Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 33 (2013)

Explicit Noether Normalization for Simultaneous Conjugation via Polynomial Identity Testing

Michael A. Forbes* Amir Shpilka†

March 7, 2013

Abstract

Mulmuley [Mul12a] recently gave an explicit version of Noether's Normalization lemma for ring of invariants of matrices under simultaneous conjugation, under the conjecture that there are deterministic black-box algorithms for polynomial identity testing (PIT). He argued that this gives evidence that constructing such algorithms for PIT is beyond current techniques. In this work, we show this is not the case. That is, we improve Mulmuley's reduction and correspondingly weaken the conjecture regarding PIT needed to give explicit Noether Normalization. We then observe that the weaker conjecture has recently been nearly settled by the authors ([FS12]), who gave quasipolynomial size hitting sets for the class of read-once oblivious algebraic branching programs (ROABPs). This gives the desired explicit Noether Normalization unconditionally, up to quasipolynomial factors.

As a consequence of our proof we give a deterministic parallel polynomial-time algorithm for deciding if two matrix tuples have intersecting orbit closures, under simultaneous conjugation.

We also study the strength of conjectures that Mulmuley requires to obtain similar results as ours. We prove that his conjectures are stronger, in the sense that the computational model he needs PIT algorithms for is equivalent to the well-known algebraic branching program (ABP) model, which is provably stronger than the ROABP model.

Finally, we consider the depth-3 diagonal circuit model as defined by Saxena [Sax08], as PIT algorithms for this model also have implications in Mulmuley's work. Previous work (such as [ASS12] and [FS12]) have given quasipolynomial size hitting sets for this model. In this work, we give a much simpler construction of such hitting sets, using techniques of Shpilka and Volkovich [SV09].

1 Introduction

Many results in mathematics are non-constructive, in the sense that they establish that certain mathematical objects exist, but do not give an efficient or explicit construction of such objects, and often further work is needed to find constructive arguments. Motivated by the recent results of Mulmuley [Mul12a] (henceforth "Mulmuley", but theorem and page numbering will refer to the full version [Mul12b]), this paper studies constructive versions of the Noether Normalization Lemma from commutative algebra. The lemma, as used in this paper, can be viewed as taking a commutative ring R, and finding a smaller subring $S \subseteq R$ such that S captures many of the interesting properties of

*Email: miforbes@mit.edu, Department of Electrical Engineering and Computer Science, MIT CSAIL, 32 Vassar St., Cambridge, MA 02139, Supported by NSF grant CCF-0939370.

†Faculty of Computer Science, Technion — Israel Institute of Technology, Haifa, Israel, shpilka@cs.technion.ac.il. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 257575.

ISSN 1433-8092

R (see Section 1.2 for a formal discussion). Like many arguments in commutative algebra, the usual proof of this lemma does not focus on the computational considerations of how to find (a concise representation of) the desired subring S. However, the area of computational commutative algebra (eg, [DK02, CLO07]) offers methods showing how many classic results can be made constructive, and in certain cases, the algorithms are even efficient.

While constructive methods for Noether Normalization are known using Gröbner bases (cf. [CLO07]), the Gröbner basis algorithms are not efficient in the worst-case (as show by Mayr and Meyer [MM82]), and are not known to be more efficient for the problems we consider. Diverging from the Gröbner basis idea, Mulmuley recently observed that a constructive version of Noether Normalization is really a problem in *derandomization*. That is, given the ring R, if we take a sufficiently large set of "random" elements from R, then these elements will generate the desired subring S of R. Indeed, the usual proof of Noether Normalization makes this precise, with the appropriate algebraic meaning of "random". This view suggests that random sampling from R is sufficient to construct S, and this sampling will be efficient if R itself is explicitly given. While this process uses lots of randomness, the results of the derandomization literature in theoretical computer science (eg, [IW97, IKW02, KI04]) give strong conjectural evidence that randomness in efficient algorithms is not necessary. Applied to the problems here, there is thus strong conjectural evidence that the generators for the subring S can be constructed efficiently, implying that the Noether Normalization Lemma can be made constructive.

Motivated by this connection, Mulmuley explored what are the minimal derandomization conjectures necessary to imply an explicit form of Noether Normalization. The existing conjectures come in two flavors. Most derandomization hypotheses concern boolean computation, and as such are not well-suited for algebraic problems (for example, a single real number can encode infinitely many bits), but Mulmuley does give some connections in this regime. Other derandomization hypotheses directly concern algebraic computation, and using them Mulmuley gives an explicit Noether Normalization Lemma, for some explicit rings of particular interest. In particular, Mulmuley proves that it would suffice to derandomize the *polynomial identity testing* (PIT) problem in certain models, in order to obtain a derandomization of the Noether Normalization Lemma. Mulmuley actually views this connection as an evidence that derandomizing PIT for these models is a difficult computational task (Mulmuley, p. 3) and calls this the GCT chasm. Although Mulmuley conjectures that it could be crossed he strongly argues that this cannot be achieved with current techniques (Mulmuley, p. 3):

> On the negative side, the results in this article say that black-box derandomization of PIT in characteristic zero would necessarily require proving, either directly or by implication, results in algebraic geometry that seem impossible to prove on the basis of the current knowledge.

In this work, we obtain a derandomization of Noether's Normalization Lemma for the problems discussed in Mulmuley's Theorems 1.1 and 1.2, using existing techniques. These results alone have been suggested by Mulmuley (in public presentation) to contain the "impossible" problems mentioned above. This suggests that problems cannot be assumed to be difficult just because they originated in algebraic-geometric setting, and that one has to consider the finer structure of the problem.

In addition, just as Mulmuley's techniques extend to Noether Normalization of arbitrary quivers (Mulmuley's Theorem 4.1), our results also extend to this case, but we omit the straightforward details. However, we do not give any results for Noether Normalization of general explicit varieties as discussed in Mulmuley's Theorem 1.5, and indeed that seems difficult given that Mulmuley's

Theorem 1.6 gives an equivalence between general PIT and Noether Normalization for a certain explicit variety.[1]

We start by briefly describing the PIT problem. For more details, see the survey by Shpilka and Yehudayoff [SY10].

1.1 Polynomial Identity Testing

The PIT problem asks to decide whether a polynomial, given by an algebraic circuit, is the zero polynomial. An algebraic circuit is a directed acyclic graph with a single sink (or root) node, called the *output*. Internal nodes are labeled with either a $\times$- or $+$-gate, which denote multiplication and addition respectively. The source (or leaf) nodes, are labeled with either variables x_i, or elements from a given field $\mathbb{F}$. An algebraic circuit computes a polynomial in the ring $\mathbb{F}[x_1, \ldots, x_n]$ in the natural way (as there are no cycles): each internal node in the graph computes a function of its children (either $\times$ or $+$), and the circuit itself outputs the polynomial computed by the output node. Algebraic circuits give the most natural and succinct way to represent a polynomial.

Given an algebraic circuit C, the PIT problem is to test if the polynomial f it computes is identically zero, as a polynomial in $\mathbb{F}[x_1, \ldots, x_n]$. Schwartz [Sch80] and Zippel [Zip79] (along with the weaker result by DeMillo and Lipton [DL78]) showed that if $f \not\equiv 0$ is a polynomial of degree $\leq d$, and $\alpha_1, \ldots, \alpha_n \in S \subseteq \mathbb{F}$ are chosen uniformly at random, then $f(\alpha_1, \ldots, \alpha_n) = 0$ with probability $\leq d/|S|$. It follows then that we can solve PIT efficiently using randomness, as evaluating an algebraic circuit can be done efficiently. The main question concerning the PIT problem is whether it admits an efficient *deterministic* algorithm, in the sense that it runs in polynomial time in the size of the circuit C. Heuristically, this problem can be viewed as replacing any usage of the Schwartz-Zippel result with a deterministic set of evaluations.

One important feature of the above randomized PIT algorithm is that it only uses the circuit C by evaluating it on given inputs. Such algorithms are called *black-box* algorithms, as they treat the circuit as merely a black-box that computes some low-degree polynomial (that admits some small circuit computing it). This is in contrast to *white-box* algorithms, that probe the structure of C in some way. White-box algorithms are thus less restricted, whence deriving a black-box algorithm is a stronger result. For the purposes of this paper, instead of referring to deterministic black-box PIT algorithms, we will use the equivalent notion of a *hitting set*, which is a small set $\mathcal{H} \subseteq \mathbb{F}^n$ of evaluation points such that for any non-zero polynomial f computed by a small algebraic circuit, f must evaluate to non-zero on some point in $\mathcal{H}$. A standard argument (see [SY10]) shows that small hitting sets *exist*, the main question is whether small *explicit* hitting sets, which we now define, exist. As usual, the notion of explicit must be defined with respect to an infinite family of objects, one object for each value of n. For clarity, we abuse notation and do not discuss families, with the understanding that any objects we design will belong to an unspecified family, and that there is a single (uniform) algorithm to construct these objects that takes as input the relevant parameters.

Definition 1.1. *Let $\mathcal{C} \subseteq \mathbb{F}[x_1, \ldots, x_n]$ be a class of polynomials. A set $\mathcal{H} \subseteq \mathbb{F}^n$ is a **hitting set for** $\mathcal{C}$ if for all $f \in \mathcal{C}$, $f \equiv 0$ iff $f|_{\mathcal{H}} \equiv 0$. The hitting set $\mathcal{H}$ is $t(n)$**-explicit** if there is an algorithm such that given an index into $\mathcal{H}$, the corresponding element of $\mathcal{H}$ can be computed in $t(n)$-time, assuming unit cost arithmetic in $\mathbb{F}$.*

[1] The main reason we could obtain our results is that for the variety we consider, there are explicitly known generators for the ring of invariants (as given in Theorem 1.3), and these generators are computationally very simple. For general explicit varieties, obtaining such explicit generators is an open problem, and even if found, the generators would not likely be as computational simple as the generators of Theorem 1.3. We refer the reader to Mulmuley's Section 10 where these issues are discussed further.

That is, we mean that the algorithm can perform field operations (add, subtract, multiply, divide, zero test) in $\mathbb{F}$ in unit time, and can start with the constants 0 and 1. We will also assume the algorithm has access to an arbitrary enumeration of $\mathbb{F}$. In particular, when $\mathbb{F}$ has characteristic 0, without loss of generality the algorithm will only produce rational numbers.

1.2 Noether Normalization for the Invariants of Simultaneous Conjugation

Mulmuley showed that when R is a particular ring, then the problem of finding the subring S given by Noether Normalization can be reduced to the black-box PIT problem, so that explicit hitting sets (of small size) would imply a constructive version of Noether Normalization for this ring. The ring considered here and in Mulmuley's Theorems 1.1 and 1.2 is the ring of invariants of matrices, under the action of simultaneous conjugation.

Definition 1.2. *Let $\vec{M}$ denote a vector of r matrices, each[2] $[\![n]\!] \times [\![n]\!]$, whose entries are distinct variables. Consider the action of $\mathrm{GL}_n(\mathbb{F})$ by simultaneous conjugation on $\vec{M}$, that is,*

$$(M_1, \ldots, M_r) \mapsto (PM_1P^{-1}, \ldots, PM_rP^{-1}) .$$

Define $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ to be the subring of $\mathbb{F}[\vec{M}]$ consisting of polynomials in the entries of $\vec{M}$ that are invariant under the action of $\mathrm{GL}_n(\mathbb{F})$. That is,

$$\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})} := \{f | f(\vec{M}) = f(P\vec{M}P^{-1}), \forall P \in \mathrm{GL}_n(\mathbb{F})\} .$$

Note that $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is in fact a ring. When $\mathbb{F}$ has characteristic zero, the following result gives an explicit set of generators for the ring of invariants. When $\mathbb{F}$ has positive characteristic, the result is known not to hold (see [KP00, §2.5]) so we will only discuss characteristic zero fields.

Theorem 1.3 ([Pro76, Raz74, For86])**.** *Let $\mathbb{F}$ be a field of characteristic zero. Let $\vec{M}$ denote a vector of r matrices, each $[\![n]\!] \times [\![n]\!]$, whose entries are distinct variables. The ring $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ of invariants is generated by the invariants $\mathcal{T} := \{\mathrm{trace}(M_{i_1} \cdots M_{i_\ell}) | \vec{i} \in [\![r]\!]^\ell, \ell \in [n^2]\}$.*

Further, the ring $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is not generated by the invariants $\{\mathrm{trace}(M_{i_1} \cdots M_{i_\ell}) | \vec{i} \in [\![r]\!]^\ell, \ell \in [\lceil n^2/8 \rceil]\}$.

That is, every invariant can be represented as a (multivariate) polynomial, with coefficients in $\mathbb{F}$, in the above generating set. Note that the above generating set is indeed a set of invariants, because the trace is cyclic, so the action of simultaneous conjugation by P cancels out.

The above result is explicit in two senses. The first sense is that all involved field constants can be efficiently constructed. The second is that for any $f \in \mathcal{T}$ and $\vec{A}$, $f(\vec{A})$ can be computed quickly. In particular, any $f \in \mathcal{T}$ can be computed by a $\mathsf{poly}(n, r)$-sized algebraic circuit, as matrix multiplication and trace can be computed efficiently by circuits. We encapsulate these notions in the following definition.

Definition 1.4. *A set $\mathcal{P} \subseteq \mathbb{F}[x_1, \ldots, x_n]$ of polynomials has $t(n)$-**explicit**[3] $\mathcal{C}$-**circuits**, if there is an algorithm such that given an index into $\mathcal{P}$, a circuit $C \in \mathcal{C}$ can be computed in $t(n)$-time, assuming unit cost arithmetic in $\mathbb{F}$, such that C computes the indexed $f \in \mathcal{P}$.*

[2] In this work we will most often index vectors and matrices starting at zero, and will indicate this by the use of $[\![n]\!]$, which denotes the set $\{0, \ldots, n-1\}$. Also, $[n]$ will be used to denote the set $\{1, \ldots, n\}$.

[3] It is important to contrast this with the vague notion of being *mathematically* explicit. For example, a non-trivial n-th root of unity is mathematically explicit, but is not computationally explicit from the perspective of the rational numbers. Conversely, the lexicographically first function $f : \{0,1\}^{\lfloor \log \log n \rfloor} \to \{0,1\}$ with the maximum possible circuit complexity among all functions on $\lfloor \log \log n \rfloor$ bits, is computationally explicit, but arguably not mathematically explicit. While will we exclusively discuss the notion of computational explicitness in this paper, the constructions are all mathematically explicit, including the Forbes-Shpilka [FS12] result cited as Theorem 1.12.

In particular, the above definition implies that the resulting circuits C have size at most $t(n)$. The class of circuits $\mathcal{C}$ can be the class of all algebraic circuits, or some restricted notion, such as algebraic branching programs, which are defined later in this paper. Thus, in the language of the above definition, the set of generators $\mathcal{T}$ has $\mathsf{poly}(n, r)$-explicit algebraic circuits.

However, the above result is unsatisfactory in that the set of generators $\mathcal{T}$ has size $\exp(\mathsf{poly}(n, r))$, which is unwieldy from a computational perspective. One could hope to find a smaller set of generators, but the lower bound in the above theorem seems a barrier in that direction. The number of generators is relevant here, as we will consider three computational problems where these generators are useful, but because of their number the resulting algorithms will be exponential-time, where one could hope for something faster. To define these problems, we first give the following standard definition from commutative algebra.

