Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 118 (2010)

Extracting Roots of Arithmetic Circuits by Adapting Numerical Methods

Maurice Jansen [*]

January 23, 2011

Abstract

Let $R = \mathbb{F}[x_1, x_2, \ldots, x_n]$. For a polynomial $f \in R[y]$, we say that a polynomial $p \in R$ is a *root* of f, if $f(p) = 0$. We study the relation between the arithmetic circuit sizes of f and p for general circuits and algebraic branching programs. An algebraic branching program (ABP) is given by a layered directed acyclic graph with source σ and sink τ, whose edges are labeled by variables or field constants. It computes the sum of weights of all paths from σ to τ, where the weight of a path is defined as the product of edge-labels on the path. For the size of an ABP we count the number of nodes in the underlying graph.

We address the following fundamental question: suppose the polynomial f can be computed by an ABP of size s. Is the ABP size of every root p of f guaranteed to be bounded by a polynomial in s ? For general circuits it is known that the circuit size of any root p of a polynomial f with circuit size s is at most $poly(s, deg(p), m)$, where m is the *multiplicity* of p in f, i.e. m is the largest number such that $(p - y)^m$ divides f. This bound follows from a result about factors of arithmetic circuits independently obtained by Kaltofen [1] and Bürgisser [2].

In this paper, we study the above question for ABPs for the case where f is assumed to factor as $f = p_0 \cdot (p_1 - y)(p_2 - y) \ldots (p_r - y)$, for $p_0, p_1, \ldots, p_r \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ with $p_0 \neq 0$ and $|\{p_1, p_2, \ldots, p_r\}| = r$, and where p_1 is *degree-dominant* in the sense that $mindeg(p_1) > max_{2 \leq i \leq r} deg(p_i)$. For this situation, provided $\mathbb{F}$ has characteristic zero, we show that p_1 can be computed by an ABP of size polynomial in s.

To prove the above result, we view the question as a problem of computing eigenvalues. Roughly, the $p_i s$ are made to appear as the eigenvalues of some matrix over the field $\mathbb{F}(x_1, x_2, \ldots, x_n)$ of rational functions. This problem is then solved by adapting the numerical method of power iteration to our situation. Using power iteration makes the computation amenable to be coded out as an ABP, since ABPs can efficiently compute iterated matrix multiplication.

In this work we adapt techniques which are well-known from numerical analysis, for use in the area of arithmetic circuit complexity. Staying with this theme, we also improve the above mentioned $poly(s, deg(p), m)$ bound for the circuit size of a root p of a polynomial f computed by an (unrestricted) arithmetic circuit of size s. Rather than applying Ref. [1, 2], we develop a discrete analogue of Newton's Method.

[*]Laboratory for Foundations of Computer Science, School of Informatics, The University of Edinburgh. Email: `maurice.julien.jansen@gmail.com`. This work was conducted while the author was a Postdoctoral Fellow at the Institute for Theoretical Computer Science of Tsinghua University, and it was supported in part by the National Natural Science Foundation of China Grant 60553001, 61073174, 61033001 and the National Basic Research Program of China Grant 2007CB807900, 2007CB807901.

ISSN 1433-8092

1 Introduction

For informal use, let us say an arithmetic circuit class $\mathcal{C}$ is *closed under taking roots*, if roots of (families of) polynomials in $\mathcal{C}$ also belong to the class $\mathcal{C}$. Important consequences follow for classes that enjoy this property either completely, or for which a 'fairly decent' root extraction lemma can be proved. Most notably, such a lemma is a crucial tool for the conditional derandomization of *polynomial identity testing* (PIT) for the class $\mathcal{C}$. For the latter well-known problem one is given an arithmetic circuit Φ, and the problem is to decide whether the polynomial computed by Φ is identical to the zero polynomial or not. Due to a result independently obtained by Kaltofen [1] and Bürgisser [2], we know that the class VP of *poly* degree polynomial families computable by *poly* size arithmetic circuits is even closed under taking *factors*, which implies the closure under taking roots. In their seminal paper on PIT, Kabanets and Impagliazzo [3] use this to give a deterministic subexponential time algorithm for identity testing 'VP-circuits', under the assumption that there exist some explicit polynomial f_n that requires super-polynomial arithmetic circuit size.

For more restricted classes $\mathcal{C}$, it is interesting to consider the question whether PIT for $\mathcal{C}$ can be achieved deterministically under any weaker assumptions. When using the framework of Ref. [3], the situation where $\mathcal{C}$ is closed under taking roots is ideal, since any loss incurred at the root extraction stage is directly reflected in the quality of the resulting hardness to randomness conversion. Examples of research efforts that follow this approach are the works by Dvir, Shpilka and Yehudayoff [4] and Jansen [5].

In Ref. [4] a root extraction lemma is proved for constant depth arithmetic circuits with $O(1)$ loss in the depth, that works well under the promise that the computed polynomials are of low degree. Consequently, a corresponding hardness to randomness conversion is obtained that applies to a low degree promise version of PIT for depth $d - O(1)$ circuits, assuming the existence of an explicit polynomial that is hard for arithmetic circuits of constant depth d. For ABPs a root extraction lemma is proved in Ref. [5], again with parameters working well only for low degree polynomials. Using this, it is proved that a certain low degree promise version of PIT for ABPs can be solved deterministically in subexponential time, assuming some explicit polynomial is hard for ABPs. In this paper we make progress towards showing that the arithmetic circuit class VDET of polynomial families computable by *poly* size ABPs is closed under taking roots. The latter statement, *if true*, would yield a deterministic subexponential time PIT algorithm for VDET, under the assumption that there exists some explicit family of polynomials that requires ABPs of super-polynomial size.

Already implicit in Ref. [4, 5] was the use of a discrete analogue of Newton's Method. We will revisit this, to give a self-contained proof of the fact that VP is closed under taking roots. The resulting argument may serve as a conceptual simplification in Ref. [3], in the sense that calling upon the more involved works Ref. [1, 2] is avoided. For ABPs however, it is hard to imagine that this technique will ultimately lead to an optimal root extraction lemma. In this paper we investigate a new approach. We cast the problem as a task of computing eigenvalues, and adapt the method of power iteration to our domain. This way, since ABPs can efficiently compute matrix multiplication, we avoid the explosion in ABP size seemingly inherent to adaptations of Newton's Method.

In the continuous domain, given a real $s \times s$ matrix M, say with real eigenvalues $\lambda_1 > \lambda_2 > \ldots > \lambda_s > 0$ and a corresponding independent set of unit eigenvectors $v_1, v_2, \ldots, v_s$, a well-known heuristic for finding an approximation to the largest eigenvalue λ_1 is to apply power iteration. Here, starting with some vector u that is typically selected at random, writing u in the eigenbasis as $u = a_1v_1 + a_2v_2 + \ldots + a_sv_s$, for certain scalars a_i, one applies a large power of M to u to obtain

$M^e u = a_1\lambda_1^e v_1 + a_2\lambda^e v_2 + \ldots + a_s\lambda^e v_s$. After normalization, the term $a_1\lambda_1^e v_1$ will be the dominant one, and thus the normalized sum will converge to v_1 as $e \to \infty$. Once an approximation $\tilde{v_1}$ to v_1 is obtained, one may approximate λ_1 by computing, for some nonzero component $(\tilde{v_1})_\ell$, the ratio $(M\tilde{v_1})_\ell/(\tilde{v_1})_\ell$.

