Electronic Colloquium on Computational Complexity, Report No. 67 (2015)

On hardness of multilinearization, and VNP-completeness in characteristics two

Pavel Hrubeš*

April 21, 2015

Abstract

For a boolean function $f : \{0,1\}^n \to \{0,1\}$, let $\hat{f}$ be the unique multilinear polynomial such that $f(x) = \hat{f}(x)$ holds for every $x \in \{0,1\}^n$. We show that, assuming VP $\neq$ VNP, there exists a polynomial-time computable f such that $\hat{f}$ requires super-polynomial arithmetic circuits. In fact, this f can be taken as a monotone 2-CNF, or a product of affine functions.

This holds over any field. In order to prove the results in characteristics two, we design new VNP-complete families in this characteristics. This includes the polynomial EC_n counting edge covers in a graph, and the polynomial $\mathrm{mclique}_n$ counting cliques in a graph with deleted perfect matching. They both correspond to polynomial-time decidable problems, a phenomenon previously encountered only in characteristics $\neq 2$.

1 Introduction

Arithmetic circuit is a standard model for computing polynomials over a field. It resembles a boolean circuit, except that an arithmetic circuit uses $+, \times$ as basic operations. The two most familiar arithmetic complexity classes, introduced by Valiant [10], are VP and VNP, and resemble the boolean classes P/poly and NP/poly. (For more details, we point the reader to, e.g., [7, 3].) Arguably, arithmetic circuits are better understood than boolean ones: several results which hold in the arithmetic setting have no known counterpart in the boolean world. Most notably, a polynomial-size arithmetic circuit computing a polynomial of polynomially-bounded degree can be simulated by a circuit of polynomial size and $O(\log^2 n)$ depth, see [9]. In the boolean setting, this would amount to asserting P/poly = NC_2/poly. Moreover, main open problems in arithmetic complexity – such as proving super-polynomial lower bounds on circuit size of an explicit polynomial – can be seen as special cases of the corresponding boolean problems, and are therefore considered easier (at least in a finite underlying field). Hence, it would be desirable to have a means of translating results from arithmetic to boolean complexity.

One such possibility[1] is the following. With a boolean function f, associate the unique multilinear polynomial $\hat{f}$ which takes the same values as f on $0,1$-inputs. Can it be the case that $\hat{f}$ has a polynomial size *arithmetic* circuit whenever f has polynomial size *boolean* circuit? This would have quite interesting consequences, including P/poly = NC_2/poly or that, in principle, arithmetic lower bounds imply boolean lower ones. Not surprisingly, we show that this is not the case: assuming VP $\neq$ VNP, there exists a polynomial-time computable boolean function f such that $\hat{f}$ requires superpolynomial arithmetic circuits. Moreover, the function f can be very simple, a monotone 2-CNF or a product of linear functions over $\mathbb{F}_2$. The converse also holds: if VP = VNP then $\hat{f}$ has complexity polynomial in that of f. These results are similar to the VNP-dichotomy theorem in [1].

*Institute of Mathematics of ASCR, Prague, Czech Republic, pahrubes@gmail.com. Supported by ERC grant FEALORA 339691.

[1]Suggested to the author by A. Rao

ISSN 1433-8092

The above holds over any underlying field. We observe that the results are easy in characteristics different from 2, whereas characteristics 2 requires much more work. This is a frequent phenomenon in arithmetic complexity: for example, completeness results in Burgisser's monograph [2] deal almost exquisitely with char $\neq 2$, and similarly for the dichotomy in [1]. However, this is not caused by a pathological nature of char $= 2$, but rather by the lack of examples of VNP-complete families. In [10], Valiant has shown that the permanent polynomial, perm_n, is VNP-complete over any field of characteristics $\neq 2$, and the Hamiltonian cycle polynomial, HC_n, is complete over any field. The permanent counts the number of perfect matchings in a bipartite graph. In view of its simplicity, it has become synonymous with VNP in char $\neq 2$. HC_n counts the number of Hamiltonian cycles in a graph, and is much more complicated than perm_n. One difference is the difficulty of the underlying decision problems: we can decide in polynomial time whether a graph has a perfect matching, whereas testing for a Hamiltonian cycle is NP-hard. This means that it is easier to deduce completeness of other polynomials by a reduction to perm_n, and an abundance of such families was presented in [2]. To the author's knowledge, HC_n was the only previously known VNP-complete family in characteristics two.

In this paper, we fill the gap by providing several new examples of VNP-complete families in characteristics two. This includes the polynomial clique^*_n which counts cliques of all sizes in a graph, the polynomial $\mathrm{mclique}_n$ which counts n-cliques in $2n$-vertex graph with a deleted matching, or the edge cover polynomial. The latter families correspond to polynomial-time decision problems. We do not deduce VNP-completeness from the completeness of HC_n, but rather employ the $\oplus$P-completeness proof of $\oplus$2SAT, as given by Valiant in [11].

2 Preliminaries

Polynomials and arithmetic circuits Let $\mathbb{F}$ be field. A polynomial f over $\mathbb{F}$ in variables $x_1, \dots, x_n$ is a finite sum of the form $\sum_J c_J x^J$, where $J = \langle j_1, \dots, j_n \rangle \in \mathbb{N}^n$, $c_J \in \mathbb{F}$ and x^J denotes the monomial $\prod_{i\in[n]} x_i^{j_i}$. The degree of a monomial x^J is $\sum_{i\in[n]} j_i$, and the degree of a polynomial is the maximum degree of a monomial with a non-zero coefficient.

The standard model for computing polynomials over $\mathbb{F}$ is that of *arithmetic circuit*. An arithmetic circuit starts from the variables $x_1, \dots, x_n$ and elements of $\mathbb{F}$, and computes f by means of the ring operations $+, \times$. The exact definition can be found in, e.g., in [7]. We denote

$$\mathrm{C}(f) := \text{the size of a smallest arithmetic circuit computing } f\,.$$

The classes VP, VNP, completeness and hardness VP and VNP are the two most interesting complexity classes in arithmetic computation. The definitions are explained in greater detail in [7, 2, 3], and we give just the main points.