Definition 1.5. *Let R be a commutative ring, and S a subring. Then R is **integral** over S if every element in R satisfies some monic polynomial with coefficients in S.*

As an example, the algebraic closure of $\mathbb{Q}$ (the algebraic numbers) is integral over $\mathbb{Q}$. In this work the rings R and S will be rings of polynomials, and it is not hard to see that all polynomials in R vanish at a point iff all polynomials in S vanish at that point. This can be quite useful, especially if S has a small list of generators. The statement of Noether Normalization is exactly that of providing such an S with a small list of generators. The question we consider here is how to find an explicit such S for the ring of invariants under simultaneous conjugation, where S should be given by its generators.

Question 1.6. *Let $\mathbb{F}$ be an algebraically closed field of characteristic zero. Is there a small set of polynomials $\mathcal{T}' \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ with explicit algebraic circuits, such that $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is integral over the subring S generated by $\mathcal{T}'$?*

We will in fact mostly concern ourselves with the next problem, which has implications for the first, when $\mathbb{F}$ is algebraically closed. We first give the following definition, following Derksen and Kemper [DK02].

Definition 1.7. *A subset $\mathcal{T}' \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is a set of **separating invariants** if for all $\vec{A}, \vec{B} \in (\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$, there exists an $f \in \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ such that $f(\vec{A}) \neq f(\vec{B})$ iff there exists an $f' \in \mathcal{T}'$ such that $f'(\vec{A}) \neq f'(\vec{B})$.*

As before, we will ask whether we can find an explicit construction.

Question 1.8. *Let $\mathbb{F}$ have characteristic zero. Is there a small set of separating invariants $\mathcal{T}' \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ with explicit algebraic circuits?*

Mulmuley used the tools of geometric invariant theory [MFK94], as done in Derksen and Kemper [DK02], to note that, over algebraically closed fields, any set $\mathcal{T}'$ of separating invariants will also generate a subring that $(\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$ is integral over. Thus, any positive answer to Question 1.8 will give a positive answer to Question 1.6. Hence, we will focus on constructing explicit separating invariants (over any field of characteristic zero).

Note that relaxations of Question 1.8 can be answered positively. If we only insist on explicit separating invariants (relaxing the insistence on having few invariants), then the exponentially-large set of generators $\mathcal{T}$ given in Theorem 1.3 suffices as these polynomials have small circuits and as they generate the ring of invariants, they have the required separation property. In contrast, if we only insist of a small set of separating invariants (relaxing the explicitness), then Noether Normalization essentially shows that a *non-explicit* set of separating invariants $\mathcal{T}'$ of size $\mathsf{poly}(n, r)$

exists, basically by taking a random $\mathcal{T}'$. More constructively, Mulmuley observed that Gröbner basis techniques can construct a small set of separating invariants $\mathcal{T}'$, but this set is still not explicit as such algorithms take exponential-space, so are far from efficient. In the particular case of $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$, Mulmuley showed that the construction can occur in PSPACE unconditionally, or even PH, assuming the Generalized Riemann Hypothesis. Thus, while there are explicit sets of separating invariants, and there are small sets of separating invariants, existing results do not achieve these two properties simultaneously.

The third problem is more geometric, as opposed to algebraic. Given a tuple of matrices $\vec{A}$, we can consider the orbit of $\vec{A}$ under simultaneous conjugation as a subset of $(\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$. A natural computational question is to decide whether the orbits of $\vec{A}$ and $\vec{B}$ intersect. However, from the perspective of algebraic geometry it is more natural to ask of the *orbit closures* intersect. That is, we now consider $\vec{A}$ and $\vec{B}$ as lying in $(\overline{\mathbb{F}}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$, where $\overline{\mathbb{F}}$ is the algebraic closure of $\mathbb{F}$. Then, we consider the orbit closures of $\vec{A}$ and $\vec{B}$ in this larger space, where this refers to taking the orbits in $(\overline{\mathbb{F}}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$ and closing them with respect to the Zariski topology. This yields the following question.

Question 1.9. *Let $\mathbb{F}$ be a field of characteristic zero. Is there an efficient deterministic algorithm (in the unit cost arithmetic model) that, given $\vec{A}, \vec{B} \in (\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$, decides whether the orbit closures of $\vec{A}$ and $\vec{B}$ under simultaneous conjugation have an empty intersection?*

Mulmuley observed that by the dictionary of geometric invariant theory [MFK94], $\vec{A}$ and $\vec{B}$ have a non-empty intersection of their orbit closures iff they are not distinguishable by any set of separating invariants. Thus, any explicit set $\mathcal{T}'$ of separating invariants, would answer this question, as one could test if f agrees on $\vec{A}$ and $\vec{B}$ (as f is easy to compute, as it has a small circuit), for all $f \in T'$. Thus, as before, Question 1.9 can be solved positively by a positive answer to Question 1.8.

The main results of this paper provide positive answers to Questions 1.6, 1.8 and 1.9.

1.3 Mulmuley's results

Having introduced the above questions, we now summarize Mulmuley's results that show that these questions can be solved positively if one assumes that there exist explicit hitting sets for a certain subclass of algebraic circuits, which we now define. We note that Mulmuley defines this model using linear, and not affine functions. However, we define the model using affine functions as this allows the model to compute any polynomial (and not just homogeneous polynomials), potentially with large size. However, this is without loss of generality, as derandomizing PIT is equally hard in the linear and the affine case, via standard homogenization techniques, see Lemma 4.1.

Definition 1.10 (Mulmuley)**.** *A polynomial $f(x_1, \ldots, x_n)$ is computable by a width w, depth d, **trace of a matrix power** if there exists a matrix $A(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]^{[\![w]\!]\times[\![w]\!]}$ whose entries are affine functions in $\vec{x}$ such that $f(\vec{x}) = \mathrm{trace}(A(\vec{x})^d)$. The **size**[4] of a trace of a matrix power is nwd.*

As matrix multiplication and trace both have small algebraic circuits, it follows that traces of matrix powers have small circuits. Further, as a restricted class of algebraic circuits, we can seek to deterministically solve the PIT problem for them, and the hypothesis that this is possible is potentially weaker than the corresponding hypothesis for general algebraic circuits. However, this hypothesis, in its black-box form, is strong enough for Mulmuley to derive implications for the above questions.

[4] One could consider the size to be $nw \log d$ because of the repeated squaring algorithm. However, in this paper our size measure is more natural as all such d will be small.

Theorem (Part of Mulmuley's Theorems 1.1 and 3.6)**.** *Let $\mathbb{F}$ be a field of characteristic zero. Assume that there is a $t(m)$-explicit hitting set, of size $s(m)$, for traces of matrix power over $\mathbb{F}$ of size m. Then there is a set $\mathcal{T}'$ of separating invariants, of size $\mathsf{poly}(s(\mathsf{poly}(n,r)))$, with $\mathsf{poly}(t(\mathsf{poly}(n,r)))$-explicit traces of matrix powers. Further, for algebraically closed $\mathbb{F}$, $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is integral over the ring generated by $\mathcal{T}'$.*

We briefly summarize the proof idea. It is clear from the definitions that the generating set $\mathcal{T}$ for $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is a set of separating invariants, albeit larger than desired. The proof will recombine these invariants into a smaller set, by taking suitable linear combinations. Specifically, suppose $\vec{A}$ and $\vec{B}$ are separable, and thus $\vec{\mathcal{T}}(\vec{A}) \neq \vec{\mathcal{T}}(\vec{B})$, where $\vec{\mathcal{T}}(\vec{A})$ denotes the sequence of evaluations $(f(\vec{A}))_{f\in\mathcal{T}}$. Standard arguments about inner-products show then that $\langle\vec{\mathcal{T}}(\vec{A}),\vec{\alpha}\rangle \neq \langle\vec{\mathcal{T}}(\vec{B}),\vec{\alpha}\rangle$ for random values of $\vec{\alpha}$. As a linear combination of invariants is also an invariant, it follows that $f_{\vec{\alpha}}(\vec{M}) = \langle\vec{\mathcal{T}}(\vec{M}),\vec{\alpha}\rangle$ is an invariant, and will separate $\vec{A}$ and $\vec{B}$ for random $\vec{\alpha}$. Intuitively, one can non-explicitly derandomize this to yield a set of separating invariants by taking sufficiently many choices of $\vec{\alpha}$ and union bounding over all $\vec{A}$ and $\vec{B}$.

Note that finding such $\vec{\alpha}$ is equivalent to asking for a hitting set for the class of polynomials $\{f_{\vec{x}}(\vec{A}) - f_{\vec{x}}(\vec{B}) : \vec{A},\vec{B} \in (\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}\}$, so explicitly derandomizing PIT would give an explicit set of separating invariants, as was desired. However, as is, the above reduction is unsatisfactory in two ways. Primarily, the resulting set of separating invariants would still be exponentially large, as one cannot, by a counting argument, construct small hitting sets for the above class of polynomials unless one can exploit structure in vectors $\vec{\mathcal{T}}(\vec{A})$. Second, the resulting invariants $f_{\vec{\alpha}}(\vec{M})$ will not have small circuits, unless, as before, one can exploit the structure of $\vec{\mathcal{T}}(\vec{A})$, but now using the structure to compute the exponentially-large sum $\langle\vec{\mathcal{T}}(\vec{A}),\vec{\alpha}\rangle$ in sub-exponential time. Both of these problems can be overcome by showing that indeed the vector $\vec{\mathcal{T}}(\vec{A})$ does have structure, in particular that it can be encoded into the coefficients of a small circuit. The circuit class that Mulmuley uses is the trace of matrix powers model.

Assuming various plausible conjectures and using the requisite results in derandomization literature, Mulmuley showed that small explicit hitting sets exist, removing the need to outright conjecture the existence of such hitting sets. This thus established the conjectural existence of small explicit sets of separating invariants by the above theorem. We list one such conditional result here, noting that all such conditional results Mulmuley derived gave sets of separating invariants of quasi-polynomial size, or worse.

Theorem (Part of Mulmuley's Theorems 1.2 and 5.1)**.** *Let $\mathbb{F}$ be a field of characteristic zero. Suppose there is a multilinear polynomial (family) $f(x_1,\ldots,x_n) \in \mathbb{Z}[x_1,\ldots,x_n]$ with coefficients containing at most $\mathsf{poly}(n)$ bits, such that $f(\vec{x})$ can be computed in $\exp(n)$-time, but f cannot be computed by an algebraic circuit of size $\mathcal{O}(2^{\epsilon n})$ and depth $\mathcal{O}(n^{\epsilon})$ for some $\epsilon > 0$. Then $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ has a $\mathsf{poly}(n,r)^{\mathsf{polylog}(n,r)}$-size set of separating invariants, with $\mathsf{poly}(n,r)^{\mathsf{polylog}(n,r)}$-explicit traces of matrix powers.*

While the above result is conditional, unconditional results can also be derived, if randomness is allowed. That is, by exploiting the connection between separating invariants and closed orbit intersections mentioned above, and using that PIT can be solved using randomness, Mulmuley obtains the following randomized algorithm.

Theorem (Mulmuley's Theorem 3.8)**.** *Let $\mathbb{F}$ be a field of characteristic zero. There is an algorithm, running in randomized $\mathsf{polylog}(n,r)$-time using $\mathsf{poly}(n,r)$-processors (RNC), in the unit cost arithmetic model, such that given $\vec{A},\vec{B} \in (\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$, one can decide whether the orbit closures of $\vec{A}$ and $\vec{B}$ under simultaneous conjugation have an empty intersection.*

Using the just mentioned conjectures, Mulmuley can also partially derandomize the above algorithm, but not to within polynomial time.

1.4 Our Results

We study further the connection raised by Mulmuley regarding the construction of separating invariants and the black-box PIT problem. In particular, we more carefully study the classes of algebraic circuits arising in the reduction from Noether Normalization to PIT. Two models are particularly important, and we define them now.

Definition 1.11 (Nisan [Nis91])**.** *A **algebraic branching program with unrestricted weights** of depth d and width $\leq w$, on the variables $x_1, \ldots, x_n$, is a directed acyclic graph such that*

- *The vertices are partitioned in $d+1$ layers $V_0, \ldots, V_d$, so that $V_0 = \{s\}$ (s is the source node), and $V_d = \{t\}$ (t is the sink node). Further, each edge goes from V_{i-1} to V_i for some $0 < i \leq d$.*
- *$\max |V_i| \leq w$.*
- *Each edge e is weighted with a polynomial $f_e \in \mathbb{F}[\vec{x}]$.*

Each s-t path is said to compute the polynomial which is the product of the labels of its edges, and the algebraic branching program itself computes the sum over all s-t paths of such polynomials.

- *In an **algebraic branching program (ABP)**, for each edge e the weight $f_e(\vec{x})$ is an affine function. The **size** is nwd.*
- *In a **read-once oblivious ABP (ROABP)** of (individual) degree $< r$, we have $n := d$, and for each edge e from V_{i-1} to V_i, the weight is a univariate polynomial $f_e(x_i) \in \mathbb{F}[x_i]$ of degree $< r$. The **size** is dwr.*

In the definition of ROABPs we will exclusively focus on individual degree, and thus will use the term "degree" (in Section 6 we will use the more usual total degree, for a different class of circuits). The ROABP model is called *oblivious* because the variable order $x_1 < \cdots < x_d$ is fixed. The model is called *read-once* because the variables are only accessed on one layer in the graph.

The ABP model is a standard algebraic model that is at least as powerful as algebraic formulas, as shown by Valiant [Val79], and can be simulated by algebraic circuits. As shown by Berkowitz [Ber84], the determinant can be computed by a small ABP over any field. See Shpilka and Yehuydayoff [SY10] for more on this model.

The ROABP model arose in prior work of the authors ([FS12]) as a natural model of algebraic computation capturing several other existing models. This model can also be seen as an algebraic analogue of the boolean model of computation known as the read-once oblivious branching program model, which is a non-uniform analogue of the complexity class RL. See Forbes and Shpilka [FS12] for more of a discussion on the motivation of this class.

Note that a polynomial computed by an ROABP of size s can be computed by an ABP of size $\mathsf{poly}(s)$. The converse is not true, as Nisan [Nis91] gave exponential lower bounds for the size of non-commutative ABPs computing the determinant, and non-commutative ABPs encompass ROABPs, while as mentioned above Berkowitz [Ber84] showed the determinant can be computed by small ABPs. Thus the ROABP model is strictly weaker in computational power than the ABP model.

While there are no efficient (white-box or black-box) PIT algorithms for ABPs, we established in prior work ([FS12]) a quasi-polynomial sized hitting set for ROABPs. This hitting set will be at the heart of our main result.

Theorem 1.12 ([FS12]). *Let $\mathcal{C}$ be the set of d-variate polynomials computable by depth d, width $\leq w$, degree $< r$ ROABPs. If $|\mathbb{F}| \geq \mathsf{poly}(d, w, r)$, then $\mathcal{C}$ has a $\mathsf{poly}(d, w, r)$-explicit hitting set $\mathcal{H} \subseteq \mathbb{F}^d$, of size $\leq \mathsf{poly}(d, w, r)^{\mathcal{O}(\lg d)}$. Further, if $\mathbb{F}$ has characteristic zero then $\mathcal{H} \subseteq \mathbb{Q}^d$.*

Our contributions are split into four sections.

