Real Interactive Proofs for VPSPACE *

Martijn Baartse[1] and Klaus Meer[2]

1 **Computer Science Institute, BTU Cottbus-Senftenberg**
Platz der Deutschen Einheit 1
D-03046 Cottbus, Germany
2 **Computer Science Institute, BTU Cottbus-Senftenberg**
Platz der Deutschen Einheit 1
D-03046 Cottbus, Germany
meer@b-tu.de

Abstract

We study interactive proofs in the framework of real number complexity as introduced by Blum, Shub, and Smale. The ultimate goal is to give a Shamir like characterization of the real counterpart $\mathrm{IP}_{\mathbb{R}}$ of classical IP. Whereas classically Shamir's result implies IP = PSPACE = PAT = PAR, in our framework a major difficulty arises from the fact that in contrast to Turing complexity theory the real number classes $\mathrm{PAR}_{\mathbb{R}}$ and $\mathrm{PAT}_{\mathbb{R}}$ differ and space resources considered alone are not meaningful. It is not obvious to see whether $\mathrm{IP}_{\mathbb{R}}$ is characterized by one of them - and if so by which.

In recent work the present authors established an upper bound $\mathrm{IP}_{\mathbb{R}} \subseteq \mathrm{MA}\exists\mathbb{R}$, where $\mathrm{MA}\exists\mathbb{R}$ is a complexity class satisfying $\mathrm{PAR}_{\mathbb{R}} \subsetneq \mathrm{MA}\exists\mathbb{R} \subseteq \mathrm{PAT}_{\mathbb{R}}$ and conjectured to be different from $\mathrm{PAT}_{\mathbb{R}}$. The goal of the present paper is to complement this result and to prove interesting lower bounds for $\mathrm{IP}_{\mathbb{R}}$. More precisely, we design interactive real protocols for a large class of functions introduced by Koiran and Perifel and denoted by $\mathrm{UniformVPSPACE}^0$. As consequence, we show $\mathrm{PAR}_{\mathbb{R}} \subseteq \mathrm{IP}_{\mathbb{R}}$, which in particular implies co-$\mathrm{NP}_{\mathbb{R}} \subseteq \mathrm{IP}_{\mathbb{R}}$, and $\mathrm{P}_{\mathbb{R}}^{Res} \subseteq \mathrm{IP}_{\mathbb{R}}$, where *Res* denotes certain multivariate Resultant polynomials.

Our proof techniques are guided by the question in how far Shamir's classical proof can be used as well in the real number setting. Towards this aim results by Koiran and Perifel on $\mathrm{UniformVPSPACE}^0$ are extremely helpful.

1998 ACM Subject Classification F.1.1 Models of Computation; F 1.2 Modes of Computation; F 1.3 Complexity Measures and Classes

Keywords and phrases Interactive Proofs, real number computation, Shamir's theorem

Digital Object Identifier 10.4230/LIPIcs.MFCS.2016.14

1 Introduction

Shamir's famous theorem [18] characterizes the set IP of languages that can be verified by an interactive protocol performed between a polynomial time probabilistic verifier and a prover of unlimited power as being equal to PSPACE.

Around the same time of Shamir's result Blum, Shub, and Smale [5] introduced a model of computation over the real numbers (for short: BSS model in the sequel) and a complexity theory for it. Since then, among other things one line of activity in research on the BSS model was to figure out whether and by what reasons important classical results in Turing

* Both authors were partially supported under projects ME 1424/7-1 and ME 1424/7-2 by the Deutsche Forschungsgemeinschaft DFG. We gratefully acknowledge the support.

complexity theory hold as well over other computational structures. Following this line, in the present paper we are interested in deriving results about the real class $\mathrm{IP}_{\mathbb{R}}$ of languages verifiable by an interactive protocol over the reals; for precise definitions see next section.

It is well known that over the reals complexity classes that are classically defined or characterized using space resources turn out to have a more subtle relation among each other than they do classically. Taken alone, space resources have no meaning at all; each decision problem can be decided in linear space using an elementary coding trick [15]. As consequence, for many equivalent characterizations especially of the class PSPACE in classical complexity it is unclear what they should become in the real number framework. Recall that PAR, PSPACE, PAT, and IP, denoting the classes of languages acceptable in parallel polynomial time with exponentially many processors, in polynomial space, in polynomial alternating time, and by interactive proofs, respectively, all are the same in Turing complexity; see the textbook [1] for references and proofs. In contrast, over $\mathbb{R}$ it is known that the first three classes mentioned above satisfy $\mathrm{PAR}_{\mathbb{R}} \subsetneq \mathrm{PSPACE}_{\mathbb{R}} \subseteq \mathrm{PAT}_{\mathbb{R}}$, where $\mathrm{PSPACE}_{\mathbb{R}}$ denotes the class of real decision problems decidable by an algorithm using both exponential time and polynomial space and the other two classes are defined by extending the classical definitions straightforwardly, see [7, 4]. As a consequence, if a new class like $\mathrm{IP}_{\mathbb{R}}$ is studied which classically gives yet another characterization of PSPACE via Shamir's result, it is not obvious where it can be located over the reals.