We will adapt the method of power iteration to construct small ABPs for roots of ABPs. Similar to the above example for the continuous domain, the method enables one to obtain an ABP for the root which is *degree-dominant*, in the sense that its minimum degree term has degree larger than the degree of any other root. This work provides a case study of how standard tools from numerical analysis can be made available in the area of arithmetic circuit complexity, and hopefully stimulates further research into this direction.

1.1 Results

Our first result is the following theorem[1]:

Theorem 1. *Let $\mathbb{F}$ be a field of characteristic zero. Let $f \in \mathbb{F}[x_1, x_2, \ldots, x_n, y]$ be a nonzero polynomial that can be computed by an ABP of size s. Suppose f factors as*

$$f = p_0(p_1 - y)(p_2 - y)\ldots(p_r - y),$$

where $\{p_0, p_1, p_2, \ldots, p_r\} \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$ with $|\{p_1, p_2, \ldots, p_r\}| = r$ and $mindeg(p_1) > max_{2\leq i\leq r} deg(p_i)$. Then p_1 has an ABP of size at most polynomial in d, r and s, where $d = \max_{i\in[r], p_i\neq 0} deg(p_i)$.

An ABP of size s computes a polynomial for which both its total degree and the individual degree of any variable is bounded by s. This implies that in the above theorem both r and d are at most s. For comparison, Lemma 2.10 of Ref. [5] yields[2] an upper bound of $s \cdot 2^{O(\log^2 deg(p_i))} r^{4+\log deg(p_i)}$ for the size of an ABP for p_i.

For our second result, define the function $\mathcal{M}(d)$ to be an upper bound on the size of an arithmetic circuit for computing the multiplication of two univariate polynomial g and h in $\mathbb{F}[z]$ of degree at most d, given the coefficients of g and h as input variables. By a result of Cantor and Kaltofen [6], one can take $\mathcal{M}(d) = O(d \log d \log\log d)$, over *any* field $\mathbb{F}$. For (unrestricted) arithmetic circuits we have the following theorem:

Theorem 2. *Let $\mathbb{F}$ be a field of characteristic zero. Let $R = \mathbb{F}[x_1, x_2, \ldots, x_n]$. Let $f \in R[y]$ be a polynomial of degree $r > 0$ that is computable by an arithmetic circuit of size s and let $p \in R$ be a nonconstant root of f for y, i.e. $f(p) \equiv 0$ and $p \notin \mathbb{F}$. Then p can be computed by an arithmetic circuit of size $O(\mathcal{M}(m)\mathcal{M}(deg(p)) \cdot deg(p) \cdot s)$, where m is the multiplicity of the root p in f.*

Due to a Lemma by Gauss (Lemma 1), in the above situation p is a root of f if and only $p - y$ is an irreducible factor of f in $\mathbb{F}[x_1, x_2, \ldots, x_n, y]$. Using Ref. [1, 2] to obtain arithmetic circuits for the factor $p - y$, as done in Ref. [3], yields a circuit for the root p of size $O(\mathcal{M}(deg(p)^3 m)(s + deg(p)\log m))$. It can be verified that our result is an improvement over the bound obtained this way.

[1]In a preliminary version of this paper, which appeared at *The 2nd Symposium on Innovations in Computer Science (ICS 2011)*, it was claimed erroneously that this could be done soly under the assumption that $p_1, p_2, \ldots, p_r$ are distinct. It is an interesting open problem whether the techniques demonstrated in the current paper can be generalized to this restricted multiplicity case.

[2]Note that this lemma is stated for *skew circuits*, but inspection of the proof yields the given bound.

2 Preliminaries

Let $\mathbb{F}$ be a field of characteristic zero. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of indeterminates. Let $R = \mathbb{F}[X]$. Let $\mathbb{G}$ denote the field of rational functions $\mathbb{F}(X)$. For a polynomial $f \in R[y]$ and $p \in R$, $f_{|y=p}$ denotes the polynomial obtained by substituting p for y in f. In case $f_{|y=p} = 0$, we say that p is a *root* of f for y. Recall the following lemma by Gauss:

Lemma 1 (Gauss)**.** *Let $f \in \mathbb{F}[X, y]$ be a nonzero polynomial, and let $p \in \mathbb{F}[X]$ be a root of f for y. Then $p - y$ is an irreducible factor of f in the ring $\mathbb{F}[X, y]$.*

In the above situation, the *multiplicity* of the root p is defined to be the largest number m such that $(p - y)^m$ divides f. The Vandermonde determinant is defined to be the polynomial $\text{Vandet}(x_1, x_2, \ldots, x_n) = \prod_{1 \le i < j \le n}(x_i - x_j)$. For two polynomials f and g, if there exists a polynomial h such that $f = gh$, we say g *divides* f, and that the division f/g is *exact*. Total degree of a polynomial f is denoted by $deg(f)$. For a vector v of polynomials, we define $deg(v) = \max_j deg(v_j)$. For a polynomial $f \in \mathbb{F}[X]$, we denote by $[f]_{=i}$, or simply $f_{=i}$, the homogeneous component of degree i. Similarly, we use the notations $f_{\le i}$, $f_{\ge i}$, $[f]_{\le i}$ and $[f]_{\ge i}$. We will also use this notation for vectors of polynomials. For example, $(f, g, h)_{\le i} = (f_{\le i}, g_{\le i}, h_{\le i})$. For a nonzero polynomial f, $mindeg(f)$ is the minimum i such that $f_{=i}$ is nonzero, We extend this to any nonzero vector v of polynomials, by letting $mindeg(v) = \min_{j, v_j \ne 0} mindeg(v_j)$.

For a matrix M, we denote by $M[[i, j]]$ the matrix obtained by removing row i and column j. Let us denote by $M[i, j]$ the matrix obtained from M by setting all entries in row i and column j to 0, except for entry $M_{i,j}$, which is set to 1. Using Laplace expansion along row i of $M[i, j]$ one immediately concludes that the following holds: for any matrix M, $\det(M[i, j]) = (-1)^{i+j} \det(M[[i, j]])$. For a $s \times s$ matrix M, $Adj(M)$ denotes the $s \times s$ *adjugate* matrix of M, defined by $Adj(M)_{ij} = \det(M[j, i])$. For any matrix M, $MAdj(M) = Adj(M)M = \det(M)I$, where I denotes the identity matrix.

An *arithmetic circuit* Ψ over variables X and field $\mathbb{F}$ is given by a directed acyclic graph whose nodes of in-degree larger than zero are labeled by $\{+, \times\}$, and with other nodes labeled by elements of $X \cup \mathbb{F}$. At each node g of Ψ we have associated a polynomial in $\mathbb{F}[X]$ computed by g, which is defined in the standard manner. The output of Ψ is the polynomial computed by some designed output gate. For the size of Ψ we count the number of edges. The size of a polynomial f, denoted by $s(f)$, is the size of the smallest arithmetic circuit computing f. We let VP stand for the class of polynomial families (f_n) for which there exists a polynomial $p(n)$ such that $deg(f_n) \le p(n)$ and $s(f_n) \le p(n)$.

