THE COMPLEXITY OF POLYNOMIALS AND THEIR COEFFICIENT FUNCTIONS

G. MALOD

Abstract. We study the link between the complexity of a polynomial and that of its coefficient functions. Valiant's theory is a good setting for this, and we start by generalizing one of Valiant's observations, showing that the class VNP is stable for coefficients functions, and that this is true of the class VP iff VP = VNP, an eventuality which would be as surprising as the equality of the classes P and NP in the Boolean case. We extend the definition of Valiant's classes to polynomials of unbounded degree, thus defining the classes $\mathrm{VP}_{\mathrm{nb}}$ and $\mathrm{VNP}_{\mathrm{nb}}$. Over rings of positive characteristic the same kind of results hold in this case, and we also prove that VP = VNP iff $\mathrm{VP}_{\mathrm{nb}} = \mathrm{VNP}_{\mathrm{nb}}$. Finally, we use our extension of Valiant's results to show that iterated partial derivatives can be efficiently computed iff VP = VNP. This is also true for the case of polynomials of unbounded degree, if the characteristic of the ring is positive.

1. Introduction

Let us start with a small exercise. Consider a function $f(x_1, \ldots, x_n)$ from $\{0,1\}^n$ to $\{0,1\}$. Define $f^*(y_1, \ldots, y_n) = \sum_{\bar{\epsilon} \in \{0,1\}^n} f(\bar{\epsilon})\ \bar{y}^{\bar{\epsilon}}$. In a way, f is the "coefficient function" of f^*. Show that $f^{**} = f$, i.e. that the $*$-operation is an involution. This means that the transformation from a function to its coefficient function and the transformation from a coefficient function to the function both have the same complexity (we are being deliberately vague here). This simple observation on Boolean functions will hold in a way for sequences of polynomials in a class called VNP, although the setting and techniques used will be different. More generally, we wish to study the link between the complexity of a polynomial and that of its coefficient function.

What is the class VNP mentioned above? It is a class defined by Valiant [Val79, Val82] which studies the complexity of polynomial sequences defined by arithmetic circuits. This theory is interesting for several reasons. As a theory of algebraic computations, it predates the famous Blum-Shub-Smale model of real Turing machines (cf. [BSS89]), and already states a "P vs NP" type of question, related to the complexity of the Permanent. It is also a good framework to study the representation of polynomials by arithmetic circuits. These representations can be quite compact (the Determinant polynomial for instance has a factorial number of monomials but a polynomial size circuit representation). Results by Kaltofen [Kal86] and von zur Gathen [vzG87] study the symbolic manipulations which can be applied to arithmetic circuits without increasing their size too much, and thus the feasibility of such a representation scheme. Valiant's algebraic complexity classes appear in this context as the relevant formalization of intractability, as our result on iterated partial derivatives illustrates. Arithmetic circuits are also linked to Boolean complexity. Circuits are often used for alternative characterizations of important classical complexity

Date: December 4, 2006.

1991 *Mathematics Subject Classification.* 03D15,68Q17.

Key words and phrases. algebraic complexity, Valiant's theory, polynomial, straight line program, circuit, coefficients, partial derivative.

classes such as P, NP and ♯P (cf. [Ven92]), or to define new classes (cf. [All04]). These characterizations can provide new insights and proofs. Valiant's theory can be seen as a general theory of arithmetic or Boolean circuits. This point of view is presented more extensively in a previous work [MP06].

We start by giving a short presentation of Valiant's theory in section 2, where we give the definitions of the classes VP and VNP, and extend them to the case of polynomial sequences of unbounded degree, defining the classes $\mathrm{VP}_{\mathrm{nb}}$ and $\mathrm{VNP}_{\mathrm{nb}}$. In section 3 we define more precisely what we mean by a coefficient function. Valiant himself had noticed in [Val82] that there is a link between the complexity of coefficient functions and his classes. Theorems 1 and 2 in section 4 can thus be seen as a generalization of his idea: we show that the class VNP is stable for coefficient functions and that the class VP cannot be stable unless VP = VNP. In section 5 we prove similar results (theorems 4 and 5) for sequences of polynomials of unbounded degree. As a surprising byproduct of the proof technique, we show that the unbounded classes $\mathrm{VP}_{\mathrm{nb}}$ and $\mathrm{VNP}_{\mathrm{nb}}$ are equal if and only if VP and VNP are equal (theorem 6). However, these last three theorems hold only over rings of positive characteristic, an assumption we use to compute efficiently binomial coefficients. Finally, we show in section 9 that the generelization of Valiant's results is useful to characterize the possibility of efficiently computing iterated partial derivatives: this possibility is also equivalent to the VP = VNP (theorem 7). Once again the same characterization holds for the unbounded class, but only over rings of positive characteristic.

2. Valiant's theory

We give here a brief introduction to Valiant's theory. Detailed information can be found in [vzG87, Bür00, MP06]. Valiant's algebraic classes revolve around the representation of polynomials over a given ring by arithmetic circuits. These polynomials are abstract, in the sense that they are defined by the sequence of their coefficients. One should remember to distinguish polynomials in this sense from polynomial functions, which are the functions defined by polynomials over a ring.

Definition 1. An *arithmetic circuit* over a ring R is a finite directed acyclic graph with vertices of in-degree 0 or 2 and exactly one vertex of out-degree 0. Vertices of in-degree 0 are called *inputs* and labeled by an element from R or a variable. The other vertices, of in-degree 2, are labeled by $\times$ or $+$ and called *computation gates*. The vertex of out-degree 0 is called the *output*.

Definition 2. The *size* of a circuit is its number of gates. The *depth* is the maximal length of a directed path from an input to an output. The *degree* of a gate is defined recursively: any input is of degree 1; the degree of a $+$ gate is the max of the incoming degrees; the degree of a $\times$ gate is the sum of the incoming degrees. The degree of the circuit is the degree of its output gate.

The polynomial computed by a circuit can easily be defined by induction. Note that the degree of the circuit mentioned here counts all the inputs, whether variable or constant. This makes the definition of the class VP below different from the one given in [Bür00]. They are in fact equivalent (cf. [MP06]).

As usual in complexity theory we are interested in asymptotics, in this case the growth of the size of the circuits representing a sequence of polynomials. We give here the definitions of Valiant's classes and the reductions used. Note that the classes depend on a chosen ring but, as we will mostly use combinatorial techniques, only the characteristic of the ring will matter for us.

The main classes of Valiant's theory are VP and VNP. The class VP represents polynomial sequences that are "easy to compute", in the sense that they have small circuits, here with the added requirement that the degree also be kept small. Although it is convenient to think of this class as an algebraic analog to the class P, this last condition makes it more similar to LOGCFL. The class VNP is defined from the class VP by allowing a sum over the values of a small circuit when a subset of its variables are substituted by Boolean words. This is the "hard to compute" class, although once again its name is a little misleading: being defined via a sum, it is much closer to the class ♯P (which can be roughly defined as counting the number of solutions of a relation in P) than it is to NP.

Definition 3. A sequence of polynomials (f_n) belongs to VP if there exists a sequence of circuits C_n of polynomially bounded size and degree such that C_n represents f_n.
A sequence of polynomials (f_n) belongs to VNP if there exists a polynomial p and a sequence $(g_n) \in \mathrm{VP}$ such that, for all n:

$$f_n(\bar{x}) = \sum_{\bar{\epsilon} \in \{0,1\}^{p(|\bar{x}|)}} g_n(\bar{x}, \bar{\epsilon}).$$

It is obvious that VP is included in VNP. Valiant's hypothesis is that this inclusion is strict; it remains a major open problem of complexity theory, which is related to similar questions in Boolean complexity (cf. [Bür00]). Following the familiar blueprint of complexity theory, we define notions of reduction and completeness.

Definition 4. A polynomial f is a *projection* of a polynomial g if $f(\bar{x}) = g(a_1, \ldots, a_m)$, where the a_i are elements of the ring or variables among $x_1, \ldots, x_n$. A sequence (f_n) is a *p-projection* of a sequence (g_n) if there exists a polynomially bounded function $t(n)$ such that f_n is a projection of $g_{t(n)}$ for all n. A sequence f is complete for a class $\mathcal{C}$ if it belongs to $\mathcal{C}$ and if any sequence g in $\mathcal{C}$ is a p-projection of f.

