, g_k)\,. \tag{4.7}$$

The homomorphism Ψ. To ensure the invertibility of all $g \in \bigcup_i L(T_i)$ we proceed as in section 3. Consider

$$h := \prod_{i \in [k]} \prod_{g \in L(T_i)} g = \prod_{i \in [\ell]} g,$$

where $g \in \bigcup_i L(T_i)$ and $\ell \leq k \cdot s$. Note that $\deg h \leq d \cdot k \cdot s$ and h is computable by $\Pi\Sigma\Pi$ circuit of size $O(s)$. Lemma 2.4 gives the relevant hitting set $\mathcal{H} \subseteq \mathbb{F}^n$ which contains an evaluation point $\boldsymbol{\alpha} = (a_1, \ldots, a_n)$ such that $h(\boldsymbol{\alpha}) \neq 0$ implying $g(\boldsymbol{\alpha}) \neq 0$, for all $g \in \bigcup_i L(T_i)$. We emphasise that, unlike the previous case, here in the blackbox setting, we *do not* have individual access of g to verify for the correct $\boldsymbol{\alpha}$. Thus, we try out all $\boldsymbol{\alpha} \in \mathcal{H}$ to see whichever works. If the input polynomial f is non-zero, then one such $\boldsymbol{\alpha}$ must exist. This search adds a multiplicative blowup of $\mathsf{poly}(s)$, since the size of $\mathcal{H}$ is $\mathsf{poly}(s)$.

Fix an $\boldsymbol{\alpha} = (a_1, \cdots, a_n) \in \mathcal{H}$ and define $\Psi : \mathbb{F}[\boldsymbol{x}] \to \mathbb{F}[\boldsymbol{x}, z]$ as $x_i \mapsto z \cdot x_i + a_i$. Denote the ring $\mathsf{R}[\boldsymbol{x}]$ where $\mathsf{R} := \mathbb{F}[z]/\langle z^D \rangle$, and $D := k \cdot (d-1) + 1$. Being 1-1, Ψ is clearly a non-zero preserving map. Moreover,

Claim 4.8. $J_{\boldsymbol{x}_k}(\boldsymbol{T}_k) = 0 \iff \Psi(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k)) = 0$, *over* $\mathsf{R}[\boldsymbol{x}]$.

Proof. As $\deg(T_i) \leq d$, each entry of the matrix can be of degree at most $d-1$; therefore $\deg(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k)) \leq k(d-1) = D-1$. Thus, $\deg_z(\Psi(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k))) < D$. Hence, the conclusion. $\square$

Equation 4.7 implies that

$$\Psi(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k)) \;=\; \Psi(T_1 \cdots T_k) \cdot \sum_{g_1 \in L(T_1),\ldots,g_k \in L(T_k)} \frac{\Psi(J_{\boldsymbol{x}_k}(g_1, \ldots, g_k))}{\Psi(g_1 \cdots g_k)}\,. \tag{4.9}$$

As T_i has product fanin s, the top-fanin in the sum in Equation 4.9 can be at most s^k. Then define,

$$\widetilde{F} \;:=\; \sum_{g_1 \in L(T_1),\ldots,g_k \in L(T_k)} \frac{\Psi(J_{\boldsymbol{x}_k}(g_1, \ldots, g_k))}{\Psi(g_1 \ldots g_k)}\,, \quad \text{over } \mathsf{R}[\boldsymbol{x}]. \tag{4.10}$$

Well-definability of $\widetilde{F}$. Note that,

$$\Psi(g_i) \bmod z \neq 0 \implies 1/\Psi(g_1 \cdots g_k) \in \mathbb{F}[[\boldsymbol{x}, z]].$$

Thus, RHS is an element in $\mathbb{F}[[\boldsymbol{x}, z]]$ and taking mod z^D it is in $\mathsf{R}[\boldsymbol{x}]$. We remark that instead of minimally reducing mod z^D, we will work with an $F \in \mathbb{F}[z, \boldsymbol{x}]$ such that $F = \tilde{F}$ over $\mathsf{R}[\boldsymbol{x}]$.

Further, we ensure that the degree of z is polynomially bounded.

Claim 4.11. *Over* $\mathsf{R}[\boldsymbol{x}]$, $\Psi(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k)) = 0 \iff F = 0$.

Proof sketch. This follows from the invertibility of $\Psi(T_1 \cdots T_k)$ in $R[\boldsymbol{x}]$. $\square$

The hitting set H'. By $J_{\boldsymbol{x}_k}(\boldsymbol{T}_k) \neq 0$, and Claims 4.8-4.11, we have $F \neq 0$ over $\mathsf{R}[\boldsymbol{x}]$. We want to find $H' \subseteq \mathbb{F}^n$, s.t. $\Psi(J_{\boldsymbol{x}_k}(\boldsymbol{T}_k))|_{\boldsymbol{x}=\boldsymbol{\alpha}} \neq 0$, for some $\boldsymbol{\alpha} \in H'$ (which will ensure the rank-preservation). Towards this, we will show (below) that F has $s^{O(\delta k)}$-size $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit over $\mathsf{R}[\boldsymbol{x}]$. Next, Theorem 2.8 provides the hitting set H' in time $s^{O(\delta^2 k \log s)}$.