It is straightforward to see from the definitions that $\mathrm{NP}_{\mathbb{R}} \subseteq \mathrm{IP}_{\mathbb{R}}$. But already the inclusion co-$\mathrm{NP}_{\mathbb{R}} \subseteq \mathrm{IP}_{\mathbb{R}}$ is far from being obvious. Shamir's proof designs an interactive protocol for the PSPACE-complete Quantified Boolean Formulas problem QBF roughly as follows: First, the problem is arithmetized in form of giving a short formula representing an algebraic expression with exponentially many terms. This expression replaces Boolean quantifiers in the original formula by sums and products, respectively, in which the quantified variables run through all values in $\{0, 1\}$. The goal of the communication protocol is to evaluate this expression interactively and randomly. Towards this aim, certain canonical univariate polynomials are attached to this expression by eliminating one after the other the leftmost operator $\sum_{x_i=0}^{1}$ or $\prod_{x_i=0}^{1}$ in it. This results in a polynomial in x_i of polynomial degree whose value in a random point is verified interactively. Though a real variant of QBF can easily be defined and is complete for $\mathrm{PAT}_{\mathbb{R}}$, Shamir's proof cannot be transformed. An arithmetization of quantifiers ranging over the reals is not possible in the same way and immediately destroys the hope of following the above approach; however, see [2] for some attempts dealing with more restricted real decision problems.

In this paper we shall therefore follow a different approach. We still are guided by the question how far Shamir's technique might lead. Instead of dealing directly with an arithmetization of computationally hard real number problems we rely on results in BSS theory that figure out how much can be done using certain oracles in real computations. Such results have been obtained by several authors; crucial for us is work by Koiran and Perifel [12]. Therein a relatively huge class $\mathrm{UniformVPSPACE}^0$ of families of polynomial functions is introduced and studied. It is a kind of uniform extension of Valiant's class VNP and covers families of polynomials with exponential degree and integer coefficients computable in PSPACE. Crucial for us will be two observations: Firstly, verifying whether the result $f_n(x)$ of a member f_n of such a family on input x equals a given y can be done within the resources of $\mathrm{IP}_{\mathbb{R}}$. This result is obtained by showing that the discrete techniques used by Shamir are sufficient to deal with $\mathrm{UniformVPSPACE}^0$. This is important in order to circumvent the above mentioned problems. Secondly, as shown in [12] $\mathrm{UniformVPSPACE}^0$ is powerful enough to deal with interesting real number problems in classes like co-$\mathrm{NP}_{\mathbb{R}}$

and larger via polynomial time BSS algorithms that access an oracle for function families in UniformVPSPACE^0. That way, a real verifier can be designed that is able to deal with problems even from class $\mathrm{PAR}_\mathbb{R}$. This leads to our main result stating $\mathrm{PAR}_\mathbb{R} \subseteq \mathrm{IP}_\mathbb{R}$. Taking into account previous results this will locate $\mathrm{IP}_\mathbb{R}$ much better within certain real number classes than it has been possible so far. It shows as well that also over the reals $\mathrm{IP}_\mathbb{R}$ under standard complexity theoretic assumptions is considerably larger than $\mathrm{NP}_\mathbb{R}$.

1.1 Previous results

Before summarizing previous results let us recall the formal definition of algebraic circuits and the class $\mathrm{PAR}_\mathbb{R}$. An algebraic circuit is a connected and directed acyclic graph having nodes either of indegree 0 (input nodes) , 1 (test nodes) or 2 (computation nodes). Nodes of indegree 2 are labelled with one of the operations $+, -, \bullet$, nodes of indegree 1 are labelled with $'\geq 0?'$ A circuit has one output node of outdegree 0. The size of a circuit is the number of its nodes, its depth is the length of the longest path from an input node to the output node. A circuit with n input nodes computes in the straightforward manner a function from $\mathbb{R}^n \mapsto \mathbb{R}$; on input $x \in \mathbb{R}^n$ it propagates values along the labels of nodes in the obvious way. The value of a test node is either 1 or 0, depending on whether its incoming value is ≥ 0 or not. We only consider circuits with one output node which is a test node. Thus, our circuits compute characteristic functions.

