HAL
archives-ouvertes.fr

Interpolation in Valiant's theory

Pascal Koiran, Sylvain Perifel

▶ **To cite this version:**

Pascal Koiran, Sylvain Perifel. Interpolation in Valiant's theory. 2007. ensl-00175862

HAL Id: ensl-00175862
https://hal-ens-lyon.archives-ouvertes.fr/ensl-00175862

Submitted on 1 Oct 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ensl-00175862, version 1 - 1 Oct 2007

Interpolation in Valiant's theory

Pascal Koiran and Sylvain Perifel

LIP*, École Normale Supérieure de Lyon.
[Pascal.Koiran,Sylvain.Perifel]@ens-lyon.fr

Abstract. We investigate the following question: if a polynomial can be evaluated at rational points by a polynomial-time boolean algorithm, does it have a polynomial-size arithmetic circuit? We argue that this question is certainly difficult. Answering it negatively would indeed imply that the constant-free versions of the algebraic complexity classes VP and VNP defined by Valiant are different. Answering this question positively would imply a transfer theorem from boolean to algebraic complexity.
Our proof method relies on Lagrange interpolation and on recent results connecting the (boolean) counting hierarchy to algebraic complexity classes. As a byproduct we obtain two additional results:

(i) The constant-free, degree-unbounded version of Valiant's hypothesis VP $\neq$ VNP implies the degree-bounded version. This result was previously known to hold for fields of positive characteristic only.

(ii) If exponential sums of easy to compute polynomials can be computed efficiently, then the same is true of exponential products. We point out an application of this result to the P=NP problem in the Blum-Shub-Smale model of computation over the field of complex numbers.

1 Introduction

Motivation – The starting point of this paper is a question raised by Christos Papadimitriou in a personal communication to Erich Kaltofen[1]:

Question (*)

If a multivariate polynomial P can be evaluated by a (boolean) polynomial-time algorithm on rational inputs, does that imply that P can be computed by a polynomial-size arithmetic circuit? In such a circuit, the only allowed operations are additions, subtractions, and multiplications.

This question can be interpreted in several ways: one should at least state in which ring the coefficients of P lie, and which constants can be used by the arithmetic circuits. Here we will focus on polynomials with integer coefficients,

* UMR 5668 ENS Lyon, CNRS, UCBL, INRIA.

[1] At the Oberwolfach complexity theory workshop where a part of this work was presented in June 2007, several participants told P.K. that they had independently thought of the same question.

and most of the paper will deal with constant-free circuits. In Section 3.4 we study the case of circuits with rational constants computing polynomials with integer coefficients (as we shall see, this is more natural than it might seem at first sight).

As pointed out by Papadimitriou, Strassen's "Vermeidung von Divisionen" (see for instance [7], chapter 7) shows that for evaluating a low-degree polynomial P, divisions would not increase exponentially the power of arithmetic circuits. It is indeed a natural question whether, more generally, all boolean operations can be replaced efficiently by additions, subtractions and multiplications. Questions of the same flavour (can "looking at bits" help for arithmetic computations?) have been studied before. In particular, Kaltofen and Villard have shown that looking at bits does help for computing the determinant [10].

Discussion – It is not clear what the correct answer to question (*) should be. In this paper we will argue that answering it either way seems difficult.

A natural strategy for obtaining a negative answer to question (*) would be to exhibit a family of polynomials that are easy to evaluate on rational inputs but hard to evaluate by arithmetic circuits. Unfortunately, there seems to be a lack of candidate polynomials. Another difficulty is that a negative answer would imply the separation of the algebraic complexity classes VP^0 and VNP^0. This observation is our main contribution to the study of question (*), and it is established in Theorem 8. The classes VP^0 and VNP^0 are constant-free versions of the classes VP (of "easily computable polynomial families") and VNP (of "easily definable polynomial families") introduced by Valiant (precise definitions are given in the next section). The separation $\mathsf{VP}^0 \neq \mathsf{VNP}^0$ seems very plausible, but it also seems very difficult to establish. As explained at the beginning of the introduction, we study in Section 3.4 the case of circuits with rational constants computing polynomials with integer coefficients. Allowing rational constants makes the hypothesis that question (*) has a negative answer stronger than in the constant-free case. Accordingly, we obtain a stronger conclusion: we can now show that the hypothesis would imply a superpolynomial lower bound on the size of arithmetic circuits computing the permanent.

Obtaining a positive answer to question (*) also seems difficult since it would imply the following transfer theorem: $\mathsf{FP} = \sharp\mathsf{P} \Rightarrow \mathsf{VP} = \mathsf{VNP}$ (assuming that $\mathsf{FP} = \sharp\mathsf{P}$, the permanent must be in FP; a positive answer to question (*) would therefore imply that the permanent is in VP, and that $\mathsf{VP} = \mathsf{VNP}$ by completeness of the permanent). Unfortunately, in spite of all the work establishing close connections between the boolean model of computation and the algebraic models of Valiant and of Blum, Shub and Smale [2,4,5,8,9,13,14,15] no such transfer theorem is known. In fact, we do not know of any hypothesis from boolean complexity theory that would imply the equality $\mathsf{VP} = \mathsf{VNP}$ (but transfer theorems in the opposite direction were established in [4]).