The computational power of traces of matrix powers: We study the model of algebraic computation, traces of matrix powers, shown by Mulmuley to have implications for derandomizing Noether Normalization. In particular, as this model is a restricted class of algebraic circuits, we can ask: how restricted is it? If this model was particularly simple, it would suggest that derandomizing PIT for this class would be a viable approach to derandomizing Noether Normalization. In contrast, if this model of computation is sufficiently general, then given the difficulty of derandomizing PIT for such general models, using Mulmuley's reduction to unconditionally derandomize Noether Normalization could be a formidable challenge. In this work, we show it is the latter case, proving the following theorem.

Theorem (Theorem 4.6). *The computational models of algebraic branching programs and traces of matrix powers are equivalent, up to polynomial blow up in size.*

Derandomizing Noether Normalization via an improved reduction to PIT: This section contains the main results of our paper. Given the above result, and the lack of progress on derandomizing PIT in such general models such as ABPs, it might seem that derandomizing Noether Normalization for simultaneous conjugation is challenging. However, we show this is not true, by showing that derandomization of black-box PIT for ROABPs suffices for derandomizing Noether Normalization for simultaneous conjugation. By then invoking our prior work on hitting sets for ROABPs cited as Theorem 1.12, we establish the following theorems, giving quasi-affirmative answers to Questions 1.6, 1.8 and 1.9. Furthermore, our results are proved unconditionally and are at least as strong as the conditional results Mulmuley obtains by assuming strong conjectures such as the Generalized Riemann Hypothesis or strong lower bound results.

Specifically, we prove the following theorem which gives an explicit set of separating invariants (see Question 1.8).

Theorem 1.13. *Let $\mathbb{F}$ be a field of characteristic zero. There is a $\mathsf{poly}(n, r)^{\mathcal{O}(\log(n))}$-sized set $\mathcal{T}_{\mathcal{H}}$ of separating invariants, with $\mathsf{poly}(n, r)$-explicit ABPs. That is, $\mathcal{T}_{\mathcal{H}} \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$, and for any $\vec{A}, \vec{B} \in (\mathbb{F}^{[n]\times[n]})^{[r]}$, $f(\vec{A}) \neq f(\vec{B})$ for some $f \in \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ iff $f'(\vec{A}) \neq f'(\vec{B})$ for some $f' \in T_{\mathcal{H}}$.*

As a consequence of Theorem 1.13 and the discussion in Subsection 1.2 we obtain the following corollary that gives a positive answer to Question 1.6. In particular, it provides a derandomization of Noether Normalization Lemma for the ring of invariants of simultaneous conjugation.

Corollary 1.14. *Let $\mathbb{F}$ be an algebraically closed field of characteristic zero. Let $\mathcal{T}_{\mathcal{H}}$ be the set guaranteed by Theorem 1.13. Then, $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is integral over the subring generated by $T_{\mathcal{H}}$.*

For deciding intersection of orbit closures, Question 1.9, the natural extension of Theorem 1.13, as argued in Subsection 1.2, would yield a quasi-polynomial-time algorithm for deciding orbit closure intersection. However, by replacing the black-box PIT results for ROABPs of Forbes and Shpilka [FS12] by the white-box PIT results by Raz and Shpilka [RS05] (as as well as follow-up work by Arvind, Joglekar and Srinivasan [AJS09]), we can obtain the following better algorithm for deciding orbit closure intersection, proving a strong positive answer to Question 1.9.

Theorem 1.15. *Let $\mathbb{F}$ be a field of characteristic zero. There is an algorithm, running in deterministic* $\mathsf{polylog}(n,r)$*-time using* $\mathsf{poly}(n,r)$*-processors (*NC*), in the unit cost arithmetic model, such that given $\vec{A},\vec{B} \in (\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$, one can decide whether the orbit closures of $\vec{A}$ and $\vec{B}$ under simultaneous conjugation have an empty intersection.*

As mentioned above, Mulmuley also gets results for Noether Normalization of arbitrary quivers (Mulmuley's Theorem 4.1) in a generalization of the results on simultaneous conjugation. The main difference is a generalization of the list of generators given in Theorem 1.3 to arbitrary quivers, as given by Le Bruyn and Procesi [LBP90]. Our improved reduction to PIT, involving ROABPs instead of ABPS, also generalizes to this case, so analogous results to the above three theorems are readily attained. However, to avoid discussing the definition of quivers, we do not list the details here.

PIT for Depth-3 Diagonal Circuits: Mulmuley's Theorem 1.4 showed that Noether Normalization for representations of $\mathsf{SL}_m(\mathbb{F})$ can be reduced, when m is constant, to black-box PIT of a subclass of circuits known as *depth-3 diagonal circuits*, see Section 6 for a definition. This class of circuits (along with a depth-4 version) was introduced in Saxena [Sax08], who gave a polynomial-time white-box PIT algorithm, via a reduction to the white-box PIT algorithm for non-commutative ABPs of Raz and Shpilka [RS05]. Saha, Saptharishi and Saxena [SSS11] (among other things) generalized these results to the depth-4 semi-diagonal model. Agrawal, Saha and Saxena [ASS12] gave (among other things) a quasipolynomial size hitting set for this model, by showing that any such circuit can be shifted so that there is a small-support monomial, which can be found via brute-force. In independent work, the present authors (in [FS12]) also established (among other things) a quasipolynomial size hitting set for this model. This was done by showing that the depth-4 semi-diagonal model is efficiently simulated by the ROABP model. Further, this was done in two ways: the first was an explicit reduction by using the duality ideas of Saxena [Sax08], and the second was to show that the diagonal model has a small space of derivatives in a certain sense, and that ROABPs can efficiently compute any polynomial with that sort of small space of derivatives. Some aspects of this model are also present in the work of Gupta-Kamath-Kayal-Saptharishi [GKKS13] showing that (arbitrary) depth-3 formulas capture, in a sense, the entire complexity of arbitrary algebraic circuits.

Here, we give a simpler proof that the depth-3 diagonal circuit model has a quasipolynomial size hitting set. This is done using the techniques of [SV09], and have some similarities with the work of Agrawal, Saha and Saxena [ASS12]. In particular, we show the entire space of derivatives is small, for depth-3 model (but not the depth-4 model). We then show that this implies such polynomials must contain a monomial of logarithmic support, which can be found via brute-force in quasipolynomial time. Unlike the work of Agrawal, Saha and Saxena [ASS12], no shifts are required for this small monomial to exist. Thus, we get the following theorem.

Theorem (Corollary 6.12)**.** *Let $\mathbb{F}$ be a field with size $\geq d+1$. Then there is a* $\mathsf{poly}(n,d,\log(s))$*-explicit hitting set of size* $\mathsf{poly}(n,d)^{\mathcal{O}(\log s)}$ *for the class of n-variate, degree $\leq d$, depth-3 diagonal circuits of size $\leq s$.*

Deciding (non-closed) orbit membership via PIT: The results mentioned thus far have taken an algebro-geometric approach to studying the orbits of tuples of matrices under simultaneous conjugation, as they take this geometric action and study the algebraic structure of its invariants. This perspective, by the very continuous nature of polynomials, can only access the orbit closures

under this group action. For example, working over $\mathbb{C}$, define

$$A_\epsilon := \begin{bmatrix} 1 & \epsilon \\ 0 & 1 \end{bmatrix} \qquad\qquad P_\delta := \begin{bmatrix} \delta & 0 \\ 0 & 1 \end{bmatrix},$$

and note that for any ϵ and for any $\delta \neq 0$, $P_\delta A_\epsilon P_\delta^{-1} = A_{\epsilon\delta}$. It follows that for any polynomial f invariant under simultaneous conjugation of 2×2 matrices, that $f(A_\epsilon)$ is independent of ϵ, as f is continuous and we can take $\epsilon \to 0$. However, for any $\epsilon \neq 0$, A_ϵ is not conjugate to $A_0 = \mathrm{I}_2$, the identity matrix, or equivalently, A_ϵ and A_0 are not in the same orbit. Thus, from the perspective of invariants A_ϵ and A_0 are the same, despite being in different orbits.

One can ask the analogue of Question 1.9, but for orbits as opposed to orbit closures. Note that by the invertibility of the group action, two orbits must either be disjoint, or equal. Thus, we equivalently ask for an algorithm to the orbit membership problem for simultaneous conjugation. That is, given $\vec{A}, \vec{B} \in (\mathbb{F}^{\mathbb{[}n\mathbb{]}\times\mathbb{[}n\mathbb{]}})^{\mathbb{[}r\mathbb{]}}$ is there an invertible $P \in \mathbb{F}^{\mathbb{[}n\mathbb{]}\times\mathbb{[}n\mathbb{]}}$ such that $\vec{B} = P\vec{A}P^{-1}$.

Several interesting cases of this problem were solved: Chistov, Ivanyos and Karpinski [CIK97] gave a deterministic polynomial time algorithm over finite fields and over algebraic number fields; Sergeichuk [Ser00] gave[5] a deterministic algorithm over any field, that runs in polynomial time when supplied with an oracle for finding roots of polynomials.[6] Chistov-Ivanyos-Karpinski also mentioned that a randomized polynomial-time algorithm for the problem follows from the work of Schwartz and Zippel [Sch80, Zip79]. In conversations with Yekhanin [Yek12], we also discovered this randomized algorithm, showing that this problem is reducible to PIT for ABPs. Because of its close relation to the rest of this work, we include for completeness the proof of the following theorem.

Theorem (Theorem A.1)**.** *Let $\mathbb{F}$ be a field of size $\geq \mathsf{poly}(n)$. Then the orbit membership problem, for simultaneous conjugation, is reducible to polynomial identity testing for ABPs. In particular, this problem has a randomized, parallel, polynomial-time algorithm.*

1.5 Notation

Given a vector of polynomials $\vec{f} \in \mathbb{F}[\vec{x}]^n$ and an exponent vector $\vec{i} \in \mathbb{N}^n$, we write $\vec{f}^{\vec{i}}$ for $f_1^{i_1} \cdots f_n^{i_n}$.

Given a polynomial $f \in \mathbb{F}[\vec{x}]$, we write $\mathfrak{C}_{\vec{x}^{\vec{i}}}(f)$ to denote the coefficient of $\vec{x}^{\vec{i}}$ in f. For a matrix $M \in \mathbb{F}[\vec{x}]^{\mathbb{[}r\mathbb{]}\times\mathbb{[}r\mathbb{]}}$, we write $\mathfrak{C}_{\vec{x}^{\vec{i}}}(M)$ to denote the $r \times r$ $\mathbb{F}$-matrix, with the $\mathfrak{C}_{\vec{x}^{\vec{i}}}$ operator applied to each entry. When we write "$f \in \mathbb{F}[\vec{x}][\vec{y}]$", we will treat f as a polynomial in the variables $\vec{y}$, whose coefficients are polynomials in the variables $\vec{x}$, and correspondingly will write $\mathfrak{C}_{\vec{y}^{\vec{j}}}(f)$ to extract the polynomial in $\vec{x}$ that is the coefficient of the monomial $\vec{y}^{\vec{j}}$ in f.

1.6 Organization

In Section 2 we give the necessary background on ROABPs. We prove our main results about explicit Noether Normalization in Section 3.

The rest of our results appear in the following order. We give the equivalence between the trace of matrix power and ABP models of computation in Section 4. We give the hitting set for depth-3 diagonal circuits in Section 6, using Hasse derivatives as defined in Section 5. In Appendix A we give the reduction from the orbit membership problem to PIT.

[5] In his paper Sergeichuk gives credit for the algorithm to Belitskiĭ [Bel83].

[6] This is not how the result is stated in [Ser00], but this is what one needs to make the algorithm efficient. In fact, to make the algorithm of Sergeichuk run in polynomial space one needs to make another assumption that would allow writing down the eigenvalues of all matrices involved in polynomial space. For example, one such assumption would be that they all belong to the some polynomial degree extension field (Grochow [Gro13]).

2 Properties of Algebraic Branching Programs

We first derive some simple properties of ABPs, as well as ROABPs, that show their tight connection with matrix products, and traces of matrix products. We begin with the following connection between an ABP with unrestricted weights, and the product of its adjacency matrices. As the lemma is proved in generality, it will apply to ABPs and ROABPs, and we will use it for both.

Lemma 2.1. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be computed by a depth d, width $\leq w$ ABP with unrestricted weights, such that the variable layers are $V_0, \ldots, V_d$. For $0 < i \leq d$, define $M_i \in \mathbb{F}[\vec{x}]^{V_{i-1} \times V_i}$ such that the (u, v)-th entry in M_i is the label on the edge from $u \in V_{i-1}$ to $v \in V_i$, or 0 if no such edge exists. Then, when treating $\mathbb{F}[\vec{x}]^{[\![1]\!] \times [\![1]\!]} = \mathbb{F}[\vec{x}]$,*

$$f(\vec{x}) = \prod_{i \in [d]} M_i(\vec{x}) := M_1(\vec{x}) M_2(\vec{x}) \cdots M_d(\vec{x}) \ .$$

Further, for an ABP, the matrix M_i has entries which are affine forms, and for an ROABP, the matrix M_i has entries which are univariate polynomials in x_i of degree $< r$.

Proof. Expanding the matrix multiplication completely, one sees that it is a sum of the product of the labels of the s-t paths such that the i-th edge goes from V_{i-1} to V_i. By the layered structure of the ABP, this is all such paths, so this sum computes $f(\vec{x})$. □

The above lemma shows that one can easily convert an ABP or ROABP into a matrix product, where the entries of matrices obey the same restrictions as the weights in the ABP or ROABP. The above lemma gives matrices with varying sizes, and it will be more convenient to have square matrices, which can be done by padding, as shown in the next lemma.

Lemma 2.2. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be computed by a depth d, width $\leq w$ ABP with unrestricted weights. Then for $i \in [d]$, there are matrices $M_i \in \mathbb{F}[\vec{x}]^{[\![w]\!] \times [\![w]\!]}$ such that in block notation,*

$$\begin{bmatrix} f(\vec{x}) & 0 \\ 0 & 0 \end{bmatrix} = \prod_{i \in [d]} M_i(\vec{x}) \ ,$$

that is, $\prod_{i \in [d]} M_i(\vec{x})$ contains a single non-zero entry located at $(0, 0)$, which contains the polynomial $f(\vec{x})$. Conversely, any such polynomial f such that

$$f(\vec{x}) = \left(\prod_{i \in [d]} M_i(\vec{x}) \right)_{(0,0)}$$

can be computed by a depth d, width w, ABP with unrestricted weights.

Further, for the specific cases of ABPs and ROABPs, the entries in the M_i are restricted: for ABPs the matrix M_i has entries which are affine forms, and for an ROABP the matrix M_i has entries which are univariate polynomials in x_i of degree $< r$.