The main result in Valiant's theory is the completeness of the Permanent sequence of polynomials for the class VNP, over rings of characteristic different from 2. The permanent of a matrix of size n with variables entries $z_{i,j}$ is defined as $\mathrm{PER}_n(z_{i,j}) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} z_{i,\sigma(i)}$. In this definition, S_n is the group of permutations of $\{1, \ldots, n\}$. This result stands in stark contrast to the fact that the Determinant sequence belongs to the class $\mathrm{VP}_{\mathrm{ws}}$, a subclass of VP. The determinant of a matrix is defined as the permanent but with positive and negative monomials depending on the sign $s(\sigma)$ of the permutation: $\mathrm{DET}_n(z_{i,j}) = \sum_{\sigma \in S_n} s(\sigma) \prod_{i=1}^{n} z_{i,\sigma(i)}$. Note that there is no such known natural complete problem for the class VP.

We now define two new classes $\mathrm{VP}_{\mathrm{nb}}$ and $\mathrm{VNP}_{\mathrm{nb}}$, which are similar to the classes VP and VNP, except that there is no bound on the degree.

Definition 5. A sequence of polynomials (f_n) belongs to $\mathrm{VP}_{\mathrm{nb}}$ if there exists a sequence of circuits C_n of polynomially bounded size such that C_n represents f_n.
A sequence of polynomials (f_n) belongs to $\mathrm{VNP}_{\mathrm{nb}}$ if there exists a polynomial p and a sequence $(g_n) \in \mathrm{VP}_{\mathrm{nb}}$ such that, for all n:

$$f_n(\bar{x}) = \sum_{\bar{\epsilon} \in \{0,1\}^{p(|\bar{x}|)}} g_n(\bar{x}, \bar{\epsilon}).$$

3. The complexity of polynomials and their coefficients

In this section we define the problem we are interested in: the links between the complexity of a polynomial and that of its coefficient function.

Let $f(x_1, \ldots, x_k)$ be a polynomial of degree d. We can represent the power of each variable x_i in a monomial[1] of f by a tuple $\bar{\epsilon}_i$ of length logarithmic in d. We can then write f as the sum $\sum_{\bar{\epsilon}_1,\ldots,\bar{\epsilon}_k} g(\bar{\epsilon}_1, \ldots, \bar{\epsilon}_k) x_1^{e_1} \cdots x_k^{e_k}$, where e_i is the number represented by $\bar{\epsilon}_i$. In this expression, g is the *total coefficient function* of f. Given a tuple $\bar{\epsilon}_1, \ldots, \bar{\epsilon}_k$ representing a monomial of f it produces the value of its coefficient. We will often write the above expression more succinctly as $f(\bar{x}) = \sum_{\bar{\epsilon}} g(\bar{\epsilon}) \bar{x}^{\bar{\epsilon}}$.

Saying that g is *the* coefficient function of f is a bit imprecise, as there could be other coefficient functions depending on the encoding of the monomials. In the case of sequences of polynomials (f_n), as in the classes introduced above, we know that the degree of f_n grows simply exponentially. We will then suppose that we have a bound $2^{p(n)}$ on this degree, and encode a monomial in the variables $x_1, \ldots, x_{q(n)}$ by the tuple $(\bar{\epsilon}_1, \ldots, \bar{\epsilon}_q(n))$, where each tuple $\bar{\epsilon}_i$ is of length $p(n)$. We will then call the sequence (g_n) defined as above as *the total coefficient function* of (f_n). For all n we can write $f_n(\bar{x}) = \sum_{\bar{\epsilon}} g_n(\bar{\epsilon}) \bar{x}^{\bar{\epsilon}}$.

We can also define *partial coefficient functions* by considering a subset of the variables. This is straightforward, and will enable us to write each polynomial f_n of a sequence as $f_n(\bar{x}, \bar{y}) = \sum_{\bar{\epsilon}} g_n(\bar{y}, \bar{\epsilon}) \cdot \bar{x}^{\bar{\epsilon}}$, where (g_n) is the partial coefficient function of (f_n) for the variables $\bar{y}$.

Our main endeavor is to investigate the links between the circuit complexity of a polynomial and that of its coefficient functions. But coefficient functions are not polynomials. In the general case they can be seen as functions $f(\bar{x}, \bar{\epsilon})$ which map a Boolean tuple $\bar{\epsilon}$ to a polynomial in the variables $\bar{x}$. However, any such function can be represented by a polynomial in the variables $\bar{\epsilon}$ and $\bar{x}$. This representation is of course not necessarily unique. We will therefore define the complexity of a function $f(\bar{x}, \bar{\epsilon})$ to be the smallest complexity of a polynomial which coincides with the f for any projection of the tuple $\bar{\epsilon}$ to Boolean values.

When we say that g is *a coefficient function* of f, we mean that there exists a choice of variables of f for which g is *the coefficient function* of f. This extends of course to sequences of polynomials. We say that a class $\mathcal{C}$ is *stable for coefficient functions* if:

- for any sequence f in $\mathcal{C}$, any coefficient function of f belongs to $\mathcal{C}$.
- for any sequence f, if there exists a coefficient function of f which belongs to $\mathcal{C}$, then f belongs to $\mathcal{C}$.

4. Polynomials of polynomially bounded degree

The Permanent sequence belongs to the class VNP. Moreover it has two striking properties, which will give us a first insight into the relation between the complexity of a polynomial and the complexity of its coefficient functions, when the degree of the polynomial is polynomially bounded.

The first property of the Permanent is that its total coefficient function belongs to the class VP. It is the function g_n defined over the variables $\epsilon_{i,j}$ (with $1 \leq i, j \leq n$):

$$g(\bar{\epsilon}) = \left(\prod_{\substack{1 \leq i,j,k,l \leq n \\ i=k \text{ iff } j \neq l}} (1 - \epsilon_{i,j}\epsilon_{k,l}) \right) \cdot \left(\prod_{i=1}^{n} \sum_{j=1}^{n} \epsilon_{i,j} \right).$$

One can see that $g_n(\bar{\epsilon})$ has value 1 if $(\epsilon_{i,j})$ is a permutation matrix (i.e. a matrix whose entries are 0 except for exactly one 1 in each row and each column) and it has value 0

[1]We will always use *monomial* to mean a product of powers of variables which appear in the development, *without* its coefficient.

otherwise. The Permanent can then be written as:

$$\mathrm{PER}(z_{i,j}) = \sum_{\bar{\epsilon}\in\{0,1\}^{n^2}} g(\bar{\epsilon})\ \bar{z}^{\bar{\epsilon}}.$$

Moreover, the permanent is the coefficient of the monomial $y_1 \cdots y_n$ in the development of the following polynomial (according to [vzG87], this observation goes back to the 19$^{\text{th}}$ century [Ham79]):

$$f_n(\bar{y}, \bar{z}) = \prod_{i=1}^{n} \left(\sum_{j=1}^{n} z_{i,j} y_j \right).$$

This is true because when one develops the above product to get a term in $y_1 \cdots y_n$, one must choose one term of the sum for each factor, so that in the end all the y_i appear. One must therefore associate a j to each i in a surjective way, thus one must choose a permutation of $\{1, \ldots, n\}$. Clearly the sequence (f_n) belongs to the class VP. As we have mentioned in the introduction, this had been also noticed by Valiant [Val82], who linked the complexity of a sequence (f_n) of polynomials with a sequence obtained by taking the coefficient of one monomial in a partial development of each f_n. We wish to extend this observation to the full coefficient function.