Summary of results – Most of our results are derived from Theorem 3, our main theorem: if the evaluation of a family of polynomials (f_n) at integer points is a problem that lies in the (non uniform) counting hierarchy, the hypothesis $\mathsf{VP}^0 = \mathsf{VNP}^0$ implies that (f_n) can be evaluated by polynomial-size

arithmetic circuits. Theorem 8, which contains our main contribution to the study of question (*), follows immediately since polynomial-time problems lie in the counting hierarchy. The proof of Theorem 3 relies on techniques from [1,6] and on Lagrange interpolation. Besides the application to question (*), we derive two additional results from Theorem 3:

- The elements of the complexity classes VP, VNP and of their constant-free versions are families of polynomials of polynomially bounded degree. We show in Theorem 5 that the collapse $\mathsf{VP}^0 = \mathsf{VNP}^0$ would imply the same collapse for the unbounded versions of VP^0 and VNP^0. For fields of positive characteristic, the same result (and its converse) was obtained with different techniques by Malod [16,17].
- Our third application of Theorem 3 is to the "$\mathsf{P} = \mathsf{NP}$?" problem in the Blum-Shub-Smale model of computation over $\mathbb{C}$ (or more generally, fields of characteristic 0). One natural strategy for separating $\mathsf{P}_\mathbb{C}$ from $\mathsf{NP}_\mathbb{C}$ would be to exhibit a problem A in $\mathsf{NP}_\mathbb{C} \setminus \mathsf{P}_\mathbb{C}$. Drawing on results from [13], we show that this strategy is bound to fail for a fairly large class of "simple" problems A, unless one can prove that $\mathsf{VP}^0 \neq \mathsf{VNP}^0$. The class of "simple" problems that we have in mind is $\mathsf{NP}_{(\mathbb{C},+,-,=)}$. This is the class of NP problems over the set of complex numbers endowed with addition, subtraction, and equality tests (there is therefore no multiplication in this structure). It contains many natural problems, such as Subset Sum and Twenty Questions [2,19], that most likely belong to $\mathsf{NP}_\mathbb{C} \setminus \mathsf{P}_\mathbb{C}$. As an intermediate result, we show in Theorem 5 that if exponential sums of easy to compute polynomials can be computed efficiently, then the same is true of exponential products.

2 Preliminaries

2.1 Valiant's Classes

In Valiant's model, one computes families of polynomials. A book-length treatment of this topic can be found in [4]. We fix a field K of characteristic zero.

An arithmetic circuit is a circuit whose inputs are indeterminates $x_1, \ldots, x_{u(n)}$ together with arbitrary constants of K; there are $+$, $-$ and $\times$-gates, and we therefore compute multivariate polynomials. The polynomial computed by an arithmetic circuit is defined in the usual way by the polynomial computed by its output gate. The size of a circuit is the number of gates.

Thus a family (C_n) of arithmetic circuits computes a family (f_n) of polynomials, $f_n \in K[x_1, \ldots, x_{u(n)}]$. The class $\mathsf{VP_{nb}}$ defined in [17] is the set of families (f_n) of polynomials computed by a family (C_n) of polynomial-size arithmetic circuits, i.e., C_n computes f_n and there exists a polynomial $p(n)$ such that $|C_n| \leq p(n)$ for all n. We will assume without loss of generality that the number $u(n)$ of variables is bounded by a polynomial function of n. The subscript "nb" indicates that there is no bound on the degree of the polynomial, in contrast with the original class VP of Valiant where a polynomial bound on the degree of the

polynomial computed by the circuit is required. Note that these definitions are nonuniform.

The class VNP is the set of families of polynomials defined by an exponential sum of VP families. More precisely, $(f_n(\bar{x})) \in \mathsf{VNP}$ if there exists $(g_n(\bar{x},\bar{y})) \in \mathsf{VP}$ and a polynomial p such that $|\bar{y}| = p(n)$ and $f_n(\bar{x}) = \sum_{\bar{\epsilon}\in\{0,1\}^{p(n)}} g_n(\bar{x},\bar{\epsilon})$. Similarly, the class $\mathsf{V\Pi P}$ is the set of families of polynomials defined by an exponential product of $\mathsf{VP_{nb}}$ families. More precisely, $(f_n(\bar{x})) \in \mathsf{V\Pi P}$ if there exists $(g_n(\bar{x},\bar{y})) \in \mathsf{VP_{nb}}$ and a polynomial p such that $|\bar{y}| = p(n)$ and $f_n(\bar{x}) = \prod_{\bar{\epsilon}\in\{0,1\}^{p(n)}} g_n(\bar{x},\bar{\epsilon})$.

We can also define constant-free circuits: the only constant allowed is then 1 (in order to allow the computation of constant polynomials). In this case, we compute polynomials with integer coefficients. If f is a polynomial with integer coefficients, we denote by $\tau(f)$ the size of a smallest constant-free circuit computing f. For classes of families of polynomials, we will use the superscript 0 to indicate the absence of constant: for instance, we will write $\mathsf{VP}^0_{\mathsf{nb}}$. For bounded-degree classes, we are to be more careful because we also want to avoid the computation of constants of exponential bitsize: we first need the following definition.