Proof. ABP $\Longrightarrow$ matrices: Lemma 2.1 furnishes M_i such that $f(\vec{x}) = \prod_i M_i(\vec{x})$. Let $M_i' \in \mathbb{F}[\vec{x}]^{[\![w]\!] \times [\![w]\!]}$ be M_i padded with zeroes to become $[\![w]\!] \times [\![w]\!]$ sized, that is,

$$M_i'(\vec{x}) := \begin{bmatrix} M_i & 0 \\ 0 & 0 \end{bmatrix}$$

where we use block-matrix notation. By the properties of block-matrix multiplication, it follows that

$$\prod_i M'_i = \prod_i \begin{bmatrix} M_i & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \prod_i M_i & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} f(\vec{x}) & 0 \\ 0 & 0 \end{bmatrix}$$

as desired. One can observe that the use of Lemma 2.1 implies that the M_i have the desired restrictions on their entries.

matrices $\implies$ ABP: For $1 < i < d$ define $M'_i := M_i$. Define M'_1 to be the 0-th row of M_1, and define M'_d to be the 0-th column of M_d. Then it follows that

$$f(\vec{x}) = \left(\prod_i M_i(\vec{x})\right)_{(0,0)} = \prod_i M'_i(\vec{x})$$

and that the M'_i have at most w rows and columns. Using the M'_i as the adjacency matrices in an ABP with unrestricted weights, it follows by Lemma 2.1 that f is computed by a depth d, width w ABP with unrestricted weights. Further, the use of this lemma shows the entry restrictions for ABPs and ROABPs are respected, to the result holds for these models as well. □

We next observe that small-size ABPs and ROABPs are respectively closed under addition and multiplication.

Lemma 2.3. *Let $f, f' \in \mathbb{F}[x_1, \ldots, x_n]$ be computed by depth d ABPs with unrestricted weights, where f is computed in width $\leq w$ and f' is computed in width $\leq w'$. Then $f + f'$ and $f - f'$ are computed by depth d, width $\leq (w + w')$ ABPs with unrestricted weights. Further, this also holds for the ABP model, and the ROABP model with degree $< r$.*

Proof. $f + f'$: Consider the ABPs with unrestricted weights for f and f'. As they have the same depth, we can align their $d + 1$ layers. Consider them together as a new ABP, by merging the two source nodes, and merging the two sink nodes. This is an ABP and has the desired depth, width and degree. Note that it computes $f + f'$, as any source-sink path must either go entirely through the ABP for f, or entirely though the ABP for f', i.e. there are no cross-terms. Thus, the sum over all paths can be decomposed into the paths for f and the paths for f', showing that the computation is $f + f'$ as desired.

Note that in the ABP case, all edge weights are still affine functions, and in the ROABP case, the edge weights are still univariate polynomials of degree $< r$ for the relevant variables, as we aligned the layers between f and f'.

$f - f'$: This follows by showing that if f' is computable by an ABP with unrestricted weights then so is $-f'$, all within the same bounds. To see this, observe that by flipping the sign of all edges from V_0 to V_1 in the computation for f', each source-sink path in the computation for f' will have its sign flipped, and thus the entire sum will be negated, computing $-f'$ as desired.

Note that the allowed edge weights for ABPs and ROABPs are closed under linear combinations, so the above computation of $-f$ is also valid in these models. □

3 Reducing Noether Normalization to Read-once Oblivious Algebraic Branching Programs

In this section we construct a small set of explicit separating invariants for simultaneous conjugation. We do so by constructing a single ROABP that encodes the entire generating set $\mathcal{T}$ for $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$,

as given by Theorem 1.3. We then use hitting sets for ROABPs to efficiently extract the separating invariants from this ROABP. We begin with the construction[7] of the ROABP.

Construction 3.1. *Let $n, r, \ell \geq 1$. Let $\vec{M}$ denote a vector of r matrices, each $[\![n]\!] \times [\![n]\!]$, whose entries are distinct variables. Define*

$$M(x) := \sum_{i \in [\![r]\!]} M_i x^i$$

and, for the ℓ variables $\vec{x}$, define

$$f_\ell(\vec{M}, \vec{x}) := \text{trace}(M(x_1) \cdots M(x_\ell)).$$

The following lemma shows that these polynomials $f_\ell(\vec{M}, \vec{x})$ can be computed by small ROABPs, when $\vec{x}$ is variable and $\vec{M}$ is constant.

Lemma 3.2. *Assume the setup of Construction 3.1. Let $\vec{A}, \vec{B} \in (\mathbb{F}^{[\![n]\!] \times [\![n]\!]})^{[\![r]\!]}$. Then $f_\ell(\vec{A}, \vec{x}) - f_\ell(\vec{B}, \vec{x})$ can be computed by a width $2n^2$, depth ℓ, degree $< r$ ROABP.*

Proof. Observe that $f_\ell(\vec{A}, \vec{x}) = \text{trace}(A(x_1) \cdots A(x_\ell)) = \sum_{i \in [\![n]\!]} (\prod_{j=1}^{\ell} A(x_j))_{(i,i)}$. By applying a permutation of indices, and appealing to Lemma 2.2, we see that $(\prod_{j=1}^{\ell} A(x_j))_{(i,i)}$ is computable by a depth ℓ, width n, degree $< r$ ROABP, for each i. Thus, appealing to Lemma 2.3 completes the claim. □

Alternatively, when $\vec{x}$ is constant, and the matrices $\vec{M}$ are variable, then $f_\ell(\vec{M}, \vec{x})$ can be computed by a small ABP.

Lemma 3.3. *Assume the setup of Construction 3.1. Let $\vec{\alpha} \in \mathbb{F}^\ell$. Then $f_\ell(\vec{M}, \vec{\alpha})$ can be computed by a width n^2, depth ℓ ABP, and this ABP is constructable in $\mathsf{poly}(n, r, \ell)$ steps.*

Proof. Observe that $f_\ell(\vec{M}, \vec{\alpha}) = \text{trace}(M(\alpha_1) \cdots M(\alpha_\ell))$, and that each $M(\alpha_i)$ is an $[\![n]\!] \times [\![n]\!]$ matrix with entries that are affine forms in the $\vec{M}$. Thus, just as in Lemma 3.2, we can compute this trace in width n^2, depth ℓ ABP by appealing to Lemma 2.2 and Lemma 2.3. It is straightforward to observe that the construction of this ABP runs in the desired time bounds, as the above lemmas are constructive. □

Our next two lemmas highlight the connection between the polynomials in Construction 3.1 and the generators of the ring of invariants provided by Theorem 1.3. Namely, they show that the generators in the set $\mathcal{T}$ of Theorem 1.3 are faithfully encoded as coefficients of the polynomial $f_\ell(\vec{M}, \vec{x})$, when viewing this polynomial as lying in the ring $\mathbb{F}[\vec{M}][\vec{x}]$. Note here that we use the $\mathfrak{C}_{\vec{x}^{\vec{i}}}$ notation as defined in Subsection 1.5.

Lemma 3.4. *Assume the setup of Construction 3.1. Then for $\vec{i} \in \mathbb{N}^\ell$, taking coefficients in $\mathbb{F}[\vec{M}][\vec{x}]$,*

$$\mathfrak{C}_{\vec{x}^{\vec{i}}}(f_\ell(\vec{M}, \vec{x})) = \begin{cases} \text{trace}(M_{i_1} \cdots M_{i_\ell}) & \text{if } \vec{i} \in [\![r]\!]^{\vec{\ell}} \\ 0 & \text{else} \end{cases}.$$

[7] There are some slightly better versions of this construction, as well as a way to more efficiently use the hitting sets of Theorem 1.12. However, these modifications make the presentation slightly less modular, and do not improve the results by more than a polynomial factor, so we do not pursue these details.

Proof. Consider $f_\ell(\vec{M},\vec{x})$ as a polynomial in $\mathbb{F}[\vec{M}][\vec{x}]$. Taking coefficients, we see that

$$\begin{aligned}\mathfrak{C}_{\vec{x}^{\vec{i}}}(f_\ell(\vec{M},\vec{x})) &= \mathfrak{C}_{\vec{x}^{\vec{i}}}\left(\operatorname{trace}(M(x_1)\cdots M(x_\ell))\right)\\ &= \mathfrak{C}_{\vec{x}^{\vec{i}}}\left(\operatorname{trace}\left(\left(\sum_{j_1\in[\![r]\!]} M_{j_1}x_1^{j_1}\right)\cdots\left(\sum_{j_\ell\in[\![r]\!]} M_{j_\ell}x_\ell^{j_\ell}\right)\right)\right)\\ &= \mathfrak{C}_{\vec{x}^{\vec{i}}}\left(\operatorname{trace}\left(\sum_{j_1,\ldots,j_\ell\in[\![r]\!]} M_{j_1}\cdots M_{j_\ell}x_1^{j_1}\cdots x_\ell^{j_\ell}\right)\right)\end{aligned}$$

by linearity of the trace,

$$\begin{aligned}&= \mathfrak{C}_{\vec{x}^{\vec{i}}}\left(\sum_{\vec{j}\in[\![r]\!]^\ell}\operatorname{trace}(M_{j_1}\cdots M_{j_\ell})\vec{x}^{\vec{j}}\right)\\ &= \begin{cases}\operatorname{trace}(M_{i_1}\cdots M_{i_\ell}) & \text{if } \vec{i}\in[\![r]\!]^{\vec{\ell}}\\ 0 & \text{else}\end{cases}.\end{aligned}$$

□

As the above lemma shows that $f_\ell(\vec{M},\vec{x})$ encodes all of the generators $\mathcal{T}$, it follows that $\vec{A}$ and $\vec{B}$ agree on the generators $\mathcal{T}$ iff they agree on $f_\ell(\vec{M},\vec{x})$.

Lemma 3.5. *Assume the setup of Construction 3.1. Let $\vec{A},\vec{B}\in(\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$ and $\ell\geq 1$. Then $\operatorname{trace}(A_{i_1}\cdots A_{i_\ell})=\operatorname{trace}(B_{i_1}\cdots B_{i_\ell})$ for all $\vec{i}\in[\![r]\!]^\ell$ iff $f_\ell(\vec{A},\vec{x})=f_\ell(\vec{B},\vec{x})$, where this second equality is as polynomials in the ring $\mathbb{F}[\vec{x}]$.*

Proof. The two polynomials $f_\ell(\vec{A},\vec{x})$, $f_\ell(\vec{B},\vec{x})$ are equal iff all of their coefficients are equal. By Lemma 3.4, this is exactly the statement that $\operatorname{trace}(A_{i_1}\cdots A_{i_\ell})=\operatorname{trace}(B_{i_1}\cdots B_{i_\ell})$ for all $\vec{i}\in[\![r]\!]^\ell$. □

Hence, the polynomials $f_\ell(\vec{M},\vec{x})$ capture the generators $\mathcal{T}$ of $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ thus in a sense capturing the entire ring $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ also.

Corollary 3.6. *Assume the setup of Construction 3.1. Let $\mathbb{F}$ be a field of characteristic zero. Let $\vec{A},\vec{B}\in(\mathbb{F}^{[\![n]\!]\times[\![n]\!]})^{[\![r]\!]}$. Then $f(\vec{A})=f(\vec{B})$ for all $f\in\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ iff $f_\ell(\vec{A},\vec{x})=f_\ell(\vec{B},\vec{x})$ for all $\ell\in[n^2]$, where the second equality is as polynomials in the ring $\mathbb{F}[\vec{x}]$.*

Proof. By Lemma 3.5, $f_\ell(\vec{A},\vec{x})=f_\ell(\vec{B},\vec{x})$ for $\ell\in[n^2]$ iff $g(\vec{A})=g(\vec{B})$ for all g of the form $g(\vec{M})=\operatorname{trace}(M_{i_1}\cdots M_{i_\ell})$ for $\vec{i}\in[\![r]\!]^\ell$ and $\ell\in[n^2]$. This set of g is exactly the set $\mathcal{T}$ of Theorem 1.3, and by that result $\mathcal{T}$ generates $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$, which implies that the polynomials in $\mathcal{T}$ agree on $\vec{A}$ and $\vec{B}$ iff all the polynomials in $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ agree on $\vec{A}$ and $\vec{B}$. Thus $f_\ell(\vec{A},\vec{x})=f_\ell(\vec{B},\vec{x})$ for $\ell\in[n^2]$ iff $f(\vec{A})=f(\vec{B})$ for all $f\in\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$. □

Thus having reduced the question of whether $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ separates $\vec{A}$ and $\vec{B}$, to the question of whether some $f_\ell(\vec{M},\vec{x})$ separates $\vec{A}$ and $\vec{B}$, we now seek to remove the need for the indeterminates $\vec{x}$. Specifically, we will replace them by the evaluation points of a hitting set, as shown in the next construction.

Construction 3.7. *Assume the setup of Construction 3.1. Let $\mathcal{H}\subseteq\mathbb{F}^{n^2}$ be a $t(n,r)$-explicit hitting set for width $\leq 2n^2$, depth n^2, degree $<r$ ROABPs. Define*

$$\mathcal{T}_{\mathcal{H}} := \{f_\ell(\vec{M},\vec{\alpha})|\vec{\alpha}\in\mathcal{H},\ell\in[n^2]\}\,,$$

where if $\ell<n^2$ we use the first ℓ variables of α for the values of $\vec{x}$ in the substitution.

We now prove the main theorems, showing how to construct small sets of explicit separating invariants. We first do this for an arbitrary hitting set, then plug in the hitting set given in our previous work as stated in Theorem 1.12.

Theorem 3.8. *Assume the setup of Construction 3.7. Let $\mathbb{F}$ be a field of characteristic zero. Then $\mathcal{T}_{\mathcal{H}}$ is a set of size $n^2|\mathcal{H}|$ of homogeneous separating invariants with $\mathsf{poly}(t(n,r),n,r)$-explicit ABPs. That is, $\mathcal{T}_{\mathcal{H}} \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$, and for any $\vec{A},\vec{B} \in (\mathbb{F}^{[n]\times[n]})^{[r]}$, $f(\vec{A}) \neq f(\vec{B})$ for some $f \in \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ iff $f'(\vec{A}) \neq f'(\vec{B})$ for some $f' \in \mathcal{T}_{\mathcal{H}}$, and each such f' is computed by an explicit ABP.*

Proof. $\mathcal{T}_{\mathcal{H}}$ has explicit ABPs: We can index $\mathcal{T}_{\mathcal{H}}$ by $\ell \in [n^2]$ and an index in $\mathcal{H}$. Given an index in $\mathcal{H}$ we can, by the explicitness of $\mathcal{H}$, compute the associated $\vec{\alpha} \in \mathcal{H}$ in $t(n,r)$ steps. By Lemma 3.3 we can compute an ABP for $f_\ell(\vec{M},\vec{\alpha})$ in $\mathsf{poly}(n,r)$ steps, as $\ell \leq n^2$, as desired.

$f \in \mathcal{T}_{\mathcal{H}}$ are homogeneous: This is clear by construction.

$|T_{\mathcal{H}}| = n^2|\mathcal{H}|$: For each $\ell \in [n^2]$, we use one invariant per point in $\mathcal{H}$.

$T_{\mathcal{H}} \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$: For any ℓ, Lemma 3.4 shows that the coefficients of $f_\ell(\vec{M},\vec{x})$ (when regarded as polynomials in $\mathbb{F}[\vec{M}][\vec{x}]$) are traces of products of the matrices in $\vec{M}$, so these coefficients are invariant under simultaneous conjugation of $\vec{M}$. It follows then for any α, $f_\ell(\vec{M},\alpha)$ is a linear combination of invariants, and thus is an invariant. As $T_{\mathcal{H}}$ consists of exactly such polynomials, it is contained in the ring of all invariants.