An algebraic branching program (ABP) over X and $\mathbb{F}$ is a 4-tuple $\Phi = (G, w, \sigma, \tau)$, where $G = (V, E)$ is a weighted directed acyclic graph for which the vertex set V can be partitioned into *levels* $L_0, L_1, \ldots, L_\ell$, where $L_0 = \sigma$ and $L_\ell = \tau$. Vertices σ and τ are called the source and sink of Φ, respectively. Edges may only go between consecutive levels L_i and L_{i+1}. The subgraph induced by L_i and L_{i+1} is called a *layer* of Φ. The weight function $w : E \to X \cup \mathbb{F}$ assigns variables or field constants to the edges of G. For a path p in G, we extend the weight function by $w(p) = \prod_{e \in p} w(e)$. Let $P_{i,j}$ denote the collection of all directed paths p from i to j in G. The program Φ computes the polynomial $\hat{\Phi} := \sum_{p \in P_{\sigma,\tau}} w(p)$. The size of Φ is defined to be $|V|$. For nodes v and w in Φ, $\Phi_{v,w}$ denotes the subprogram of Φ with source v and sink w.

We will also consider "multi-output" ABPs. In this case the last layer of the A consists of several sink nodes $\tau_1, \tau_2, \ldots, \tau_m$. The output of the ABP is given by the tuple of polynomials $(f_1, f_2, \ldots, f_m)$ computed by the subprograms $A_{\sigma,\tau_1}, A_{\sigma,\tau_2}, \ldots, A_{\sigma,\tau_m}$.

ABPs are convenient when dealing with substitution. It is easily seen that if g can be computed by an ABP A_g of size s_g and f is computed by an ABP A_f of size s_f, then $f_{|x_i=g}$ can be computed by an ABP of size $O(e_f s_g)$, where $e_f = O(s_f^2)$ is the number of edges in A_f. For the analysis, we define absolute constants $\gamma_1 = 3, \gamma_2 = 5, \gamma_3 = 12$. We use a result by Mahajan and Vinay and a result by Kaltofen and Koiran.

Theorem 3 (See Theorem 2 in [8]). *The determinant of an $n \times n$ matrix can be computed by an ABP of size $O(n^{\gamma_1})$ with $O(n^{\gamma_2})$ many edges.*

Lemma 2 (See[3] [7]). *Suppose $|\mathbb{F}|$ is infinite. Let $f, g \in \mathbb{F}[X]$ be given that both are computable by ABPs of size at most s. Assuming the division f/g is exact, then f/g can be computed by an ABP of size $O(s^{\gamma_3})$.*

The following lemma is proved by the well-known trick of 'splitting' nodes in order to keep track of degree components.

Lemma 3. *Let $d \geq 0$ be an integer. Let Φ be an ABP of size s with e many edges computing the polynomial $f \in \mathbb{F}[X]$. Then there exist an ABP Ψ of size $O(ds)$ and $O(de)$ many edges computing $[f]_{\leq d}$. Similarly for $f_{\geq d}$ and $f_{=d}$. Also, a similar statement holds for multi-output ABPs.*

Finally, we use a result by Kaltofen and Singer for computing formal partial derivatives. We use the notation $\frac{\partial^k f}{\partial^k y}$ to denote the formal partial derivative of f of order k w.r.t. the variable y.

Theorem 4 (Theorem 3.1 in [10]). *For any integer $k \geq 0$, if $f \in \mathbb{F}[X, y]$ can be computed by an arithmetic circuit of size s, then $\frac{\partial^k f}{\partial^k y}$ can be computed by an arithmetic circuit of size $O(\mathcal{M}(k) \cdot s)$.*

3 Standard Form ABPs, Valiant Matrices and Homogenizations

Definition 1. *Let $f \in \mathbb{F}[X, y]$ be a polynomial whose degree in y equals r, and write $f = \sum_{i=0}^{r} C_r(x) y^r$. We say an ABP Φ with source σ and sink τ computing f is in standard form, if it has the following structure:*

- *There is a set of distinct nodes $\{b_0, b_1, \ldots, b_r\}$, such that for each $i \in \{0, 1, \ldots, r\}$, there is an edge from the source σ to b_i with label 1. These are the only edges adjacent to the source.*
- *There are distinct nodes $c_0, c_1, \ldots, c_r$. The subprograms in the set $\{\Phi_{b_i, c_i} : i \in [r]\}$ are disjoint as graphs. For every $i \in \{0, 1, \ldots, r\}$, the subprogram Φ_{σ, c_i} computes $C_i(x)$.*
- *There is a path $c_r = a_0, a_1, \ldots, a_{r-1}, a_r = \tau$, where each edge (a_i, a_{i+1}) is labeled with the variable y. These are the only occurrences of y variables in Φ.*
- *All remaining edges are labeled with the constant one. These simply realize that for every $0 \leq i < r$, there is one single path of weight 1 from c_i to a_{r-i}.*
- *the length of every path from σ to τ is even.*

The following lemma follows along similar lines as Lemma 3:

[3]This follows from From Lemma 1 and Lemma 2 in Ref. [7]. We get an extra quadratic blow-up of s, since the DAG we use for ABPs must be leveled.

Lemma 4. *Let $f \in \mathbb{F}[X, y]$ be computed by an ABP Φ of size s, and let $r = deg_y(f)$. Then f can be computed by an ABP Ψ in standard from of size $O(sr^2)$. This means in particular that the variable y appears exactly r times on an edge in Ψ.*

Given an ABP Φ of size s computing f, we can construct a matrix $M(\Phi)$ of order s, whose entries are variables and field elements, such that $\det(M) = f$, as done in [11]. Namely, thinking of Φ as a graph, one adds a loop back from τ to σ with label 1, and one puts a self loop on all nodes other than σ and τ with label 1. Wlog. we make the following convention:

Convention 1. *We assume nodes in Φ always carry a unique number $\in [s]$, which we then use to index columns/rows, and we assume that nodes $a_0, a_1, \ldots, a_r$ are numbered $1, 2, \ldots, r+1$, respectively.*

Let $M(\Phi)$ be the adjacency matrix of the weighted graph obtained this way, which we call the *Valiant matrix associated to* Φ. Assuming wlog. that the length of every path from σ to τ in Φ is even, then $\det(M(\phi)) = f$. In our notation, we will use variable names of nodes of Φ to index the matrix $M(\Phi)$, and also do this for s-vectors operated on. For example, for the standard form ABP Φ for Definition 1, the entry $M(\Phi)_{a_0 a_1}$ equals y.

We let $\Pi : \mathbb{F}^s \to \mathbb{F}^s$ denote the projection mapping $\Pi(a_1, a_2, \ldots, a_s)^T = (0, a_2, \ldots, a_{r+1}, 0 \ldots, 0)^T$, and we let $\Sigma : \mathbb{F}^s \to \mathbb{F}^s$ denote the mapping that performs a cyclic shift by one as follows: $\Sigma(a_1, a_2, \ldots, a_s)^T = (a_2, a_3, \ldots, a_s, a_1)^T$. As is standard practice, we will also use Π and Σ to denote matrices that realize these mappings relative to the standard basis.