Definition 1. *Let C be an arithmetic circuit. The* formal degree *of a gate of C is defined by induction:*

- *the formal degree of an input is 1;*
- *the formal degree of a gate $+$ or $-$ is the maximum of the formal degrees of its inputs;*
- *the formal degree of a gate $\times$ is the sum of the formal degrees of its inputs.*

Now, the formal degree of a circuit is the formal degree of the output gate.

We are now able to define constant-free degree-bounded Valiant's classes. A family of polynomials (f_n) belongs to VP^0 if it is computable by a family of circuits of size and formal degree bounded by a polynomial function of n. The class VNP^0 is then defined accordingly by a sum of VP^0 families, in the same way as VNP is defined from VP.

Remark 1. The hypothesis $\tau(\mathrm{PER}_n) = n^{O(1)}$ used in [6] is implied by the hypothesis $\mathsf{VNP}^0 \subset \mathsf{VP}^0_{\mathsf{nb}}$ and hence by $\mathsf{VP}^0 = \mathsf{VNP}^0$. As mentioned in [6], the converse $\tau(\mathrm{PER}_n) = n^{O(1)} \Rightarrow \mathsf{VNP}^0 = \mathsf{VP}^0$ is not known to hold, because the family (PER_n) is not known to be VNP^0-complete in a constant-free context (the proof of completeness of Valiant [20] indeed uses the constant $1/2$). We will mostly be concerned by the hypothesis $\mathsf{VP}^0 = \mathsf{VNP}^0$, but we will come to the hypothesis $\tau(\mathrm{PER}_n) = n^{O(1)}$ in Section 3.4 when dealing with circuits with constants.

2.2 Counting Classes

In this paper we will encounter several counting classes, in particular the counting hierarchy defined below. Let us first see two classes of functions, $\sharp\mathsf{P}$ and GapP.

Definition 2. – *The class* $\sharp\mathsf{P}$ *is the set of functions* $f : \{0,1\}^* \to \{0,1\}^*$ *such that there exist a language* $A \in \mathsf{P}$ *and a polynomial* $p(n)$ *satisfying*

$$f(x) = \#\{y \in \{0,1\}^{p(|x|)} : (x,y) \in A\}.$$

– *A function* f *is in* GapP *if it is the difference of two functions in* $\sharp\mathsf{P}$.

Returning to classes of languages we recall the definition of the counting hierarchy, introduced by Wagner [21]. It contains all the polynomial hierarchy PH and is contained in PSPACE. It is defined via the "majority" operator $\mathbf{C}$ as follows.

Definition 3. – *If* K *is a complexity class, the class* $\mathbf{C}.K$ *is the set of languages* A *such that there exist a language* $B \in K$ *and a polynomial* $p(n)$ *satisfying*

$$x \in A \iff \#\{y \in \{0,1\}^{p(|x|)} : (x,y) \in B\} \geq 2^{p(|x|)-1}.$$

– *The* i*-th level* $\mathsf{C_iP}$ *of the counting hierarchy is defined recursively by* $\mathsf{C_0P} = \mathsf{P}$ *and* $\mathsf{C_{i+1}P} = \mathbf{C}.\mathsf{C_iP}$. *The counting hierarchy* CH *is the union of all these levels* $\mathsf{C_iP}$.

Level 1 of CH, that is, $\mathbf{C}.P$, is also called PP. Since Valiant's classes are nonuniform, we will rather work with nonuniform versions of these boolean classes, as defined now following Karp and Lipton [11].

Definition 4. *If* K *is a complexity class, the class* K/poly *is the set of languages* A *such that there exist a language* $B \in K$*, a polynomial* $p(n)$ *and a family of words (called advices)* $(a_n)_{n\geq 0}$ *satisfying*

– *for all* $n \geq 0$, $|a_n| \leq p(n)$;
– *for all word* x, $x \in A \iff (x, a(|x|)) \in B$.

Remark that the advice only depends on the size of x*: it must therefore be the same for all words of same length.*

2.3 Sequences of Integers

Our aim now is to introduce a notion of complexity of a sequence of integers. In order to avoid dealing with the sign of integers separately, we assume that we can retrieve it from the boolean encoding of the integers. For example, the sign could be given by the first bit of the encoding and the absolute value by the remaining bits.

Definition 5. *A* sequence of exponential bitsize *is a sequence of integers* $(a(n,k))$ *such that there exists a polynomial* $p(n)$ *satisfying:*

1. $a(n,k)$ *is defined for* $n,k \in \mathbb{N}$ *and* $0 \leq k < 2^{p(n)}$;
2. *for all* $n > 1$*, for all* $k < 2^{p(n)}$*, the bitsize of* $a(n,k)$ *is* $\leq 2^{p(n)}$.

From $a(n,k)$, the following language is then defined:

$$\mathrm{Bit}(a) = \{(1^n, k, j, b) |\ \textit{the } j\textit{-th bit of } a(n,k) \textit{ is } b\},$$

The reader should be aware that the above definition and the next one are not quite the same as in [6]: we use a unary encoding for n instead of a binary encoding.

Definition 6. *A sequence $a(n,k)$ of exponential bitsize is* definable in $\mathsf{CH/poly}$ *if the language* $\mathrm{Bit}(a)$ *is in* $\mathsf{CH/poly}$.