$\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ separates $\iff$ $T_{\mathcal{H}}$ separates: By Corollary 3.6, we see that for any $\vec{A}$ and $\vec{B}$, there is an $f \in \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ with $f(\vec{A}) \neq f(\vec{B})$ iff there is some $\ell \in [n^2]$ such that $f_\ell(\vec{A},\vec{x}) - f_\ell(\vec{B},\vec{x}) \neq 0$. By Lemma 3.2, $f_\ell(\vec{A},\vec{x}) - f_\ell(\vec{B},\vec{x})$ is computable by a width $\leq 2n^2$, depth ℓ, degree $< r$ ROABP, which by the addition of $n^2 - \ell$ dummy variables (say, to the end of $\vec{x}$), can be considered as a depth n^2 ROABP. Thus, as $\mathcal{H}$ is a hitting set for ROABPs of the relevant size, for any $\ell \in [n^2]$, $f_\ell(\vec{A},\vec{x}) - f_\ell(\vec{B},\vec{x}) \neq 0$ iff there is some $\vec{\alpha} \in \mathcal{H}$ such that $f_\ell(\vec{A},\vec{\alpha}) - f_\ell(\vec{B},\vec{\alpha}) \neq 0$, and thus iff there is an $\alpha \in \mathcal{H}$ such that the invariant $f'(\vec{M}) := f_\ell(\vec{M},\vec{\alpha}) \in T_{\mathcal{H}}$ separates $\vec{A}$ and $\vec{B}$. □

As done in Mulmuley's Theorem 3.6, we can conclude that the ring of invariants is integral over the subring generated by the separating invariants. This uses the following theorem of Derksen and Kemper [DK02] (using the ideas of geometric invariant theory [MFK94]), which we only state in our specific case, but does hold more generally.

Theorem 3.9 (Theorem 2.3.12, Derksen and Kemper [DK02], stated by Mulmuley in Theorem 2.11)**.** *Let $\mathbb{F}$ be an algebraically closed field of characteristic zero. Let $\mathcal{T}' \subseteq \mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ be a finite set of homogeneous separating invariants. Then $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is integral over the subring S generated by $\mathcal{T}'$.*

Combining Theorem 3.8 and Theorem 3.9 yields the following corollary.

Corollary 3.10. *Assume the setup of Construction 3.7. Let $\mathbb{F}$ be an algebraically closed field of characteristic zero. Then $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ is integral over the subring generated by $\mathcal{T}_{\mathcal{H}}$, a set of $n^2|\mathcal{H}|$ invariants with $\mathsf{poly}(t(n,r),n,r)$-explicit ABPs.*

Continuing with the dictionary of geometric invariant theory [MFK94], we can obtain the following deterministic black-box algorithm for testing of two orbit closures intersect.

Corollary 3.11. *Assume the setup of Construction 3.7. Let $\mathbb{F}$ be a field of characteristic zero. There is an algorithm, running in deterministic $\mathsf{poly}(n,r,t(n,r),|\mathcal{H}|)$-time, in the unit cost arithmetic model, such that given $\vec{A},\vec{B} \in (\mathbb{F}^{[n]\times[n]})^{[r]}$, one can decide whether the orbit closures of $\vec{A}$ and $\vec{B}$ under simultaneous conjugation have an empty intersection. Further, this algorithm is black-box, as it only compares $f(\vec{A})$ and $f(\vec{B})$ for various polynomials f.*

Proof. As observed in Mulmuley's Theorem 3.8, the orbit closures of $\vec{A}$ and $\vec{B}$ intersect iff all invariants in $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ agree on $\vec{A}$ and $\vec{B}$. Our Theorem 3.8 shows this question can be answered by testing if $\vec{A}$ and $\vec{B}$ agree with respect to all $f \in \mathcal{T}_{\mathcal{H}}$, and this can be tested in $\mathsf{poly}(n, r, |\mathcal{H}|, t(n, r))$-time, as ABPs can be evaluated quickly. □

Thus, the above results, Theorem 3.8, Corollary 3.10, and Corollary 3.11 give positive results to Questions 1.8, 1.6, and 1.9 respectively, assuming small explicit hitting sets for ROABPs. Plugging in the hitting sets results of Forbes and Shpilka [FS12] as cited in Theorem 1.12, we obtain Theorem 1.13 and Corollary 1.14.

However, using the hitting set of Theorem 1.12 does not allow us to deduce the efficient algorithm for orbit closure intersection claimed in Theorem 1.15 as the hitting set is too large. To get that result, we observe that deciding the orbit closure intersection problem does not require black-box PIT, and that white-box PIT suffices. Thus, invoking the white-box results of Raz and Shpilka [RS05], and the follow-up work by Arvind, Joglekar and Srinivasan [AJS09], we can get the desired result.

Theorem (Theorem 1.15)**.** *Let $\mathbb{F}$ be a field of characteristic zero. There is an algorithm, running in deterministic $\mathsf{polylog}(n, r)$-time using $\mathsf{poly}(n, r)$-processors (NC), in the unit cost arithmetic model, such that given $\vec{A}, \vec{B} \in (\mathbb{F}^{[\![n]\!] \times [\![n]\!]})^{[\![r]\!]}$, one can decide whether the orbit closures of $\vec{A}$ and $\vec{B}$ under simultaneous conjugation have an empty intersection.*

Proof. As observed in Mulmuley's Theorem 3.8, the orbit closures of $\vec{A}$ and $\vec{B}$ intersect iff all invariants in $\mathbb{F}[\vec{M}]^{\mathrm{GL}_n(\mathbb{F})}$ agree on $\vec{A}$ and $\vec{B}$. Our results, Corollary 3.6 and Lemma 3.2, show there is a non-empty intersection iff an n^2-sized set of $\mathsf{poly}(n, r)$-size ROABPs all compute the zero polynomial.

Raz and Shpilka [RS05] gave a polynomial-time algorithm (in the unit-cost arithmetic model) for deciding if a non-commutative ABP computes the zero polynomial, and ROABPs are a special case of the non-commutative ABP model because the oblivious nature ensures that all multiplications of the variables are in the same order and thus commutativity is never exploited. Thus, by applying this algorithm to all of the ROABPs $f_\ell(\vec{A}, \vec{x}) - f_\ell(\vec{B}, \vec{x})$, we can decide if the orbit closures of $\vec{A}$ and $\vec{B}$ intersect in polynomial time, thus in P.

Further, Arvind, Joglekar and Srinivasan [AJS09] observed that the Raz-Shpilka [RS05] algorithm can be made parallel (within NC^3) while still running in polynomial-time, by using parallel linear algebra. Arvind, Joglekar and Srinivasan [AJS09] also gave an alternate, characteristic-zero specific, parallel algorithm for this problem (within NC^2). Hence, one can test each of "$f_\ell(\vec{A}, \vec{x}) - f_\ell(\vec{B}, \vec{x}) \equiv 0$?" in parallel, and then return the "and" of all of these tests, again in parallel. Thus, this gives a parallel polynomial-time (NC) algorithm for testing if orbit closures interesect. □

We comment briefly on space-bounded boolean computation, and its relation with this work. As noted in Forbes and Shpilka [FS12], the ROABP model is a natural algebraic analogue of space-bounded boolean computation and the hitting sets given by Forbes and Shpilka [FS12] can be seen as an algebraic analogue of the boolean pseudorandom generator (PRG) given by Nisan [Nis92]. First, we note that by this analogy, and the fact that subsequent work by Nisan [Nis94] showed that the PRG of Nisan [Nis92] can be made polynomial-time with additional space, one expects that the quasi-polynomial-time blackbox identity test (or hitting set) of Forbes and Shpilka [FS12] can be made into a parallel polynomial-time whitebox identity test for ROABPs, which would bring the proof of Theorem 1.15 more in line with the other parts of this paper. However, the NC^3 version of the Raz-Shpilka [RS05] algorithm is simpler than any modification of the Forbes-Shpilka [FS12] result, we do not pursue the details here.

By this connection, it follows that one can convert the white-box PIT problem for ROABPs into a derandomization question in small-space computation (once the bit-lengths of the numbers involved are bounded). Thus, one could avoid the algorithms of Raz-Shpilka [RS05] and Arvind, Joglekar and Srinivasan [AJS09], and simply use the algorithm of Nisan [Nis94]. However, this is less clean, and furthermore this booleanization cannot give similar results to Theorem 3.8, since one cannot a priori bound the bit-length of the matrices $\vec{A}$ and $\vec{B}$.

We note here that the above result for testing intersection of orbit closures is stated in the unit-cost arithmetic model for simplicity. At its core, the result uses linear algebra, which can be done efficiently even when the bit-lengths of the numbers are considered. Thus, it seems likely the above algorithm is also efficient with respect to bit-lengths, but we do not pursue the details here.

4 Equivalence of Trace of Matrix Powers, Determinants and ABPs

In this section we study the class of polynomials computed by small traces of matrix powers, to gain insight into the strength of the derandomization hypotheses that Mumuley requires for his implications regarding Noether Normalization. In particular, we show that a polynomial can be computed as a small trace of matrix power iff it can be computed by a small ABP, as defined in Definition 1.11.

We first show that from the perspective of the hardness of derandomizing PIT, a trace of matrix power can be either defined using linear or affine forms, so that we are not losing generality in our equivalence results below, when using the affine definition. Specifically, we want to relate the complexity of PIT for $\mathrm{trace}(A(\vec{x})^d)$, for an affine matrix $A(\vec{x}) = A_0 + \sum_{i=1}^n A_i x_i$, to the complexity of PIT of $\mathrm{trace}(A'(\vec{x}, z)^d)$, where A' is the homogenized version $A'(\vec{x}, z) := A_0 z + \sum_{i=1}^n A_i x_i$. Clearly, one can reduce the homogenized linear case back to the affine case, by taking $z = 1$, in both the black-box and white-box PIT model. We now consider reducing the affine case to the linear case. Note that this is trivial in the white-box model of PIT, as given the trace of matrix power $\mathrm{trace}(A(\vec{x})^d)$ we can easily construct the trace of matrix power $\mathrm{trace}(A'(\vec{x}, z)^d)$ by replacing constants by the appropriate multiple of z. The next lemma shows that these two traces are also polynomially equivalent in the black-box model.

Lemma 4.1. *Let $A(x_1, \ldots, x_n) = A_0 + \sum_{i=1}^n A_i x_i$ be matrix of affine forms. Define its homogenization $A'(\vec{x}, z) = A_0 z + \sum_{i=1}^n A_i x_i$. Then for any $\vec{\alpha}$ and β, $\mathrm{trace}(A'(\vec{\alpha}, \beta)^d)$ can be computed using $\mathsf{poly}(d)$ queries to $\mathrm{trace}(A(\vec{x})^d)$.*

Proof. $\underline{\beta \neq 0}$: Observe that by homogeneity and linearity of the trace, we have that $\mathrm{trace}(A'(\vec{\alpha}, \beta)^d) = \beta^d \,\mathrm{trace}(A(\vec{\alpha}/\beta)^d)$, where $\vec{\alpha}/\beta$ is the resulting of dividing $\vec{\alpha}$ by β coordinate-wise. Thus, only 1 query is needed in this case.

$\underline{\beta = 0}$: Writing $y\vec{x}$ for the coordinate-wise multiplication of y on $\vec{x}$, we can expand the trace of matrix power in the variable y, so that $\mathrm{trace}(A(y\vec{x})^d) = \sum_j \mathfrak{C}_{y^j}(\mathrm{trace}(A(y\vec{x})^d))$, where $\mathfrak{C}_{y^j}$ extracts the relevant coefficient in y, resulting in a polynomial in $\vec{x}$. It follows from polynomial interpolation that in $d+1$ queries to $\mathrm{trace}(A(y\vec{x})^d)$ we can compute $\mathfrak{C}_{y^j}(\mathrm{trace}(A(y\vec{x})^d))$ for any j. Observing that $\mathrm{trace}(A'(\vec{\alpha}, 0)^d) = \mathfrak{C}_{y^d}(\mathrm{trace}(A(y\vec{x})^d))$ yields the result. □

Thus as $\mathrm{trace}(A'(\vec{x}, z)^d) = 0$ iff $\mathrm{trace}(A(\vec{x})^d) = 0$, solving black-box PIT for $\mathrm{trace}(A'(\vec{x}, z)^d)$ solves it for $\mathrm{trace}(A(\vec{x})^d)$, and the above lemma gives the needed query-access reduction. Thus, as the linear and affine models are equivalent with respect to PIT, we now only discuss traces of matrix powers in the affine case, and seek to show this model is computationally equivalent (not just with respect to PIT) to the ABP model. We first establish that traces of matrix powers can efficiently simulate ABPs. To do this, we first study matrix powers and how they interact with traces.

Lemma 4.2. *Let $x_0, \dots, x_{d-1}$ be formally non-commuting variables, and let R be any commutative ring. Define $A(\vec{x}) \in R[x_0, \dots, x_{d-1}]^{[\![d]\!] \times [\![d]\!]}$ by*

$$A(\vec{x})_{i,j} = \begin{cases} x_i & \text{if } j = i+1 \pmod d \\ 0 & \text{else} \end{cases}$$

Then $\operatorname{trace}(A(\vec{x})^d) = \sum_{i \in [\![d]\!]} x_i x_{i+1} \cdots x_{(i-1 \bmod d)}$.

Proof. The matrix A defines an adjacency matrix on a d-vertex graph, which is the directed cycle with weights $x_0, x_1, \dots, x_{d-1}$ ordered cyclically. Raising A to the d-power corresponds to the adjacency matrix for the length-d walks on the length-d cycle. The only such walks are single traversals of the cycle, starting and ending at some vertex i, and these walks have weight $x_i x_{i+1} \cdots x_{(i-1 \bmod d)}$. Taking the trace of A^d corresponds to summing the weights of these walks, giving the desired formula. □

Alternate proof of Lemma 4.2. By definition,

$$(A^d)_{i,i} = \sum_{i_1, \dots, i_{d-1}} A_{i,i_1} A_{i_1,i_2} \cdots A_{i_{d-1},i}.$$

It follows that the only nonzero contribution is when $i_j = i_{j-1} + 1$ for all j, when defining $i_0 = i_d = i$ and working modulo d, and that this yields $x_i x_{i+1} \cdots x_{(i-1 \bmod d)}$. The claim follows by summing over i. □

As this result holds even when the variables x_i are non-commuting, we can use this lemma over the ring of matrices and thus embed matrix multiplication (over varying matrices) to matrix powering (of a single matrix).

Corollary 4.3. *Let R be a commutative ring, and let $M_1, \dots, M_d \in R^{[\![n]\!] \times [\![n]\!]}$ be matrices. Define the larger matrix $A \in R^{[\![nd]\!] \times [\![nd]\!]}$ by treating A as a block matrix in $(R^{[\![n]\!] \times [\![n]\!]})^{[\![d]\!] \times [\![d]\!]}$, so that*

$$A_{i,j} = \begin{cases} M_i & \text{if } j = i+1 \pmod d \\ 0 & \text{else} \end{cases}$$

Then $\operatorname{trace}(A^d) = d \operatorname{trace}(M_1 \cdots M_d)$.

Proof. The properties of block-matrix multiplication imply that we can treat the M_i as lying in the non-commutative ring of matrices, and thus we apply Lemma 4.2 to the trace of A^d to see that

$$\begin{aligned} \operatorname{trace}(A^d) &= \sum_{i=1}^{d} \operatorname{trace}(M_i M_{i+1} \cdots M_{(i-1 \bmod d)}) \\ &= d \operatorname{trace}(M_1 M_2 \cdots M_d) \end{aligned}$$

where the second equality uses that trace is cyclic. □

This lemma shows that the trace of a matrix power can embed the trace of a matrix product (up to the factor d), and Lemma 2.2 shows that traces of matrix products capture ABPs. This leads to the equivalence of traces of matrix powers and ABPs.