▶ **Definition 1.** A probem $L \subseteq \mathbb{R}^\infty := \bigsqcup_{i\geq 1} \mathbb{R}^i$ belongs to class $\mathrm{PAR}_\mathbb{R}$ iff there exists a family $\{C_n\}_{n\in\mathbb{N}}$ of algebraic circuits of depth polynomially bounded in n, a constant $s \in \mathbb{N}$, and a vector $c \in \mathbb{R}^s$ of real constants such that

- **(i)** each C_n has $n + s$ input nodes;
- **(ii)** for all $n \in \mathbb{N}$ the circuit C_n computes the characteristic function of $L \cap \mathbb{R}^n$, when the last s input nodes are assigned the constant values from c, i.e., $x \in L \cap \mathbb{R}^n \Leftrightarrow C_n(x, c) = 1$;
- **(iii)** the family $\{C_n\}_n$ is PSPACE uniform, i.e., there is a Turing machine working in polynomial space which for each $n \in \mathbb{N}$ computes a descripition of C_n.

If no constant vector c is involved we obtain the constant free version of $\mathrm{PAR}_\mathbb{R}$ denoted by $\mathrm{PAR}^0_\mathbb{R}$.

The above definition basically is from [6]. There are equivalent ones explicitly involving BSS machines [4]. The vector c used above then plays the role of the machine constants of such a BSS algorithm.

There has so far not been much work on interactive proofs in the BSS model. It started with a paper by Ivanov and de Rougemont [11] where Shamir's result was shown to hold as well in the additive real number model. In this model, multiplications are not allowed. The interaction was restricted to exchanging bits. One side result in this paper was that the classes $\mathrm{PAR}_\mathbb{R}$ and $\mathrm{IP}_\mathbb{R}$ are provably different [1].

In [2] the real class $\mathrm{IP}_\mathbb{R}$ was introduced. The above mentioned problem in $\mathrm{IP}_\mathbb{R} \setminus \mathrm{PAR}_\mathbb{R}$ from [11] is one that can be formalized by using polynomially alternating existential quantifications over the reals and arbitrary Boolean quantifiers. It is therefore kind of natural trying to relate $\mathrm{IP}_\mathbb{R}$ to another real complexity class $\mathrm{MA}\exists\mathbb{R}$ introduced and studied by Cucker and

[1] in [11] $\mathrm{IP}_\mathbb{R}$ is not introduced formally, but it is shown that there exists a problem not in $\mathrm{PAR}_\mathbb{R}$ but within a class that easily is seen to be a subclass of $\mathrm{IP}_\mathbb{R}$ as defined below. However, this example does NOT show $\mathrm{PAR}_\mathbb{R} \subseteq \mathrm{IP}_\mathbb{R}$

Briquel in [8]. It is a class which does not make sense over finite alphabets and can be located between $\mathrm{PSPACE}_{\mathbb{R}}$ and $\mathrm{PAT}_{\mathbb{R}}$.

▶ **Definition 2.** ([8]) The class $\mathrm{MA}\exists\mathbb{R}$ consists of all decision problems $A \subseteq \mathbb{R}^\infty$ for which there exists a problem $L \in \mathrm{P}_{\mathbb{R}}$ together with a polynomial p such that an $x \in \mathbb{R}^\infty$ belongs to A if and only if the following formula holds:

$$\forall_B z_1 \exists_{\mathbb{R}} y_1 \ldots \forall_B z_{p(|x|)} \exists_{\mathbb{R}} y_{p(|x|)} (x, y, z) \in L \ .$$

Here, $y = (y_1, \ldots, y_{p(|x|)})$ and $z = (z_1, \ldots, z_{p(|x|)})$. The subscripts $B, \mathbb{R}$ for the quantifiers indicate whether a quantified variable ranges over $B := \{0, 1\}$ or $\mathbb{R}$, respectively.

Note that $\mathrm{MA}\exists\mathbb{R}$ contains the real polynomial hierarchy $\mathrm{PH}_{\mathbb{R}}$, i.e., problems with a fixed number of both existential and universal real quantifiers and even its supclasses $\mathrm{PAR}_{\mathbb{R}} \subsetneq \mathrm{PSPACE}_{\mathbb{R}}$, see [8]. It is not known to capture $\mathrm{PAT}_{\mathbb{R}}$; however, $\mathrm{MA}\exists\mathbb{R}$ reflects the special structure of quantifiers mentioned above in relation to the problem that witnesses $\mathrm{PAR}_{\mathbb{R}} \neq \mathrm{IP}_{\mathbb{R}}$. In fact, we have

▶ **Theorem 3.** *([2])* $\mathrm{IP}_{\mathbb{R}} \subseteq \mathrm{MA}\exists\mathbb{R}$.