The first observation suggests that the class VNP is a good class for our endeavor. If the coefficient function of a polynomial sequence belongs to the class VNP, one can easily see from the definition that the polynomial sequence also belongs to the class VNP. The fact that the total coefficient function of the permanent is VP, coupled with the VNP-completeness of the permanent, would enable us to write the coefficient function of a polynomial sequence in VNP as a sum of projections of the coefficient function of the Permanent, thus showing that it is also in VNP. The class VNP is therefore stable, when going from a polynomial to one of its coefficient function or the other way around. Now, if one could show a similar result for the class VP, the fact that the sequence (f_n) of the second property is in VP would imply that its coefficient function also belongs to VP, and then taking the right coefficient would show that the permanent belongs to VP, and thus that VP = VNP.

We will follow the outline given above, but because we would like our results to hold independently of the field characteristic, we shall use the Hamiltonian instead of the permanent. Proving the same kind of results with the Hamiltonian adds slight technical difficulties, which we solve in the following lemmas.

4.1. **Properties of the Hamiltonian.**

Lemma 1. *The total coefficient function of* HC_n *belongs to the class* VP.

Proof. We wish to write the Hamiltonian in the following form:

$$\mathrm{HC}_n(z_{i,j}) = \sum_{\bar{\epsilon}\in\{0,1\}^{n^2}} h_n(\bar{\epsilon})\bar{z}^{\bar{\epsilon}}.$$

The function h_n starts by checking that $(\epsilon_{i,j})$ is a permutation matrix in the same way as the function g_n did for the permanent. It then checks that the given permutation is a cycle of length n by computing the successive images of the integers $1, \ldots, n$ under the iteration of the permutation. Consider the successive powers of the matrix $E = (\epsilon_{i,j})$. The function h_n must check that:

- for j ranging from 1 to $n-1$, all the coefficients on the diagonal of E^j are equal to 0.

- all the coefficients on the diagonal of E^n are equal to 1.

One can easily represent the sequence of functions (h_n) by a sequence of polynomials of bounded degree. □

Lemma 2. *There exists a polynomial sequence (t_n) belonging to the class* VP *such that, for all n, the polynomial* HC_n *is the coefficient of a monomial from t_n.*

Proof. We begin by defining a triple-indexed family of polynomials $(T_{p,q,r})$ with $p \in \mathbb{N}$ and $1 \leq q, r \leq n$. These polynomials are defined over the variables $x_{i,j}$, y_k and z_l, with $1 \leq i, j, k, l \leq n$ by the following rules:

- $T_{1,i,j} = x_{i,j} y_i z_j$.
- $T_{p+1,i,j} = \sum_{k=1}^{n} T_{p,i,k} \cdot T_{1,k,j}$.

One can then show by an easy induction on p that:

$$T_{p,i,j} = \sum_{1 \leq m_1, \ldots, m_{p-1} \leq n} x_{i,m_1} \left(\prod_{i=1}^{p-2} x_{m_i, m_{i+1}} \right) x_{m_{p-1},j} \left(\prod_{i=1}^{p-1} y_{m_i} z_{m_i} \right) y_i \, z_j,$$

and that there exists a circuit of size $3n^2 + n + (p-1)(2n^3 - n^2)$ computing all the $T_{k,i,j}$ for $1 \leq i, j \leq n$ and $1 \leq k \leq p$.

Let $t_n = T_{n,1,1}$. The polynomial sequence (t_n) does belong to the class VP (it is computable by a circuit of polynomial size and it is easy to check that its degree is polynomial). Moreover we can write:

$$T_{n,1,1} = \sum_{1 \leq m_1, \ldots, m_{n-1} \leq n} x_{1,m_1} x_{m_1,m_2} \cdots x_{m_{n-2},m_{n-1}} x_{m_{n-1},1} \left(\prod_{i=1}^{n-1} y_{m_i} z_{m_i} \right) y_1 \, z_1,$$

so that the coefficient of the monomial $y_1 z_1 \cdots y_n z_n$ is the Hamiltonian of size n. Indeed, one must choose pairwise distinct integers $m_1, \ldots, m_{n-1}$ between 2 and n. The associated variables $x_{i,j}$ will correspond to a cycle of length n yielding the monomial $x_{1,m_1} x_{m_1,m_2} \cdots x_{m_{n-2},m_{n-1}} x_{m_{n-1},1}$. □

4.2. **Stability of the classes** VP **and** VNP**.**

Theorem 1. *Let f be a sequence of polynomials. The following properties are equivalent:*

1. *f belongs to the class* VNP.
2. *any coefficient function of f belongs to the class* VNP.
3. *there exists a coefficient function of f which belongs to the class* VNP.

Proof. (1) ⇒ (2). We start by considering just one polynomial $f(x_1, \ldots, x_m, y_1, \ldots, y_n)$ of degree d. Suppose that f is the image of the Hamiltonian polynomial HC_q under a projection σ. Recall the expression of HC_q by its total coefficient function:

$$\mathrm{HC}_q(\bar{z}) = \sum_{\bar{\epsilon}} h_q(\bar{\epsilon}) \prod_{i,j} z_{i,j}^{\epsilon_{i,j}}.$$

The simple idea here is to identify, for a given monomial of f, all the monomials of HC_q which will be projected unto it via σ. We represent a monomial in the variables $y_1, \ldots, y_n$ by the tuple $(\bar{\eta}_1, \ldots, \bar{\eta}_n)$. Let us first build a circuit which computes the polynomial $u(\bar{\epsilon}, \bar{\eta})$, whose value is 1 if and only if $\bar{\eta}$ is the monomial in the variables $\bar{y}$ resulting from applying the projection σ to the monomial $\bar{z}^{\bar{\epsilon}}$. For $1 \leq i, j \leq q$, if $z_{i,j}$ is projected unto a variable in $\bar{y}$, we put the tuple $(\bar{\alpha}_1, \ldots, \bar{\alpha}_n)$, all of whose elements are 0, except for $\alpha_{k,0} = \epsilon_{i,j}$. We then take the sum of all these tuples in the following way: the sum of $(\bar{\alpha}_1, \ldots, \bar{\alpha}_n)$ and $(\bar{\beta}_1, \ldots, \bar{\beta}_n)$ is the tuple $(\bar{\gamma}_1, \ldots, \bar{\gamma}_n)$ such that $\bar{\gamma}_i$ is the binary representation of the sum

of the numbers represented by $\bar{\alpha}_i$ and $\bar{\beta}_i$. This circuit computes the powers $\bar{\alpha}_1, \ldots, \bar{\alpha}_n$ of the variables $y_1, \ldots, y_n$ in the image of $\bar{z}^{\bar{\epsilon}}$ by the projection σ. The circuit computing $u(\bar{\epsilon}, \bar{\eta})$ ends with an equality test of $\bar{\alpha}$ and $\bar{\eta}$. This computation can be done by iterated additions of integers, and thus in logarithmic depth (cf. [Vol99]), which guarantees that the degree stays polynomial. For all i and j, define now a polynomial $c_{i,j}$:

- if $z_{i,j}$ is mapped to a variable y_k, $c_{i,j}$ is the constant polynomial 1.
- if $z_{i,j}$ is mapped to a constant a, $c_{i,j}$ is the polynomial $\epsilon_{i,j}\, a \ + 1 - \epsilon_{i,j}$
- if $z_{i,j}$ is mapped to a variable x_k, $c_{i,j}$ is the polynomial $\epsilon_{i,j}\, x_k \ + 1 - \epsilon_{i,j}$.

The product of the polynomials $c_{i,j}$ is the polynomial $v(\bar{\epsilon}, \bar{x})$ whose value is the coefficient of the monomial in the variables $\bar{y}$ resulting from the application of the projection σ to $\bar{z}^{\bar{\epsilon}}$. It is easy to check that the size and degree of all these circuits is polynomial in q. The coefficient function $g(\bar{x}, \bar{\eta})$ of f for the variables $\bar{y}$ is then:

$$g(\bar{x}, \bar{\eta}) = \sum_{\bar{\epsilon} \in \{0,1\}^{q^2}} h_q(\bar{\epsilon})\ u(\bar{\epsilon}, \bar{\eta})\ v(\bar{\epsilon}, \bar{x}).$$

Suppose now that $g = (g_n)$ is a coefficient function of a sequence $f = (f_n)$ belonging to VNP. We know that f is then a p-projection of the Hamiltonian and we can therefore do the above construction for each g_n, thus showing that g also belongs to VNP.