Claim 4.12 (Main size bound)**.** $F \in \mathsf{R}[\boldsymbol{x}]$ *has* $\Sigma\wedge\Sigma\Pi^{[\delta]}$*-circuit of size* $(s3^{\delta})^{O(k)}$.

The proof studies the two parts of Equation 4.10—

1. The numerator $\Psi(J_{\boldsymbol{x}_k}(g_1, \ldots, g_k))$ has $O(3^{\delta}2^k k! ks)$-size $\Sigma\wedge\Sigma\Pi^{[\delta-1]}$-circuit (see Lemma 4.15), and
2. $1/\Psi(g_1 \cdots g_k)$, for $g_i \in L(T_i)$ has $(s3^{\delta})^{O(k)}$-size $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit; both over $\mathsf{R}[\boldsymbol{x}]$ (see Lemma 4.16).

We need the following two claims to prove the numerator size bound.

Claim 4.13. *Let* $g_i \in L(T_i)$*, where* $T_i \in \Pi\Sigma\Pi^{[\delta]}$ *of size atmost* s*, then the polynomial* $J_{\boldsymbol{x}_k}(g_1, \ldots, g_k)$ *is computable by* $\Sigma^{[k!]}\Pi^{[k]}\Sigma\Pi^{[\delta-1]}$ *of size* $O(k!\, ks)$.

Proof Sketch. Each entry of the matrix has degree at most $\delta - 1$. Trivial expansion gives $k!$ top-fanin where each product (of fanin k) has size $\sum_i \mathsf{size}(g_i)$. As, $\mathsf{size}(T_i) \leq s$, trivially each $\mathsf{size}(g_i) \leq s$. Therefore, the total size is $k! \cdot \sum_i \mathsf{size}(g_i) = O(k!\, ks)$. $\square$

Claim 4.14. *Let* $g \in \Sigma\Pi^{\delta}$*, then* $\Psi(g) \in \Sigma\Pi^{\delta}$ *of size* $3^{\delta} \cdot \mathsf{size}(g)$ *(for* $n \gg \delta$*).*

Proof Sketch. Each monomial $\boldsymbol{x}^{\boldsymbol{e}}$ of degree δ, can produce $\prod_i (e_i + 1) \leq ((\sum_i e_i + n)/n)^n \leq (\delta/n + 1)^n$-many monomials, by AM-GM inequality as $\sum_i e_i \leq \delta$. As $\delta/n \to 0$, we have $(1 + \delta/n)^n \to e^{\delta}$. As $e < 3$, the upper bound follows. $\square$

Lemma 4.15 (Numerator size)**.** $\Psi(J_{\boldsymbol{x}_k}(g_1, \ldots, g_k))$ *is computable by* $\Sigma\wedge\Sigma\Pi^{[\delta-1]}$ *of size* $O(3^{\delta}\, 2^k k\, k! s) =: s_2$.

Proof. In Claim 4.13 we showed that $J_{\boldsymbol{x}_k}(g_1, \ldots, g_k) \in \Sigma^{[k!]}\Pi^{[k]}\Sigma\Pi^{[\delta-1]}$ of size $O(k! ks)$. Moreover, for a $g \in \Sigma\Pi^{[\delta-1]}$, we have $\Psi(g) \in \Sigma\Pi^{[\delta-1]}$ of size at most $3^{\delta} \cdot \mathsf{size}(g)$, over $\mathsf{R}[\boldsymbol{x}]$ due to Claim 4.14).

Combining these, one concludes that $\Psi(J_{\boldsymbol{x}_k}(g_1, \ldots, g_k)) \in \Sigma^{[k!]}\Pi^{[k]}\Sigma\Pi^{[\delta-1]}$, of size $O(3^{\delta}\, k! ks)$. We *convert* the Π-gate to $\wedge$ gate using waring identity (Lemma 2.10) which blowsup the size by a multiple of 2^{k-1}. Thus, $\Psi(J_{\boldsymbol{x}_k}(g_1, \ldots, g_k)) \in \Sigma\wedge\Sigma\Pi^{[\delta-1]}$ of size $O(3^{\delta}\, 2^k k\, k! s)$. $\square$

In the following lemma, using power series expansion of expressions like $1/(1-a\cdot z)$, we conclude that $1/\Psi(g)$ has a small $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit, which would further imply the same for $1/\Psi(g_1\cdots g_k)$.