A family of polynomials $\{f_n\} = \{f_n\}_{n\in\mathbb{N}}$ is in VP, if f_n has polynomially bounded degree and circuit size. The family is in VNP, if $f_n(x) = \sum_{u\in\{0,1\}^m} g_{t(n)}(u, x)$ where $t : \mathbb{N} \to \mathbb{N}$ is polynomially bounded and $\{g_n\}$ is a family in VP. A polynomial $f(x_1, \dots, x_n)$ is a *projection* of $g(y_1, \dots, y_m)$, if there exist $a_1, \dots a_m \in \mathbb{F} \cup \{x_1, \dots, x_n\}$ such that $f(x_1, \dots, x_n) = g(a_1, \dots, a_m)$. $\{g_n\}$ is a p-projection of $\{f_n\}$, if there exists a polynomially bounded $t : \mathbb{N} \to \mathbb{N}$ such that g_n is a projection of $f_{t(n)}$ for every n. A family $\{f_n\}$ is VNP-*complete*, if it is in VNP and every family in VNP is a p-projection of $\{f_n\}$. As customary, we will often identify a family $\{f_n\}$ with the polynomial f_n.

The best known VNP-complete polynomials are the permanent and the Hamiltonian cycle polynomial

$$\mathrm{perm}_n := \sum_{\sigma} \prod_{i=1}^{n} x_{i,\sigma(i)}\,,\ \mathrm{HC}_n := \sum_{\sigma'} \prod_{i=1}^{n} x_{i,\sigma(i)}\,,$$

where σ ranges over permutations of $[n]$ and σ' over all cycles in S_n (i.e., every monomial in HC_n corresponds to a Hamiltonian cycle in the complete directed graph on n vertices). Valiant [10] has shown that the permanent family is VNP complete over any field of characteristic different from 2, and HC_n is VNP-complete over any field.

Our last definition is less standard. We will say that a family $\{f_n\}$ is *hard for* VNP if for every family $\{g_n\} \in$ VNP, there exists a polynomially bounded $t : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$ such that

$$\mathrm{C}(g_n) = O(n^c \cdot \mathrm{C}(f_{t(n)}))\,.$$

Clearly, it is enough to take for $\{g_n\}$ a VNP-complete family. We do not require that g_n is somehow reducible to $f_{t(n)}$, only that the arithmetic complexity of g_n is polynomially bounded by that of $f_{t(n)}$. In Section 3.1, we will compare this with the more common notion of c-reduction.

Notation For $v = \langle v_1, \ldots, v_n\rangle \in \{0,1\}^n$, $|v| = \sum_{i=1}^n v_i \in \mathbb{N}$ denotes the number of 1's in v. If $x = \langle x_1, \ldots, x_n\rangle$ is a vector of variables, we define the polynomials x^v and x_v as

$$x^v := \prod_{i:v_i=1} x_i\,,\; x_v := \prod_{i:v_i=0} (1-x_i)\,. \tag{1}$$

We usually write x as $\{x_1, \ldots, x_n\}$, identifying $v \in \{0,1\}^n$ with a function from x to $\{0,1\}$.

Multilinearization A polynomial f in variables $x_1, \ldots, x_n$ is *multilinear*, if $f = \sum_{v\in\{0,1\}^n} c_v x^v$. In other words, every monomial containing x_i^k with $k > 1$ has zero coefficient in f. Let f be a function $f : \{0,1\}^n \to \mathbb{F}$. The *multilinearization of* f is the unique multilinear polynomial $\hat{f}$ over $\mathbb{F}$ which satisfies $\hat{f}(v) = f(v)$ for every $v \in \{0,1\}^n$. The multilinearization can be explicitly written as

$$\hat{f}(x_1, \ldots, x_n) = \sum_{v\in\{0,1\}^n} f(v) x^v x_v\,. \tag{2}$$

A boolean function $f : \{0,1\}^n \to \{0,1\}$ is automatically a function $f : \{0,1\}^n \to \mathbb{F} \supseteq \{0,1\}$, and the definition applies also in this case. However, $\hat{f}$ significantly depends on the ambient field $\mathbb{F}$.

2.1 Main results

We are interested in the arithmetic circuit complexity of computing $\hat{f}$, provided f itself is easy to compute. This is interesting in two cases. First, when $f : \{0,1\}^n \to \{0,1\}$ is a boolean function with a small boolean circuit, or second, f is a polynomial computable by a small arithmetic circuit. The two cases are not unrelated, since a boolean circuit can be simulated by an arithmetic circuit on 0, 1-inputs (e.g., replace $\neg x$ by $1 - x$, $x \wedge y$ by $x \cdot y$ and $x \vee y$ by $xy - x - y + 1$).

A monotone 2-CNF is a booloean formula of the form $\bigwedge_{\langle i,j\rangle\in A}(x_i \vee x_j)$ for some $A \subseteq [n] \times [n]$. In the next section, we prove the following:

Theorem 1. *Let $\mathbb{F}$ be an arbitrary field. For every n, there exists a boolean function $\alpha_n : \{0,1\}^n \to \{0,1\}$ which can be computed by a monotone 2-CNF but the family $\{\hat{\alpha}_n\}$ is hard for* VNP*. Moreover, the 2-CNF is polynomial-time constructible and the family $\{\hat{\alpha}_n\}$ is* VNP*-complete in* $\mathrm{char}(\mathbb{F}) \neq 2$.

This implies:

Corollary 2. *Assume that* VP $\neq$ VNP*. Then there exists $\{f_n\} \in$* VP *such that $\{\hat{f}_n\} \notin$* VP*.*

Theorem 1 and Corollary 2 show that boolean functions or polynomials cannot be efficiently multilinearized, unless VP = VNP. The converse also holds[2]:

Proposition 3. *Assume that $\{f_n\}$ is i) a family of polynomials in* VNP*, or ii) a family of boolean function which is in P/poly. Then $\{\hat{f}_n\}$ is in* VNP*.*

[2]Instead of P/poly, one could have ♯P/poly.

Proof. i) Equation (2) can be written as $\hat{f} = \sum_{v\in\{0,1\}^n} \left(f(v)\prod_{i=1}^n (x_i v_i + (1-x_i)(1-v_i))\right)$. This shows that $\{\hat{f}_n\} \in$ VNP. ii) If $f : \{0,1\}^n \to \{0,1\}$ has a boolean circuit of size s, we can find a polynomial f_1 with an arithmetic circuit of size $O(s)$ such that $f(u) = f_1(u)$ for every $u \in \{0,1\}^n$. However, this polynomial may have an exponential degree. Instead, encode the boolean circuit as a 3-CNF in $m = O(s)$ new variables, obtaining a polynomial of degree $O(s)$ so that $f_1(u) = \sum_{v\in\{0,1\}^m} f_2(u, v)$ holds for every $u \in \{0,1\}^n$, and proceed as in i). □

Other contributions of this paper are the following.

Multilinearization of linear products In Theorem 7, Section 4, we consider $\hat{f}$ for f defined as a product of affine functions. We show that this is hard for VNP already when each affine function depends on two variables only. The exception is the two-element field where three variables are necessary.