Remark 2. We shall also meet sequences with more than two parameters (n,k), for example $a(n, \alpha^{(1)}, \ldots, \alpha^{(n)})$ for some integers $\alpha^{(i)}$. In order to see it as a sequence with two parameters, $(\alpha^{(1)}, \ldots, \alpha^{(n)})$ will be considered as the encoding of a single integer. The parameter n might also be given as a subscript, as in $f_n(k)$, which should better be written $f(n,k)$.

Let us now propose a similar definition for families of polynomials.

Definition 7. *Let $(f_n(x_1, \ldots, x_{u(n)}))$ be a family of polynomials with integer coefficients. We say that (f_n) can be* evaluated in $\mathsf{CH/poly}$ at integer points *if the following conditions are satisfied:*

1. *the number $u(n)$ of variables is polynomially bounded;*
2. *the degree of f_n as well as the bitsize of the coefficients of f_n are bounded by $2^{p(n)}$ for some polynomial $p(n)$;*
3. *the language $\{(1^n, i_1, \ldots, i_{u(n)}, j, b) |$ the j-th bit of $f_n(i_1, \ldots, i_{u(n)})$ is $b\}$ is in* $\mathsf{CH/poly}$.

Remark 3. The same definition can be made for other complexity classes than $\mathsf{CH/poly}$. For instance, if we replace $\mathsf{CH/poly}$ by P we obtain the notion of "polynomial time evaluation at integer points". This notion will be useful for the study of question (*).

The following lemma is obvious from these definitions.

Lemma 1. *The family $(f_n(x_1, \ldots, x_{u(n)}))$ can be evaluated in* $\mathsf{CH/poly}$ *at integer points if and only if the sequence of integers $a(n, i_1, \ldots, i_{u(n)}) = f_n(i_1, \ldots, i_{u(n)})$ is definable in* $\mathsf{CH/poly}$.

The following theorem of [1, Theorem 4.1] will also be useful due to its corollary below.

Theorem 1. *Let BitSLP be the following problem: given a constant-free arithmetic circuit computing an integer N, and given $i \in \mathbb{N}$ in binary, decide whether the i-th bit of the binary representation of N is 1. Then BitSLP is in* CH.

Corollary 1. *If $(f_n) \in \mathsf{VP}^0_{\mathsf{nb}}$ then it can be evaluated in* $\mathsf{CH/poly}$ *at integer points.*

The results of this paper rely on the following link between Valiant's classes and the counting hierarchy, [6, Lemmas 2.5 and 2.12].

Lemma 2. *If* $\mathsf{VP}^0 = \mathsf{VNP}^0$ *then* $\mathsf{CH/poly} = \mathsf{P/poly}$.

In particular, Lemma 2 was used to show that big sums and products are computable in the counting hierarchy, [6, Theorem 3.7]. As already mentioned, the context is not exactly the same as in [6] because we use a unary encoding. We now give a version of this result which is just an easy "scaling up" of [6, Theorem 3.7] (it is enough to define $a'(2^{p(n)}, k) = a(n, k)$ and to apply the result of Bürgisser).

Theorem 2. *Let* $p(n)$ *be a polynomial and suppose* $a = (a(n,k))_{n\in\mathbb{N},k\leq 2^{p(n)}}$ *is definable in* $\mathsf{CH/poly}$. *Consider the sequences*

$$b(n) = \sum_{k=0}^{2^{p(n)}} a(n,k) \text{ and } d(n) = \prod_{k=0}^{2^{p(n)}} a(n,k).$$

Then $(b(n))_{n\in\mathbb{N}}$ *and* $(d(n))_{n\in\mathbb{N}}$ *are definable in* $\mathsf{CH/poly}$.

Suppose now that $(s(n))_{n\in\mathbb{N}}$ *and* $(t(n))_{n\in\mathbb{N}}$ *are definable in* $\mathsf{CH/poly}$. *Then the sequence of products* $(s(n)t(n))_{n\in\mathbb{N}}$, *and, if* $t(n) > 0$, *the sequence of quotients* $(\lfloor s(n)/t(n)\rfloor)_{n\in\mathbb{N}}$, *are definable in* $\mathsf{CH/poly}$.

3 Interpolation

We now begin the main technical developments.

3.1 Coefficients

The following lemma is Valiant's criterion [20], see also [4, Prop. 2.20] and [12, Th. 2.3].

Lemma 3. *Let* $a : (1^n, i) \mapsto a(1^n, i)$ *be a function of* $\mathsf{GapP/poly}$, *where* n *is given in unary and* i *in binary. Let* $p(n)$ *be a polynomial and define the following sequence of polynomials:*

$$f_n(x_1, \ldots, x_{p(n)}) = \sum_{i=0}^{2^{p(n)}-1} a(1^n, i) x_1^{i_1} \cdots x_{p(n)}^{i_{p(n)}},$$

where i_j *is the* j*-th bit in the binary expression of* i.

Then $(f_n) \in \mathsf{VNP}^0$.

Here is a "scaled up" generalization of [6, Th. 4.1(2)] to multivariate polynomials.