(2) $\Rightarrow$ (3). Obvious.

(3) $\Rightarrow$ (1). Assume $g = (g_n)$ is the coefficient function of $f = (f_n)$ for the variables $\bar{y}$: $f_n(\bar{x}, \bar{y}) = \sum_{\bar{\epsilon}} g_n(\bar{x}, \bar{\epsilon}) \bar{y}^{\bar{\epsilon}}$. If g belongs to VNP, there exists a sequence of polynomials (h_n) in VP such that $g_n(\bar{x}, \bar{z}) = \sum_{\bar{\eta}} h_n(\bar{x}, \bar{z}, \bar{\eta})$. We can then express $f_n(\bar{x}, \bar{y})$ as $\sum_{\bar{\epsilon}, \bar{\eta}} h_n(\bar{x}, \bar{\epsilon}, \bar{\eta}) \bar{y}^{\bar{\epsilon}}$, thus showing that (f_n) belongs to VNP, since the monomial $\bar{y}^{\bar{\epsilon}}$ is computable by an arithmetic circuit of polynomial size and degree. □

Theorem 2. *The following properties are equivalent:*

(1) VP *is stable for coefficient functions.*
(2) VP = VNP.

Proof. (1) $\Rightarrow$ (2). The sequence (t_n) from lemma 2 belongs to VP. Therefore if VP is stable for coefficient functions, then the coefficient function of (t_n) for the variables $(\bar{y}, \bar{z})$ also belongs to VP. Thus the polynomial HC_n, which is the coefficient of the monomial $y_1 z_1 \cdots y_n z_n$, is computable by a circuit of polynomial size and degree. By completeness of the Hamiltonian, VP= VNP.

(2) $\Rightarrow$ (1). There are two parts to show that VP is stable for coefficient functions. Suppose first that a sequence f belongs to VP $\subseteq$ VNP. By theorem 1, any coefficient function of f also belongs to VNP and, assuming VP = VNP, to VP. This proves one direction of the stability property. Suppose now that a coefficient function of f belongs to VP. Then f belongs to VNP and, assuming VP = VNP, also to VP. Thus VP is stable for coefficients functions. □

5. Polynomials of unbounded degree

The results from the previous section depend on the existence of a VNP-complete sequence of polynomials with specific properties. In the case of the classes of unbounded degree, we do not know of any natural complete sequence of polynomials. We will use techniques inspired by [Bür00] (section 5.6) to define a generic complete sequence and later prove the necessary properties.

5.1. **A $\mathrm{VP}_{\mathrm{nb}}$-complete sequence.** We wish to build a family $G_l^m(\bar{a}, \bar{b}, \bar{y})$ which, given appropriate inputs for $\bar{a}$, $\bar{b}$ and $\bar{y}$, can simulate any computation of length smaller than l over m inputs. The definition is by induction:

- $G_{-m}^m = 1, G_{-m+1}^m = y_1, \ldots, G_0^m = y_m$.
- for $l \geq 1$,

$$G_l^m = \left(\sum_{i=-m}^{l-1} a_{l,i} G_i^m \right) \cdot \left(\sum_{i=-m}^{l-1} b_{l,i} G_i^m \right).$$

Lemma 3. *The sequence $(G_n^n(\bar{a}, \bar{b}, \bar{y}))$ is complete for $\mathrm{VP}_{\mathrm{nb}}$.*

Proof. We must first check that (G_n^n) belongs to $\mathrm{VP}_{\mathrm{nb}}$. It is easy to show by induction that all the polynomials G_l^m can be simultaneously computed by a circuit of size $4lm + 2l^2 - 3l + m$. Thus G_n^n can be computed by a circuit of polynomial size.

We will now show that any sequence in $\mathrm{VP}_{\mathrm{nb}}$ is a p-projection of (G_n^n). We start by considering a single polynomial f computed by an arithmetic circuit. It is easier in this case to view the circuit as a *straight line program* (SLP). This is done by taking an ordering of the gates of the circuit which is compatible with the computation order. The SLP is then a sequence $C = (c_1, \ldots, c_t)$ of instructions of the form $c_i = (\sigma_i, u_i, v_i)$. The operation σ_i is an addition or a multiplication. The integer indexes u_i and v_i denote either one of the n inputs (if they take a value between $-n+1$ and 0) or the result of a previous instruction (if they take a value between 1 and $i-1$).

Let k be a number greater than both n and t. Then we can show that f is a projection of G_k^k, by specializing the generic computation expressed by G_k^k. For i ranging from $-n+1$ to 0, we project the variable y_i of G_k^k to the input of C of index i. The values affected to the variables $a_{i,j}$ and $b_{i,j}$ will depend on the computation c_i:

- if $\sigma_i = +$ and $u_i \neq v_i$, a_{i,u_i}, a_{i,v_i} $b_{i,-k}$ are mapped to 1, while the other variables $a_{i,j}$ and $b_{i,j}$ are mapped to 0.
- if $\sigma_i = +$ and $u_i = v_i$, a_{i,u_i} is mapped to 2, $b_{i,-k}$ is mapped to 1 and the other variables $a_{i,j}$ and $b_{i,j}$ to 0.
- if $\sigma_i = *$, a_{i,u_i} and b_{i,v_i} are mapped to 1, the other variables $a_{i,j}$ and $b_{i,j}$ to 0.

For $t < i \leq k$, $a_{i,t}$ and $b_{i,-k}$ are mapped to 1 and the other variables to 0.

The construction above can be done for each f_n of a sequence (f_n) belonging to $\mathrm{VP}_{\mathrm{nb}}$, showing that it is a p-projection of (G_n^n) and completing the completeness proof. □

5.2. **A $\mathrm{VNP}_{\mathrm{nb}}$-complete sequence.** Define the polynomial D_n as:

$$D_n = \sum_{p=0}^{n} \left(c_p \sum_{\bar{\epsilon} \in \{0,1\}^{n-p}} G_n^n(\bar{a}, \bar{b}, y_1, \ldots, y_p, \epsilon_1, \ldots, \epsilon_{n-p}) \right).$$

Lemma 4. *The sequence (D_n) is complete for $\mathrm{VNP}_{\mathrm{nb}}$.*

Proof. Let us start by checking that the sequence (D_n) belongs to $\mathrm{VNP}_{\mathrm{nb}}$. Define the polynomial H_n as:

$$H_n(\bar{a}, \bar{b}, y_1, \ldots, y_n, e_1, \ldots, e_n) = \sum_{p=0}^{n} c_p e_1 \cdots e_p G_n^n(\bar{a}, \bar{b}, y_1, \ldots, y_p, e_{p+1}, \ldots, e_n).$$

It is obvious that (H_n) belongs to $\mathrm{VP}_{\mathrm{nb}}$. Observing that $D_n = \sum_{\bar{\epsilon}} H_n(\bar{a}, \bar{b}, y_1, \ldots, \bar{y}, \bar{\epsilon})$ shows that (D_n) does indeed belong to $\mathrm{VNP}_{\mathrm{nb}}$.