The main purpose of this paper is to complement this upper bound result by obtaining non-trivial lower bounds for $\mathrm{IP}_{\mathbb{R}}$ as well. In Section 2 we introduce the main concepts and define $\mathrm{IP}_{\mathbb{R}}$ and $\mathrm{UniformVPSPACE}^0$. Section 3 gives the result showing that all function families in $\mathrm{UniformVPSPACE}^0$ can be evaluated interactively within $\mathrm{IP}_{\mathbb{R}}$. The final section applies this theorem to prove our main result, namely that some further real complexity classes are included in $\mathrm{IP}_{\mathbb{R}}$, $\mathrm{PAR}_{\mathbb{R}}$ being the most interesting among them.

One remark concerning the contribution of this paper seems in charge. There is not one big new technical result presented here. Different variants of Theorem 10 below have been known before, see [16] and [14]; we present the proof again because of self-containment and because reformulating the results in the cited papers in the way we need them would not save much space. We believe the value of the present paper is the combination of several pieces of previous works in a way that has not been done so far. This in particular refers to using the class $\mathrm{UniformVPSPACE}^0$ in relation with interactive proofs and realizing that discrete techniques are sufficient to deal with it in a certain sense. That way, we obtain the strongest result on real interactive proofs so far. This result in our opinion definitely is interesting by itself and means significant progress concerning the question how a seminal result of classical complexity theory looks like in the real number framework.

2 Basic notions and results

In this section we recall the definitions of the two main complexity classes considered in this paper, namely $\mathrm{IP}_{\mathbb{R}}$ as well as $\mathrm{UniformVPSPACE}^0$. The former was defined in [2], the latter in [12].

2.1 The model for interaction and some variants

As underlying algorithm model we work in the Blum-Shub-Smale BSS model over $\mathbb{R}$ [4, 5]. Decision problems considered in this model are subsets of $\mathbb{R}^\infty := \bigsqcup_{i \geq 1} \mathbb{R}^i$. The model allows to perform the basic arithmetic operations $+, -, \bullet$ and test instructions of the form 'is $x \geq 0$?' at unit cost; an $x \in \mathbb{R}^i$ has (algebraic) size i. Below, in addition we allow both the verifier and the prover to exchange real numbers at unit cost.

The prover P is a BSS machine unlimited in computational power. The verifier V is a randomized polynomial time BSS algorithm. It is important to point out that randomization (still) is discrete, i.e., V generates a sequence of random bits $r = (r_1, r_2, \ldots)$ during its computation. The computation proceeds as follows:

- Given an input $x \in \mathbb{R}^n$ of size $|x| = n$ and (some of) the random bits of r the verifier V computes a real $V(x, r) =: w_1 \in \mathbb{R}$ and sends it to P;
- using x and w_1 the prover P sends a real $P(x, w_1) =: p_1 \in \mathbb{R}$ back to V;
- let $(w_1, p_1, w_2, \ldots, p_i)$ denote the information sent forth and back after i rounds, then in round $i+1$ V computes a real $V(x, r, w_1, p_1, \ldots, p_i) =: w_{i+1}$ and sends it to P; P then computes a real $P(x, w_1, p_1, \ldots, p_i, w_{i+1}) =: p_{i+1}$ and sends it to V;
- the communication halts after a polynomial number $m = poly(|x|)$ of rounds. Then V computes its final result $V(x, r, w_1, \ldots, p_{m-1}) =: w_m \in \{0, 1\}$ representing its decision to reject or accept the input, respectively.

We denote the result of an interaction between V and P on input x and V using r as random string by $(P, V)(x, r)$. All computations by V have to be finished in (real) polynomial time; thus, in particular the number of random bits generated as well as the number of rounds is polynomially bounded in the size $|x|$ of x.

▶ **Definition 4.** a) A language $L \subseteq \mathbb{R}^\infty$ has an interactive protocol if there exists a polynomial time randomized verifier V such that

(i) if $x \in L$ there exists a prover P such that $\Pr_{r \in \{0,1\}^*} \{(P, V)(x, r) = 1\} \geq \frac{2}{3}$ and

(ii) if $x \notin L$, then for all provers P it holds $\Pr_{r \in \{0,1\}^*} \{(P, V)(x, r) = 1\} \leq \frac{1}{3}$.

Above, the length of r can be polynomially bounded in the length of x.

b) The class $\mathrm{IP}_\mathbb{R}$ contains all $L \subseteq \mathbb{R}^\infty$ which have an interactive protocol.