Lemma 4. *Let*

$$f_n(x_1,\dots,x_n) = \sum_{\alpha^{(1)},\dots,\alpha^{(n)}} a(n,\alpha^{(1)},\dots,\alpha^{(n)}) x_1^{\alpha^{(1)}} \cdots x_n^{\alpha^{(n)}},$$

where the integers $\alpha^{(i)}$ range from 0 to 2^n-1 and $a(n,\alpha^{(1)},\dots,\alpha^{(n)})$ is a sequence of integers of absolute value $< 2^{2^n}$ definable in $\mathsf{CH/poly}$.

If $\mathsf{VP}^0 = \mathsf{VNP}^0$ *then* $(f_n) \in \mathsf{VP}^0_{\mathsf{nb}}$.

Proof. Expand a in binary: $a(n,\alpha^{(1)},\dots,\alpha^{(n)}) = \sum_{i=0}^{2^n} a_i(n,\bar{\alpha})2^i$. Let h_n be the following polynomial:

$$h_n(x_{1,1},x_{1,2},\dots,x_{1,n},x_{2,1},\dots,x_{n,n},z_1,\dots,z_n) =$$

$$\sum_{i=0}^{2^n}\sum_{\bar{\alpha}} a_i(n,\bar{\alpha}) z_1^{i_1}\cdots z_n^{i_n} x_{1,1}^{\alpha_1^{(1)}} x_{1,2}^{\alpha_2^{(1)}} \cdots x_{1,n}^{\alpha_n^{(1)}} x_{2,1}^{\alpha_1^{(2)}} \cdots x_{n,n}^{\alpha_n^{(n)}}.$$

Then we have:

$$h_n(x_1^{2^0},x_1^{2^1},\dots,x_1^{2^n},x_2^{2^0},\dots,x_n^{2^n},2^{2^0},2^{2^1},\dots,2^{2^n}) = f_n(x_1,\dots,x_n).$$

Since $\mathsf{VP}^0 = \mathsf{VNP}^0$, by Lemma 2 the nonuniform counting hierarchy collapses, therefore computing the i-th bit $a_i(n,\bar{\alpha})$ of $a(n,\bar{\alpha})$ on input $(1^n,\bar{\alpha},i)$ is in $\mathsf{GapP/poly}$ (and even in $\mathsf{P/poly}$). By Lemma 3, $(h_n) \in \mathsf{VNP}^0$. By the hypothesis $\mathsf{VP}^0 = \mathsf{VNP}^0$, $(h_n) \in \mathsf{VP}^0$ and thus using repeated squaring for computing big powers yields $(f_n) \in \mathsf{VNP}^0_{\mathsf{nb}}$. □

3.2 Interpolation

Let us now state two lemmas on interpolation polynomials.

Lemma 5 (multivariate Lagrange interpolation). *Let $p(x_1,\dots,x_n)$ be a polynomial of degree $\le d$. Then*

$$p(x_1,\dots,x_n) = \sum_{0\le i_1,\dots,i_n\le d} p(i_1,\dots,i_n) \prod_{k=1}^{n}\Bigg(\prod_{j_k\neq i_k}\frac{x_k-j_k}{i_k-j_k}\Bigg),$$

where the integers j_k range from 0 to d.

Proof. The proof goes by induction on the number n of variables. For $n=1$, this is the usual Lagrange interpolation formula: we have

$$p(x) = \sum_{i=0}^{d} p(i) \prod_{j\neq i}\frac{x-j}{i-j}$$

because both polynomials are of degree $\le d$ and coincide on at least $d+1$ distinct points.

For $n+1$, the induction case $n=1$ yields

$$p(x_1,\ldots,x_{n+1}) = \sum_{i_{n+1}=0}^{d} p(x_1,\ldots,x_n,i_{n+1}) \prod_{j_{n+1}\neq i_{n+1}} \frac{x_{n+1}-j_{n+1}}{i_{n+1}-j_{n+1}}.$$

By induction hypothesis, this is equal to

$$\sum_{i_{n+1}=0}^{d} \left(\sum_{0\leq i_1,\ldots,i_n\leq d} p(i_1,\ldots,i_n) \prod_{k=1}^{n} \prod_{j_k\neq i_k} \frac{x_k-j_k}{i_k-j_k} \right) \prod_{j_{n+1}\neq i_{n+1}} \frac{x_{n+1}-j_{n+1}}{i_{n+1}-j_{n+1}}$$

which is the desired result. □

Lemma 6. *Let $a(n) = \prod_{i=0}^{2^n-1} \prod_{j\neq i}(i-j)$, where j ranges from 0 to 2^n-1. Let $p_{i_1,\ldots,i_n}(\bar{x})$ be the following family of polynomials:*

$$p_{i_1,\ldots,i_n}(x_1,\ldots,x_n) = \prod_{k=1}^{n} \left(a(n) \prod_{j_k\neq i_k} \frac{x_k-j_k}{i_k-j_k} \right),$$

where the integers j_k range from 0 to 2^n-1 and the integers i_k are given in binary and range from 0 to 2^n-1. Then the coefficients of $p_{i_1,\ldots,i_n}$ are integers definable in the counting hierarchy, as is $a(n)$.