Consider a polynomial $f(\bar{z})$ which can be written as a Valiant sum $\sum_{\bar{\alpha}} g(\bar{z}, \bar{\alpha})$. We know that g is a projection of G_k^k if k is greater than the circuit complexity of g. We can

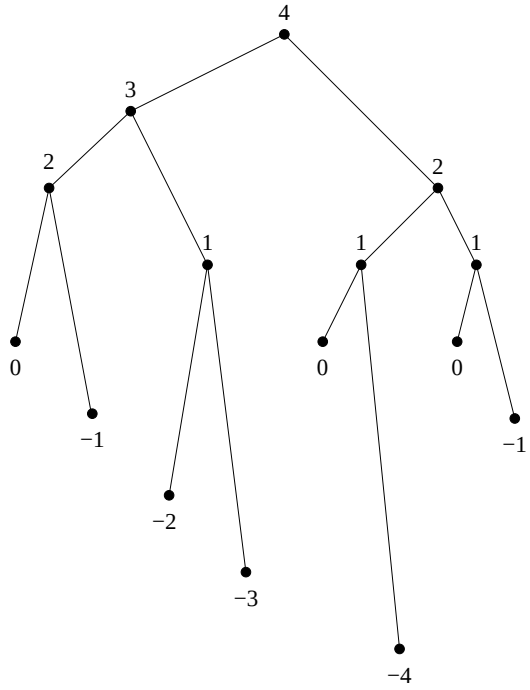

FIGURE 1. A tree in the non-commutative development of G_4^4

then write $g(\bar{z}, \bar{\alpha})$ as $G_k^k(\bar{u}, \bar{v}, \bar{0}, \bar{z}, \bar{\alpha})$, by projecting the variables $\bar{a}$ and $\bar{b}$ of G_k^k to some appropriate tuples $\bar{u}$ and $\bar{v}$, and the variables $\bar{y}$ to the tuples $\bar{0}$ (which is used to fill up extraneous variables), $\bar{z}$ and $\bar{\alpha}$. Call q the sum of the length of the tuples $\bar{0}$ and $\bar{z}$. Then the length of the tuple $\bar{\alpha}$ is $k - q$. Using the projections defined above and mapping all variables c_p to 0 except c_q we can show that f is a projection of D_k. It is straightforward to apply this construction to a sequence (f_n) in $\mathrm{VNP}_{\mathrm{nb}}$ to finish the completeness proof. □

That was the easy part. We now need to show that (D_n) has properties similar to those of the Hamiltonian. We start by studying the total coefficient function of the polynomials G_n^n and D_n.

5.3. The total coefficient function of G_n^n.

5.3.1. *Monomials in the non-commutative development.* Let us develop G_n^n, without using the commutativity of the product.

$$G_n^n = \left(\sum_{i=-n}^{n-1} a_{n,i} G_i \right) \cdot \left(\sum_{i=-n}^{n-1} b_{n,i} G_i \right).$$

We must choose a term in the left sum and one in the right sum. Each term will be a G_l^m for some value of l, in which case we repeat the process, or a variable y_i, in which case we stop. We thus identify a monomial from the non-commutative development with a binary tree with nodes labeled by integers in the range $(-n, n)$, such that the label of a node is always strictly greater than that of its sons, and a node is a leaf iff its label is less than or equal to 0. Figure 1 gives an example of such a tree.

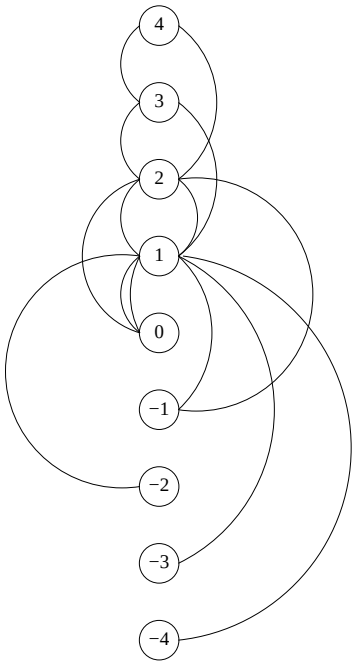

FIGURE 2. The graph corresponding to the tree of Figure 1

5.3.2. *Monomials in the commutative development.* The monomial associated to a tree is obtained in the following manner: the power of a variable $a_{i,j}$ (resp. $b_{i,j}$) is the number of left (resp. right) edges going from a node labeled i to a node labeled j. The power of a variable y_i is the number of leaves labeled $-n+i$. In the case of the tree in Figure 1, the monomial is:

$$a_{4,3}a_{3,2}a_{2,1}a_{2,0}a_{1,0}^2a_{1,-2}b_{4,2}b_{2,1}b_{2,-1}b_{1,0}b_{1,-1}b_{1,-3}b_{1,-3}b_{1,-4}y_1y_2y_3^2y_4^3.$$

Because of commutativity, different trees can produce the same monomial. We therefore introduce a new structure, which is the same for trees yielding the same monomial. Given a tree, we identify the nodes bearing the same label, but we keep left and right edges distinct. We obtain a graph with $2n+1$ nodes labeled from $-n$ up to n, with left and right edges between nodes, as shown in Figure 2. Any such graph which originates from a tree, and therefore which does correspond to a monomial, must then satisfy the following conditions:

(1) node n has exactly one left edge and one right edge.
(2) for any node $i \geq 1$, the number of left edges linking i to nodes of strictly smaller label is equal to the number of right edges linking i to nodes of strictly smaller label and is equal to the total number of edges (left and right) linking i to nodes of strictly greater label.
(3) any node $i \leq 0$ is linked only to nodes whose label is strictly positive.

As we did with trees, we can define the corresponding monomial. The power of a variable $a_{i,j}$ (resp. $b_{i,j}$) is the number of left (resp. right) edges going from the node labeled i to the node labeled j. The power of a variable x_i is the number of edges going to the node $-n+i$. We now have a one-to-one correspondence between monomials appearing in G_n^n and graphs satisfying the above conditions. Translating those conditions to relations between

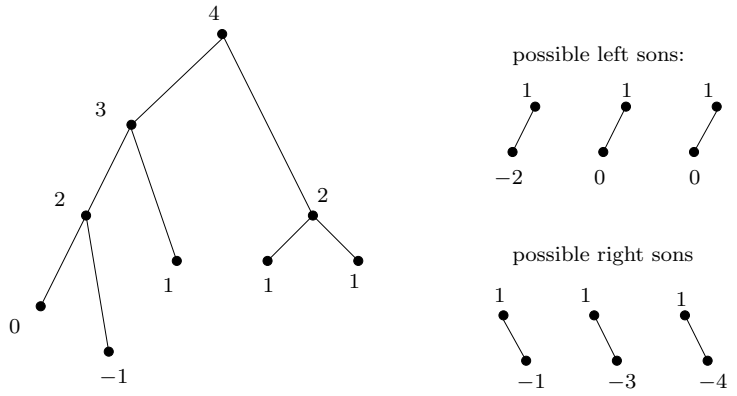

FIGURE 3. A partial tree compatible with the graph of Figure 2.

the powers of variables $\bar{a}$, $\bar{b}$ and $\bar{x}$, we now define a function which decides if a Boolean word coding a monomial in those variables does appear in G_n^n.

Denote by $A_{i,j}$ (respectively $B_{i,j}$) the number of left edges (respectively right edges) from node i to node j, we can now write the conditions above in the following way:

(1) $\sum_{i=-n}^{n-1} A_{n,i} = \sum_{i=-n}^{n-1} B_{n,i} = 1$.
(2) For all $i \geq 1$, $\sum_{j<i} A_{i,j} = \sum_{j<i} B_{i,j} = \sum_{j>i}(A_{j,i} + B_{j,i})$.
(3) Pour tout $i \leq 0$, pour tout $j < i$, $A_{i,j} = B_{i,j} = 0$.

Thus the numbers $\bar{A}$ and $\bar{B}$ determine the graph, and hence the monomial, which can be computed from those numbers. The power of a variable $a_{i,j}$ is exactly $A_{i,j}$. The power of the variable $b_{i,j}$ is $B_{i,j}$. The power of the variable y_i is $\sum_{j\geq 1}(A_{j,i} + B_{j,i})$. In other words there is an exact link between the powers and numbers of edges in the graph.

Let us use a tuple $\bar{\alpha}_{i,j}$ (respectively $\bar{\beta}_{i,j}$ and $\bar{\eta}_i$) to encode the powers of the variable $a_{i,j}$ (respectively $b_{i,j}$ and y_i). A monomial of G_n^n can then be encoded by a tuple $\bar{\alpha}, \bar{\beta}, \bar{\eta}$. We can easily use the link mentioned above to describe a polynomial size circuit $c(\bar{\alpha}, \bar{\beta}, \bar{\eta})$ that checks whether a tuple $\bar{\alpha}, \bar{\beta}, \bar{\eta}$ does represent a monomial of G_n^n. This done by checking the conditions on the numbers, using iterated additions of n numbers of n bits and checking equality of the results, in logarithmic depth. Note that the tuple $(\bar{\alpha}, \bar{\beta})$ determines the monomial and thus the powers of the variables $\bar{y}$, and those powers can be computed from $(\bar{\alpha}, \bar{\beta})$ in logarithmic depth.