In the above definitions private coins are used, i.e., we do not allow the prover to know the outcome of V's random choices. One could change this requirement and let the verifier only send the random bits; what the verifier computes out of it then could be as well computed by the allmighty prover. Such protocols are called Arthur-Merlin protocols. Another modification uses one-sided instead of two-sided error in the acceptance condition for V. Then, for $x \in L$ there must be a prover such that V accepts with probability 1. For sake of completeness we show below that these modifications do not change the class $\mathrm{IP}_\mathbb{R}$. Both the result and its proof are the same as in the Turing model.

▶ **Definition 5.** The class $\widetilde{\mathrm{IP}_\mathbb{R}}$ is defined similar to $\mathrm{IP}_\mathbb{R}$, but with the following modifications:

(i) The verifier V uses *public coins*, i.e., it only sends the random bits r generated in each round to P.

(ii) The verifier accepts with one-sided error: A language L is in $\widetilde{\mathrm{IP}_\mathbb{R}}$ iff there is a verifier V such that $\forall x \in L$ there exists a prover P such that $\Pr_{r \in \{0,1\}^*} \{(P, V)(x) = 1\} = 1$. And $\forall x \notin L$ $\forall P$ it holds $\Pr_{r \in \{0,1\}^*} \{(P, V)(x) = 1\} \leq \frac{1}{3}$.

▶ Proposition 6. $\mathrm{IP}_\mathbb{R} = \widetilde{\mathrm{IP}_\mathbb{R}}$.

Proof. [2] The inclusion $\widetilde{\mathrm{IP}_\mathbb{R}} \subseteq \mathrm{IP}_\mathbb{R}$ being clear let $L \in \mathrm{IP}_\mathbb{R}$ and let V be a corresponding

[2] The fact that public coins are as powerful as private ones was first shown in [10]. An easier proof that also replaces two-sided by one-sided error was given by J. Kilian. We could not figure out whether the proof was published, it is however refered to in [9]. For sake of completeness we follow this proof below.

verifier accepting L with private coins and two-sided error. Without loss of generality in each communication round V generates one random bit. A new verifier $\tilde{V}$ using public coins and accepting L with one-sided error is obtained as follows. On input x $\tilde{V}$ expects from a prover to provide information about the communication between V and an optimal prover for it. More precisely, define a protocol tree T coding the protocol between V and an optimal prover on x as follows. An edge in T represents one communication round after a coin toss has been made by V. Since one bit is generated in each round T is binary, the outgoing edges of each node represent the communication for results 0 and 1, respectively. For m communication rounds the probability that V accepts x is $1/2^m$ times the number of accepting paths.

In its communication on x with a prover the new verifier $\tilde{V}$ descends a path of T top down as follows. Let r be the current node of T traversed, r_1, r_2 its left and right child, respectively. $\tilde{V}$ asks the prover for the numbers R, R_1, R_2 of accepting paths the communication between an optimal prover and V would generate when starting in r, r_1, and r_2, respectively. If r is the root and the number reported by the prover is $< \frac{2}{3} \cdot 2^m$ the verifier rejects right away. For an arbitrary node r it checks whether $R = R_1 + R_2$ and rejects if the equation is violated. Otherwise, $\tilde{V}$ moves to r_i with probability $\frac{R_i}{R}$ for $i = 1, 2$. The protocol continues until a leaf is reached. If the path traversed is accepting for the protocol followed by V, then $\tilde{V}$ accepts, otherwise it rejects.

$\tilde{V}$ obviously uses public coins; its random decisions are known to the prover because it is informed about the child of r that is picked by $\tilde{V}$. To see that $\tilde{V}$ accepts L with one-sided error first note that for $x \in L$ an optimal prover will always give the correct numbers R, R_1, R_2 and thus $\tilde{V}$ ends with probability 1 in an accepting leaf because there must exist such a leaf in T. Let us then assume $x \notin L$ and let P be an arbitrary prover.

Claim: For each node r in T the following holds: if there are R accepting paths from r on for an optimal prover and V, but the current prover P claims there are $R' > R$ accepting paths, then $\tilde{V}$ will realize an error with probability $\geq 1 - \frac{R}{R'}$.

Proof of claim: By induction on the height h of r. Let $h = 1$ and let r have children r_1, r_2 being leaves. If $R = 0$, then no matter whether $R' = 1$ or $R' = 2$ both paths are rejecting and $\tilde{V}$ realizes it with probability $1 = 1 - \frac{0}{R'}$. If $R = 1$ then $R' = 2$ and $\tilde{V}$ chooses the rejecting path with probability $\frac{1}{2} = 1 - \frac{1}{2}$.