Theorem 4.4. *Let $\mathbb{F}$ be a field. If a polynomial f is computable by a width w, depth d ABP, then for any $d' \geq d$ such that $\mathrm{char}(\mathbb{F}) \nmid d'$, f can be computed by a width wd', depth d' trace of matrix power. In particular, $d' \in \{d, d+1\}$ suffices.*

Conversely, if a polynomial f is computable by a width w, depth d trace of matrix power, then f can also be computed by a width w^2, depth d ABP.

Proof. ABP $\implies$ trace of matrix power: By Lemma 2.2 there are $[\![w]\!] \times [\![w]\!]$ matrices $M_1, \ldots, M_d$ whose entries are affine forms in $\vec{x}$, such that $f(\vec{x}) = \mathrm{trace}(M_1 \cdots M_d)$. For $i \in [d']$, define new matrices M'_i such that for $i = 1$, $M'_i := M_i/d'$, for $1 < i \leq d$, we set $M'_i := M_i$, and for $i > d$, define $M'_i := \mathrm{I}_w$, the $[\![w]\!] \times [\![w]\!]$ identity matrix. By linearity of the trace, it follows that $f(\vec{x}) = d' \cdot \mathrm{trace}(M_1 \cdots M_d M_{d+1} \cdots M_{d'})$. Noting that the M'_i have entries that are all affine forms, Corollary 4.3 implies that there is an $[\![wd']\!] \times [\![wd']\!]$ matrix $A(\vec{x})$ whose entries are affine forms in $\vec{x}$, such that $\mathrm{trace}(A(\vec{x})^{d'}) = d' \cdot \mathrm{trace}(M_1 \cdots M_{d'}) = f(\vec{x})$, as desired. Noting that d and $d+1$ cannot both be divisible by the characteristic of $\mathbb{F}$ completes the claim.

trace of matrix power $\implies$ ABP: Suppose $f(\vec{x}) = \mathrm{trace}(A(\vec{x})^d)$, where $A(\vec{x})$ is a $[\![w]\!] \times [\![w]\!]$ matrix of affine forms. Note then that for each i, the (i,i)-th entry of $A(\vec{x})^d$ is computable by a width w, depth d ABP, as established by Lemma 2.2, after applying the suitable permutation of indices. As the trace is the summation over i of these functions, we can apply Lemma 2.3 to get the result. □

Thus the above shows that, up to polynomial factors in size, ABPs and traces of matrix powers compute the same polynomials. We note that there is also an equivalent computational model, defined by the determinant, which we mention for completeness.

Definition 4.5. *A polynomial $f(x_1, \ldots, x_n)$ is a width w **projection of a determinant** if there exists a matrix $A(\vec{x}) \in \mathbb{F}[x_1, \ldots, x_n]^{[\![w]\!] \times [\![w]\!]}$ whose entries are affine functions in $\vec{x}$ such that $f(\vec{x}) = \det(A(\vec{x}))$. The **size** of a projection of a determinant is nw.*

Valiant [Val79] (see also [MP08]) showed any small ABP can be simulated by a small projection of a determinant. Conversely, phrased in the language of this paper, Berkowitz [Ber84] gave a small ABP for computing a small projection of a determinant. Thus, the projection of determinant model is also equivalent to the ABP model. We summarize these results in the following theorem.

Theorem 4.6. *The computational models of algebraic branching programs, traces of matrix powers, and projections of determinants are equivalent, up to polynomial blow up in size.*

In particular, this implies that derandomizing black-box PIT is equally hard for all three of these computational models.

5 Hasse Derivatives

In this section we define Hasse derivatives, which are a variant of (formal) partial derivatives but work better over finite fields. For completeness, we will derive various properties of Hasse derivatives. We start with the definition.

Definition 5.1. *Let R be a commutative ring, and $R[\vec{x}]$ be the ring of n-variate polynomials. For a vector $\vec{u} \in R^n$ and $k \geq 0$, define $\partial_{\vec{u}^k}(f) : R[\vec{x}] \to R[\vec{x}]$, the **$k$-th Hasse derivative of f in direction** $\vec{u}$, by $\partial_{\vec{u}^k}(f) = \mathfrak{C}_{y^k}(f(\vec{x} + \vec{u}y)) \in R[\vec{x}]$, where $\vec{x} + \vec{u}y$ is defined as the vector whose i-th coordinate in $x_i + u_i y$.*

*If $\vec{u} = \vec{e}_i$, then we use $\partial_{x_i^k}$ to denote $\partial_{\vec{e}_i^k}$, and will call this **the k-th Hasse derivative with respect to the variable** x_i. For $\vec{i} \in \mathbb{N}^n$, we will define $\partial_{\vec{x}^{\vec{i}}} := \partial_{x_1^{i_1}} \cdots \partial_{x_n^{i_n}}$.*

Note that for $k = 1$, this is usual (formal) partial derivative. We now use this definition to establish some basic properties of the Hasse derivative. In particular, the below commutativity property shows that the definition of $\partial_{\vec{x}^i}$ is not dependent on the order of the variables. Note that while Hasse derivatives are linear operators on $R[\vec{x}]$, they are not linear in the direction $\vec{u}$ of the derivative, and so several of these properties will be stated for more than two terms.

Lemma 5.2. *For $f, g \in R[\vec{x}]$, $\vec{u}, \vec{v} \in R^n$, $\alpha, \beta \in R$, and $k, \ell, k_1, \ldots, k_m \geq 0$, then*

1. $\partial_{\vec{u}^0}(f) = f$
2. $\partial_{\vec{u}^k}(\alpha f + \beta g) = \alpha \partial_{\vec{u}^k}(f) + \beta \partial_{\vec{u}^k}(g)$
3. $f(\vec{x} + \vec{u}_1 y_1 + \cdots + \vec{u}_m y_m) = \sum_{k_1,\ldots,k_m \geq 0} \partial_{\vec{u}_m^{k_1}} \cdots \partial_{\vec{u}_m^{k_m}}(f) y_1^{k_1} \cdots y_m^{k_m}$
4. $\partial_{\vec{u}_1^{k_1}} \cdots \partial_{\vec{u}_m^{k_m}}(f) = \mathfrak{C}_{y_1^{k_1} \cdots y_m^{k_m}}(f(\vec{x} + \vec{u}_1 y_1 + \cdots + \vec{u}_m y_m))$
5. $\partial_{\vec{u}^k} \partial_{\vec{v}^\ell}(f) = \partial_{\vec{v}^\ell} \partial_{\vec{u}^k}(f)$
6. $\partial_{\left(\sum_{j=1}^m \alpha_j \vec{u}_j\right)^k}(f) = \sum_{k_1+\cdots+k_m=k} \left(\prod_j \alpha_j^{k_j}\right) \partial_{\vec{u}_1^{k_1}} \cdots \partial_{\vec{u}_m^{k_m}}(f)$
7. $\partial_{\vec{u}^{k_1}} \cdots \partial_{\vec{u}^{k_m}}(f) = \binom{k_1+\cdots+k_m}{k_1,\ldots,k_m} \partial_{\vec{u}^{k_1+\cdots+k_m}}$

Proof. (1): This is trivial.

(2): This follows from the linearity of $\mathfrak{C}_{y^k}(\cdot)$, and so

$$\mathfrak{C}_{y^k}(\alpha f(\vec{x} + \vec{u}y) + \beta g(\vec{x} + \vec{u}y)) = \alpha\, \mathfrak{C}_{y^k}(f(\vec{x} + \vec{u}y)) + \beta\, \mathfrak{C}_{y^k}(g(\vec{x} + \vec{u}y)))$$

(3): This will be proven by induction on m.

$m = 1$: This is the definition of the Hasse derivative.

$m > 1$: By induction, we have that

$$f(\vec{x} + \vec{u}_2 y_2 + \cdots + \vec{u}_m y_m) = \sum_{k_2,\ldots,k_m \geq 0} [\partial_{\vec{u}_2^{k_2}} \cdots \partial_{\vec{u}_m^{k_m}}(f)](\vec{x}) y_2^{k_2} \cdots y_m^{k_m}.$$

Making the substitution $\vec{x} \leftarrow \vec{x} + \vec{u}_1 y_1$ we obtain

$$f(\vec{x} + \vec{u}_1 y_1 + \vec{u}_2 y_2 + \cdots + \vec{u}_m y_m) = \sum_{k_2,\ldots,k_m \geq 0} [\partial_{\vec{u}_2^{k_2}} \cdots \partial_{\vec{u}_m^{k_m}}(f)](\vec{x} + \vec{u}_1 y_1) y_2^{k_2} \cdots y_m^{k_m}$$

and so by expanding $\partial_{\vec{u}_2^{k_2}} \cdots \partial_{\vec{u}_m^{k_m}}(f)$ into its Hasse derivatives, we obtain

$$f(\vec{x} + \vec{u}_1 y_1 + \vec{u}_2 y_2 + \cdots + \vec{u}_m y_m) = \sum_{k_1,k_2,\ldots,k_m \geq 0} [\partial_{\vec{u}_1^{k_1}} \partial_{\vec{u}_2^{k_2}} \cdots \partial_{\vec{u}_m^{k_m}}(f)](\vec{x}) y_1^{k_1} y_2^{k_2} \cdots y_m^{k_m}$$

as desired.

(4): This is a restatement of (3).

(5): This follows immediately from (4), taking $m = 2$, as $\mathfrak{C}_{z^\ell y^k}(f(\vec{x} + \vec{v}z + \vec{u}y)) = \mathfrak{C}_{y^k z^\ell}(f(\vec{x} + \vec{u}y + \vec{v}z))$.

(6): By (3) we have that

$$f(\vec{x} + \vec{u}_1 y_1 + \cdots + \vec{u}_m y_m) = \sum_{k_1,\ldots,k_m \geq 0} [\partial_{\vec{u}_1^{k_1}} \cdots \partial_{\vec{u}_m^{k_m}}(f)](\vec{x}) y_1^{k_1} \cdots y_m^{k_m}$$

and applying the substitution $y_j \leftarrow \alpha_j y$ we obtain

$$f(\vec{x} + (\alpha_1 \vec{u}_1 + \cdots + \alpha_m \vec{u}_m)y) = \sum_{k_1,\ldots,k_m \geq 0} \alpha_1^{k_1} \ldots \alpha_m^{k_m} [\partial_{\vec{u}_1^{k_1}} \cdots \partial_{\vec{u}_m^{k_m}}(f)](\vec{x}) y^{k_1+\cdots+k_m}$$

and thus taking the coefficient of y^k yields the result.

(7): By (4), we see that

$$\begin{aligned}
\partial_{\vec{u}^{k_1}} \cdots \partial_{\vec{u}^{k_m}}(f) &= = \mathfrak{C}_{y_1^{k_1} \cdots y_m^{k_m}}(f(\vec{x} + \vec{u}y_1 + \cdots + \vec{u}y_m)) \\
&= \mathfrak{C}_{y_1^{k_1} \cdots y_m^{k_m}}(f(\vec{x} + \vec{u}(y_1 + \cdots + y_m))) \\
&= \mathfrak{C}_{y_1^{k_1} \cdots y_m^{k_m}}\left(\sum_k \partial_{\vec{u}^k}(f)(\vec{x}) \cdot (y_1 + \cdots + y_m)^k\right) \\
&= \sum_k \partial_{\vec{u}^k}(f)(\vec{x}) \cdot \mathfrak{C}_{y_1^{k_1} \cdots y_m^{k_m}}\left((y_1 + \cdots + y_m)^k\right) \\
&= \binom{k_1 + \cdots + k_m}{k_1, \ldots, k_m} \partial_{\vec{u}^{k_1+\cdots+k_m}}(f)
\end{aligned}$$

□

We now recover the action of a partial derivative on a monomial.

Lemma 5.3. *For any $\ell \in [n]$, $k \geq 0$, and $i_\ell \geq 0$,*

$$\partial_{x_\ell^k}(x_1^{i_1} \cdots x_\ell^{i_\ell} \cdots x_n^{i_n}) = \binom{i_\ell}{k} x_1^{i_1} \cdots x_{\ell-1}^{i_{\ell-1}} x_\ell^{i_\ell - k} x_{\ell+1}^{i_{\ell+1}} \cdots x_n^{i_n}$$

Proof. We do the case where $\ell = 1$, as the general case is symmetric. Thus we want to understand the coefficient of y^k in $(x_1 + y)^{i_1} x_2^{i_2} \ldots x_n^{i_n}$. The binomial theorem tells us the coefficient of y^k in $(x_1 + y)^{i_1}$ is $\binom{i_1}{k} x_1^{i_1 - k}$ (even in the case that $i_1 = 0$, interpreted correctly). Plugging this into the rest of the monomial yields the result. □

We can now use the above properties to establish the product rule for Hasse derivatives.

Lemma 5.4 (Product Rule)**.** *For $f, g \in R[\vec{x}]$, $\vec{u} \in R^n$ and $k \geq 0$,*

$$\partial_{\vec{u}^k}(fg) = \sum_{i+j=k} \partial_{\vec{u}^i}(f) \partial_{\vec{u}^j}(g)$$

Proof.

$$\begin{aligned}
(fg)(\vec{x} + \vec{u}y) &= f(\vec{x} + \vec{u}y) g(\vec{x} + \vec{u}y) \\
&= \left(\sum_i \partial_{\vec{u}^i}(f) y^i\right)\left(\sum_j \partial_{\vec{u}^j}(g) y^j\right) \\
&= \sum_k \sum_{i+j=k} \partial_{\vec{u}^i}(f) \partial_{\vec{u}^j}(g) y^k
\end{aligned}$$

and result follows by taking the coefficient of y^k. □

We now will establish the chain rule for Hasse derivatives. As Hasse derivatives necessarily take multiple derivatives all at once, the chain rule we derive will be more complicated than the usual chain rule for (formal) partial derivatives, which was only for a single partial derivative. This formula, and its variants, are sometimes called Faà di Bruno's formula. The below formula is written with vector exponents, as explained in Subsection 1.5, and the ∂ operators applied to vectors of polynomials are defined coordinate-wise.