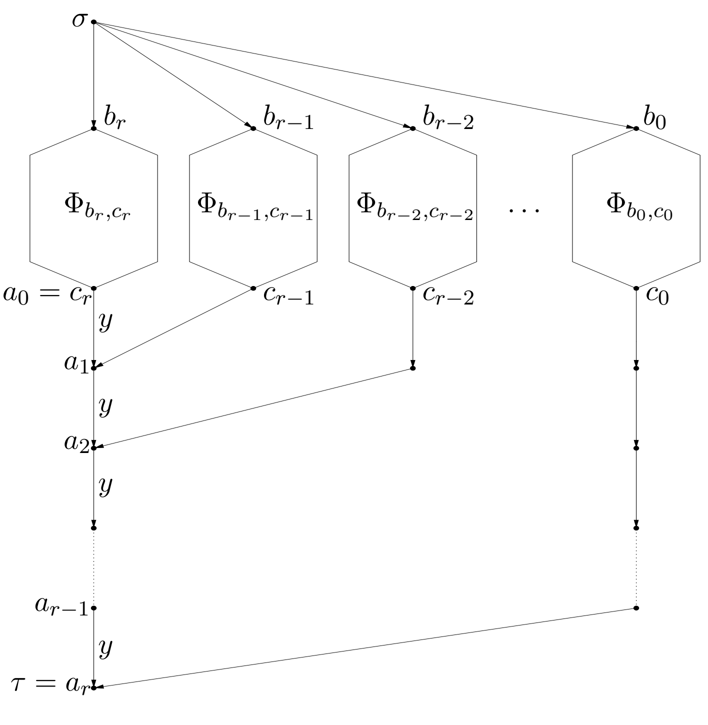

Figure 1: Schematic depiction of an ABP Φ in standard form computing $f = \sum_{i=0}^{r} C_r(x) y^r$. For each i, the subprogram Φ_{b_i,c_i} computes $C_i(x)$.

Proposition 1. *In the above situation, we can write for some matrix A with entries in $\mathbb{F} \cup X$, $M(\Phi) = A - yB$, where $-B = \Sigma\Pi$.*

Proof. $-B$ is a $0,1$-matrix that has a 1 on entry $-B_{ij}$ iff edge (i,j) is labeled with y. With out numbering convention, $-B$ contain precisely r ones, and these are on entries $-B_{12}, -B_{23}, \ldots, -B_{r,r+1}$. This implies $-B$ can be written as a projection on to coordinates $2, 3, \ldots, r+1$, followed by a shift, i.e. $-B = \Sigma\Pi$. □

3.1 A Closed Form for Eigenvectors Related to the Valiant Matrix

The following proposition is easily proved using Laplace expansion:

Proposition 2. *Let M be a singular matrix of order m. For any fixed i, if we define the m-vector v by taking $v_j = \det(M[i,j])$, then $Mv = 0$.*

Next we derive the main lemma of this subsection.

Lemma 5. *Let Φ be an ABP of size s in standard form computing the polynomial $f = \sum_{i=0}^{r} C_i(X)y^r \in \mathbb{F}[X,y]$ of degree r in y. Let nodes $a_0, b_0, c_0, a_1, b_1, c_1, \ldots, a_r, b_r, c_r$ be given as in Definition 1, which implies the subprogram Φ_{σ,c_i} computes $C_i(X)$. Let $M(\Phi)$ be the associated standard form Valiant matrix. Suppose that $q \in \mathbb{F}[X]$ is such that $f_{|y=q} \equiv 0$. Let N be the matrix obtained by setting $y = q$ in $M(\Phi)$, i.e. $N = M(\Phi)_{|y=q}$. Suppose we define the s-vector v by $(v)_j = \det(N[c_r, j])$, for all $j \in [s]$. Then the following hold:*

1. *$Nv = 0$.*
2. *$deg(\det(N[c_i, j])) \leq sdeg(q)$.*
3. *$\forall j \in [r]$, $(v)_{a_j} = q^{r-j} \cdot (-1)^{j+1} C_r(X)$.*

By our labeling convention, the last item mean $\forall j \in [r]$, $(v)_{j+1} = q^{r-j} \cdot (-1)^{j+1} C_r(X)$, and hence also $\forall j \in [r]$, $(\Pi v)_{j+1} = q^{r-j} \cdot (-1)^{j+1} C_r(X)$.

Proof. We recall the notion of a cycle cover for later use. A cycle cover C in a directed graph $G = (V,E)$ with n vertices is a set of disjoint simple cycles $C_1, C_2, \ldots, C_i$ such that every vertex in G is contained in some cycle C_i. For weighted G, the weight of a cycle C is taken to be the product of weights of edges in C. For a simple cycle C we define its sign $sgn(C)$ to be -1 if C is of even length, and 1 otherwise. For the cycle cover C, define $sgn(C) = \prod_i sgn(C_i)$.

Observe that $\det(N) = \det(M(\Phi)_{|y=q}) = f_{|y=q} \equiv 0$. Hence the first property follows from Proposition 2. The second property is clear. To verify the last property, let $j \in [r]$ be arbitrary. Consider the matrix $N = M(\Phi)_{|y=q}$. Let G be the weighted graph corresponding to $M(\Phi)$. We can think of the matrix $N[c_r, a_j]$ as the adjacency matrix of a graph H formed by doing the following to G:

- replacing all y-labels in G by q.
- Removing all edges out of c_r, including the self loop.
- Removing all edges into a_j, including the self loop.
- Adding the edge From c_r to a_j with label one.

Then $\det(N[c_r, a_j]) = \sum_C sgn(C)w(C)$, where the sum is over all cycle covers in H. Observe that since c_r and a_j do not have self-loops, any cycle cover C in H must include the edge (c_r, a_j). So the cycle covers are of the following structure: 1) From the source σ there is a path to c_r, 2) The edge (c_r, a_j) is taken, 3) $r - j$ edges with label q are taken, 4) Finally, the loop back from $a_r = \tau$ to the source σ is taken, and self-loops with label 1 are taken for all vertices not included in above cycle.

All of the above described cycles starting at the source σ are of the same length. In case $j = 1$ the length equals the same 'big cycle length' as in $M(\Phi)$, which is odd. For general j, by considering how many edges we skip with the edge (c_r, a_j) one can conclude that $sgn(C) = (-1)^{j+1}$. Hence $\det(N[c_r, a_j]) = \sum_C sgn(C)w(C) = (-1)^{j+1} \sum_C w(C)$. The expression $\sum_C w(C)$ equals the sum of weights of all paths from σ to τ that go over (c_r, a_j). Since these paths all go over c_r, this sum factors as $C_r(X, z)$ (weight of all paths from σ to c_r) times q^{r-j} (weights of path "(c_r, a_j), followed by going from a_j to τ"). $\square$

4 Proof of Theorem 1

We first prove the following lemma:

Lemma 6. *Let $f \in \mathbb{F}[X, y]$ be a polynomial of degree $r > 1$ computed by a standard form ABP Φ of size s. Suppose f factors as*

$$f = \prod_{i \in [r]} (p_i - y),$$

where $\{p_1, p_2, \dots, p_r\} \subset \mathbb{F}[X]$ is a set of pairwise distinct nonzero polynomials and such that

$$mindeg(p_1) > max_{2 \leq i \leq r} deg(p_i).$$

Then p_1 can be computed by an ABP of size $O(d^{\gamma_3^2+2\gamma_3} r^{\gamma_3^2} s^{(\gamma_1+\gamma_2+4)\gamma_3^2+4\gamma_3})$, where $d = \max_{i \in [r]} deg(p_i)$ and γ_1, γ_2 and γ_3 are the absolute constants introduced in Section 2.

Proof. We will show the method for obtaining p_1.

4.1 Towards Computing Eigenvectors

Consider the associated Valiant matrix $M(\Phi)$. Note that $f_{|y=0} \neq 0$. By Proposition 1 we can write

$$M(\Phi) = A - yB.$$

Then $f = \det(A - yB)$. We have the following two properties:

1. $-B = \Sigma\Pi$
2. A is invertible over $\mathbb{F}(X)$.