For arbitrary h let the correct number of accepting paths from r, r_1, r_2 on be R, R_1, R_2, respectively. Let R', R'_1, R'_2 denote the (larger) numbers claimed by P. According to the induction hypothesis if the protocol starts in r_i the verifier $\tilde{V}$ realizes an error with probability $\geq (1 - \frac{R_i}{R'_i}), i = 1, 2$. In node r it chooses the left child with probability $\frac{R'_1}{R'}$ and the right one with probability $\frac{R'_2}{R'}$. The error probability thus is $(1 - \frac{R_1}{R'_1}) \cdot \frac{R'_1}{R'} + (1 - \frac{R_2}{R'_2}) \cdot \frac{R'_2}{R'} = 1 - \frac{R}{R'}$.

Finally, each $x \notin L$ is rejected by the original verifier V and any prover with probability $\geq \frac{2}{3}$, i.e., at most $\frac{1}{3}$ of all paths starting at the root of T are accepting. $\tilde{V}$ either rejects directly if the prover claims $R < \frac{2}{3} \cdot 2^m$ accepting paths or it rejects with probability $\geq 1 - \frac{1/3}{2/3} = \frac{1}{2}$ by the claim. Running the protocol for $\tilde{V}$ once more increases this probability to at least $\frac{3}{4} > \frac{2}{3}$ as required. □

2.2 UniformVPSPACE^0

The following class of functions was introduced and studied by Koiran and Perifel in [12] and kind of generalizes the famous Valiant class VNP. Informally, it consists of uniform families of polynomials with integer coefficients which depend on polynomially many variables, potentially an exponential degree and whose coefficients can be computed in PSPACE. Though originally defined over arbitrary fields we restrict ourselves to the real numbers.

▶ **Definition 7.** (see [12])

(a) A family $\{f_n\}_{n\in\mathbb{N}}$ of real polynomials belongs to UniformVPSPACE^0 iff the following conditions are satisfied: There exists a polynomial p such that

- **(i)** each f_n depends on $u(n)$ variables, where $u(n)$ is bounded from above by $p(n)$;
- **(ii)** the total degree of each f_n is bounded by $2^{p(n)}$;
- **(iii)** the coefficients of each f_n are integers which are bounded in their bitsize by $2^{p(n)}-1$;
- **(iv)** the coefficient function a is PSPACE computable. More precisely, a gets as arguments triples (n,α,i), where $n\in\mathbb{N}$ is given in unary, $\alpha=(\alpha_1,\ldots,\alpha_{u(n)})$ is a list of binary numbers representing a monomial $x^\alpha = x_1^{\alpha_1}\cdot x_2^{\alpha_2}\cdot\ldots\cdot x_{u(n)}^{\alpha_{u(n)}}$, and i is a binary number. Then $a(n,\alpha,i)\in\{0,1\}$ gives the i-th bit of the coefficient of monomial x^α in f_n. In particular, the value $a(n,\alpha,0)$ gives the sign of this monomial. [3]

The functions f_n thus have the following representation:

$$f_n(x_1,\ldots,x_{u(n)}) = \sum_\alpha \left[(-1)^{a(n,\alpha,0)} \left(\sum_{i=1}^{2^{p(n)}} 2^{i-1} a(n,\alpha,i) \right) x^\alpha \right].$$

(b) A family $\{f_n\}_{n\in\mathbb{N}}$ of polynomials belongs to class UniformVPAR iff it can be computed by a PSPACE-uniform family of arithmetic circuits of polynomial depth, compare Definition 1. If the family is constant free we obtain class UniformVPAR^0.

Again, the superscript '0' indicates that a class is defined without involving additional real constants. It is relatively straightforward to see that both notions above characterize the same set of families:

▶ **Lemma 8.** *([12]) It holds* UniformVPSPACE^0 = UniformVPAR^0.

The result implies in particular that if a family of functions $\{f_n\}\in$ UniformVPAR is defined by a family of circuits using a constant vector $c\in\mathbb{R}^s$, then one obtains another family of functions $\{g_n\}\in$ UniformVPSPACE^0 such that for all $x\in\mathbb{R}^{u(n)}$ of suitable input size we have $g_n(x,c)=f_n(x)$. This will be needed below.

3 Lower bound for IP$_\mathbb{R}$

In this section we prove our main technical theorem. Basically it shows that function families in UniformVPSPACE^0 can be represented by certain formulas having a very particular structure. The latter strongly resembles the structure of formulas arising via arithmetization of discrete quantified boolean formulas as outlined in the introduction. Of course, the new kind of formulas involve real variables. The special structure obtained allows to verify the values of such functions in a way similar to Shamir's original interactive proof for the QBF problem. Using additional results about class UniformVPSPACE^0 then makes it possible to derive real interactive proofs for interesting real number problems, foremost for all problems in PAR$_\mathbb{R}$.