5.3.3. *Coefficient of a monomial.* Recalling that our aim is to describe the coefficient function of G_n^n, we now need to compute the number of monomials in the non-commutative development which become identical with commutativity. In other words we need to compute the number of trees which yield a given graph. If we wish to go from a graph to a tree, we must split each node and give to each copy of this node a left-hand son and a right-hand son. Figure 3 gives an example of this process. The number of ways of doing this for the left-hand sons of node i is:

$$\binom{\sum_{j<i} A_{i,j}}{A_{i,i-1}} \binom{\sum_{j<i-1} A_{i,j}}{A_{i,i-2}} \cdots \binom{A_{i,-n} + A_{i,-n+1}}{A_{i,-n+1}} \binom{A_{i,-n}}{A_{i,-n}}.$$

This number must be multiplied by the equivalent for the right-hand sons, and we must multiply all the numbers obtained for all the nodes $i \geq 1$, thus obtaining the coefficient

value of the monomial. By simplifying the binomial coefficients, we can write it as:

$$\prod_{i=1}^{n}\left(\frac{\left(\sum_{j=-n}^{i-1}A_{i,j}\right)!}{\prod_{j=-n}^{i-1}A_{i,j}!}\cdot\frac{\left(\sum_{j=-n}^{i-1}B_{i,j}\right)!}{\prod_{j=-n}^{i-1}B_{i,j}!}\right),$$

however we will actually use the expression involving the binomial coefficients. Suppose we have a "magic" gate which takes two n bit numbers as input and computes their binomial coefficient. Then it is easy to see that we can build a circuit $v(\bar{\alpha},\bar{\beta})$, of polynomial size and degre, which uses binomial gates to compute the coefficient of the monomial encoded by $(\bar{\alpha},\bar{\beta},\bar{\epsilon})$. As remarked in section 5.3.2, this only depends on the tuples $\bar{\alpha}$ and $\bar{\beta}$.

5.4. **The polynomial D_n.** Going back to the definition of D_n, it is cumbersome but straightforward to define the function computing its coefficients. The polynomial D_n has the same variables as G_n^n, to which we added the variables $c_0,\ldots,c_n$. Let us encode the power of the variable c_i by the tuple $\bar{\gamma}_i$. We now wish to express the function $d_n(\bar{\alpha},\bar{\beta},\bar{\gamma},\bar{\eta})$ such that: $D_n=\sum_{\bar{\alpha},\bar{\beta},\bar{\gamma},\bar{\eta}} d(\bar{\alpha},\bar{\beta},\bar{\gamma},\bar{\eta})\ \bar{a}^{\bar{\alpha}}\,\bar{b}^{\bar{\beta}}\,\bar{c}^{\bar{\gamma}}\,\bar{y}^{\bar{\eta}}$. Thanks to sections 5.3.2 and 5.3.3, we can write G_n^n as $\sum_{\bar{\alpha},\bar{\beta},\bar{\eta}} v(\bar{\alpha},\bar{\beta})\,c(\bar{\alpha},\bar{\beta},\bar{\eta})\ \bar{a}^{\bar{\alpha}}\,\bar{b}^{\bar{\beta}}\,\bar{y}^{\bar{\eta}}$. The definition of D_n,

$$D_n=\sum_{k=0}^{n}\left(c_k\sum_{\bar{\epsilon}\in\{0,1\}^{n-k}}G_n^n(\bar{a},\bar{b},y_1,\ldots,y_k,\epsilon_1,\ldots,\epsilon_{n-k})\right),$$

yields

$$\begin{aligned}D_n &= \sum_{k=0}^{n}\left(c_k\sum_{\bar{\epsilon}\in\{0,1\}^{n-k}}\sum_{\bar{\alpha},\bar{\beta},\bar{\eta}}v(\bar{\alpha},\bar{\beta})\,c(\bar{\alpha},\bar{\beta},\bar{\eta})\ \bar{a}^{\bar{\alpha}}\,\bar{b}^{\bar{\beta}}\,y_1^{\bar{\eta}_1}\cdots y_k^{\bar{\eta}_k}\,\epsilon_1^{\bar{\eta}_{k+1}}\cdots\epsilon_{n-k}^{\bar{\eta}_n}\right)\\ &= \sum_{k=0}^{n}\sum_{\bar{\alpha},\bar{\beta},\bar{\epsilon}}\left(v(\bar{\alpha},\bar{\beta})\,c(\bar{\alpha},\bar{\beta},\bar{\epsilon})\ c_k\,\bar{a}^{\bar{\alpha}}\,\bar{b}^{\bar{\beta}}\,y_1^{\bar{\eta}_1}\cdots y_k^{\bar{\eta}_k}\sum_{\bar{\epsilon}\in\{0,1\}^{n-k}}\epsilon_1^{\bar{\eta}_{k+1}}\cdots\epsilon_{n-k}^{\bar{\eta}_n}\right).\end{aligned}$$

As we have seen, the powers of the variables $\bar{y}$ can be computed from the tuple $\bar{\alpha},\bar{\beta}$. The value of the sum $\sum_{\bar{\epsilon}}\epsilon_1^{\bar{\eta}_{k+1}}\cdots\epsilon_{n-k}^{\bar{\eta}_n}$ thus only depends on the tuple $\bar{\alpha},\bar{\beta}$. Call u the function which given a number k and the tuples $\bar{\eta}_i$ outputs the number of indexes i between $k+1$ and n such that $\bar{\eta}_i$ is the tuple $\bar{0}$. The value of the preceding sum is then $2^{u(k,\bar{\eta})}$. Given a binary representation of k, this function can be computed by a circuit in the following way. First test in parallel whether each $\bar{\eta}_i$ is the tuple $\bar{0}$. Then test in parallel for each i if it is strictly greater than k and multiply the result by that of the test on $\bar{\eta}_i$ to get the values δ_i. Finally add all the binary representations of δ_i to output the binary representation of $u(k,\bar{\eta})$. This can be done in logarithmic depth via iterated additions. The value $2^{u(k,\bar{\eta})}$ can then be computed by a circuit of polynomial depth.

The coefficient function of D_n starts by checking that the power of exactly one of the variables c_k is 1 and that the other powers are 0, and it computes the binary representation of the corresponding number k. All this can be done by a circuit in logarithmic depth, once again using iterated additions. It then computes the coefficient function of G_n^n, multiplies it by $2^{u(k,\bar{\epsilon})}$ and checks that the powers $\bar{\eta}_i$ of the variables y_i for i strictly greater than k are 0. This last step can also be accomplished in logarithmic depth. Thus if we take the computation of the binomial coefficients for granted, the coefficient function of D_n can be computed by a circuit of logarithmic depth.

But taking the computation of the binomial coefficient for granted is wishful thinking: we need to be able to compute $\binom{i}{j}$ with i and j taking values as high as 2^n. This should

not come as a surprise, since G_n^n is $\mathrm{VP}_{\mathrm{nb}}$-complete, and therefore can be projected to the polynomial:

$$(x+y)^{2^n} = \sum_{i=0}^{2^n} \binom{2^n}{i} x^i\, y^{2^n-i}.$$

The coefficients of this polynomial must then be written as a (simply exponential) sum over the coefficient function of G_n^n. Computing binomial coefficients efficiently is however possible when the characteristic is positive.

6. Rings of positive characteristic

6.1. **Fast computation of binomial coefficients.** In the case of rings of positive characteristic, we can make good use of the following theorem.

Theorem 3 (Lucas, 1878)**.** *Let m and n be two integers, written in base p as $m = m_0 + m_1 p + \cdots + m_d p^d$ and $n = n_0 + n_1 p + \cdots + n_d p^d$, then:*

$$\binom{n}{m} \equiv \binom{n_0}{m_0}\binom{n_1}{m_1}\cdots\binom{n_d}{m_d} \quad (mod\ p).$$

We show that the total coefficient function of the sequence (D_n) belongs to VP. Its membership in $\mathrm{VP}_{\mathrm{nb}}$ is actually sufficient to show the stability of the class $\mathrm{VNP}_{\mathrm{nb}}$, but this stronger result will become useful when we turn to the class $\mathrm{VP}_{\mathrm{nb}}$.