Lemma 4.16 (Denominator size). *Let $g_i \in L(T_i)$. Then, $1/\Psi(g_1\cdots g_k)$ can be computed by a $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit of size $s_1 := (s3^\delta)^{O(k)}$, over $\mathsf{R}[\boldsymbol{x}]$.*

Proof. Let $g \in L(T_i)$ for some i. Assume, $\Psi(g) = A - z\cdot B$, for some $A \in \mathbb{F}$ and $B \in \mathsf{R}[\boldsymbol{x}]$ of degree δ, with $\mathsf{size}(B) \le 3^\delta \cdot s$, from Claim 4.14. Note that, over $\mathsf{R}[\boldsymbol{x}]$,

$$\frac{1}{\Psi(g)} \;=\; \frac{1}{A(1-\frac{B}{A}\cdot z)} \;=\; \frac{1}{A}\cdot\sum_{i=0}^{D-1}\left(\frac{B}{A}\right)^i\cdot z^i\,. \tag{4.17}$$

As, B^i has a trivial $\wedge\Sigma\Pi^{[\delta]}$-circuit (over $\mathsf{R}[\boldsymbol{x}]$) of size $\le 3^\delta\cdot s + i$; summing over $i\in[D-1]$, the overall size is at most $D\cdot 3^\delta\cdot s + O(D^2)$. As $D < k\cdot d$, we conclude that $1/\Psi(g)$ has $\Sigma\wedge\Sigma\Pi^{[\delta]}$ of size $\mathsf{poly}(s\cdot k\cdot d3^\delta)$, over $\mathsf{R}[\boldsymbol{x}]$. Multiplying k-many such products directly gives an upper bound of $(s\cdot 3^\delta)^{O(k)}$, using Lemma 2.11 (basically, waring identity). $\square$

Proof of Claim 4.12. Combining Lemmas 4.15-4.16, observe that $\Psi(J_{x_k}(\cdot)/\Psi(\cdot)$ has $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit of size at most $(s_1\cdot s_2)^2 = (s\cdot 3^\delta)^{O(k)}$, over $\mathsf{R}[\boldsymbol{x}]$, using Lemma 2.11. Summing up at most s^k many terms (by defn. of F), the size still remains $(s\cdot 3^\delta)^{O(k)}$. $\square$

Degree bound. As, syntactic degree of T_i are bounded by d, and Ψ maintain $\deg_{\boldsymbol{x}} = \deg_z$, we must have $\deg_z(\Psi(J_{x_k}(g_1,\ldots,g_k)) = \deg_{\boldsymbol{x}}(J_{x_k}(g_1,\ldots,g_k)) \le D-1$. Note that, Lemma 4.15 actually works over $\mathbb{F}[\boldsymbol{x},z]$ and thus there is no additional degree-blow up (in z). However, there is some degree blowup in Lemma 4.16, due to Equation 4.17.

Note that Equation 4.17 shows that over $\mathsf{R}[\boldsymbol{x}]$,

$$\frac{1}{\Psi(g)} = \left(\frac{1}{A^D}\right)\cdot\left(\sum_{i=0}^{D-1} A^{D-1-i}z^i\cdot B^i\right) =: \frac{p(\boldsymbol{x},z)}{q},$$

where $q = A^D$. We think of $p \in \mathbb{F}[\boldsymbol{x},z]$ and $q\in\mathbb{F}$. Note, $\deg_z(\Psi(g)) \le \delta$ implies $\deg_z(p) \le \deg_z((B\,z)^{D-1}) \le \delta\cdot(D-1)$.

Finally, denote $1/\Psi(g_1\cdots g_k) =: P_{g_1,\ldots,g_k}/Q_{g_1,\ldots,g_k}$, over $\mathsf{R}[\boldsymbol{x}]$. This is just multiplying k-many (p/q)'s; implying a degree blowup by a multiple of k. In particular – $\deg_z(P_{(\cdot)}) \le \delta\cdot k\cdot(D-1)$ Thus, in Equation 4.10, summing up s^k-many terms gives an expression (over $\mathsf{R}[\boldsymbol{x}]$):

$$F \;=\; \sum_{g_1\in L(T_1),\ldots,g_k\in L(T_k)} \Psi(J_{x_k}(g_1,\ldots,g_k))\cdot\left(\frac{P_{g_1,\ldots,g_k}}{Q_{g_1,\ldots,g_k}}\right) \;=:\; \frac{P(\boldsymbol{x},z)}{Q}\,.$$

Verify that $Q\in\mathbb{F}$. The degree of z also remains bounded by

$$\max_{g_i\in L(T_i),i\in[k]} \deg_z(P_{g_1,\ldots,g_k}) + \delta k \le \mathsf{poly}(s).$$

Using the degree bounds, we finally have $P \in \mathbb{F}[\boldsymbol{x}, z]$ as a $\Sigma\wedge\Sigma\Pi^{[\delta]}$-circuit (over $\mathbb{F}(z)$) of size $n^{O(\delta)}\,(s3^{\delta})^{O(k)} = 3^{O(\delta k)}s^{O(k+\delta)} =: s_3$.

We want to *construct* a set $H' \subseteq \mathbb{F}^n$ such that the action $P(H', z) \neq 0$. Using [For15] (Theorem 2.8), we conclude that it has $s^{O(\delta \log s_3)} = s^{O(\delta^2 k \log s)}$ size hitting set which is constructible in a similar time. Hence, the construction of Φ follows, making $\Phi(f)$ a $k+2$ variate polynomial. Finally, by the obvious degree bounds of $\boldsymbol{y}, z, t$ from the definition of Φ, we get the blackbox PIT algorithm with time-complexity $s^{O(\delta^2 k \log s)}$; finishing Theorem 1.2(b).