Lemma 7. *Let nodes $\{a_i, b_i, c_i : i \in [r]\}$ in Φ be given as in Definition 1. Let $C_i(X)$ be the polynomial computed by the subprogram Φ_{σ, c_i}. Clearly, we have that $f = \sum_{i=0}^r C_i(X) y^i$. For all $i \in [r]$, we define the column vector v_i by letting for every $j \in [s]$, $(v_i)_j = \det(N_{p_i}[c_r, j])$, where $N_{p_i} = N_{p_i} = A - p_i B$. Then the following hold:*

1. For every $i \in [r]$, $N_{p_i} v_i = 0$.

2. *For every $i \in [r]$, $deg(v_i) \leq sd$.*
3. *Letting e_i be the standard basis vector with 1 in the ith position, then for*

$$V := [v_1, v_2, \ldots, v_r, e_1, e_{r+2}, \ldots, e_s],$$

it holds that

$$\det(V) = (\pm 1) \cdot \mathrm{Vandet}(p_1, p_2, \ldots, p_r).$$

4. $\mathrm{Vandet}(p_1, p_2, \ldots, p_r)^2$ *can be computed by an ABP of size $O(r^{5+\gamma_2} ds)$, where γ_2 is the absolute constant introduced in Section 2.*

Proof. The first and second item immediately follow from Lemma 5, Items 1 and 2. Note that $C_r(X) = coef(y^r, f_0) = (-1)^r$. By Lemma 5, Item 3, the matrix $V' = [v_1, v_2, \ldots, v_r]$ consisting of the r column vectors $v_1, v_2, \ldots, v_r$ contains (modulo multiplying columns by -1) the $r \times r$ Vandermonde matrix $(p_i^j)_{0 \leq j \leq r-1, 1 \leq i \leq r}$. as a submatrix on rows in the set $J = \{a_j : j \in [r]\}$. By convention, $J = \{2, 3, \ldots, r+1\}$. Hence $\det(V) = (\pm 1) \cdot \mathrm{Vandet}(p_1, p_2, \ldots, p_r)$. The proof of the fourth property we treat in the next subsection. □

Remark 1. *Note that for $v_1, v_2 \ldots, v_r$ as in the above, for*

$$V' := [\Pi v_1, \Pi v_2, \ldots, \Pi v_r, e_1, e_{r+2}, \ldots, e_s],$$

it also holds that

$$\det(V') = (\pm 1) \cdot \mathrm{Vandet}(p_1, p_2, \ldots, p_r).$$

as $\det(V') = \det(V)$. *Also note that* $\det(V') \neq 0$, *as $p_1, p_2, \ldots, p_r$ are pairwise distinct.*

4.2 A Small ABP for Computing $\mathrm{Vandet}(p_1, p_2, \ldots, p_r)^2$

This subsection is dedicated to proving Item 4 of Lemma 7. Define the polynomial

$$T_i(x_1, x_2, \ldots, x_r) = x_1^i + x_2^i + \ldots + x_r^i.$$

We use the fact[4] that

$$\det \begin{pmatrix} T_0 & T_1 & \ldots & T_{r-1} \\ T_1 & T_2 & \ldots & T_r \\ \vdots & & & \\ T_{r-1} & T_r & \ldots & T_{2r-2} \end{pmatrix} \quad (1)$$

$$= \mathrm{Vandet}(x_1, x_2, \ldots, x_r)^2. \quad (2)$$

The strategy is to express each T_i as a 'small' formula of $S_r^j(x_1, x_2, \ldots, x_r)$, where S_r^j is the elementary symmetric polynomial in r variables of degree j, i.e.

$$S_r^j(x_1, x_2, \ldots, x_r) = \sum_{I \subset [r], |I| = j} \prod_{i \in I} x_i.$$

[4]This follows by multiplying the Vandermonde matrix with nodes $x_1, x_2, \ldots, x_r$ by it transpose.

It is well-known that the T_is and S_r^js are related through the Newton Identities.

At first sight it may look like we have run into a circular argument. How do we plug in the p_is? This bootstrapping problem is resolved by observing that, if we succeed in the above[5], regardless of not having small ABPs for the p_is, we readily have small ABPs for any $S_r^j(p_1, p_2, \ldots, p_r)$. Namely, consider the following remark and subsequent derivation:

Remark 2. *For every j, $S_r^j(p_1, p_2, \ldots, p_r)$ equals the coefficient of y^{r-j} of f modulo a factor of ± 1. Hence an ABP Φ_j computing $S_r^j(p_1, p_2, \ldots, p_r)$ of size at most s is easily obtained from the standard form ABP Φ.*

4.2.1 A Formal Power Series Identity Related to the Newton Identities

Let w be an new variable. We have the following lemma:

Lemma 8. *Provided the characteristic of $\mathbb{F}$ is zero, we have the following identity in the ring of formal power series $\mathbb{F}[[X]]$: $\sum_{\ell \geq 1} \frac{1}{\ell}(\sum_{j=1}^{r}(-1)^j S_r^j(x_1, x_2, \ldots, x_r)w^j)^\ell = \sum_{n \geq 1} \frac{-w^n}{n} T_n(x_1, x_2, \ldots, x_r)$.*

Proof. We recall the definitions of the formal power series (FPS) *exp*(w) and $\log(1-w)$. These are given by $\exp(w) = \sum_{n \geq 0} \frac{w^n}{n!}$. and $\log(1-w) = -\sum_{n \geq 1} \frac{w^n}{n}$. We will use that $\exp(\log(1 - wx_j)) = 1 - wx_j$. Hence

$$\begin{aligned}
&\prod_{j \in [r]} (1 - wx_j) \\
= \;& \prod_{j \in [r]} \exp(\log(1 - wx_j)) \\
= \;& \exp(\sum_{j \in [r]} \log(1 - wx_j)) \\
= \;& \exp(-\sum_{j \in [r]} \sum_{n \geq 1} (wx_j)^n/n) \\
= \;& \exp(-\sum_{n \geq 1} T_n(x_1, x_2, \ldots, x_r)w^n/n)
\end{aligned}$$

Hence, by multiplying out the l.h.s. we get that

$$\sum_{j=1}^{r}(-1)^j S_n^j(x_1, x_2, \ldots, x_r)w^j \quad = \quad \exp(-\sum_{n \geq 1} T_n(x_1, x_2, \ldots, x_r)w^n/n) - 1.$$

Now we use that for $g(w) := -\sum_{n \geq 1} T_n(x_1, x_2, \ldots, x_r)w^n/n$, it holds that $\log(1+(\exp(g(w))-1)) =$

[5]In [12] the converse is achieved to get small depths formulas for S_r^j.

$g(w)$. Thus applying $\log(1+w)$ to both sides of the above equation yields that

$$\begin{aligned} -\sum_{n\geq 1} T_n(x_1,x_2,\ldots,x_r)w^n/n &= \\ \log(1-\sum_{j=1}^{r}(-1)^j S_n^j(x_1,x_2,\ldots,x_r)w^j) &= \\ \sum_{\ell\geq 1}\frac{1}{\ell}(\sum_{j=1}^{r}(-1)^j S_n^j(x_1,x_2,\ldots,x_r)w^j))^\ell. \end{aligned}$$

□

In the following, we truncate the expression on the l.h.s. in the above lemma, discarding terms that cannot possibly contribute to the coefficient of w^i. Then we do some circuit manipulations to extract the coefficient of w^i, and this way we obtain an ABP computing T_i in terms of the S_r^js.