VNP**-completeness in characteristics 2** In Section 5 we provide new examples of VNP-complete families in characteristics two. In Theorem 10, we first prove VNP-completeness of the clique polynomial

$$\text{clique}^*_n = \sum_{A\subseteq[n]} \prod_{i<j\in[n]} x_{i,j}\,.$$

We use it to deduce completeness of other polynomials in Theorem 13. We focus on families based on polynomial-time decision problems, as well as polynomials whose coefficients can be expressed in terms of CNF's. In particular, the polynomial DS_n is used in the proof of Theorem 1. In Section 6, we discuss structural properties of the VNP-families in a greater detail.

3 Multilinearization of 2-CNFs

In this section, we prove Theorem 1. In order to appreciate the power of multilinearization, let us first sketch a simple proof of Corollary 2 in $\mathrm{char}(\mathbb{F}) \neq 2$. Let f_n be the polynomial

$$f_n := \prod_{i\in[n]} \sum_{j\in[n]} x_{ij} z_j\,.$$

Then $\hat{f}_n = (\prod_{i\in[n]} z_i) \cdot \mathrm{perm}_n + g$, where g has degree $< 2n$. $(\prod_{i\in[n]} z_i) \cdot \mathrm{perm}_n$ is homogeneous of degree $2n$, and so $(\prod_{i\in[n]} z_i) \cdot \mathrm{perm}_n$ is the $2n$-homogeneous part of $\hat{f}_n$. To conclude VNP-hardness, it is enough to recall the following:

Lemma 4. *For $k \in \mathbb{N}$, let $f^{(k)}$ be the k-homogeneous part of the polynomial f. Then $f^{(0)}, \ldots, f^{(k)}$ can be simultaneously computed by a circuit of size $O(C(f)k^2)$.*

This fact traces back to Strassen [8], and appears in various places, including [7].

To prove Theorem 1, we need an appropriate 2-CNF, and the following lemma. The lemma shows that from a multilinear polynomial $f(x, y)$, we can easily compute other polynomials such as $\sum_{v\in\{0,1\}^n} f(v, y)$.

Lemma 5. *Let $f(x, y)$ be a multilinear polynomial in two disjoint sets of variables x, y, with $x = \{x_1, \ldots, x_n\}$ and $C(f(x, y)) = s$. For every $r \leq n$, the following can be computed by circuits of size $O(sn^2)$:*

(i). $\sum_{v\in\{0,1\}^n} f(v, y)x^v$, $\sum_{v\in\{0,1\}^n,|v|=r} f(v, y)x^v$,

(ii). $\sum_{v\in\{0,1\}^n} f(v, y)$, $\sum_{v\in\{0,1\}^n,|v|=r} f(v, y)$

Moreover, if $\mathrm{char}(\mathbb{F}) \neq 2$, *we have* $\sum_{v\in\{0,1\}^n} f(v, y) = 2^n f(1/2, \ldots, 1/2, y)$.

In characteristics $\neq 2$, the "moreover" part was observed in [6].

Proof. We will suppress the dependance on y, writing $f(x)$ instead of $f(x,y)$. Accordingly, degree of f is taken with respect to the variables x. Since f is multilinear, it can be written as (v ranges over $\{0,1\}^n$)

$$f(x) = \sum_v f(v)x^v x_v = \sum_v \left(f(v) \prod_{i:v_i=1} x_i \prod_{i:v_i=0} (1-x_i) \right) . \tag{3}$$

In $\mathrm{char}(\mathbb{F}) \neq 2$, if we set $x_1,\dots,x_n$ to $1/2$, we obtain $x^v x_v = 2^{-n}$, for every v. Hence, $f(1/2,\dots,1/2) = 2^{-n}\sum_v f(v)$, concluding the "moreover" part.

To prove (i), recall Lemma 4 and another useful fact, again due to Strassen [8]: if a polynomial g has degree d and can be computed by a circuit *with division gates* of size s, it can be computed by a circuit without divisions of size $O(sd^2)$. (Strictly speaking, this holds in infinite fields; in finite fields the complexity may be slightly larger [4].) This said, we claim that

$$\sum_v f(v)x^v = f(x_1/(1+x_1),\dots,x_n/(1+x_n)) \prod_{i\in[n]} (1+x_i) . \tag{4}$$

This follows from (3): we have

$$\prod_{i:v_i=1} \frac{x_i}{1+x_i} \prod_{i:v_i=0} \left(1 - \frac{x_i}{1+x_i}\right) = \prod_{i:v_i=1} x_i \cdot (\prod_{i\in[n]} (1+x_i))^{-1} = x^v (\prod_{i\in[n]} (1+x_i))^{-1} ,$$

giving (4). This shows that $\sum_v f(v)x^v$ has circuit complexity $O(sn^2)$. Furthermore, $\sum_{|v|=r} f(v)x^v$ is the r-homogeneous part of $\sum_v f(v)x^v$ – this would give circuit complexity $O(sn^4)$. In order to obtain the $O(sn^2)$ bound, it is enough to reproduce the division elimination proof directly. In (4), replace $(1+x_i)^{-1}$ by its truncated power series, namely, with $\lambda(x_i) = \sum_{j=0}^{n-1} (-1)^j x_i^j$. Then $\sum_{|v|=r} f(v)x^v$ is the r-homogeneous part of $f(x_1\lambda(x_1),\dots,x_n\lambda(x_n)) \prod_{i\in[n]}(1+x_i)$.

(ii) follows from (i) by setting $x_1,\dots,x_n := 1$. □

Proof of Theorem 1. Consider the DS_n polynomial defined in (6), Section 5.2, where we will prove its VNP-completeness over any field. It depends on $m = n(n+1)/2$ variables $x = \{x_i, x_{j,k} : i \in [n], j < k \in [n]\}$. The definition can be rewritten as $\mathrm{DS}_n = \sum_{v\in\{0,1\}^m} \alpha_n(v)x^v$, where α_n is the boolean function

$$\alpha_n(y) := \bigwedge_{i<j\in[n]} ((\neg y_{i,j} \vee \neg y_i) \wedge (\neg y_{i,j} \vee \neg y_j)) .$$

By Lemma 5 part (i), we have $\mathrm{C}(\mathrm{DS}_n) = O(\mathrm{C}(\hat\alpha_n)m^2))$, and hence $\{\hat\alpha_n\}$ is VNP-hard. α_n is not monotone but rather antimonotone (i.e., all variables are negated). However, switching $\neg y_a$ to y_a in α_n amounts to switching y_a to $1-y_a$ in $\hat\alpha_n$, and has negligible effect on complexity. We can achieve that α_n depends on n variables by reindexing the family.