We could also give the final hitting set for the general problem.

Solution to Problem 4.6. We know that

$$C(T_1, \ldots, T_m) = 0 \iff E := \Phi(C(T_1, \ldots, T_m)) = 0.$$

Since, H' can be constructed in $s^{O(\delta^2\, k\, \log s)}$-time, it is trivial to find hitting set for $E|_{H'}$ (which is just a $k+2$-variate polynomial with the aformentioned degree bounds). The final hitting set for E can be constructed in $s'^{O(k)} \cdot s^{O(\delta^2\, k\, \log s)}$-time. □

Remark.
1. As Jacobian Criterion (Fact 4.4) holds when the characteristic is $> d^{\mathsf{trdeg}}$, it is easy to conclude that our theorem holds for all fields of char $> d^k$.

2. The above proof gives an efficient reduction from blackbox PIT for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ circuits to $\Sigma\wedge\Sigma\Pi^{[\delta]}$ circuits. In particular, a poly-time hitting set for $\Sigma\wedge\Sigma\Pi^{[\delta]}$ circuits would put PIT for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ in P.

3. Also, DiDI-technique (of Theorem Theorem 1.1) directly gives a blackbox algorithm, but the complexity is *exponentially* worse (in terms of k in the exponent) for its recursive blowups.

4.2 PIT for $\Sigma^{[k]}\Pi\Sigma\wedge$

As we remarked earlier, the proof of Theorem 1.2(a) is similar to the one we discussed in section 4.1. Here we sketch the proof, stating some relevant changes. Similar to Theorem 1.2(b), we generalize this theorem and prove for a much bigger class of polynomials.

Problem 4.18. *Let $\{T_i \,|\, i \in [m]\}$ be $\Pi\Sigma\wedge$ circuits of (syntactic) degree at most d and size s. Let the transcendence degree of T_i's, $\mathsf{trdeg}_{\mathbb{F}}(T_1, \ldots, T_m) =: k \ll s$. Further, $C(x_1, \ldots, x_m)$ be a circuit of size + degree $< s'$. Design a blackbox-PIT algorithm for $C(T_1, \ldots, T_m)$.*

It is trivial to see that $\Sigma^{[k]}\Pi\Sigma\wedge$ is a very special case of the above settings. We will use the same idea (& notation) as in Theorem 1.2(b), using the Jacobian technique. The main idea is to come up with Ψ map, and correspondingly the hitting set H'. If $g \in L(T_i)$, then $\text{size}(g) \leq O(dn)$. The D (and hence $R[\boldsymbol{x}]$) remains as before. Claims 4.8-4.11 hold similarly. We will construct the hitting set H' by showing that F has a small $\Sigma\wedge\Sigma\wedge$ circuit over $R[\boldsymbol{x}]$.

Note that, Claim 4.13 remains the same for $\Sigma\wedge\Sigma\wedge$ (implying the same size blowup). However, Claim 4.14, the size blowup is $O(d\,\mathsf{size}(g))$, because each monomial x^e can only produce $d+1$ many monomials. Therefore, similar to Lemma 4.16, one can show that $\Psi(J_{x_k}(g_1,\ldots,g_k)) \in \Sigma\wedge\Sigma\wedge$, of size $O(2^k k! kds)$. Similarly, the size in Lemma 4.15 can be replaced by $s^{O(k)}$. Therefore, we get (similar to Claim 4.12):

Claim 4.19. *$F \in R[\mathbf{x}]$ has $\Sigma\wedge\Sigma\wedge$ -circuit of size $s^{O(k)}$.*

Next, the degree bound also remains the same. Following the same footsteps, it is not hard to see that while degree bound on z remains $\mathsf{poly}(ksd)$. Therefore, $P \in \mathbb{F}[\mathbf{x}, z]$ has $\Sigma\wedge\Sigma\wedge$ -circuit of size $s^{O(k)}$.

We want to *construct* a set $H' \subseteq \mathbb{F}^n$ such that the action $P(H', z) \neq 0$. By Lemma 2.9, we conclude that it has $s^{O(k \log\log s)}$ size hitting set which is constructible in a similar time. Hence, the construction of map Φ and the theorem follows (from z-degree bound).

Solution to Problem 4.18. We know that

$$C(T_1,\ldots,T_m) = 0 \iff E := \Phi(C(T_1,\ldots,T_m)) = 0.$$

Since, H' can be constructed in $s^{O(k\log\log s)}$ time, it is trivial to find hitting set for $E|_{H'}$ (which is just a $k+2$-variate polynomial with the aforementioned degree bounds). The final hitting set for E can be constructed in $s'^{O(k)} \cdot s^{O(k\log\log s)}$ time. $\square$

5 Conclusion

This work introduces the powerful DiDI-technique and solves three open problems in PIT for depth-4 circuits, namely $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$ (blackbox) and $\Sigma^{[k]}\Pi\Sigma\wedge$ (both whitebox and blackbox). Here are some immediate questions of interest which require rigorous investigation.