Lemma 5.5 (Chain Rule)**.** *For $f \in R[\vec{x}]$ and $g_1, \ldots, g_n \in R[\vec{y}]$,*

$$\partial_{\vec{u}^k}(f(g_1,\ldots,g_n)) \quad = \quad \sum_{\sum_{j=1}^k j|\vec{\ell}_j|=k} \left(\prod_{j=1}^k [\partial_{\vec{u}^j}(\vec{g})(\vec{y})]^{\vec{\ell}_j}\right) \binom{\vec{\ell}_1+\cdots+\vec{\ell}_k}{\vec{\ell}_1,\ldots,\vec{\ell}_k} \left[\partial_{\vec{x}^{\sum_{j=1}^k \vec{\ell}_j}}(f)\right](\vec{g}(\vec{y}))$$

Proof.

$$\begin{aligned}
\partial_{\vec{u}^k}(f(g_1,\ldots,g_n)) &= \mathfrak{C}_{z^k}\left((f\circ\vec{g})(\vec{y}+\vec{u}z)\right) \\
&= \mathfrak{C}_{z^k}\left(f\left(\sum_{\ell}\partial_{\vec{u}^\ell}(\vec{g})(\vec{y})z^\ell\right)\right) \\
&= \mathfrak{C}_{z^k}\left(f\Big(\vec{g}(\vec{y})+\partial_{\vec{u}}(\vec{g})(\vec{y})z+\cdots+\partial_{\vec{u}^k}(\vec{g})(\vec{y})z^k\Big)\right)
\end{aligned}$$

We now seek to take derivatives "in the direction of $\partial_{\vec{u}^j}(\vec{g})(\vec{y})$" from the point $\vec{x} := \vec{g}(\vec{y})$, treating each j as a different direction and each z^j as a different variable. However, this is a subtle operation, as up until now we have taken the directions of our derivatives as independent of the point of derivation. To make this subtlety clear, we now study the above equation, by "undoing" the substitutions $\vec{x} \leftarrow \vec{g}(\vec{y})$, and $z_j \leftarrow z^j$, and working with derivatives in the ring $R[\vec{y}][\vec{x}]$. We will then later "redo" these substitutions. A simpler form of this logic was used to establish Lemma 5.2.6. We start about by applying Lemma 5.2.3 to expand out f in the directions $\partial_{\vec{u}^j}(\vec{g})(\vec{y})$.

$$\begin{aligned}
&f\Big(\vec{x}+\partial_{\vec{u}}(\vec{g})(\vec{y})z_1+\cdots+\partial_{\vec{u}^k}(\vec{g})(\vec{y})z_k\Big) \\
&\quad = \sum_{\ell_1,\ldots,\ell_k\geq 0}\left[\partial_{[\partial_{\vec{u}}(\vec{g})(\vec{y})]^{\ell_1}}\cdots\partial_{[\partial_{\vec{u}^k}(\vec{g})(\vec{y})]^{\ell_k}}(f)\right](\vec{x})\cdot z_1^{\ell_1}\cdots z_k^{\ell_k}
\end{aligned}$$

and by Lemma 5.2.6, we can decompose the derivative $\partial_{\vec{u}^j}(\vec{g})(\vec{y})$ as linear combinations of the $\partial_{x_i^k}$ derivatives, so that as operators on $R[\vec{y}][\vec{x}]$,

$$\partial_{[\partial_{\vec{u}^j}(\vec{g})(\vec{y})]^{\ell_j}} = \sum_{\ell_{j,1}+\cdots+\ell_{j,n}=\ell_j}\left(\prod_{i=1}^n [\partial_{\vec{u}^j}(g_i)(\vec{y})]^{\ell_{j,i}}\right)\left[\partial_{x_1^{\ell_{j,1}}}\cdots\partial_{x_n^{\ell_{j,n}}}\right]$$

As these operators are acting on $R[\vec{y}][\vec{x}]$ we can treat all polynomials in $\vec{y}$ as constants, in particular the terms involving the $\partial_{\vec{u}^j}(g_i)(\vec{y})$ above. This allows us to move all of the operators past the terms involving the g_i, to obtain,

$$= \sum_{\substack{\sum_{i=1}^n \ell_{j,i}=\ell_j \\ \ell_j\geq 0 \\ j\in[k]}}\left(\prod_{i=1,j=1}^{n,k}[\partial_{\vec{u}^j}(g_i)(\vec{y})]^{\ell_{j,i}}\right)\left[\partial_{x_1^{\ell_{1,1}}}\cdots\partial_{x_n^{\ell_{1,n}}}\cdots\partial_{x_1^{\ell_{k,1}}}\cdots\partial_{x_n^{\ell_{k,n}}}(f)\right](\vec{x})\cdot z_1^{\ell_1}\cdots z_k^{\ell_k}$$

By invoking Lemma 5.2.7 we can clump derivatives by variable,

$$= \sum_{\substack{\sum_{i=1}^{n} \ell_{j,i} = \ell_j \\ \ell_j \geq 0 \\ j \in [k]}} \left(\prod_{i=1,j=1}^{n,k} [\partial_{\vec{u}^j}(g_i)(\vec{y})]^{\ell_{j,i}} \right) \left(\prod_{i=1}^{n} \binom{\ell_{1,i} + \cdots + \ell_{k,i}}{\ell_{1,i}, \ldots, \ell_{k,i}} \right)$$

$$\left[\partial_{x_1^{\sum_{j=1}^{k} \ell_{j,1}}} \cdots \partial_{x_n^{\sum_{j=1}^{k} \ell_{j,n}}} (f) \right] (\vec{x}) \cdot z_1^{\ell_1} \cdots z_k^{\ell_k}$$

We can "redo" the substitutions $\vec{x} \leftarrow \vec{g}(\vec{y})$, and $z_j \leftarrow z^j$, to obtain that,

$$\partial_{\vec{u}^k}(f(g_1, \ldots, g_n)) = \mathfrak{C}_{z^k} \left(f\left(\vec{g}(\vec{y}) + \partial_{\vec{u}}(\vec{g})(\vec{y}) z + \cdots + \partial_{\vec{u}^k}(\vec{g})(\vec{y}) z^k \right) \right)$$

$$= \sum_{\substack{\sum_{i=1}^{n} \ell_{j,i} = \ell_j \\ \sum_{j=1}^{k} j \ell_j = k}} \left(\prod_{i=1,j=1}^{n,k} [\partial_{\vec{u}^j}(g_i)(\vec{y})]^{\ell_{j,i}} \right) \left(\prod_{i=1}^{n} \binom{\ell_{1,i} + \cdots + \ell_{k,i}}{\ell_{1,i}, \ldots, \ell_{k,i}} \right)$$

$$\left[\partial_{x_1^{\sum_{j=1}^{k} \ell_{j,1}}} \cdots \partial_{x_n^{\sum_{j=1}^{k} \ell_{j,n}}} (f) \right] (\vec{g}(\vec{y}))$$

and rewriting things in vector notation,

$$= \sum_{\sum_{j=1}^{k} j |\vec{\ell}_j| = k} \left(\prod_{j=1}^{k} [\partial_{\vec{u}^j}(\vec{g})(\vec{y})]^{\vec{\ell}_j} \right) \binom{\vec{\ell}_1 + \cdots + \vec{\ell}_k}{\vec{\ell}_1, \ldots, \vec{\ell}_k} \left[\partial_{\vec{x}^{\sum_{j=1}^{k} \vec{\ell}_j}} (f) \right] (\vec{g}(\vec{y})) \qquad \square$$

6 Hitting Sets for Depth-3 Diagonal Circuits

In this section we construct hitting sets for the depth-3 diagonal circuit model, as defined by Saxena [Sax08]. In this section we will use some additional notation. For a vector $\vec{e} \in \mathbb{N}^n$, we define $\mathrm{Supp}(\vec{e})$ to be its support $S \subseteq [n]$, $|\vec{e}|_0 := |\mathrm{Supp}(\vec{e})|$ and $|\vec{e}|_\times := \prod_{\ell=1}^{n}(e_\ell + 1)$. We now define the depth-3 diagonal circuit model.

Definition 6.1 (Saxena [Sax08])**.** *A polynomial $f(x_1, \ldots, x_n)$ is computable by a **depth-3 diagonal circuit** if*

$$f(\vec{x}) = \sum_{\ell=1}^{s} \vec{L}_\ell^{\vec{e}_\ell}(\vec{x}) \,,$$

*where each $\vec{L}_\ell$ is a vector of affine functions. The **size** is $n \sum_{\ell=1}^{s} |\vec{e}_\ell|_\times$.*

The hitting sets will actually be for any polynomial whose space of partial derivatives is low-dimensional. We now define this, using the notion of Hasse derivatives from Section 5 so the results apply over any characteristic.

Definition 6.2. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$. The **dimension of Hasse derivatives** of f, denoted $|\partial(f)|$, is defined as*

$$|\partial(f)| := \dim\{\partial_{\vec{x}^{\vec{i}}}(f) | \vec{i} \in \mathbb{N}^n\} \,.$$

The dimension is taken in the $\mathbb{F}$-vector space $\mathbb{F}[x_1, \dots, x_n]$. Note that by Lemma 5.2 it follows that all iterated Hasse derivatives, even with arbitrary directions, are contained in the above space. This dimension is also well-behaved in various respects, such as being sub-additive, which we now establish.

Lemma 6.3. *Let $f, g \in \mathbb{F}[x_1, \dots, x_n]$. Then $|\partial(f+g)| \leq |\partial(f)| + |\partial(g)|$.*

Proof. As Hasse derivatives are linear (Lemma 5.2) it follows that

$$\operatorname{span}\{\partial_{\vec{x}^{\vec{i}}}(f+g) | \vec{i} \in \mathbb{N}^n\} \subseteq \operatorname{span}\left(\{\partial_{\vec{x}^{\vec{i}}}(f) | \vec{i} \in \mathbb{N}^n\}, \{\partial_{\vec{x}^{\vec{i}}}(g) | \vec{i} \in \mathbb{N}^n\}\right) ,$$

and thus taking dimensions finishes the claim. □

We now work to proving that depth-3 diagonal circuits have low-dimensional spaces of partial derivatives. We first study the partial derivatives of a single monomial.

Lemma 6.4. *Let $\vec{x}^{\vec{i}} \in \mathbb{F}[x_1, \dots, x_n]$. Then $|\partial(\vec{x}^{\vec{i}})| \leq |\vec{i}|_\times$.*

Proof. Let $\vec{j} \in \mathbb{N}^n$, then

$$\partial_{\vec{x}^{\vec{j}}}(\vec{x}^{\vec{i}}) = \prod_{\ell \in [n]} \binom{i_\ell}{j_\ell} x_\ell^{i_\ell - j_\ell} ,$$

where $\binom{i_\ell}{j_\ell} = 0$ if $i_\ell < j_\ell$. Thus, all possible Hasse derivatives are some non-zero scalar multiple of $\vec{x}^{\vec{k}}$ for any $\vec{k} \leq \vec{i}$, and there are $|\vec{i}|_\times$ such $\vec{k}$, giving the desired bound. □

Note that this upper bound is an equality over characteristic zero, but not over finite characteristic, as seen by x^p in characteristic p, where $|\partial(x^p)| = 2$ but $|p|_\times = p + 1$. However, this slack does not qualitatively affect the results. We now extend this dimension bound by using the chain rule.

Lemma 6.5. *Let $f \in \mathbb{F}[\vec{y}]$, and let $\vec{L} \in \mathbb{F}[\vec{x}]$ be a vector of affine forms. Then $|\partial(f \circ \vec{L})| \leq |\partial(f)|$.*

Proof. Consider some derivative $\partial_{\vec{x}^{\vec{i}}}(f \circ \vec{L}) = \partial_{x_1^{i_1}} \cdots \partial_{x_n^{i_n}}(f \circ \vec{L})$. By the chain rule (Lemma 5.5) we see that $\partial_{x_n^{i_n}}(f \circ \vec{L})$ is a linear combination of Hasse derivatives of f, evaluated at $\vec{L}(\vec{x})$, as the Hasse derivatives of $\vec{L}(\vec{x})$ are constants. Since Lemma 5.2 shows that derivatives of derivatives are (scalar multiples of) derivatives, we see that we can induct downwards on ℓ to obtain that $\partial_{x_\ell^{i_\ell}} \cdots \partial_{x_n^{i_n}}(f \circ \vec{L})$ is a linear combination of Hasse derivatives of f, evaluated at $\vec{L}(\vec{x})$. Taking $\ell = 1$, and noting that evaluating the Hasse derivatives of f at $\vec{L}(\vec{x})$ cannot increase dimension, the claim follows. □

Combing the above result with sub-additivity of the dimension of derivatives, we can now bound the dimension of depth-3 diagonal circuits. Such bounds are not attainable for the depth-4 diagonal circuit model (which we did not define), as that model contains the polynomial $(\sum_{\ell=1}^n x_\ell^2)^n$, which has an exponentially large space of derivatives. As mentioned in the introduction, there are other works ([ASS12] and [FS12]) that handle the depth-4 case, using other techniques.

Lemma 6.6. *Let $f(\vec{x}) = \sum_{\ell=1}^s \vec{L}_\ell(\vec{x})^{\vec{e}_\ell}$ be a depth-3 diagonal circuit, where $\vec{L}_\ell(\vec{x})$ is an affine function. Then $|\partial(f)| \leq \sum_{\ell=1}^s |\vec{e}_\ell|_\times$.*

Proof. By sub-additivity of dimension of Hasse derivatives (Lemma 6.3) it suffices to prove the claim for $s = 1$. Thus consider some $f(\vec{x}) = \vec{L}(\vec{x})^{\vec{e}}$. Note that $f(\vec{x}) = g(\vec{L}(\vec{x}))$, where $g(\vec{y}) = \vec{y}^{\vec{e}}$. The claim then follows from Lemma 6.5 and Lemma 6.4. □

We now seek to show that any polynomial with low-dimensional derivatives must have a small-support monomial. To do so, we introduce the notion of a monomial ordering (see [CLO07] for more on monomial orderings) and establish facts about its interactions with derivatives.

Definition 6.7. *A **monomial ordering** is a total order $\prec$ on the non-zero monomials in $\mathbb{F}[\vec{x}]$ such that*

- *For all $\vec{i} \in \mathbb{N}^n$, $1 \prec \vec{x}^{\vec{i}}$.*
- *For all $\vec{i}, \vec{j}, \vec{k} \in \mathbb{N}^n$, $\vec{x}^{\vec{i}} \prec \vec{x}^{\vec{j}}$ implies $\vec{x}^{\vec{i}+\vec{k}} \prec \vec{x}^{\vec{j}+\vec{k}}$*

For concreteness, one can consider the lexicographic ordering on monomials, which is easily seen to be a monomial ordering. We now observe that derivatives are monotonic with respect to monomial orderings, except when the characteristic prevents it. That is, over characteristic p we have $x^{p-1} \prec x^p$ (in any ordering), but $\partial_x(x^p) = 0$, which is not included in the ordering.

Lemma 6.8. *Let $\prec$ be a monomial ordering on $\mathbb{F}[\vec{x}]$. Let $\vec{x}^{\vec{i}} \prec \vec{x}^{\vec{j}}$ be monomials. Then for any $\vec{k}$, if $\partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{i}}), \partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{j}}) \neq 0$ then $\partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{i}}) \prec \partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{j}})$, where we abuse notation and ignore (non-zero) coefficients in the ordering.*

Proof. By Lemma 5.3, the assumptions of $\partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{i}}), \partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{j}}) \neq 0$ imply that $\vec{k} \leq \vec{i}, \vec{j}$, and that

$$\partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{i}}) = a\vec{x}^{\vec{i}-\vec{k}}, \qquad \partial_{\vec{x}^{\vec{k}}}(\vec{x}^{\vec{j}}) = b\vec{x}^{\vec{j}-\vec{k}},$$

where $a, b \neq 0$ are constants. As the monomial ordering is total, and is monotonic over multiplication, it follows $\vec{x}^{\vec{i}-\vec{k}} \prec \vec{x}^{\vec{j}-\vec{k}}$, as desired. □

Monotonicity then implies that the largest monomial of a polynomial f will continue to be largest when taking derivatives, as long as it is not annihilated. Treating polynomial as vectors, this then gives us a diagonal system of vectors, from which we can deduce the following rank bound.