To prove VNP-*completeness* in char $\neq 2$, consider the function

$$g_n(x,y,x_0) := x_0 \wedge \alpha_n(y) \wedge \bigwedge_{i\in[n], j<k\in[n]} ((\neg y_i \vee x_i) \wedge (\neg y_{j,k} \vee x_{j,k}) .$$

It is easy to see that $\sum_{v\in\{0,1\}^m} \hat g_n(x,v,x_0) = x_0\mathrm{DS}_n$. Hence, by the "moreover" part of Lemma 5 , we have $x_0\mathrm{DS}_n = 2^n \hat g_n(x,1/2,\dots,1/2,x_0)$ and hence $\mathrm{DS}_n = \hat g_n(x,1/2,\dots,1/2,2^n)$. That is, DS_n is a projection of $\hat g_n$. The variables x,x_0 occur in g_n only positively and y only negatively. However, the y variables are all in the scope of the boolean sum, and replacing $\neg y_a$ by y_a in g_n yields the same result. □

3.1 Comments

In the proof, we used the polynomial DS_n, since it can be easily expressed in terms of a 2-CNF. In characteristics $\neq 2$, we could have used the permanent instead. We can write $\mathrm{perm}_n(x) = \sum_{|v|=n} x^v f_n(v)$, where f_n is an antimonotone 2-CNF. Namely,

$$f_n(y) = \bigwedge_{i_1 \neq i_2, j \in [n]} ((\neg y_{i_1,j} \vee \neg y_{i_2,j}) \wedge (\neg y_{j,i_1} \vee \neg y_{j,i_2})) \ .$$

This would give *hardness* of $\hat{f}_n$ by Lemma 5 part (i). To obtain VNP-completeness, one can use the *partial permanent* polynomial, defined by

$$\mathrm{perm}^*_n := \sum_{\beta} \prod_{i \in \mathrm{dom}(\beta)} x_{i,\beta(i)} \ ,$$

where β ranges over injective partial functions from $[n]$ to $[n]$ (the empty product equals 1). That the family perm^*_n is VNP-complete in char $\neq 2$ was shown in [5, 2]. The advantage of perm^*_n is that $\mathrm{perm}^*_n = \sum_v x^v f_n(v)$ with v ranging over all of $\{0,1\}^{n^2}$. Furthermore, Theorem 1 in char $= 2$ can be proved directly using Proposition 9

The difference between hardness and completeness in Theorem 1 is due to the restricted nature of p-projections, and the family $\hat{\alpha}_n$ is complete with respect to more general reductions. In Lemma 5, we need to compute $\sum_{v \in \{0,1\}^n} f(v,y)$ from $f(x,y)$ when f is multilinear. In characteristics different from two, this can be done by the projection $x := 1/2, \dots, 1/2$. In general, the Lemma chiefly relies on computing homogeneous components of $f(h,y)$, where h is a substitution from VP. In infinite field, this will be accommodated by the more general *c-reduction* (introoduced in [2]). In this reduction, we think of f as an oracle and a computation can apply $+$, $\times$ or f to previously computed values. By means of interpolation, the homogeneous components of f can be obtained from f via c-reductions (see [2]). We note:

Remark 6. *(i). The polynomial $\hat{\alpha}_n$ from Theorem 1 can be evaluated in polynomial time on every* $0,1$-*input. Hence, the family cannot be* VNP*-complete in $\mathbb{F}_2$ unless* $\oplus$P*/poly $\subseteq$ P/poly (this is both with respect to p-projections and c-reductions).*

(ii). If $\mathbb{F}$ is infinite, but of arbitrary characteristics, $\hat{\alpha}_n$ is VNP*-complete with respect to c-reductions.*

4 Multilinearization of linear products

Here, we consider hardness of multlinearization of products of affine functions. An *affine function* over a field $\mathbb{F}$ is a polynomial of the form $\sum_{i=1}^n a_i x_i + a_0$ with $a_0, \dots, a_n \in \mathbb{F}$. Its *width* is the number of non-zero a_i's. The following theorem shows that products of functions of small width are hard to multilinearize.

Theorem 7. *Assume that $\mathbb{F}$ is of size at least three. Then*

(i). for every n, there exists a polynomial f_n in n variables which is a product of affine functions of width 2, but $\{\hat{f}_n\}$ is hard for VNP.

If $\mathbb{F} = \mathbb{F}_2$, then

(ii) the above holds with affine functions of width 3,

(iii) if f is a product of affine functions, each depending on at most 2 variables, then $C(\hat{f}) = O(n)$.

We deduce parts *(i)* and *(ii)* from Theorem 1. Let $\alpha = \alpha_n = \bigwedge_{\langle i,j \rangle \in A} (x_i \vee x_j)$ be the hard 2-CNF in n variables.

Proof of part (i). This is implied by the following:

Claim. *There exists $h(x_1, x_2)$ which is a product of three affine functions of width 2 such that for every $x_1, x_2 \in \{0,1\}$, $x_1 \vee x_2 = h(x_1, x_2)$.*

Proof. Assume that $\mathrm{char}(\mathbb{F}) \neq 2$. Then take the product $2(1 - x_1/2)(1 - x_2/2)(x_1 + x_2)$. If $\mathrm{char}(\mathbb{F}) = 2$ but $|\mathbb{F}| > 2$ then $\mathbb{F}$ contains the 4-element field $\mathbb{F}_4$. Choose two distinct non-zero $a, b \in \mathbb{F}_4$ and take the product $(ax_1 + bx_2)^3$. This works because $t^4 = t$ for every $t \in \mathbb{F}_4$. □

Instead of the 2-CNF α, we can take the product $\prod_{\langle i,j\rangle \in A} h(x_i, x_j)$. □

Proof of part (ii). With a disjunction $x_1 \vee x_2$, we associate L_{x_1,x_2}, a system of the three equations

$$z_{01} = x_1 + 1\,,\ z_{10} = x_2 + 1\,,\ z_{11} = x_1 + x_2 + 1\,,$$

where z_{01}, z_{10}, z_{11} are fresh variables. For the hard 2-CNF α, let $L := \bigcup_{\langle i,j\rangle \in A} L_{x_i,x_j}$. Setting $k := |A|$, the system L depends on $3k$ extra variables z.

Claim. *For every $x \in \{0,1\}^n$ the following are equivalent:*

(i). $\alpha(x) = 1$

(ii). there exists $z \in \{0,1\}^{3k}$ with $|z| = k$ such that x, z is a solution of L over $\mathbb{F}_2$, and such a z is unique.