1. Can the exponent in Theorem 1.1 be improved to $O(k)$? Currently, it is exponential in k.

2. Can we improve Theorem 1.2(b) to $s^{O(\log\log s)}$ (like in Theorem 1.2(a))?

3. Can we design a polynomial-time PIT for $\Sigma^{[k]}\Pi\Sigma\Pi^{[\delta]}$?

4. Design a polynomial time PIT for $\Sigma\wedge\Sigma\Pi^{[\delta]}$ circuits (i.e. unbounded top-fanin)?

5. Can we solve PIT for $\Sigma^{[k]}\Pi\Sigma\mathsf{M}_2$ circuits efficiently (polynomial/quasipolynomial-time), where $\Sigma\mathsf{M}_2$ denotes bivariate polynomials?

6. Can we design an efficient PIT for rational functions of the form $\Sigma\,(1/\Sigma\wedge\Sigma)$ or $\Sigma\,(1/\Sigma\Pi)$ (for *un*bounded top-fanin)?

References

[AFS$^+$18] Matthew Anderson, Michael A Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity testing and lower bounds for read-k oblivious algebraic branching programs. *ACM Transactions on Computation Theory (TOCT)*, 2018. Preliminary version in the IEEE 31st Computational Complexity Conference (CCC'16). 7

[AGKS15] Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM Journal on Computing*, 2015. 7

[Agr05] Manindra Agrawal. Proving lower bounds via pseudo-random generators. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*. Springer, 2005. 2

[AGS19] Manindra Agrawal, Sumanta Ghosh, and Nitin Saxena. Bootstrapping variables in algebraic circuits. *Proceedings of the National Academy of Sciences*, 2019. Preliminary version in Symposium on Theory of Computing, 2018 (STOC'18). 3

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of mathematics*, 2004. 2

[ALM$^+$98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM (JACM)*, 1998. 2

[And20] Robert Andrews. Algebraic Hardness Versus Randomness in Low Characteristic. In *35th Computational Complexity Conference (CCC 2020)*, 2020. 3

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM (JACM)*, 1998. Preliminary version in 33rd Annual Symposium on Foundations of Computer Science (FOCS'92). 2

[ASS13] Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth-Δ formulas. In *Proceedings of the* 45th *Annual ACM symposium on Theory of computing (STOC'13)*, pages 321–330, 2013. 5

[ASSS16] Manindra Agrawal, Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. Jacobian hits circuits: Hitting sets, lower bounds for depth-D occur-k formulas and depth-3 transcendence degree-k circuits. *SIAM Journal on Computing*, 2016. Preliminary version in 44th Symposium on Theory of Computing, 2018 (STOC'12). 4, 5, 7, 8, 20, 22

[AV08] Manindra Agrawal and V Vinay. Arithmetic Circuits: A Chasm at Depth Four. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 67–75. IEEE, 2008. 3

[BMS13] Malte Beecken, Johannes Mittmann, and Nitin Saxena. Algebraic independence and blackbox identity testing. *Information and Computation*, 2013. Preliminary version in 38^{th} International Colloquium on Automata, Languages and Programming (ICALP'11). 5, 6, 7, 8

[BOT88] Michael Ben-Or and Prasoon Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation. In *Proceedings of the* 20^{th} *Annual ACM symposium on Theory of computing (STOC'88)*, 1988. 3, 10

[BS21] Pranav Bisht and Nitin Saxena. Poly-time blackbox identity testing for sum of log-variate constant-width ROABPs. *Computational Complexity*, 2021. 7

[CCG12] Enrico Carlini, Maria Virginia Catalisano, and Anthony V. Geramita. The solution to the Waring problem for monomials and the sum of coprime monomials. *Journal of Algebra*, 2012. 12

[CKR$^+$20] Prerona Chatterjee, Mrinal Kumar, C Ramya, Ramprasad Saptharishi, and Anamay Tengse. On the Existence of Algebraically Natural Proofs. In *IEEE* 61^{st} *Annual Symposium on Foundations of Computer Science (FOCS'20)*, 2020. 2

[CKS18] Chi-Ning Chou, Mrinal Kumar, and Noam Solomon. Hardness vs randomness for bounded depth arithmetic circuits. In 33^{rd} *Computational Complexity Conference (CCC'18)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018. 2, 6

[DDOS14] Zeev Dvir, Rafael Mendes De Oliveira, and Amir Shpilka. Testing equivalence of polynomials under shifts. In *International Colloquium on Automata, Languages, and Programming*. Springer, 2014. 2

[DDS21a] Pranjal Dutta, Prateek Dwivedi, and Nitin Saxena. Deterministic Identity Testing Paradigms for Bounded Top-Fanin Depth-4 Circuits. In *Proceedings of the 36th Annual Computational Complexity Conference (CCC 2021)*, 2021. 1