Proposition 3. *Let $u_1, u_2, \ldots, u_r$ be a set of new variables. For any $i \in [r]$ the following statements are true. Let $E(u_1, u_2, \ldots, u_i, w) = \sum_{1\leq \ell\leq i}\frac{1}{\ell}(\sum_{j=1}^{i}(-1)^j u_j w^j)^\ell$. Then*

1. *There exists an ABP $\Gamma(u_1, u_2, \ldots, u_i, w)$ with $O(i^3)$ many edges computing E.*
2. *There exists an ABP $\Gamma'(u_1, u_2, \ldots, u_i)$ with $O(i^4)$ many edges computing the coefficient of w^i in E.*
3. *Say Γ' computes the polynomial E'. Then*

$$\begin{aligned} &E'(S_r^1(x), S_r^2(x), \ldots, S_r^i(x)) \\ &\quad = -T_i(x_1, x_2, \ldots, x_r)/i. \end{aligned}$$

Proof. The first item is left as an easy exercise. Then second item then follows by applying Lemma 3. The last item follows from Lemma 8. □

4.2.2 Putting It Together

By Proposition 3,

$$\begin{aligned} &E'(S_r^1(p_1, \ldots, p_r), \ldots, S_r^i(p_1, \ldots, p_r)) \\ &\qquad = -T_i(p_1, \ldots, p_r)/i. \end{aligned}$$

By Remark 2 and comments thereafter, we conclude that for any $i \in [r]$, we have an ABP computing $T_i(p_1, \ldots, p_r)$ of size $O(r^5 ds)$. The $r \times r$ determinant can be computed by an ABP with $O(r^{\gamma_2})$ many edges by Theorem 3. Hence using Equation (2) we obtain an ABP for computing $\mathrm{Vandet}(p_1, p_2, \ldots, p_r)^2$ of size $O(r^{5+\gamma_2} ds)$. □

4.3 Selecting a Good Starting Vector u

Let $s \times s$ matrix $V = [v_1, v_2, \ldots, v_r, e_1, e_{r+2}, \ldots, e_s]$ be given by Lemma 7. We have the following straightforward proposition:

Proposition 4. $\forall \alpha \in \mathbb{F}$, *$Adj(A)Bv_i = \frac{\det(A)}{p_i} \cdot v_i$.*

Proof. By Lemma 7, Item 1, it holds that $Av_i = p_i B v_i$, for any $i \in [r]$. Therefore, $Adj(A)Bv_i = \frac{\det(A)}{p_i} \cdot v_i$. $\square$

For each $i \in [r]$, let $w_i = \Pi v_i$. Let $V' = [w_1, w_2, \ldots, w_r, e_1, e_{r+2}, \ldots, e_s]$. As we observed in Remark 1, we have that $\det(V') = \det(V)$.

Lemma 9. *There exists a nonsingular matrix M and nonzero polynomial g such that*

- *for each $i \in [r]$, $Mw_i = gp_i w_i$.*
- *M has entries $\in \mathbb{F}[X]$ that are of degree at most s^2.*
- *Each entry of M can be computed by an ABP of size $O(s^{\gamma_1+\gamma_2})$.*
- *g has degree at most s^2 and can be computed by an ABP of size $O(s^{\gamma_1+\gamma_2})$.*

Proof. Since $-B = \Sigma\Pi$, we get by Proposition 4 that

$$Adj(A)\Sigma\Pi v_i = -\frac{\det(A)}{p_i} v_i$$

Applying the projection Π both on the left and right of the above equation, and using that $\forall v, \Pi\Pi v = \Pi v$, we get that

$$(\Pi Adj(A)\Sigma\Pi)\Pi v_i = -\frac{\det(A)}{p_i}\Pi v_i.$$

We want to invert the action of $Adj(A)\Sigma$ on $Im(\Pi)$ and scale up by $\det(A)$. Note that $\Pi Adj(A)\Sigma\Pi$ is obtained from $Adj(A)\Sigma$ by setting to zero entries outside the minor on rows/columns $2, 3, \ldots, r+1$. Let P be the matrix

$$P = \Pi Adj(A)\Sigma\Pi + diag(c_1, c_2, \ldots, c_s).$$

where $c_1 = 1, c_2 = 0, c_3 = 0, \ldots, c_{r+1} = 0, c_{r+2} = 1, c_{r+3} = 1, \ldots, c_s = 1$. By the above, for each $i \in [r]$,

$$Pw_i = -\frac{\det(A)}{p_i} w_i.$$

We claim that P is nonsingular. Namely, as we observed before, $\det(A) \neq 0$. We know that $w_1, w_2, \ldots, w_s, e_1, e_{r+2}, \ldots, e_s$ is a basis, since $\det(V') \neq 0$. Note that P acts as an identity on $e_1, e_{r+2}, \ldots, e_s$. We conclude that P applied to any vector in this basis results in a nonzero scaling of the vector, and hence P is nonsingular. Let

$$M = -\det(A)Adj(P).$$

and $g = \det(P)$. Then $M = -\det(A)gP^{-1}$. Hence

$$Mw_i = gp_i w_i.$$

Using Theorem 3 it is straightforward to verify the bounds on the degrees of M and g and the sizes of their respective ABPs.

$\square$

The following lemma now follows easily:

Lemma 10. *$\exists i \in \{2, 3, \dots, r+1\}$ such that*

$$\det(V')^2 M e_i = a_1 w_1 + a_2 w_2 + \ldots + a_r w_r,$$

where M is the matrix given by Lemma 9, and

1. *$\forall i, a_i \in \mathbb{F}[X]$,*
2. *$a_1 \neq 0$, and*
3. *$\forall i$, $deg(a_i) \leq (2d+1)s^2$.*

Proof. We have that $w_1, w_2, \dots, w_r$ forms a basis of $Range(\Pi)$, since $rank(\Pi) = r$ and $w_1, w_2, \dots, w_r \in Range(\Pi)$ are independent. By Lemma 9, $w_1, w_2, \dots, w_r$ forms a basis of $MRange(\Pi)$. Hence $dim(MRange(\Pi)) = r$. The set $\{e_2, e_3, \dots, e_{r+1}\}$ is also a basis of $Range(\Pi)$. Hence for every e_i with $i \in \{2, 3, \dots, r+1\}$, we can write $\det(V')^2 M e_i = a_{1,i} w_1 + a_{2,i} w_2 + \ldots + a_{r,i} w_r$, for certain $a_{1,i}, a_{2,i}, \dots, a_{r,i} \in \mathbb{G}$. Suppose that for every $i \in [s]$, $a_{1,i} = 0$. This means that $MRange(\Pi) \subseteq span(w_2, \dots, w_r)$. Hence $dim(MRange(\Pi)) < r$, which is a contradiction.

Now let i be such that $a_{1,i} \neq 0$. The coefficients $a_{1,i}, a_{2,i}, \dots, a_{r,i}$ can be obtained as the first r components of the vector $(V')^{-1} \det(V')^2 M e_i = \det(V') \cdot Adj(V') M e_i$. Note that this implies all a_i are in $\mathbb{F}[X]$, as all of $V', Adj(V')$ and M only have polynomial entries. The degrees of entries in $Adj(V')$, and also $deg(\det(V'))$, can be bounded by ds^2, since entries of V' have degree at most sd due to Item 2, Lemma 7. By Lemma 9 entries of M have degrees bounded by s^2. Hence any a_i has degree at most $(2d+1)s^2$. □

Let i be given by the above lemma, and fix the vector $u = \det(V')^2 M e_i$. This vector will be the starting point for applying power iteration. To stress, this is an element of $\mathbb{F}[X]^s$, since V', M and e_i only contain polynomial entries.