Proof. As a first step, note that the coefficient of the monomial $x_1^{\alpha_1}\cdots x_n^{\alpha_n}$ in p_n is equal to the product of the coefficients of the monomials $x_k^{\alpha_k}$ in the univariate polynomials $a(n)\prod_{j_k\neq i_k} \frac{x_k-j_k}{i_k-j_k}$. Hence we just have to check that these different coefficients of univariate polynomials are themselves definable in the counting hierarchy. Let us first focus on the univariate polynomial $\prod_{j_k\neq i_k}(x_k-j_k)$, that is, let us forget the multiplicative term $b(n,i_k) = a(n)/\prod_{j_k\neq i_k}(i_k-j_k)$ for the moment.

We use the same argument as [6, Cor. 3.9]. Namely, we remark that the coefficients of this polynomial are bounded in absolute value by $2^{2^{n^2}}$. Therefore in the univariate polynomial $\prod_{j_k\neq i_k}(x_k-j_k)$ we can replace the variable x_k by $2^{2^{n^2}}$ and there will be no overlap of the coefficients of the different powers of x_k, thus we can recover the coefficients of the monomial from the value of this product. By the first part of Theorem 2, we can evaluate in the counting hierarchy the polynomial at the point $2^{2^{n^2}}$, because it is a product of exponential size. So the coefficients are definable in the counting hierarchy.

It is now enough to note that the first part of Theorem 2 implies that $a(n)$ as well as $b(n,i_k) = a(n)/\prod_{j_k\neq i_k}(i_k-j_k)$ are also definable in the counting hierarchy. □

Remark that the sequence $a(n)$ of Lemma 6 is introduced only so as to obtain integer coefficients. We will then divide by $a(n)$ in the next proofs.

3.3 Main Results

Let us now state the main theorem.

Theorem 3. *Let $(f_n(x_1, \ldots, x_{u(n)}))$ be a family of multivariate polynomials. Suppose (f_n) can be evaluated in $\mathsf{CH/poly}$ at integer points. If $\mathsf{VP}^0 = \mathsf{VNP}^0$ then $(f_n) \in \mathsf{VP}^0_{\mathsf{nb}}$.*

Proof. The goal is to use the interpolation formula of Lemma 5:

$$f_n(x_1, \ldots, x_{u(n)}) = \sum_{0 \leq i_1, \ldots, i_{u(n)} \leq d} b_{i_1, \ldots, i_{u(n)}}(\bar{x}), \tag{1}$$

where $b_{i_1,\ldots,i_{u(n)}}(\bar{x}) = f_n(i_1, \ldots, i_{u(n)}) \prod_{k=1}^{u(n)} \prod_{j_k \neq i_k} \frac{x_k - j_k}{i_k - j_k}$. We will show that the coefficients of $b_{i_1,\ldots,i_{u(n)}}$ and f_n are definable in $\mathsf{CH/poly}$. The conclusion of the theorem will then follow from Lemma 4.

In order to show that the coefficients of $b_{i_1,\ldots,i_{u(n)}}$ are definable in $\mathsf{CH/poly}$, we note that the polynomial $p_{i_1,\ldots,i_n}$ and the sequence $a(n)$ of Lemma 6 satisfy the relation

$$b_{i_1,\ldots,i_{u(n)}}(\bar{x}) = a(u(n))^{-u(n)} f_n(i_1, \ldots, i_{u(n)}) p_{i_1,\ldots,i_{u(n)}}(\bar{x}).$$

By Lemma 6, the coefficients of $p_{i_1,\ldots,i_{u(n)}}(\bar{x})$ are definable in CH. By hypothesis, (f_n) can be evaluated in $\mathsf{CH/poly}$ at integer points. This implies by Lemma 1 that $f_n(i_1, \ldots, i_{u(n)})$ is definable in $\mathsf{CH/poly}$. This is also the case of the product $f_n(i_1, \ldots, i_{u(n)}) p_{i_1,\ldots,i_{u(n)}}(\bar{x})$ by Theorem 2. Now, the same theorem enables us to divide by $a(u(n))^{u(n)}$, thereby showing that the coefficients of $b_{i_1,\ldots,i_{u(n)}}(\bar{x})$ are definable in $\mathsf{CH/poly}$. It then follows from (1) and another application of Theorem 2 that the coefficients of f_n are definable in $\mathsf{CH/poly}$. Therefore by Lemma 4, $(f_n) \in \mathsf{VP}^0_{\mathsf{nb}}$ under the hypothesis $\mathsf{VP}^0 = \mathsf{VNP}^0$. □

We now derive some consequences of Theorem 3.

Theorem 4. *Let $(f_n(\bar{x}, \bar{\epsilon})) \in \mathsf{VP}^0_{\mathsf{nb}}$. Let*

$$g_n(\bar{x}) = \sum_{\bar{\epsilon}} f_n(\bar{x}, \bar{\epsilon}) \text{ and } h_n(\bar{x}) = \prod_{\bar{\epsilon}} f_n(\bar{x}, \bar{\epsilon}).$$

If $\mathsf{VNP}^0 = \mathsf{VP}^0$ then (g_n) and (h_n) are in $\mathsf{VP}^0_{\mathsf{nb}}$.