Proof. L_{x_1,x_2} is set up so that the following hold. If $x_1, x_2, z_{01}, z_{10}, z_{11} \in \{0,1\}$ is a solution and $x_1 \vee x_2 = 0$ then $|z_{01}, z_{10}, z_{11}| = 3$. If $x_1 \vee x_2 = 1$ then $|z_{01}, z_{10}, z_{11}| = 1$. Hence, every solution x, z of L satisfies $|z| \geq k$ and equality holds iff $\alpha(x) = 1$. □

We can rewrite L as $\ell_1 = 1, \ldots, \ell_m = 1$, where every ℓ_i is a linear function of width ≤ 3. Define $g(x,z) := \prod_{i \in [m]} \ell_i$. The Claim entails that $\hat{\alpha}(x)$ can be written as $\hat{\alpha}(x) = \sum_{z \in \{0,1\}^{3k}, |z| = k} g(x,z)$. Therefore, $\hat{g}$ is VNP-hard by Lemma 5 part *(ii)*. □

Proof of part (iii). Assume that f is in variables $x_1, \ldots, x_n$ and $f = f_1 f_2 \cdots f_s$ where each f_i is an affine function depending on at most 2 variables. Consider the graph G on vertices $x_1, \ldots, x_n$ defined as follows: there is an edge between $x_i \neq x_j$ iff there exists $k \in [s]$ such that f_k depends on both x_i and x_j (i.e., $f_k = x_i + x_j$ or $f_k = x_i + x_j + 1$). Suppose G has connected components $G_1, \ldots, G_r$. Then $f = g_1 \cdots g_r$, where for every i, g_i is the product of the f_j's depending on some variable from G_i. Since $g_1, \ldots, g_r$ depend on disjoint sets of variables, we have $\hat{f} = \hat{g}_1 \cdots \hat{g}_r$, and it is enough to multilinearize each g_i separately. It is therefore sufficient to consider the case when G is connected. But then there exist at most two $u \in \{0,1\}^n$ such that $f(u) = 1$. For if we fix $x_1 \in \{0,1\}$, the equations $f_1 = 1, \ldots, f_s = 1$ have at most one solution: a simple path from x_1 to x_k in G determines x_k uniquely. Writing $\hat{f} = \sum_{v \in \{0,1\}^n} x^v x_v f(v) = \sum_{v: f(v) \neq 0} f(v) x^v x_v$ gives a circuit of size $O(n)$. □

We note that *(ii)* and *(iii)* of the theorem can be stated in a greater generality.

Remark 8. *(i). Parts (ii) and (iii) hold for any field $\mathbb{F}$, if f and f_n are taken as* boolean functions *defined as conjunctions of affine functions over $\mathbb{F}_2$.*

(ii). Given a set of linear equations over $\mathbb{F}_2$, we can count the number of solutions in polynomial time. Hence, the multilinearization in (ii) is easy to evaluate on every 0, 1*-input, and cannot be* VNP*-complete (unless* $\oplus$P/poly $\subseteq$ P/poly*).*

5 VNP completeness in characteristics two

In this section, we present new VNP-complete families in characteristics two. We emphasize that completeness is understood with respect to p-projections. The main tool is the following proposition, implicit in [11]. In this paper, Valiant proved $\oplus$P-completeness of $\oplus$2SAT, as well as of several other problems, including counting vertex covers in special kinds of bipartite graphs $\mod 2$. (An *antimonotone* 2-CNF is obtained by negating all variables in a monotone 2-CNF.)

Proposition 9 ([11])**.** *Let $f(x)$ be an n-variate boolean function computable by a circuit of size s. Then there exists a monotone (similarly, antimonotone) 2-CNF $g(x,y)$ in $m = O(s)$ auxiliary variables y such that for every $x \in \{0,1\}^n$, $f(x) = \sum_{y\in\{0,1\}^m} g(x,y) \mod 2$.*

Proof sketch. First, it is enough to consider the case of f being a 3-CNF and, second, a single disjunction of three variables or their negations. Consider the disjunction $f(x,y,z) = \neg x \vee \neg y \vee \neg z$. Then take the 2-CNF $g(x,y,z,u)$ which is the conjunction of $u \vee x, u \vee y, u \vee z$. The key observation is that if $f(x,y,z) = 1$, then $g(x,y,z,u) = 1$ has unique solution $u = 1$, and if $f(x,y,z) = 0$ then every $u \in \{0,1\}$ satisfies $g(x,y,z,u) = 1$. Hence, $f(x,y,z) = \sum_{u\in\{0,1\}} g(x,y,z,u) \mod 2$, allowing to rewrite a 3-CNF as a 2-CNF. To convert a 2-CNF to a monotone one, we can replace $x \vee \neg y$ with the conjunction $x \vee \bar{y}$, $y \vee \bar{y}$, $\neg y \vee \neg\bar{y}$, where the last disjunct can be treated as before. $\square$

In Section 5.1, we use the proposition to prove VNP-completeness of our first polynomial, clique_n^*. In Section 5.2, we use clique_n^* to conclude completeness of several other families.

5.1 Completeness of clique^*

The polynomial clique^* is defined as

$$\mathrm{clique}_n^* := \sum_{A\subseteq[n]} \prod_{i<j\in A} x_{i,j}\,, ,$$

where the empty products equal 1. Interpreting the variables as edges in a (simple and undirected) graph on n vertices, clique_n^* counts the number of cliques of all sizes. The polynomial has constant term $n+1$. In some contexts, it is more convenient to have the constant term equal 1, as in $(\mathrm{clique}_n^* - n)$. In this modification, VNP-completeness of clique_n^* in char $\neq 2$ was proved in [2].

In the rest of this section, we show:

Theorem 10. *The family $\{\mathrm{clique}_n^*\}$ is* VNP*-complete over any field.*

It is convenient to think of clique_n^* and similar polynomials in terms of edge-weighted graphs. Let $G = (V,E)$ be a (simple undirected) graph whose edges are weighted by a variable from a set x or an element of $\mathbb{F}$, via the function $w : E \to \mathbb{F} \cup x$. For $E' \subseteq E$, the weight of E' is defined as the product of weights in E' (empty products equal 1). A clique is a subset A of V such that every two distinct vertices in A are connected by an edge. The weight of a clique is the weight of its edge-set (hence, a clique of size ≤ 1 has weight 1). This guarantees that clique_n^* equals the sum of weights of all cliques in the complete graph on vertices $[n]$, where an edge between i,j, $i<j$, is weighted by $x_{i,j}$.