We start with the definition of these specially structured formulas.

▶ **Definition 9.** Let $x_1,x_2,\ldots$ be a countable set of variable symbols.

(a) A *binary polynomial formula* over the reals is a formula p which can be built in finitely many steps according to the following rules:

[3] Note that since a only attains values in $\{0,1\}$ it can be seen as decision problem and thus the PSPACE requirement makes sense.

(i) $p = 1$ and $p = x_i$ for $i = 1, 2, \ldots$ are binary polynomial formulas ;
(ii) if p_1, p_2 are binary polynomial formulas, then so are $p_1 + p_2, p_1 - p_2, p_1 \cdot p_2$;
(iii) if p is a binary polynomial formula depending freely on x_i, then both $\sum_{x_i \in \{0,1\}} p(\ldots, x_i, \ldots)$ and $\prod_{x_i \in \{0,1\}} p(\ldots, x_i, \ldots)$ are binary polynomial formulas (with x_i bounded by summation and multiplication, respectively).

All formulas - and no others - that can be obtained in finitely many steps applying the rules i) to iii) are binary polynomials formulas.

(b) The size of a binary polynomial formula is defined as the number of construction steps used in a) to generate it.

(c) A binary polynomial formula p in the canonical way represents a real polynomial function. It depends on the free variables, i.e., on those x_i that have been introduced via rule i) but have not been bound by a Boolean summation or multiplication applying rule iii).

The following theorem shows that families of functions in UniformVPSPACE^0 are basically the same as families of polynomials given via uniform families of binary polynomial formulas. Similar statements in different variants are already in [16] and [14]. We present the proof for sake of self-containment and because reformulating the results of those papers in the way we need them would likely not save much space.

▶ **Theorem 10.** *Let $\{f_n\}_n$ be a family of polynomial functions. Then $\{f_n\}_n$ belongs to class* UniformVPSPACE^0 *if and only if there exists a polynomial time Turing algorithm which on input $n \in \mathbb{N}$ (in unary) computes a binary polynomial formula p_n which represents f_n. By computing p_n we mean that the algorithm computes a scheme how to generate p_n according to the steps defined above.*

Proof. For the if-direction let $\{p_n\}_n$ be a family of binary polynomial formulas which are uniformly generated by a polynomial time Turing machine. Then it is easy to see that p_n can be computed by a PSPACE-uniform family of arithmetic circuits of polynomial depth. Since the formulas only involve the constant 1 the circuits are constant-free as well. The summation and multiplication operators in a formula can be simulated in parallel by the circuit, thus the polynomially many construction steps for the formula result in a polynomial depth for the circuit. It follows that the polynomial family $\{p_n\}_n$ belongs to class UniformVPAR^0. Lemma 8 now implies the 'if'-direction.

For the only-if direction, let a family $\{f_n\}_n \in$ UniformVPSPACE^0 be given and consider one of its members

$$f_n(x_1, \ldots, x_{u(n)}) = \sum_{\alpha} \left[(-1)^{a(n,\alpha,0)} \left(\sum_{i=1}^{2^{p(n)}} 2^{i-1} a(n, \alpha, i) \right) x^{\alpha} \right].$$

Without loss of generality we assume $u = p$. Our task is to show that the different parts in this representation can be rewritten in form of binary polynomial formulas.

Step 1: Let us start with constructing binary polynomial formulas for the numbers 2^{i-1}. To catch the necessary ideas we first give an unsuccessful approach: It is $2^{i-1} = \sum_{j_1=0}^{1} \sum_{j_2=0}^{1} \cdots \sum_{j_{i-1}=0}^{1} 1$, but the length of this binary formula is i. Since parameter i in the above sum for representing f_n is running from 1 to $2^{p(n)}$ the corresponding formula becomes too long. Instead, consider the binary representation of $i =: (i_1, \ldots, i_{p(n)})$. We define a binary polynomial formula for $G_1(i_1, \ldots, i_{p(n)}) := 2^{i-1}$. Its main building block is a formula for the

characteristic function

$$F_1(j_1,\ldots,j_{p(n)},i_1,\ldots,i_{p(n)}) = \begin{cases} 1 & \text{if } 0 \neq (j_1,\ldots,j_{p(n)}) < (i_1,\ldots,i_{p(n)}) \\ -1 & \text{if } 0 = i \\ 0 & \text{otherwise} \end{cases},$$

where the ordering $<$ is to be understood as ordering of the integers represented in binary by the corresponding tuples. Once a binary polynomial formula for F_1 is available one for G_1 is obtained via

$$G_1(i_1,\ldots,i_{p(n)}) = \prod_{j_1=0}^{1} \cdots \prod_{j_{p(n)}=0}^{1} \big(F_1(j_1,\ldots,j_{p(n)},i_1,\ldots,i_{p(n)})+1\big);$$

this follows from the definition of F_1 since the above product contributes a factor 2 for each $0 \neq j < i$, a factor 0 if $i = 0$ and a factor 1 in the other cases.