Proof. By Corollary 1 (f_n) can be evaluated in $\mathsf{CH/poly}$ at integer points. Now, using Lemma 1 before and after the first part of Theorem 2 shows that (g_n) and (h_n) can also be evaluated in $\mathsf{CH/poly}$ at integer points. The result then follows by Theorem 3. □

The following is now immediate.

Theorem 5. *The hypothesis $\mathsf{VP}^0 = \mathsf{VNP}^0$ implies that $\mathsf{VP}^0_{\mathsf{nb}} = \mathsf{VNP}^0_{\mathsf{nb}}$ and $\mathsf{VP}^0_{\mathsf{nb}} = \mathsf{V\Pi P}^0$.*

Remark 4. It is not clear whether the converse of the first implication in Theorem 5 ($\mathsf{VP}^0 = \mathsf{VNP}^0 \Longrightarrow \mathsf{VP}^0_{\mathsf{nb}} = \mathsf{VNP}^0_{\mathsf{nb}}$) holds true. This is related to the issue of large constants in arithmetic circuits: it seems difficult to rule out the possibility that some polynomial family in VNP^0 (for instance, the permanent or the hamiltonian) does not lie in VP^0 but is still computable by polynomial-size arithmetic circuits using integer constants of exponential bit size.

The converse does hold if arbitrary constants are allowed: we indeed have $\mathsf{VP}_{\mathsf{nb}} = \mathsf{VNP}_{\mathsf{nb}} \Longrightarrow \mathsf{VP} = \mathsf{VNP}$. But in this non-constant-free context, it is not clear whether $\mathsf{VP} = \mathsf{VNP}$ unconditionally implies $\mathsf{VP}_{\mathsf{nb}} = \mathsf{VNP}_{\mathsf{nb}}$: indeed, in this context the generalized Riemann hypothesis would be needed to make the proof of Lemma 2 work (see [6] for details).

As mentioned in the introduction, another corollary concerns a transfer theorem with classes of algebraic complexity in the BSS model. Blum, Shub and Smale [2,3] have defined the classes P and NP over the real and complex fields. It was extended to arbitrary structures by Poizat [18]. Here we use nonuniform versions of these classes, hence the notations $\mathbb{P}$ and $\mathbb{NP}$.

Theorem 7 below proves that, over a field of characteristic zero, if we separate (the nonuniform versions of) P and NP thanks to a "simple" NP problem, then we separate (the constant-free versions of) VP and VNP. The class of "simple" problems here is NP where the multiplication is not allowed, i.e., the only operations are $+, -$ and $=$. It contains in particular Twenty Questions and Subset Sum. We will need a result from [13]:

Theorem 6. *Let K be a field of characteristic zero. If $\mathsf{VP}^0_{\mathsf{nb}} = \mathsf{V\Pi P}^0$ then $\mathbb{NP}_{(K,+,-,=)} \subseteq \mathbb{P}_{(K,+,-,\times,=)}$.*

By Theorem 5, the following is immediate.

Theorem 7. *Let K be a field of characteristic zero. If $\mathsf{VP}^0 = \mathsf{VNP}^0$ then $\mathbb{NP}_{(K,+,-,=)} \subseteq \mathbb{P}_{(K,+,-,\times,=)}$.*

At last, as a corollary of Theorem 3 again, we obtain the following result concerning question (*), suggesting that it will be hard to refute. As pointed out in the introduction, this result does not give any evidence concerning the answer to question (*) since the separation $\mathsf{VP}^0 \neq \mathsf{VNP}^0$ is very likely to be true.

Theorem 8. *If question (*) has a negative answer then $\mathsf{VP}^0 \neq \mathsf{VNP}^0$. More precisely, let (f_n) be a family of multivariate polynomials which can be evaluated in polynomial time at integer points (in the sense of Remark 3). If $\mathsf{VP}^0 = \mathsf{VNP}^0$ then $(f_n) \in \mathsf{VP}^0_{\mathsf{nb}}$.*

3.4 Arithmetic Circuits with Constants

In this section we investigate another interpretation of question (*): we still consider polynomials with integer coefficients, but we allow rational constants in our circuits (it turns out that the constant $1/2$ plays a special role due to its appearance in the completeness proof for the permanent). The hypothesis that

question (*) has a negative answer is then stronger, and we obtain a stronger conclusion than in Theorem 8. Namely, we can conclude that $\tau(\mathrm{PER}_n) \neq n^{O(1)}$ instead of $\mathsf{VNP}^0 \neq \mathsf{VP}^0$ (see Remark 1). We recall that τ, the constant-free arithmetic circuit complexity of a polynomial, is defined in Section 2.1.

Theorem 9. *As explained above, we consider here polynomials with integer coefficients but circuits with rational constants. If question (*) has a negative answer, then $\tau(\mathrm{PER}_n)$ is not polynomially bounded.*

More precisely, let (f_n) be a family of multivariate polynomials which can be evaluated in polynomial time at integer points (in the sense of Remark 3). If $\tau(\mathrm{PER}_n)$ is polynomially bounded, (f_n) can be evaluated by a family of polynomial-size arithmetic circuits that use only the constant $1/2$.

It is easy to see that Theorem 9 follows from a slight modification of the different lemmas above. Lemma 2 is replaced by the following stronger lemma, from [6].