[DDS21b] Pranjal Dutta, Prateek Dwivedi, and Nitin Saxena. Demystifying the border of depth-3 algebraic circuits. *Proceedings of the 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2021)*, 2021. 5

[DL78] Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 1978. 3, 9

[DS07] Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM Journal on Computing*, 2007. 4

[DS22] Pranjal Dutta and Nitin Saxena. Separated borders: Exponential-gap fanin-hierarchy theorem for approximative depth-3 circuits. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, 2022. 6

[DSS22] Pranjal Dutta, Nitin Saxena, and Amit Sinhababu. Discovering the Roots: Uniform Closure Results for Algebraic Classes Under Factoring. *J. ACM*, 2022. Preliminary version in the *50th Annual ACM Symposium on Theory of Computing (STOC 2018)*. 2, 7

[DST21] Pranjal Dutta, Nitin Saxena, and Thomas Thierauf. A Largish Sum-Of-Squares Implies Circuit Hardness and Derandomization. In *Proceedings of the 11th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, 2021. 2

[DSY10] Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM Journal on Computing*, 2010. Preliminary version in the *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*. 2, 7

[FGS18] Michael A Forbes, Sumanta Ghosh, and Nitin Saxena. Towards blackbox identity testing of log-variate circuits. In *Proceedings of the 45th International Colloquium on Automata, Languages and Programming (ICALP 2018)*, 2018. 7

[FGT19] Stephen Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-NC. *SIAM Journal on Computing*, 2019. Preliminary version in Proceedings of the 48^{th} Annual ACM symposium on Theory of Computing (STOC'16). 2

[For14] Michael A. Forbes. *Polynomial identity testing of read-once oblivious algebraic branching programs*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2014. 10

[For15] Michael A Forbes. Deterministic divisibility testing via shifted partial derivatives. In *Proceedings of the 56^{th} Annual Symposium on Foundations of Computer Science (FOCS'15)*, 2015. 4, 5, 6, 11, 26

[FS13] Michael A Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In 54^{th} *Annual Symposium on Foundations of Computer Science (FOCS'13)*, 2013. 5, 7, 11

[FSS14] Michael A Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Proceedings of the 46^{th} Annual ACM symposium on Theory of computing (STOC'14)*, 2014. 7, 11

[FSV18] Michael A Forbes, Amir Shpilka, and Ben Lee Volk. Succinct hitting sets and barriers to proving lower bounds for algebraic circuits. *Theory of Computing*, 2018.

Preliminary version in the *51st Annual ACM Symposium on Theory of Computing (STOC 2019)*. 2

[GGOW16] Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. In 57*th* *Annual Symposium on Foundations of Computer Science (FOCS'16)*, 2016. 7

[GKKS16] Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth three. *SIAM Journal on Computing*, 2016. Preliminary version in the *54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*. 3

[GKS17] Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Identity Testing for Constant-Width, and Any-Order, Read-Once Oblivious Arithmetic Branching Programs. *Theory of Computing*, 2017. Preliminary version in the *31st Annual Computational Complexity Conference (CCC 2016)*. 4, 7, 11

[GKSS22] Zeyu Guo, Mrinal Kumar, Ramprasad Saptharishi, and Noam Solomon. Derandomization from Algebraic Hardness. *SIAM Journal on Computing*, 2022. Preliminary version in the *60th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2019)*. 3

[GKST17] Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf. Deterministic identity testing for sum of read-once oblivious arithmetic branching programs. *Computational Complexity*, 2017. Preliminary version in the *30th Annual Computational Complexity Conference (CCC 2015)*. 7

[GOS22] Abhibhav Garg, Rafael Oliveira, and Akash Kumar Sengupta. Robust Radical Sylvester-Gallai Theorem for Quadratics. In *38th International Symposium on Computational Geometry (SoCG 2022)*, 2022. 3

[Gro15] Joshua A Grochow. Unifying known lower bounds via geometric complexity theory. *Computational Complexity*, 2015. Preliminary version in the IEEE 29*th* Computational Complexity Conference (CCC'14). 2

[GS20] Abhibhav Garg and Nitin Saxena. Special-case algorithms for blackbox radical membership, Nullstellensatz and transcendence degree. In *Proceedings of the* 45*th* *International Symposium on Symbolic and Algebraic Computation*, 2020. 6

[Guo21] Zeyu Guo. Variety Evasive Subspace Families. In *36th Computational Complexity Conference (CCC 2021)*, 2021. 6

[Gup14] Ankit Gupta. Algebraic Geometric Techniques for Depth-4 PIT & Sylvester-Gallai Conjectures for Varieties. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2014. 5, 6, 7

[HS80] Joos Heintz and Claus-Peter Schnorr. Testing polynomials which are easy to compute. In *Proceedings of the* 12^{th} *annual ACM symposium on Theory of computing (STOC'80)*, 1980. 2

[JKY16] A Grochow Joshua, D Mulmuley Ketan, and Qiao Youming. Boundaries of VP and VNP. In *Proceedings of the 43rd International Colloquium on Automata, Languages and Programming (ICALP 2016)*, 2016. 2