Lemma 11. *Let $f(x)$ be an antimonotone 2-CNF in variables $x = \{x_1,\dots,x_n\}$. Then there exists a graph $G = (V,E)$ with $|V| = O(n)$ and a weight function $w : E \to \mathbb{F} \cup x$, such that*

$$\sum_{u\in\{0,1\}^n} f(u)x^u = \sum_A w(A)\,, \tag{5}$$

where A ranges over all cliques of G.

Proof. Assume that f can be written as a conjunction of clauses $\mathcal{C} = C_1, \ldots, C_m$, where each C_i is of the form $\neg x_i \vee \neg x_j$ with $i, j \in \{1, \ldots n\}$. Let G be the graph whose vertices are $x_0, x_1, \ldots, x_n$, where x_0 is a new variable not appearing in $\mathcal{C}$. There is an edge between x_i and x_j, $i \neq j$, iff every clause in $\mathcal{C}$ is consistent with the assignment $x_i, x_j := 1$. (In other words, $\mathcal{C}$ does not contain $\neg x_{i'} \vee \neg x_{j'}$ for any $i', j' \in \{i, j\}$). This guarantees a one-to-one correspondence between cliques of G containing x_0 and satisfying assignments of $\mathcal{C}$: $v \in \{0,1\}^n$ satisfies $\mathcal{C}$ iff $A_v \cup \{x_0\}$ is a clique in G, where $A_v := \{x_i : v_i = 1, i \in \{1, \ldots n\}\}$. Let us weigh the graph as follows: an edge between x_0 and x_i is weighted by x_i and all other edges by 1. Hence, the weight of $A_v \cup \{x_0\}$ is $\prod_{i \in A_v} x_i = x^v$. All cliques not containing x_0 have weight 1. In other words, the sum of weights of cliques in G equals

$$\sum_{v \in \{0,1\}^n} x^v f(v) + a\,,$$

for some $a \in \mathbb{F}$. We can add to G an isolated edge with weight $-a-2$ to obtain G' with the required property. $\square$

We can now prove the theorem.

Proof of Theorem 10. Clearly, the family is in VNP. The family is complete in char $\neq 2$ as shown in [2], and it remains to deal with char $= 2$. We deduce its completeness from VNP-completeness of HC_n. The only property of HC_n we use is the following: it can be written as $\mathrm{HC}_n = \sum_{v \in \{0,1\}^{n^2}} f(v) x^v$, where x is the vector of its n^2variables and $f : \{0,1\}^{n^2} \to \{0,1\}$ is a boolean function of polynomial circuit size. By means of Proposition 9, we can write

$$\mathrm{HC}_n = \sum_{v \in \{0,1\}^{n^2}, u \in \{0,1\}^m} g(v,u) x^v\,,$$

where g is an antimonotone 2-CNF, m is polynomial in n, and the summation is in characteristics 2. Lemma 11 shows that the polynomial

$$\sum_{v \in \{0,1\}^{n^2}, u \in \{0,1\}^m} g(v,u) x^v y^u$$

is a projection of clique^*_k, with k polynomial in n. Setting the variables y to 1 means that also HC_n is a projection of clique^*_k $\square$

5.2 Other VNP-complete families

Let clique_n and $\mathrm{mclique}_n$ be the polynomials

$$\mathrm{clique}_n := \sum_{A \subseteq [2n], |A| = n} \prod_{i < j \in [2n]} x_{i,j}\,, \quad \mathrm{mclique}_n := \mathrm{clique}_n(x_{1,n+1}, \ldots, x_{n,2n} := 0)\,.$$

They are both homogeneous of degree $n(n-1)/2$. clique_n counts the number of cliques of size n in a $2n$-vertex graph. We can think of $\mathrm{mclique}_n$ as counting n-cliques in a special kind of graph, which we call a *graph with forbidden matching*. This is a graph on $2n$ vertices $a_1, \ldots, a_n$, $b_1, \ldots, b_n$ such that there is no edge between a_i and b_i for every $i \in [n]$. We note that completeness of clique could be proved directly via parsimonius reductions to 3-SAT. mclique is more interesting, because the corresponding decision problem is in polynomial time:

Observation 12. *Given a $2n$-vertex graph G with forbidden matching, we can decide in polynomial time whether it contains a clique of size n.*

Proof. We assume that the forbidden matching is part of the input (otherwise, we can find it in polynomial time by finding a perfect matching in the complementary graph). Note that every n-clique in G must contain precisely one of the vertices a_i, b_i for every $i \in [n]$. Identifying a_i with i and b_i with $i+n$, we then see that G has an n-clique iff the following clauses are satisfiable

$$x_i \vee x_{i+n}\ ,\ i \in [n]\ ,\ \neg x_j \vee \neg x_k\ , \text{ for all } j \neq k \in [2n] \text{ such that } j,k \text{ are not incident}\,.$$

This is a set of 2-clauses and its satisfiability can be determined in polynomial time. $\square$

We also define the *subgraph counting polynomial* and *disjoint subgraph polynomial* by

$$\mathrm{CS}_n := \sum_{A\subseteq[n],B\subseteq A^{(2)}} \left(\prod_{i\in A} x_i \prod_{\langle j,k\rangle\in B} x_{j,k}\right)\ ,\ \mathrm{DS}_n := \sum_{A\subseteq[n],B\subseteq([n]\setminus A)^{(2)}} \left(\prod_{i\in A} x_i \prod_{\langle j,k\rangle\in B} x_{j,k}\right)\ . \tag{6}$$

Here $A^{(2)} := \{\langle j,k\rangle : j < k \in A\}$. The motivation is the following: if $B \subseteq A^{(2)}$ then B can be viewed as a set of edges on vertices A, and so (B, A) is a subgraph of the complete n-vertex graph.

Finally, we present two polynomials counting edge-coverings of a graph

$$\mathrm{EC}^*_n := \sum_{B} \prod_{\langle j,k\rangle\in B} x_{j,k}\ ,\ \mathrm{EC}_n := \sum_{|B|=\lceil 3n/4\rceil} \prod_{\langle j,k\rangle\in B} x_{j,k}\ ,$$

where B ranges over $B \subseteq [n]^{(2)}$ which form an edge cover of $[n]$ – that is, such that $v(B) = [n]$, where $v(B) := \{i,j : \langle i,j\rangle \in B\}$. The factor $3/4$ in EC_n is rather arbitrary. In the proof, it matters that $1/2 < 3/4 < 1$. Note that any n-vertex graph, $n > 1$, has a minimal edge cover of size between $n/2$ and $n-1$, where an edge cover of size $n/2$ is a perfect matching.