Lemma 7. *If $\tau(\mathrm{PER}_n) = n^{O(1)}$ then $\mathsf{CH/poly} = \mathsf{P/poly}$.*

Then Lemma 4 is replaced by the following result, whose proof relies on an inspection of Valiant's proof [20] of VNP-completeness of the permanent, see [6].

Lemma 8. *Let*

$$f_n(x_1, \ldots, x_n) = \sum_{\alpha^{(1)}, \ldots, \alpha^{(n)}} a(n, \alpha^{(1)}, \ldots, \alpha^{(n)}) x_1^{\alpha^{(1)}} \cdots x_n^{\alpha^{(n)}},$$

where the integers $\alpha^{(i)}$ range from 0 to $2^n - 1$ and $a(n, \alpha^{(1)}, \ldots, \alpha^{(n)})$ is a sequence of integers of absolute value $< 2^{2^n}$ definable in $\mathsf{CH/poly}$.

If $\tau(\mathrm{PER}_n) = n^{O(1)}$ then there exists a polynomial $p(n)$ such that $\tau(2^{p(n)} f_n) = n^{O(1)}$.

Finally, Theorem 3 becomes the following.

Lemma 9. *Let $(f_n(x_1, \ldots, x_{u(n)}))$ be a family of multivariate polynomials (with integer coefficients). Suppose (f_n) can be evaluated in $\mathsf{CH/poly}$ at integer points. If $\tau(\mathrm{PER}_n) = n^{O(1)}$ then there exists a polynomial $p(n)$ such that $\tau(2^{p(n)} f_n) = n^{O(1)}$.*

Theorem 9 follows since the coefficient $2^{p(n)}$ can be cancelled by multiplying by the constant $2^{-p(n)}$, which can be computed from scratch from the constant $1/2$.

Acknowledgments

We would like to thank Erich Kaltofen and Christos Papadimitriou for sharing their thoughts on question (*).

References

1. E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. Bro Miltersen. On the complexity of numerical analysis. In *IEEE Conference on Computational Complexity*, pages 331–339, 2006.
2. L. Blum, F. Cucker, M. Shub, and S. Smale. *Complexity and Real Computation*. Springer-Verlag, 1998.
3. L. Blum, M. Shub, and S. Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bulletin of the American Mathematical Society*, 21(1):1–46, 1989.
4. P. Bürgisser. *Completeness and Reduction in Algebraic Complexity Theory*. Number 7 in Algorithms and Computation in Mathematics. Springer, 2000.
5. P. Bürgisser. The complexity of factors of multivariate polynomials. *Foundations of Computational Mathematics*, 4(4):369–396, 2004.
6. P. Bürgisser. On defining integers in the counting hierarchy and proving lower bounds in algebraic complexity. In *Proc. STACS 2007*, pages 133–144, 2007. Full version: ECCC Report No. 113, August 2006.
7. P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic Complexity Theory*. Springer, 1997.
8. H. Fournier and P. Koiran. Are lower bounds easier over the reals? In *Proc. 30th ACM Symposium on Theory of Computing*, pages 507–513, 1998.
9. H. Fournier and P. Koiran. Lower bounds are not easier over the reals: Inside PH. In *Proc. 27th International Colloquium on Automata, Languages and Programming*, volume 1853 of *Lecture Notes in Computer Science*, pages 832–843. Springer, 2000.
10. E. Kaltofen and G. Villard. On the complexity of computing determinants. Computational Complexity, 13(3-4):91-130, 2004.
11. R. Karp and R. Lipton. Turing machines that take advice. *L'Enseignement Mathématique*, 28:191–209, 1982.
12. P. Koiran. Valiant's model and the cost of computing integers. *Computational Complexity*, 13(3-4):131–146, 2004.
13. P. Koiran and S. Perifel. Valiant's model: from exponential sums to exponential products. In *Proc. MFCS 2006*, volume 4162 of *Lecture Notes in Computer Science*, pages 596–607. Springer-Verlag, 2006.
14. P. Koiran and S. Perifel. VPSACE and a transfer theorem over the complex field. In *Proc. MFCS 2007*, volume 4708 of *Lecture Notes in Computer Science*, pages 359-370, 2007.
15. P. Koiran and S. Perifel. VPSACE and a transfer theorem over the reals. In *Proc. STACS 2007*, volume 4393 of *Lecture Notes in Computer Science*, pages 417–428. Springer-Verlag, 2007. long version: http://prunel.ccsd.cnrs.fr/ensl-00103018.
16. G. Malod. The complexity of polynomials and their coefficient functions. In *Proc. 22nd IEEE Conference on Computational Cmplexity*, pages 193–204, 2007.
17. Guillaume Malod. *Polynômes et coefficients*. PhD thesis, Université Claude Bernard Lyon 1, July 2003. Available from http://tel.archives-ouvertes.fr/tel-00087399.
18. B. Poizat. *Les petits cailloux*. Aléas, 1995.
19. M. Shub and S. Smale. On the intractability of Hilbert's Nullstellensatz and an algebraic version of "P=NP". *Duke Mathematical Journal*, 81(1):47–54, 1995.
20. L. G. Valiant. Completeness classes in algebra. In *Proc. 11th ACM Symposium on Theory of Computing*, pages 249–261, 1979.
21. K. W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Inform.*, 23(3):325–356, 1986.