[JQS10] Maurice Jansen, Youming Qiao, and Jayalal Sarma. Deterministic Black-Box Identity Testing π-Ordered Algebraic Branching Programs. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010*, 2010. 7

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 2004. Preliminary version in the Proceedings of the 35^{th} Annual ACM symposium on Theory of computing (STOC'03). 2

[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 (ICALP'15)*, 2015. 7

[KMSV13] Zohar S Karnin, Partha Mukhopadhyay, Amir Shpilka, and Ilya Volkovich. Deterministic identity testing of depth-4 multilinear circuits with bounded top fan-in. *SIAM Journal on Computing*, 2013. Preliminary version in the Proceedings of the 42^{nd} ACM symposium on Theory of computing (STOC'10). 5

[Koi12] Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theoretical Computer Science*, 2012. 3

[KPT15] Pascal Koiran, Natacha Portier, and Sébastien Tavenas. A Wronskian approach to the real τ-conjecture. *Journal of Symbolic Computation*, 2015. 7

[KRST] Mrinal Kumar, C. Ramya, Ramprasad Saptharishi, and Anamay Tengse. If VNP Is Hard, Then so Are Equations for It. In *39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*. 2

[KS01] Adam R Klivans and Daniel Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the* 33^{rd} *Annual ACM symposium on Theory of computing (STOC'01)*, 2001. 3, 10, 21

[KS06] Adam Klivans and Amir Shpilka. Learning restricted models of arithmetic circuits. *Theory of computing*, 2006. Preliminary version in the 16^{th} Annual Conference on Learning Theory (COLT'03). 2

[KS07] Neeraj Kayal and Nitin Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 2007. Preliminary version in the 21st Computational Complexity Conference (CCC'06). 4, 6

[KS09] Zohar S Karnin and Amir Shpilka. Reconstruction of generalized depth-3 arithmetic circuits with bounded top fan-in. In 24th *Annual IEEE Conference on Computational Complexity (CCC'09)*. IEEE, 2009. 2

[KS11] Zohar S Karnin and Amir Shpilka. Black box polynomial identity testing of generalized depth-3 arithmetic circuits with bounded top fan-in. *Combinatorica*, 2011. Preliminary version in the 23rd Annual IEEE Conference on Computational Complexity (CCC'08). 4

[KS16] Mrinal Kumar and Shubhangi Saraf. Sums of Products of Polynomials in Few Variables: Lower Bounds and Polynomial Identity Testing. In 31st *Conference on Computational Complexity, CCC 2016*, LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. 3, 5, 7

[KS17] Mrinal Kumar and Shubhangi Saraf. Arithmetic Circuits with Locally Low Algebraic Rank. *Theory Comput.*, 2017. Preliminary version in the 31st Conference on Computational Complexity (CCC'16). 5, 7

[KSS14] Swastik Kopparty, Shubhangi Saraf, and Amir Shpilka. Equivalence of polynomial identity testing and deterministic multivariate polynomial factorization. In *IEEE 29th Conference on Computational Complexity (CCC'14)*. IEEE, 2014. 2

[KST19] Mrinal Kumar, Ramprasad Saptharishi, and Anamay Tengse. Near-optimal Bootstrapping of Hitting Sets for Algebraic Circuits. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 639–646, 2019. 3

[LFKN92] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM (JACM)*, 1992. 2

[LMP19] Guillaume Lagarde, Guillaume Malod, and Sylvain Perifel. Non-commutative computations: lower bounds and polynomial identity testing. *Chic. J. Theor. Comput. Sci.*, 2019. 7

[Lov79] László Lovász. On determinants, matchings, and random algorithms. In *Fundamentals of Computation Theory (FCT'79)*, 1979. 2

[LST21] Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial Lower Bounds Against Low-Depth Algebraic Circuits. *Accepted in the 62nd Annual Symposium on Foundations of Computer Science (FOCS), 2021*, 2021. 5, 6

[Mah13] Meena Mahajan. Algebraic Complexity Classes. *CoRR*, 2013. Pre-print available at `arXiv:1307.3863`. 9, 10

[Muk16] Partha Mukhopadhyay. Depth-4 identity testing and Noether's normalization lemma. In *International Computer Science Symposium in Russia (CSR'16)*. Springer, 2016. 5, 6

[Mul12a] Ketan D Mulmuley. Geometric complexity theory V: Equivalence between blackbox derandomization of polynomial identity testing and derandomization of Noether's normalization lemma. In *IEEE 53rd Annual Symposium on Foundations of Computer Science (FOCS'12)*. IEEE, 2012. 2

[Mul12b] Ketan D Mulmuley. The GCT program toward the P vs. NP problem. *Communications of the ACM*, 2012. 2

[MVV87] Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Comb.*, 1987. Preliminary version in the Proceedings of the 19^{th} Annual ACM symposium on Theory of Computing (STOC'87). 2

[Niv69] Ivan Niven. Formal Power Series. *The American Mathematical Monthly*, 1969. 7