Theorem 13. *The families* clique_n, $\mathrm{mclique}_n$, CS_n *and* DS_n *are* VNP *complete over any field.* EC^*_n *and* EC_n *are* VNP*-complete in characteristics equal to two.*

We divide the proof into its constituent parts.

clique_n and $\mathrm{mclique}_n$. This is by reduction to clique^*. Given an edge-weighted graph G on vertices $a_1,\ldots,a_n$, consider the following graph H on $2n$ vertices $a_1,\ldots,a_n,b_1,\ldots,b_n$. H is the union of G, a complete graph on $b_1,\ldots,b_n$, as well as all edges $< a_i, b_j >$ such that $j \neq i$. All edges in $H \setminus G$ have weight one. Every n-clique of H must contain precisely one of the vertices a_i, b_i for every $i \in [n]$, and is of the form $\{a_i : i \in A\} \cup \{b_i : i \in [n]\setminus A\}$, where $\{a_i : i \in A\}$ is a clique in G. This provides a one-to-one correspondence between cliques of G and n-cliques of H, preserving clique-weight. This shows that clique^*_n is a projection of $\mathrm{mclique}_n$ and hence $\{\mathrm{mclique}_n\}$ is VNP-complete. By definition, $\mathrm{mclique}_n$ is a projection of clique_n and hence also $\{\mathrm{clique}_n\}$ is VNP-complete. $\square$

To prove the rest of the theorem, we first note:

Claim. *The family* $\mathrm{clique}^*_n|_{\bar{x}+1} := \sum_{A\subseteq[n]} \prod_{i<j\in A}(1 + x_{i,j})$ *is* VNP*-complete.*

Proof. In general, if $a \in \mathbb{F}$ and $\{f_n\}$ is VNP-complete then so is $\{f_n|_{\bar{x}+a}\}$. Here, $f_{\bar{x}+a}$ denotes the polynomial obtained by substituting $z := z + a$, for every variable z in f. First, if h is a projection of g then $h_{\bar{x}+a}$ is a projection of $g_{\bar{x}+a}$. (For, if $h(x_1,\ldots,x_n) = g(q(y_1),\ldots,q(y_n))$ with $q(y_i) \in \mathbb{F} \cup \{x_1,\ldots,x_n\}$ then $h(x_1 + a,\ldots,x_n + a) = g(q'(y_1) + a,\ldots,q'(y_n) + a)$, where: $q'(y_i) := q(y_i)$, if $q(y_i)$ is a variable, and $q'(y_i) = q(y_i) - a$ if $q(y_i) \in \mathbb{F}$). Second, VNP-completeness of $\{f_n\}$ gives that $\{f_n|_{\bar{x}-a}\}$ is a p-projection of $\{f_n\}$ and so $\{f_n\}$ is a p-projection of $\{f_n|_{\bar{x}+a}\}$. $\square$

CS_n and DS_n. $\mathrm{clique}^*_n|_{\bar{x}+1}$ can be rewritten as

$$\mathrm{clique}^*_n|_{\bar{x}+1} = \sum_{A\subseteq[n]} \prod_{i<j\in A} (1 + x_{i,j}) = \sum_{A\subseteq[n],B\subseteq A^{(2)}} \prod_{\langle i,j\rangle\in B} x_{ij}\,. \tag{7}$$

This is precisely the polynomial obtained by setting $x_1,\ldots,x_n$ to 1 in CS_n or DS_n. $\square$

Edge covers EC_n^*. We work in characteristics two. We can further rewrite (7) as

$$\mathrm{clique}_n^*|_{\bar{x}+1} = \sum_{B\subseteq[n]^{(2)}} \sum_{A^2\supseteq B} \prod_{\langle j,k\rangle\in B} x_{j,k} = c(B) \sum_{B\subseteq[n]^{(2)}} \prod_{\langle j,k\rangle\in B} x_{j,k}\,,$$

where $c(B)$ is the number of sets $A \subseteq [n]$ with $B \subseteq A^{(2)}$. Hence, $c(B) = 2^{n-|v(B)|}$. In characteristics 2, the only non-zero terms are those with $v(B) = [n]$ corresponding to edge covers. □

Edge covers EC_n. This will be by reduction to EC_n^*. Given an edge-weighted graph G on n vertices, it is enough to find an edge-weighted graph H with $m = O(n^2)$ vertices such that the sum of weights of edge-covers of G equals the sum of weights of edge-covers of size $3m/4$ of H.

Given N and k, let $G_{N,k}$ be the following graph on $2N + 2k + 1$ vertices. The vertices are partitioned into sets $\{a\}, A_1, A_2$, and B_1, B_2 with $|A_1| = |A_2| = N$ and $|B_1| = |B_2| = k$. Its $2N + k$ edges consist of all edges between a and A_1, a perfect matching between A_1 and A_2, and a perfect matching between B_1 and B_2. Every edge cover of $G_{N,k}$ must contain the two matchings and at least one edge between a and A_1. Hence, every edge cover has size at least $N+k+1$ and the number of edge covers of size $N+k+r$ is exactly $\binom{N}{r}$ if $0 < r \le N$. Furthermore, if $N = 2^q - 1$ for some $q \in \mathbb{N}$ then $\binom{N}{r}$ is *odd* for every $r \in [N]$.

Let H be the disjoint union of G and $G_{N,k}$, where N is the smallest $N > n(n-1)/2$ of the form $N = 2^q-1$, $q \in \mathbb{N}$. Edges in $G_{N,k}$ are weighted by 1. We claim that, in characteristics 2,

$$\sum_{E \text{ edge cover of } G} w(E) = \sum_{E' \text{ edge cover of } H, |E'|=2N+k} w(E')\,.$$

This is because every edge cover E of G with $|E| = s$ can be extended to exactly $\binom{N}{N-s}$ covers E' of E with $|E'| = 2N + k$ and $E = E' \cap G$. The weight of E' equals the weight of E and $\binom{N}{N-s}$ is odd. The graph H has $v = n + 2N + 2k + 1$ vertices. If we choose $k = N - 3(n + 1)/2$, the sum ranges over E' of size $3v/4$. (Without loss of generality, we assumed that n is odd.) □

This concludes the proof of Theorem 13. We remark that:

Remark 14. *By similar reductions, one can obtain* VNP*-completeness of analogous families defined on bipartite graphs. Namely, polynomials counting bicliques*

$$\sum_{A_1,A_2\subseteq[n]} \prod_{i\in A_1, j\in A_2} x_{i,j}\,, \quad \sum_{A_1\dot\cup A_2=[n]} \prod_{i\in A_1, j\in A_2} x_{i,j}\,,$$

as well as polynomials counting edge covers in a bipartite graph.