Theorem 6.9. *Let $f \in \mathbb{F}[\vec{x}]$ be a polynomial, and let $\prec$ be any monomial ordering in $\mathbb{F}[\vec{x}]$. Let $\vec{x}^{\vec{i}}$ be the largest monomial (with respect to $\prec$) with a non-zero coefficient in f. Then $|\vec{i}|_0 \leq \log |(\partial(f)|$.*

Proof. Consider the set of vectors $A \subseteq \mathbb{N}^n$ defined by

$$A := \{\vec{j} : \forall \ell \in [n], j_\ell \in \{0, i_\ell\}\} ,$$

so that all vectors in A have support contained in $\mathrm{Supp}(\vec{i})$. For a fixed $\vec{j} \in A$, linearity and Lemma 5.3 imply that

$$\partial_{\vec{x}^{\vec{j}}}(\vec{x}^{\vec{i}}) = \prod_{\ell \in \mathrm{Supp}(\vec{i}) \setminus \mathrm{Supp}(\vec{j})} x_\ell^{i_\ell} ,$$

which in particular is non-zero. Write f as $f = a\vec{x}^{\vec{i}} + g$, where all monomials in g are less than $\vec{x}^{\vec{i}}$ and $a \neq 0$. By Lemma 6.8 it follows that all non-zero monomials in $\partial_{\vec{x}^{\vec{j}}}(g)$ are less than $\partial_{\vec{x}^{\vec{j}}}(\vec{x}^{\vec{i}})$, so $a \prod_{\ell \in \mathrm{Supp}(\vec{i}) \setminus \mathrm{Supp}(\vec{j})} x_\ell^{i_\ell}$ is the leading term of $\partial_{\vec{x}^{\vec{j}}}(f)$. Thus, ranging $\vec{j}$ over all vectors in A, we get $2^{|\vec{i}|_0}$ derivatives of f, each with a different leading monomial. Thus, these polynomials are linearly independent, so it must be that $2^{|\vec{i}|_0} \leq |\partial(f)|$, giving the desired bound. □

Just as in Lemma 6.4, the characteristic of the underlying field affects the tightness of this result. In particular, in characteristic zero, one can improve this result to $|\vec{i}|_\times \leq \log |\partial(f)|$.

The above lemma shows that any f with a small-dimensional space of derivatives must have a small-support monomial. We now give a construction aimed at hitting any polynomial with such small-support monomials. Note that this will beat the union bound, in the sense that a union-bound argument for creating hitting sets against polynomials with small-support monomials will not yield small hitting sets as there are too many such polynomials. However, there is still a small hitting set, as we now construct.

Construction 6.10. *Let $n, d, m \geq 1$. Let $\mathbb{F}$ be a field of size $\geq d+1$. Let $S \subseteq \mathbb{F}$ with $|S| = d+1$. Define $\mathcal{H}' \subseteq \mathbb{F}^n$ by*

$$\mathcal{H}' := \{\vec{\alpha} : \vec{\alpha} \in S^n, |\vec{\alpha}|_0 \leq m\} .$$

We now establish the desired properties of this construction.

Theorem 6.11. *Assume the setup of Construction 6.10. Then $\mathcal{H}'$ is $\mathsf{poly}(n, d, m)$-explicit and has size $|\mathcal{H}'| \leq (nd)^m$, and for any $f \in \mathbb{F}[x_1, \ldots, x_n]$ of total degree $\leq d$, with $|\partial(f)| \leq 2^m$, $f = 0$ iff $f|_{\mathcal{H}'} \equiv 0$.*

Proof. $\mathcal{H}'$ is explicit: This is clear from construction.

$|\mathcal{H}'| \leq (nd)^m$: This is clear from construction.

$f = 0 \implies f|_{\mathcal{H}'} \equiv 0$: This is clear.

$f \neq 0 \implies f|_{\mathcal{H}'} \not\equiv 0$: By Theorem 6.9 it follows that f has a non-zero coefficient on a monomial $\vec{x}^{\vec{i}}$ where $\vec{i}$ has support $T \subseteq [n]$, with $|T| \leq \log |\partial(f)| \leq m$. Let $\vec{y}$ be a vector of variables indexed by T, and define $f_T(\vec{y})$ as $f(\vec{x})$ under the substitution that $x_\ell \leftarrow 0$ for $\ell \notin T$ and $x_\ell \leftarrow y_\ell$ for $\ell \in T$. As $\vec{x}^{\vec{i}}$ is not annihilated by this substitution, it follows that $f_T \neq 0$ and is of total degree $\leq d$. By construction $\mathcal{H}'|_T$ contains all tuples in $S^{|T|}$. As $|S| \geq d+1$ it follows from polynomial interpolation that $(f_T)|_{(\mathcal{H}'|_T)} \not\equiv 0$, and thus f has a non-zero evaluation $\vec{\alpha} \in S^n$ with $\mathrm{Supp}(\alpha) \subseteq T$. By construction of $\mathcal{H}'$, $\vec{\alpha} \in \mathcal{H}'$, and thus $f|_{\mathcal{H}'} \not\equiv 0$ as desired. □

By combining Theorem 6.11 with Lemma 6.6 we obtain the following hitting set for diagonal circuits.

Corollary 6.12. *Let $\mathbb{F}$ be a field with size $\geq d+1$. Then there is a $\mathsf{poly}(n, d, \log(s))$-explicit hitting set of size $\mathsf{poly}(n, d)^{\mathcal{O}(\log s)}$ for the class of n-variate, degree $\leq d$, depth-3 diagonal circuits of size $\leq s$.*

7 Acknowledgments

The first author would like to thank Peter Bürgisser for explaining the work of Mulmuley, and the Complexity Theory week at Mathematisches Forschungsinstitut Oberwolfach for facilitating that conversation. He would also like to thank Sergey Yekhanin for the conversation that led to Theorem A.1, and Scott Aaronson for some helpful comments.

The authors are grateful to Josh Grochow [Gro13] for bringing the works of Chistov-Ivanyos-Karpinski [CIK97], Sergeichuk [Ser00] and Belitskiĭ [Bel83] to their attention, and for explaining these works to them.

References

[AJS09] V. Arvind, P. S. Joglekar, and S. Srinivasan. Arithmetic Circuits and the Hadamard Product of Polynomials. In FSTTCS, pages 25–36, 2009. 9, 17, 18

[ASS12] M. Agrawal, C. Saha, and N. Saxena. Quasi-polynomial hitting-set for set-depth-Δ formulas. Electronic Colloquium on Computational Complexity (ECCC), 19(113), 2012. 1, 10, 25

[Bel83] G. R. Belitskiĭ. Normal forms in a space of matrices. Analysis in Infinitedimensional Spaces and Operator Theory, pages 3–15, 1983. In Russian. 11, 27

[Ber84] S. J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. Inf. Process. Lett., 18(3):147–150, 1984. 8, 20, 30

[BvzGH82] A. Borodin, J. von zur Gathen, and J. E. Hopcroft. Fast parallel matrix and gcd computations. Information and Control, 52(3):241–256, 1982. 30

[CIK97] A. L. Chistov, G. Ivanyos, and M. Karpinski. Polynomial time algorithms for modules over finite dimensional algebras. In Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC), pages 68–74, 1997. 11, 27, 30

[CLO07] D. Cox, J. Little, and D. O'Shea. Ideals, varieties, and algorithms. Undergraduate Texts in Mathematics. Springer, New York, third edition, 2007. An introduction to computational algebraic geometry and commutative algebra. 2, 26

[DK02] H. Derksen and G. Kemper. Computational invariant theory. Invariant Theory and Algebraic Transformation Groups, I. Springer-Verlag, Berlin, 2002. Encyclopaedia of Mathematical Sciences, 130. 2, 5, 16

[DL78] R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. Inf. Process. Lett., 7(4):193–195, 1978. 3

[For86] E. Formanek. Generating the ring of matrix invariants. In Ring theory (Antwerp, 1985), volume 1197 of Lecture Notes in Math., pages 73–82. Springer, Berlin, 1986. 4

[FS12] M. A. Forbes and A. Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. Electronic Colloquium on Computational Complexity (ECCC), 19:115, 2012. 1, 4, 8, 9, 10, 17, 25

[GKKS13] A. Gupta, P. Kamath, N. Kayal, and R. Saptharishi. Arithmetic circuits: A chasm at depth three. Electronic Colloquium on Computational Complexity (ECCC), 20(26), 2013. 10

[Gro13] J. Grochow, 2013. Private communication. 11, 27

[IKW02] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. J. Comput. Syst. Sci., 65(4):672–694, 2002. 2

[IW97] R. Impagliazzo and A. Wigderson. P=BPP if E requires exponential circuits: Derandomizing the XOR lemma. In STOC, pages 220–229, 1997. 2

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

[KP00] H. Kraft and C. Procesi. Classical invariant theory, a primer. Lecture Notes, 2000. 4

[LBP90] L. Le Bruyn and C. Procesi. Semisimple representations of quivers. Trans. Amer. Math. Soc., 317(2):585–598, 1990. 10

[MFK94] D. Mumford, J. Fogarty, and F. Kirwan. Geometric invariant theory, volume 34 of Ergebnisse der Mathematik und ihrer Grenzgebiete (2) [Results in Mathematics and Related Areas (2)]. Springer-Verlag, Berlin, third edition, 1994. 5, 6, 16

[MM82] E. W. Mayr and A. R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. Adv. in Math., 46(3):305–329, 1982. 2

[MP08] G. Malod and N. Portier. Characterizing valiant's algebraic complexity classes. J. Complexity, 24(1):16–38, 2008. 20

[Mul87] K. Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. Combinatorica, 7(1):101–104, 1987. 30

[Mul12a] K. Mulmuley. Geometric complexity theory V: Equivalence between blackbox derandomization of polynomial identity testing and derandomization of Noether's normalization lemma. In FOCS, pages 629–638, 2012. 1

[Mul12b] K. Mulmuley. Geometric complexity theory V: Equivalence between blackbox derandomization of polynomial identity testing and derandomization of Noether's normalization lemma. arXiv, abs/1209.5993, 2012. Full version of the FOCS 2012 paper. 1

[Nis91] N. Nisan. Lower bounds for non-commutative computation. In STOC, pages 410–418, 1991. 8

[Nis92] N. Nisan. Pseudorandom generators for space-bounded computation. Combinatorica, 12(4):449–461, 1992. 17

[Nis94] N. Nisan. RL$\subseteq$SC. Computational Complexity, 4:1–11, 1994. 17, 18

[Pro76] C. Procesi. The invariant theory of $n \times n$ matrices. Advances in Math., 19(3):306–381, 1976. 4

[Raz74] Ju. P. Razmyslov. Identities with trace in full matrix algebras over a field of characteristic zero. Izv. Akad. Nauk SSSR Ser. Mat., 38:723–756, 1974. 4

[RS05] R. Raz and A. Shpilka. Deterministic polynomial identity testing in non-commutative models. Computational Complexity, 14(1):1–19, 2005. 9, 10, 17, 18

[Sax08] N. Saxena. Diagonal circuit identity testing and lower bounds. In ICALP, pages 60–71, 2008. 1, 10, 24

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. J. Assoc. Comput. Mach., 27(4):701–717, 1980. 3, 11, 30

[Ser00] V. V. Sergeichuk. Canonical matrices for linear matrix problems. Linear Algebra Appl., 317(1-3):53–102, 2000. 11, 27

[SSS11] C. Saha, R. Saptharishi, and N. Saxena. A case of depth-3 identity testing, sparse factorization and duality. Electronic Colloquium on Computational Complexity (ECCC), 18:21, 2011. 10

[SV09] A. Shpilka and I. Volkovich. Improved polynomial identity testing for read-once formulas. In APPROX-RANDOM, pages 700–713, 2009. 1, 10

[SY10] A. Shpilka and A. Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. Foundations and Trends in Theoretical Computer Science, 5(3-4):207–388, 2010. 3, 8

[Val79] L. G. Valiant. Completeness classes in algebra. In STOC, pages 249–261, 1979. 8, 20

[Yek12] S. Yekhanin. Personal Communication, 2012. 11, 30

[Zip79] R. Zippel. Probabilistic algorithms for sparse polynomials. In EUROSAM, pages 216–226, 1979. 3, 11, 30

A Orbit Intersection Reduces to PIT

In this section, we study the (non-closed) orbit intersection problem, as compared with the orbit closure intersection problem studied in Section 3. Unlike with orbit closures, the orbits with non-empty intersections must be equal, because the group action is invertible. Thus, the orbit intersection problem is equivalent to the orbit membership problem. As mentioned before, Chistov, Ivanyos, and Karpinski [CIK97] observed a randomized algorithm for the orbit membership problem, based on the Schwartz-Zippel lemma [Sch80, Zip79]. In conversations with Yekhanin [Yek12], we also discovered this result, which we include for completeness, given its closeness to the other questions studied in this work.

Theorem A.1. *Let $\mathbb{F}$ be a field of size $> n$. There is a reduction, running in deterministic $\mathsf{polylog}(n,r)$-time using $\mathsf{poly}(n,r)$-processors in the unit cost arithmetic model, from the orbit membership problem of $(\mathbb{F}^{[n]\times[n]})^{[r]}$ under simultaneous conjugation, to polynomial identity testing of ABPs. In particular, the orbit membership problem can be solved in randomized $\mathsf{polylog}(n,r)$-time with $\mathsf{poly}(n,r)$-processors (RNC), in the unit cost arithmetic model.*

Proof. Let $\vec{A},\vec{B} \in (\mathbb{F}^{[n]\times[n]})^{[r]}$. There exists an invertible P such that $\vec{B} = P\vec{A}P^{-1}$ iff there is an invertible P such that $\vec{B}P = P\vec{A}$. This second equation is a homogeneous linear equation in P, and thus the set of solutions is a vector space. Gaussian elimination can efficiently (in parallel, see Borodin, von zur Gathen and Hopcroft [BvzGH82], and the derandomization by Mulmuley [Mul87]) find a basis $\{P_i\}_{i=1}^{\ell}$ for this vector space, with $\ell \le n^2$. It follows then that such an invertible P exists iff $\{P_i\}_i$ contain an invertible matrix in their $\mathbb{F}$-span. As the non-vanishing of the determinant characterizes invertible matrices, it follows that such a P exists iff $f(x_1,\ldots,x_\ell) := \det(\sum_{i=1}^{\ell} P_i x_i)$ has a non-zero point in $\mathbb{F}^\ell$.

Clearly $f(\vec{x})$ having a non-zero point in $\mathbb{F}^\ell$ implies that $f(\vec{x})$ is non-zero as a polynomial. Conversely, as the total degree of $f(\vec{x})$ is $\le n$, and the field $\mathbb{F}$ has size $> n$, polynomial interpolation implies that if $f(\vec{x})$ is a non-zero polynomial then it has a non-zero points in $\mathbb{F}^\ell$. Thus, we see that there is a P such that $\vec{B} = P\vec{A}P^{-1}$ iff $f(\vec{x})$ is a non-zero polynomial, where by the results of Berkowitz [Ber84], $f(\vec{x})$ is computable by a $\mathsf{poly}(n,r)$-size ABP. Thus, we have reduced orbit-membership to PIT of ABPs, and done so in parallel. Using that ABPs can be evaluated efficiently in parallel, and that we can solve PIT with random evaluations by the Schwartz-Zippel lemma, we get the corresponding randomized parallel algorithm for orbit membership. □

ECCC ISSN 1433-8092
http://eccc.hpi-web.de