[Ore22] Øystein Ore. Über höhere kongruenzen. *Norsk Mat. Forenings Skrifter*, 1922. 3, 9

[OS22] Rafael Oliveira and Akash Kumar Sengupta. Radical Sylvester-Gallai Theorem for Cubics. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 212–220, 2022. 3

[PS20] Shir Peleg and Amir Shpilka. A generalized Sylvester-Gallai type theorem for quadratic polynomials. In 35^{th} *Computational Complexity Conference (CCC'20)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2020. 3, 5

[PS21] Shir Peleg and Amir Shpilka. Polynomial time deterministic identity testing algorithm for $\sum^{[3]}\prod\sum\prod^{[2]}$ circuits via Edelstein-Kelly type theorem for quadratic polynomials. In 53^{rd} *Annual ACM symposium on Theory of computing (STOC'21)*, 2021. 3, 5, 7

[PSS18] Anurag Pandey, Nitin Saxena, and Amit Sinhababu. Algebraic independence over positive characteristic: New criterion and applications to locally low-algebraic-rank circuits. *Computational Complexity*, 2018. Preliminary version in the 41^{st} International Symposium on Mathematical Foundations of Computer Science (MFCS'16). 7

[RS05] Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 2005. Preliminary version in the 19^{th} IEEE Annual Conference on Computational Complexity (CCC'04). 7, 11

[Sap13] Ramprasad Saptharishi. *Unified Approaches to Polynomial Identity Testing and Lower Bounds*. PhD thesis, PhD thesis, Chennai Mathematical Institute, 2013. 10

[Sap19a] Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. Github survey, 2019. 9

[Sap19b] Ramprasad Saptharishi. Private Communication, 2019. 6

[Sax08] Nitin Saxena. Diagonal circuit identity testing and lower bounds. In *International Colloquium on Automata, Languages, and Programming (ICALP'08)*, pages 60–71. Springer, 2008. 6, 11

[Sax09] Nitin Saxena. Progress on Polynomial Identity Testing. *Bulletin of the EATCS*, 2009. 9, 10

[Sax14] Nitin Saxena. Progress on polynomial identity testing-II. In *Perspectives in Computational Complexity*. Springer, 2014. 9

[Sch80] Jacob T Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM (JACM)*, 1980. 3, 9

[Sha92] Adi Shamir. IP= PSPACE. *Journal of the ACM (JACM)*, 1992. 2

[Shp09] Amir Shpilka. Interpolation of depth-3 arithmetic circuits with two multiplication gates. *SIAM Journal on Computing*, 2009. Preliminary version in the Proceedings of the 39^{th} Annual ACM symposium on Theory of Computing (STOC 2007). 2

[Shp19] Amir Shpilka. Sylvester-Gallai type theorems for quadratic polynomials. In *Proceedings of the 51^{st} Annual ACM SIGACT Symposium on Theory of Computing (STOC'19)*, pages 1203–1214, 2019. 5

[Sin19] Amit Kumar Sinhababu. *Power series in complexity: Algebraic Dependence, Factor Conjecture and Hitting Set for Closure of VP*. PhD thesis, PhD thesis, Indian Institute of Technology Kanpur, 2019. 7

[SS11] Nitin Saxena and Comandur Seshadhri. An almost optimal rank bound for depth-3 identities. *SIAM journal on computing*, 2011. Preliminary version in the 24^{th} IEEE Conference on Computational Complexity (CCC'09). 4

[SS12] Nitin Saxena and Comandur Seshadhri. Blackbox identity testing for bounded top-fanin depth-3 circuits: The field doesn't matter. *SIAM Journal on Computing*, 2012. Preliminary version in the 43^{rd} Annual ACM symposium on Theory of computing (STOC'11). 4, 5, 6

[SS13] Nitin Saxena and Comandur Seshadhri. From Sylvester-Gallai configurations to rank bounds: Improved blackbox identity test for depth-3 circuits. *Journal of the*

ACM (JACM), 2013. Preliminary version in the 51^{st} Annual IEEE Symposium on Foundations of Computer Science (FOCS'10). 4

[SSS13] Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. A case of depth-3 identity testing, sparse factorization and duality. *Computational Complexity*, 2013. 3, 4, 6

[SV18] Shubhangi Saraf and Ilya Volkovich. Black-box identity testing of depth-4 multilinear circuits. *Combinatorica*, 2018. Preliminary version in the Proceedings of the 43^{rd} Annual ACM symposium on Theory of computing (STOC'11). 5

[SY10] Amir Shpilka and Amir Yehudayoff. *Arithmetic circuits: A survey of recent results and open questions*. Now Publishers Inc, 2010. 9, 10

[Val79] Leslie G Valiant. Completeness classes in algebra. In *Proceedings of the 11^{th} Annual ACM symposium on Theory of computing (STOC'79)*, 1979. 2

[Vas04] Wolmer Vasconcelos. *Computational methods in commutative algebra and algebraic geometry*. 2004. 6

[Zip79] Richard Zippel. Probabilistic Algorithms for Sparse Polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, EUROSAM '79, 1979. 3, 9