Lemma 11. *u can be computed by a multi-output generalized ABP of size $O(r^{5+\gamma_2} ds + s^{\gamma_1+\gamma_2+2})$.*

Proof. By Lemma 7, we have an ABP B_1 of size $O(r^{5+\gamma_2} ds)$ computing the polynomial $\det(V')^2$. By Lemma 9 each entry of M can be computed by an ABP of size $O(s^{\gamma_1+\gamma_2})$. By putting these in series we obtain a multi-output ABP for u of size $O(r^{5+\gamma_2} ds + s^{\gamma_1+\gamma_2+2})$. □

4.4 Applying Power Iteration

Now we are ready to start applying power iteration in order to isolate the single eigenvector w_1 and consequently find the corresponding eigenvalue. We have that $u = a_1 w_1 + a_2 w_2 + \ldots + a_r w_r$, for certain $a_i \in \mathbb{F}[X]$, as given by Lemma 10.

Let $\kappa = mindeg(p_1)$. Let $B = \max_{2 \leq i \leq r} deg(a_i w_i)$. By Lemma 7 and Lemma 10, $B \leq (2d+1)s^2 + sd$. Set $e = B + 1$. When we apply M^e to our chosen starting point u, by linearity over $\mathbb{G}$ of M, we have that

$$u' := M^e u \quad = \quad \sum_{i \in [r]} a_i \, (g p_i)^e \, w_i.$$

We let

$$u'' = \frac{u'}{g^e}.$$

Then

$$u'' = \sum_{i\in[r]} a_i \, (p_i)^e \, w_i$$

Let $u''' = [u'']_{\geq \kappa e}$. Note that for any $2 \leq i \leq r$, $deg(a_i p_i^e w_i) \leq B + (\kappa - 1)e$, whereas $mindeg(a_1 p_1^e w_1) \geq \kappa e$. Since we set $e = B + 1$, we have that for any, $2 \leq i \leq r$, $deg(a_i p_i^e w_i) < \kappa e$. Hence

$$u''' = a_1 p_1^e w_1.$$

4.5 Constructing the ABP for the Eigenvalue p_1

Iterated matrix multiplication is coded easily with ABPs, which yields the following lemma:

Lemma 12. *The vector u' can be computed by a multi-output ABP of size $O(drs^{\gamma_1+\gamma_2+4})$.*

Proof. Starting with the generalized multi-output ABP computing u given by Lemma 11 of size $O(r^{5+\gamma_2}ds + s^{\gamma_1+\gamma_2+2})$, we add stages to compute the required consecutive multiplication by M. By Lemma 9, each such matrix multiplication can be achieved by adding $O(s^{\gamma_1+\gamma_2+2})$ nodes to the ABP. We therefore get that the ABP for u' has size $O(es^{\gamma_1+\gamma_2+2} + r^{5+\gamma_2}ds)$. This gives the bound $O(ds^{\gamma_1+\gamma_2+4} + r^{5+\gamma_2}ds)$, since $e = B + 1 \leq (2d+1)s^2 + ds + 1 = O(ds^2)$. Since γ_1 must always be at least 1, and $r \leq s$, we can crudely simplify this bound to $O(drs^{\gamma_1+\gamma_2+4})$. $\square$

Corollary 1. *The vector u'' can be computed by a multi-output ABP of size $O(d^{\gamma_3}r^{\gamma_3}s^{(\gamma_1+\gamma_2+4)\gamma_3+2})$.*

Proof. By Lemma 9, we can compute g^e by an ABP of size $O(es^{\gamma_1+\gamma_2}) = O(ds^{\gamma_1+\gamma_2+2})$. We then apply Lemma 2 to get for each entry of u'' an ABP of size $O(d^{\gamma_3}r^{\gamma_3}s^{(\gamma_1+\gamma_2+4)\gamma_3})$. $\square$

Note that we can bound $deg(u'') = O(d^2s^2)$. Hence Lemma 3 and Corollary 1 give that

Corollary 2. *The vector u''' can be computed by a multi-output ABP of size of size $O(d^{\gamma_3+2}r^{\gamma_3}s^{(\gamma_1+\gamma_2+4)\gamma_3+4})$.*

We apply M one more time to obtain the eigenvector corresponding to w_1. We have that

$$Mu''' = gp_1u'''.$$

We know that $Mu''' \in \mathbb{F}[X]$, since M only contains polynomial entries and $u''' \in \mathbb{F}[X]$. Hence, if ℓ is such that $(u''')_\ell$ is a nonzero component (which must exist), we get that

$$\frac{(Mu''')_\ell}{g(u''')_\ell} = p_1.$$

Both for the enumerator and denominator in the above expression we have ABPs of size $O(d^{\gamma_3+2}r^{\gamma_3}s^{(\gamma_1+\gamma_2+4)\gamma_3+4})$. Finally, we apply Lemma 2 to perform the exact division. Hence p_1 can be computed by a ABP of size $O(d^{\gamma_3^2+2\gamma_3}r^{\gamma_3^2}s^{(\gamma_1+\gamma_2+4)\gamma_3^2+4\gamma_3})$. This completes the proof of Lemma 6. $\square$

We can now prove Theorem 1. By Lemma 3, we have an ABP for the coefficient of y^r in f of size $O(rs)$. This program computes $(-1)^r p_0$. If there are p_is that are constant, any of these can be computed by ABPs with size at most $O(1)$. We use Lemma 2 to obtain an ABP of size $O(r^{\gamma_3}s^{\gamma_3})$, where we have divided out p_0 and any such linear factors. After this we we convert the ABP to standard form using Lemma 4. This blows up the size to $s' = O(r^{\gamma_3+2}s^{\gamma_3})$. We can now apply Lemma 6 to obtain an ABP for p_1 of size $O(d^{\gamma_3^2+2\gamma_3}r^{\gamma_3^2}(s')^{(\gamma_1+\gamma_2+4)\gamma_3^2+4\gamma_3})$, where γ_1, γ_2 and γ_3 are the absolute constants introduced in Section 2. We conclude p_1 has an ABP of size $poly(d, r, s)$. $\square$

5 Roots of Arithmetic Circuits and Newton's Method

For $f(y) \in \mathbb{R}[y]$ with $f(p) = 0$ for $p \in \mathbb{R}$, recall the update rule for Newton's method $y_{k+1} = y_k - \frac{f(y_k)}{f'(y_k)}$, where f' is the derivative of f. For arithmetic circuits we have the following analogue, where we compute successively better approximations $p_{\leq k}, p_{\leq k+1}, \ldots$ to a root $p \in \mathbb{F}[X]$ of $f \in \mathbb{F}[X, y]$.

Lemma 13. *Let $f \in \mathbb{F}[X, y]$ and let $f'(x, y) := \frac{\partial f}{\partial y}$. Let $p \in \mathbb{F}[X]$ be a root of f for y, and assume that $\xi_0 := f'(0, p(0)) \neq 0$. Then $\forall k \geq 1$ it holds that $p_{\leq k+1} = p_{\leq k} - \frac{1}{\xi_0} \cdot f(x, p_{\leq k})_{=k+1}$.*

Proof. Let $r = \deg_y(f)$ and write $f = \sum_{i=0}^r C_i(x) y^i$. So $f'(x, y) = \frac{\partial f}{\partial y} = \sum_{i=1}^r iC_i(x)y^{i-1}$.