6 Defining functions and complexity of decision problems

In this section, we give a different perspective on Theorem 1, and discuss our VNP-complete families in terms of the complexity of their underlying decision problems.

With a boolean function $f : \{0,1\}^n \to \{0,1\}$, we have associated the polynomial $\hat{f}$ which agrees with f on the boolean cube. There is different way how to obtain a multilinear polynomial from f, namely, as the polynomial whose coefficients are computed by f. More generally, if $f : \{0,1\}^n \to \mathbb{F}$, let f^* be the polynomial in variables $x = \{x_1, \ldots, x_n\}$

$$f^* := \sum_{v\in\{0,1\}^n} f(v)x^v\,.$$

Hence, the function f computes the coefficient of x^v in f^*. We will call f the *defining function of* f^*. We can compare this with (2): $\hat{f} = \sum_v x^v x_v f(v)$. The difference between f^* and $\hat{f}$ corresponds to generating function versus probability generating function of [2]. The two polynomials can be quite different. If 1 is the constant function from $\{0,1\}^2$ to $\{0,1\}$ then $\hat{1} = 1$ whereas $1^* = 1 + x_1 + x_2 + x_1x_2$. However, we observe that $\hat{f}$ and f^* are polynomially related.

Proposition 15. *Let s_1 and s_2 be the circuit complexity of f^* and $\hat{f}$, respectively, where $f : \{0,1\}^n \to \mathbb{F}$. Then $s_1 = O(s_2 n^2)$ and $s_2 = O(s_1)n^2$. Hence,* VNP*-hardness results of Theorem 1 and 7 hold for f^* instead of $\hat{f}$.*

Proof. The first equality was proved in Lemma 5, the second one follows similarly from (4). ☐

We believe that this is enough to reproduce the dichotomy results of [1] for both $\hat{f}$ and f^* over fields of arbitrary characteristics.

Defining functions of VNP-complete families We now discuss the defining functions of the families from Section 5. For homogeneous polynomials, we consider slightly more general defining functions. If $f(x)$ is a homogeneous polynomial of degree k, we will call g its *hom. defining function*, if $f(x) = \sum_{|v|=k} g(v)x^v$. We note:

- The defining function of perm_n^* and the hom. defining function of perm_n is an antimonotone 2-CNF. In contrast, the hom. defining function of HC_n is not in AC0.

This is because the defining function of perm_n^* (and the hom. defining function of perm_n) checks whether a bipartite graph is a partial matching. This can be expressed as an antimonotone 2-CNF as in Section 3.1. For HC_n, the homogeneous defining function decides, given a graph with n edges and n vertices, whether it is a cycle (cf. [12]). For the polynomials in Section 5, we note the following:

(i). The defining function of $(\mathrm{clique}_n^* - n)$, DS_n and EC_n^* is a 3-CNF, antimonotone 2-CNF and a monotone CNF of polynomial size, respectively.

(ii). The hom. defining function of clique_n, $\mathrm{mclique}_n$ and EC_n is a 3-CNF, antimonotone 2-CNF and a monotone CNF of polynomial size, respectively.

Underlying decision problems of VNP-complete families Let $\{f_n\}$ be a family of multilinear polynomials with 0,1-coefficients such that f_n is in m_n variables. With $\{f_n\}$, we associate the following decision problem:

Given $n \in \mathbb{N}$, $v \in \{0,1\}^{m_n}$, and $k \leq m_n$, decide whether there exists $u \in \{0,1\}^{m_n}$ such that[3] *$u \leq v$, $|u| = k$ and x^u has coefficient equal to 1 in f_n.*

In characteristics zero, this is equivalent to checking whether $f^{(k)}(v) \neq 0$, where $f^{(k)}$ is the k-homogeneous part of f. For a family consisting of homogeneous polynomials, the parameter k can be dropped. For example, the decision problem associated with perm_n^* consists in checking whether a bipartite graph has a matching of size k, and a perfect matching in the case of perm_n. Hence, we note:

- The decision problem associated with perm_n or perm_n^* is in P. For HC_n, it is NP-hard.

As for the polynomials in Section 5, we note

Proposition 16. *The decision problem associated with* $(\mathrm{clique}_n^* - n)$ *or* clique_n *is NP-hard. For the other families in Theorem 13, the decision problem is in P.*

Proof. The first part follows from NP-hardness of deciding whether a $2n$-vertex graph has an n-clique. For $\mathrm{mclique}_n$, the statement is given by Observation 12. EC_n and EC_n^* follow from the fact that a smallest edge cover can be found in polynomial time. The decision problem associated with CS_n amounts to the following: given a graph $G = (V, E)$ and $k \in \mathbb{N}$, decide whether there exists a subgraph $G' = (V', E')$ with $|V'| + |E'| = k$. Such a subgraph exists if and only if $k \leq |V| + |E|$: if $k \leq |V|$ we can remove all but $k - |V|$ edges to achieve $|V| + |E'| = k$. If $k < |V|$, remove all edges and all but k vertices. DS_n is similar. ☐

[3] $u \leq v$ means $u_i \leq v_i$ for every $i \in [m_n]$

Acknowledgement We thank Anup Rao for triggering this investigation and Amir Yehudayoff for useful discussions.

References

[1] I. Briquel and P. Koiran. A dichotomy theorem for polynomial evaluation. In *Mathematical Foundations of Computer Science*, 2009.

[2] P. Bürgisser. *Completeness and reduction in algebraic complexity theory*, volume 7 of *Algorithms and Computation in Mathematics*. Springer, 2000.

[3] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*, volume 315 of *A series of comprehensive studies in mathematics*. Springer, 1997.

[4] P. Hrubeš and A. Yehudayoff. Arithmetic complexity in ring extensions. *Theory of Computing*, 7:119–129, 2011.

[5] M. Jerrum. *On the complexity of evaluating multivariate polynomials*. PhD thesis, Dept. of Computer Science, University of Edinburgh, 1981.

[6] A. Juma, V. Kabanets, C. Rackoff, and A. Shpilka. The black-box query complexity of polynomial summation. *Comput. Complex.*, 18(1):59–79, 2009.

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

[8] V. Strassen. Vermeidung von Divisionen. *J. of Reine Angew. Math.*, 264:182–202, 1973.

[9] L. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *Siam J. Comp.*, 12:641–644, 1983.

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

[11] L. G. Valiant. Accidental algorithms. In *Proceedings of the 47th Annual IEEE Symposium Foundations of Computer Science*, pages 509–517, 2006.

[12] A. Wigderson. The complexity of graph connectivity. In *Proceedings of the 17th International Symposium on Mathematical Foundations of Computer Science*, pages 112–132, 1992.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de