Lemma 5. *Over a ring of positive characteristic, the total coefficient function of the sequence (D_n) belongs to* VP*.*

Proof. We have stressed in the previous section that apart from the binomial coefficients, the other steps in the computation of the coefficient function can be done in logarithmic depth, and therefore by circuits of polynomial size and degree. Let us now show that a binomial coefficient can also be computed by a circuit of polynomial size and degree, when the characteristic of the ring is polynomial.

Finding the decomposition of a number $m = \sum_{i=0}^n m_i 2^i$ in base p can be done by hard-coding in the circuit the decomposition in base p of the powers of 2: $2^i = \sum_{j=0}^n d_{i,j} p^j$. We can then write:

$$m = \sum_{i=0}^n m_i \sum_{j=0}^n d_{i,j} p^j = \sum_{i=0}^n \left(\sum_{j=0}^n m_i d_{i,j} p^j \right).$$

Each expression $\sum_{j=0}^n m_i d_{i,j} p^j$ can be seen as representing a number in base p (its decomposition is given by the products $m_i d_{i,j}$). Thus if we compute the representation of the sum of these numbers when i ranges from 0 to n, we will get a representation of m in base p. What we need here is another iterated sum, this time in base p. We use the simple "3 for 2" trick but in base p: given three k-digit numbers $, x, y, z$, we find two $k+1$-digit numbers u and v such that $x + y + z = u + v$. The digits u_i are obtained in parallel by computing $x_i + y_i + z_i$ modulo p. The digits v_i are obtained by computing in parallel the carry of $x_i + y_i + z_i$. Because p is fixed for us, we can simulate the operations on the digits by small circuits of constant depth, and the iterated addition can be done in logarithmic depth.

Computing a binomial coefficient of two numbers smaller than p can then be done by a circuit of constant depth (with regard to n). Each of the binomial coefficients we wish to compute for the coefficient function of G_n^n is the product of a polynomial number of such "small" binomial coefficients, and we only need a polynomial number of them. □

6.2. **Stability of the classes** $\mathrm{VP}_{\mathrm{nb}}$ **and** $\mathrm{VNP}_{\mathrm{nb}}$**.** We now have enough to prove the stability of the class $\mathrm{VNP}_{\mathrm{nb}}$.

Theorem 4 (Over a ring of positive characteristic)**.** *Let f be a sequence of polynomials. The following properties are equivalent:*

(1) *f belongs to the class* $\mathrm{VNP}_{\mathrm{nb}}$.
(2) *any coefficient function of f belongs to the class* $\mathrm{VNP}_{\mathrm{nb}}$.
(3) *there exists a coefficient function of f which belongs to the class* $\mathrm{VNP}_{\mathrm{nb}}$.

Proof. The proof is similar to the proof given for the stability of VNP, but using the sequence (D_n) instead of the Hamiltonian. □

Theorem 5. *The following properties are equivalent:*

(1) $\mathrm{VP}_{\mathrm{nb}}$ *is stable for coefficient functions.*
(2) $\mathrm{VP}_{\mathrm{nb}} = \mathrm{VNP}_{\mathrm{nb}}$.

Proof. This proof is similar to the proof of theorem 2 in the case of the class VP, with one difficulty. We need to find a sequence (F_n) belonging to $\mathrm{VP}_{\mathrm{nb}}$, such that D_n appears as the coefficient of a monomial in F_n for all n. Let us write D_n as $\sum_{\bar{\alpha},\bar{\beta},\bar{\gamma},\bar{\eta}} d(\bar{\alpha},\bar{\beta},\bar{\gamma},\bar{\eta})\ \bar{a}^{\bar{\alpha}}\,\bar{b}^{\bar{\beta}}\,\bar{c}^{\bar{\gamma}}\,\bar{y}^{\bar{\eta}}$. We can build a polynomial $F_n(\bar{u},\bar{v},\bar{w},\bar{z})$ such that:

- F_n has variables $u_{i,j,0},\ldots,u_{i,j,n}$ for each variable $a_{i,j}$ of D_n, and similarly for variables $b_{i,j}$, $c_{i,j}$ and y_i.
- if we replace $u_{i,j,k}$ by $a_{i,j}^{2^k}$, $v_{i,j,k}$ by $b_{i,j}^{2^k}$, $w_{i,j,k}$ by $c_i^{2^k}$ and $z_{i,k}$ by $y_i^{2^k}$, we get D_n.
- F_n belongs to VNP.

In essence we show that we can separate the computation of D_n in two parts. First compute all the possible powers of the variables and constants it needs. Then apply the polynomial F_n, with the property that F_n belongs to VNP. F_n is then a projection of the Hamiltonian. Since the Hamiltonian is a coefficient of a VP sequence, F_n is also a coefficient of a VP sequence, and by replacing the variables of F_n with the powers of the variables of D_n, we thus show that D_n is a coefficient of a $\mathrm{VP}_{\mathrm{nb}}$ sequence. We now have all the ingredients to repeat the proof given in the case of polynomially bounded degree. □

7. A consequence for Valiant's hypothesis

The following result is an interesting consequence of the techniques introduced for the proof in the unbounded degree case.

Theorem 6. *Over a field of non-zero characteristic,* $\mathrm{VP} = \mathrm{VNP}$ *iff* $\mathrm{VP}_{\mathrm{nb}} = \mathrm{VNP}_{\mathrm{nb}}$.

Proof. That $\mathrm{VP}_{\mathrm{nb}} = \mathrm{VNP}_{\mathrm{nb}}$ implies $\mathrm{VP} = \mathrm{VNP}$ is obvious. For the converse suppose that $\mathrm{VP} = \mathrm{VNP}$. Then the sequence (F_n) introduced earlier belongs to VP. By replacing its variables with the powers of the variables of (D_n) we deduce that (D_n) belongs to $\mathrm{VP}_{\mathrm{nb}}$. Because (D_n) is $\mathrm{VNP}_{\mathrm{nb}}$-complete, $\mathrm{VP}_{\mathrm{nb}} = \mathrm{VNP}_{\mathrm{nb}}$. □

8. In characteristic 0

We have seen that the results in positive characteristic depend on the efficient computation of binomial coefficients. It seems difficult to find such a computation in characteristic 0. Consider the polynomial sequence:

$$f_n(x_0,\ldots,x_n,y_1,y_2) = \prod_{i=0}^{n}\left(x_i(y_1+y_2)^{2^i}+1-x_i\right).$$

This sequence belongs to $\mathrm{VP}_{\mathrm{nb}}$. Moreover, if $\bar{\beta}$ is a Boolean tuple, $f_n(\bar{\beta}, y_1, y_2)$ is the polynomial $(y_1+y_2)^{\bar{\beta}} = \sum_{\bar{\alpha}} \binom{\bar{\beta}}{\bar{\alpha}}\, y_1^{\bar{\alpha}}\, y_2^{\bar{\beta}-\bar{\alpha}}$. Call $g_n(\bar{x}, \bar{\epsilon}, \bar{\eta})$ the coefficient function of f_n for the variables y_1 and y_2: $f_n(\bar{x}, y_1, y_2) = \sum_{\bar{\epsilon},\bar{\eta}} g_n(\bar{x}, \bar{\epsilon}, \bar{\eta})\, y_1^{\bar{\epsilon}}\, y_2^{\bar{\eta}}$. If $\bar{\gamma}$ is the binary representation of the result of subtracting the number encoded by $\bar{\alpha}$ to the number encoded by $\bar{\beta}$, then $g_n(\bar{\beta}, \bar{\alpha}, \bar{\gamma})$ computes the binomial coefficient $\binom{\bar{\beta}}{\bar{\alpha}}$.