The following computation is modulo the ideal I_{k+2} generated by $x_1^{k+2}, x_2^{k+2}, \ldots, x_n^{k+2}$, i.e. we identify any polynomials g and h if $[g]_{\leq k+1} = [h]_{\leq k+1}$.

$$\begin{aligned}
0 &\equiv f(x,p) \\
&\equiv f(x, p_{\leq k} + p_{=k+1}) \\
&\equiv \sum_{i=0}^{r} C_i(x) \left(p_{\leq k} + p_{=k+1}\right)^i \\
&\equiv C_0(x) + \\
&\quad \sum_{i=1}^{r} C_i(x) \left((p_{\leq k})^i + i \cdot (p_{\leq k})^{i-1} \cdot p_{=k+1}\right) \\
&\equiv \sum_{i=0}^{r} C_i(x)(p_{\leq k})^i + \\
&\quad p_{=k+1} \cdot \sum_{i=1}^{r} i \cdot C_i(x)(p_{\leq k})^{i-1} \\
&\equiv f(x, p_{\leq k}) + p_{=k+1} \cdot f'(x, p_{\leq k}) \\
&\equiv f(x, p_{\leq k}) + p_{=k+1} \cdot f'(x, p_{\leq k})_{=0}.
\end{aligned}$$

Note that $f'(x, p_{\leq k})_{=0} = f'(0, p_{\leq k}(0)) = f'(0, p(0)) = \xi_0$. We get that without going modulo I_{k+2}, the following equation is satisfied: $0 = f(x, p_{\leq k})_{=k+1} + p_{=k+1} \cdot \xi_0$. This implies the statement of the lemma. $\square$

5.1 Proof of Theorem 2

Let $f'(x, y) := \frac{\partial f}{\partial y}$. In case $f'(0, p(0)) \neq 0$, we can construct an arithmetic circuit for p by repeatedly applying Lemma 13. We compute the components of p separately, starting with p_0 and p_1, which we can easily compute within size $O(s)$. To compute $p_{=k+1}$, provided we have $p_0, p_1, \ldots, p_{=k}$ computed at gates somewhere already, we use a copy of a circuit Φ that computes the homogeneous components of f up to degree $k+1 \leq deg(p)$. This is a circuit for which, similar to the proof of Lemma 3, each node is split into $k+1$ nodes computing homogeneous components. Let $v_0, v_1, \ldots, v_{k+1}$ be the gates in Φ corresponding to the output gate of the original circuit, i.e. $f_0, f_1, \ldots, f_{=k+1}$ are computed at these gates. We can bound the size of Φ by $O(\mathcal{M}(k+1)s)$, provided we use a gadget of size $\mathcal{M}(k+1)$ that computes the coefficient map of polynomial multiplication, in order to deal with multiplication. Note that having $p_0, p_1, \ldots, p_k$ computed separately at gates is exactly the right format for feeding $p_{\leq k}$ into Φ for the variable y. A straightforward structural induction proves that after rewiring, for every $0 \leq i \leq k+1$, the gate v_i computes $f(x, p_{\leq k})_i$. Lemma 13 tells us that after rescaling the output of the gate v_{k+1} by a factor $-1/\xi_0$, we have obtained $p_{=k+1}$. We repeat the previously described construction for k up to degree $deg(p)$. This way, we obtain a circuit for p of size $O(\mathcal{M}(deg(p)) \cdot deg(p) \cdot s)$.

If $f'(0, p(0)) = 0$, then we can reduce to the above case as follows. Write $f = \sum_{i=0}^{r} C_i(x) y^i$ with $C_r(x) \not\equiv 0$. Let $f^i(x,y) = \frac{\partial^i f}{\partial^i y}$. Then $f^r(x,y) = r! \cdot C_r(x)$. Since the characteristic of $\mathbb{F}$ is zero, $r! \neq 0$, so $f^r(x,p) \not\equiv 0$. We have in this case that $f^0(x,p) \equiv 0$. Let i be the smallest integer for which $f^i(x,p) \not\equiv 0$. Then $0 < i \leq r$, and $f^{i-1}(x, p(x)) \equiv 0$. Due to Lemma 1, $f = (y-p)^m h$, for some polynomial h not divsible by $y - p$. By repeatedly computing partial derivatives one easily observes that the number i equals the multiplicity m of the root p in f.

We have that there exists $x_0 \in \mathbb{F}$ such that $f^i(x_0, p(x_0)) \neq 0$. Let $g(x,y) = f^{i-1}(x + x_0, y)$, and let $q = p(x + x_0)$. By Theorem 4, one gets that g is computable by a circuit of size $O(\mathcal{M}(m)s)$. Let $g' = \frac{\partial g}{\partial y}$. Then $g'(x,y) = f^i(x + x_0, y)$. The polynomial g is not identically zero, and $g(x, q(x)) = f^{i-1}(x + x_0, p(x + x_0)) \equiv 0$, and furthermore $g'(0, q(0)) = f^i(x_0, p(x_0)) \neq 0$. Now one proceeds as in the first case, to get a circuit for q of size $O(\mathcal{M}(m)\mathcal{M}(deg(p)) \cdot deg(p) \cdot s)$, from which one obtains a circuit for p of size $O(\mathcal{M}(m)\mathcal{M}(deg(p)) \cdot deg(p) \cdot s)$. □

Corollary 3. *The class* VP *is closed under taking roots.*

References

[1] Erich Kaltofen. Factorization of polynomials given by straight-line programs. In *Randomness and Computation*, pages 375–412. JAI Press, 1989.

[2] P. Bürgisser. The complexity of factors of multivariate polynomials. *Found. Comput. Math.*, 4(4):369–396, 2004.

[3] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity testing means proving circuit lower bounds. *Computational Complexity*, 13(1–2):1–44, 2004.

[4] Z. Dvir, A. Shpilka, and A Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. In *Proceedings of the 40th Annual STOC*, pages 741–748, 2008.

[5] M. Jansen. Weakening assumptions for deterministic subexponential time non-singular matrix completion. In *27th International Symposium on Theoretical Aspects of Computer Science (STACS 2010)*, volume 5 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 465–476, 2010.

[6] D.G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28(7):693–701, 1991.

[7] E. Kaltofen and P. Koiran. Expressing a fraction of two determinants as a determinant. In *Proceedings, The 19th International Symposium on Symbolic and Algebraic and Computation (ISSAC)*, pages 141–146, 2008.

[8] M. Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago Journal of Theoretical Computer Science*, 1997(Article 5), 1997.

[9] N. Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8(1–2):7–29, 1999.

[10] E. Kaltofen and M. Singer. Size-efficient parallel algebraic circuits for partial derivatives. In V. Shirkov, V.A. Rostovtsev, and V.P. Gerdt, editors, *Proceedings, IV International Conference on Computer Algebra in Physical Research*, pages 133–145. World Scientific, 1991.

[11] L. Valiant. Completeness classes in algebra. In *Proc. 11th Annual ACM Symposium on the Theory of Computing*, pages 249–261, 1979.

[12] A. Shpilka and A. Wigderson. Depth-3 arithmetic formulae over fields of characteristic zero. *Journal of Computational Complexity*, 10(1):1–27, 2001.