Thus the stability of the class $\mathrm{VP}_{\mathrm{nb}}$ would imply an efficient computation of the binomial coefficients. This is unlikely to be true, as it is easy to show that an efficient computation of the binomial coefficients implies an efficient computation of the factorial (i.e. $k!$ could be computed by a circuit of size polynomial in $\log k$). An efficient computation of the factorial would yield a fast algorithm for factorization. Note also that the question of the fast computation of the factorial also plays a part in the the BSS-model (cf. [BCSS98]). A recent article by Koiran also establishes links between the computation of integers and Valiant's theory [Koi05].

9. Polynomial differentiation

It is easy to see that if a polynomial can be computed by a circuit of size t, then its partial derivatives of order 1 can be each be computed by a circuit of size $4t$. More surprising is the fact that a circuit of size $4t$ is sufficient to compute all of them simultaneously [BS83]. In the case of iterated partial derivatives, there does not seem to be such efficient computations, as evidenced by the sequence (f_n) from section 4:

$$f_n(\bar{z}, \bar{y}) = \prod_{1\leq i\leq n} \left(\sum_{1\leq j\leq n} z_{i,j} y_j \right).$$

The coefficient of the monomial $y_1 \cdots y_n$ in f_n is the permanent of the variables $\bar{z}$ and is also the result of the iterated partial derivative $\partial^n f_n / \partial y_1 \cdots \partial y_n$. An efficient computation of iterated partial derivatives implies an efficient computation of the Permanent. In fact, the link between iterated partial derivation and Valiant's classes is a bit tighter, as shown in the following theorems.

Theorem 7. *The following properties are equivalent.*

(1) $\mathrm{VP} = \mathrm{VNP}$.

(2) *For all sequence $(f_n(x_1, \ldots, x_{q(n)}))$ belonging to* VP*, for all polynomial $p(n)$, for all families $p_{n,i}$, where i ranges from 0 to $q(n)$, whose sum is $p(n)$, the sequence:*

$$\left(\frac{\partial^{p(n)} f_n}{\partial x_1^{p_{n,1}} \cdots \partial x_{q(n)}^{p_{n,q(n)}}} \right)$$

belongs to VP*.*

Proof. (1) $\rightarrow$(2). Consider a single polynomial $f(x_1, \ldots, x_k) = \sum_{\bar{\alpha}} g(\bar{\alpha}) x_1^{\bar{\alpha}_1} \cdots x_k^{\bar{\alpha}_k}$. For a given tuple of numbers $(p_1, \ldots, p_k)$ of sum p, call f' the iterated partial derivative $\left(\frac{\partial^p f}{\partial x_1^{p_1} \cdots \partial x_k^{p_k}} \right)$. Call g' the total coefficient function of f': $f'(x_1, \ldots, x_k) = \sum_{\bar{\beta}} g'(\bar{\beta}) x_1^{\bar{\beta}_1} \cdots x_k^{\bar{\beta}_k}$. We wish to express g' using g. On input $(\bar{\beta}_1, \ldots, \bar{\beta}_k)$, we just add the values p_i to the number represented by the tuple $\bar{\beta}_i$ for each i (we compute this in binary representation). We then apply the function g to the result. Let b_i be the integer represented by $\bar{\beta}_i$. We then compute the products $(b_i+1) \cdots (b_i+p_i)$ for all i and multiply them by the previous result. This computes the coefficient function of f'. It is not difficult to see that apart from the computation of g, this can be done on input by a circuit of depth logarithmic

in k and p. If we now consider a sequence (f_n) in VP, then we know that its coefficient function belongs to VNP and, assuming VP = VNP, to VP. Then by applying the case of a single polynomial to f_n, we can prove that the total coefficient function of the sequence of iterated partial derivative of f_n also belongs to VP. This means that the sequence of iterated partial derivative of (f_n) belongs to VNP and thus to VP by assumption.

(2) $\rightarrow$(1). We can reason in the same way as we did for the Permanent at the beginning of this section, but using the polynomial which contains a monomial whose coefficient is the Hamiltonian instead. □

It was important to consider the full coefficient function, and not just a coefficient, in order to obtain this equivalence, which links the possibility of efficiently computing a fundamental task of symbolic computation to a major open question of complexity theory.

The same property can be obtained for classes of unbounded degree, when the characteristic is positive. The proof is similar, but we do not need to be careful with the degree of the circuits.

Theorem 8. *Over any ring of positive characteristic, the following properties are equivalent.*

(1) $\mathrm{VP}_{\mathrm{nb}} = \mathrm{VNP}_{\mathrm{nb}}$.

(2) *For all sequence* $(f_n(x_1, \ldots, x_{q(n)}))$ *belonging to* $\mathrm{VP}_{\mathrm{nb}}$*, for all polynomial* $p(n)$*, for all families* $p_{n,i}$*, where* i *ranges from* 0 *to* $q(n)$*, whose sum is* $p(n)$*, the sequence:*

$$\left(\frac{\partial^{p(n)} f_n}{\partial x_1^{p_{n,1}} \cdots \partial x_{q(n)}^{p_{n,q(n)}}} \right)$$

belongs to $\mathrm{VP}_{\mathrm{nb}}$.

Note that thanks to theorem 6, this means that for rings of positive characteristic, an efficient way to compute iterated partial derivatives for polynomials of polynomially bounded degree implies an efficient way to compute them for arbitrary polynomials.

In a previous article [MP06], we showed that the equality $\mathrm{VP}_{\mathrm{ws}} = \mathrm{VNP}$ is equivalent to the possibility of writing a permanent as a determinant of reasonable size, an old mathematical question. Our results on derivatives relate the efficient computations of iterated partial derivatives to a similar equality of complexity classes. Thus the techniques of complexity theory can give meaning both to mathematical problems and to computational ones.

References

[All04] E. Allender. Arithmetic Circuits and Counting Complexity Classes. In Jan Krajicek, editor, *Complexity of Computations and Proofs*, volume 13 of *Quaderni di Matematica*, pages 33–72. Seconda Universita di Napoli, 2004.

[BCSS98] L. Blum, F. Cucker, M. Shub, and S. Smale. *Complexity and Real Computation*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998.

[BS83] W. Baur and V. Strassen. The Complexity of Partial Derivatives. *Theor. Comput. Sci.*, 22:317–330, 1983.

[BSS89] Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc. (N.S.)*, 21(1):1–46, 1989.

[Bür00] P. Bürgisser. *Completeness and Reduction in Algebraic Complexity Theory*, volume 7 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2000.

[Ham79] J. Hammon. Question 6001. *Educ. Times*, 32(179), 1879.

[Kal86] E Kaltofen. Uniform Closure Properties of P-computable Functions. In *STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 330–337, New York, NY, USA, 1986. ACM Press.

[Koi05] P. Koiran. Valiant's Model and the Cost of Computing Integers. *Computational Complexity*, 13(3-4):131–146, 2005.

[MP06] G. Malod and N. Portier. Characterizing Valiant's Algebraic Complexity Classes. In *MFCS 2006, Proceedings of the 31st International Symposium on Mathematical Foundations of Computer Science*, volume 4162 of *Lecture Notes in Computer Science*, pages 704–716. Springer Verlag, 2006.

[Val79] L. G. Valiant. Completeness Classes in Algebra. In *STOC '79: Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 249–261, New York, NY, USA, 1979. ACM Press.

[Val82] L. G. Valiant. Reducibility by Algebraic Projections. In *Logic and Algorithmic: an International Symposium held in honor of Ernst Specker*, volume 30 of *Monographies de l'Enseignement Mathémathique*, pages 365–380, 1982.

[Ven92] H. Venkateswaran. Circuit Definitions of Nondeterministic Complexity Classes. *SIAM J. Comput.*, 21(4):655–670, 1992.

[Vol99] H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999.

[vzG87] J. von zur Gathen. Feasible Arithmetic Computations: Valiant's Hypothesis. *J. Symb. Comput.*, 4(2):137–172, 1987.

E-mail address: Guillaume.Malod@umh.ac.be

Institut de Mathématiques, Université de Mons-Hainaut, 6 av. du Champ de Mars, 7000 Mons, Belgium