Binary polynomial formulas for the cases $i = 0$ and $j = 0$ are easily obtained. The order relation $(j_1,\ldots,j_{p(n)}) < (i_1,\ldots,i_{p(n)})$ can be expressed as

$$j_{p(n)} < i_{p(n)} \ \vee \ \big\{j_{p(n)} = i_{p(n)} \wedge j_{p(n-1)} < i_{p(n-1)}\big\} \ \vee \ldots \\ \big\{j_{p(n)} = i_{p(n)} \wedge \ldots \wedge j_2 = i_2 \wedge j_1 < i_1\big\}.$$

A binary polynomial formula for the characteristic function $y < z$ of comparing two single input bits is given by $z \cdot (z - y)$; and a formula for the above Boolean combination is obtained by combining two characteristic functions χ_1, χ_2 via $\chi_1 \cdot \chi_2$ for conjunctions and via $\chi_1 + \chi_2$ for disjunctions (note here that at most one clause becomes true). That way a binary polynomial formula representing G_1 is obtained. Its length clearly is polynomially bounded in n.

Step 2: Next, a binary polynomial formula for the function

$$G_\alpha(x_1,\ldots,x_{p(n)}) := x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdot \ldots \cdot x_{p(n)}^{\alpha_{p(n)}} = x^\alpha$$

for given α is derived as follows. First, consider a single factor, for example $x_1^{\alpha_1}$, and let the binary representation of α_1 be $(\alpha_{11},\alpha_{12},\ldots,\alpha_{1p(n)})$. Now for $p(n)$ variables $t := (t_1,\ldots,t_{p(n)})$ consider the binary polynomial formula

$$\chi(\alpha_1 = 0) + (1 - \chi(\alpha_1 = 0)) \cdot \left(x_1 \cdot \prod_{t_1=0}^{1} \cdots \prod_{t_{p(n)}=0}^{1} \big(F_1(t_1,\ldots,t_{p(n)},\alpha_{11},\ldots,\alpha_{1p(n)}) \cdot (x_1 - 1) + 1\big) \right).$$

Here, $\chi(\alpha_1 = 0)$ denotes a binary polynomial formula for the characteristic function of the condition $\alpha_1 = 0$. A short moment of reflection now shows that for $\alpha_1 = 0$ the above formula results in $x_1^0 = 1$; if $\alpha_1 > 0$, then for each integer $0 < t < \alpha_1$ a factor x_1 is contributed whereas for $t = 0$ and $t \geq \alpha_1 > 0$ a factor 1 is obtained. Thus, the formula represents $x_1^{\alpha_1}$.

Since each monomial in f_n has $p(n)$ variables, the above construction can be repeated $p(n)$ many times to obtain

$$G_\alpha(x_1,\ldots,x_{p(n)}) = \prod_{j=1}^{p(n)} \Bigg[\chi(\alpha_j = 0) + \\ (1 - \chi(\alpha_j = 0)) \cdot \left(x_j \prod_{t_1=0}^{1} \cdots \prod_{t_{p(n)}=0}^{1} \big(F_1(t_1,\ldots,t_{p(n)},\alpha_{j1},\ldots,\alpha_{jp(n)}) \cdot (x_j - 1) + 1\big) \right) \Bigg],$$

i.e., a binary polynomial formula for x^α of polynomial length. Note that in the above formula the first product results from applying a polynomial number of times construction step a),ii) of Definition 9, whereas the subsequent products result from step iii).

Step 3: The representation of the coefficients $a(n,\alpha,i)$ as binary polynomial formulas is based on PSPACE-completeness of the QBF problem, i.e., the question of deciding whether a quantified Boolean formula is true [19]. By assumption, computing $a(n,\alpha,i)$ can be done in PSPACE. Thus, for each n there exists a Boolean formula $\psi_n(\alpha,i) = \exists x_1 \forall x_2 \ldots Q_m x_m \phi(n,\alpha,i)$ where the quantifiers range over $\{0,1\}, Q_m \in \{\exists,\forall\}, \phi$ is quantifier free and $a(n,\alpha,i) = 1$ iff $\psi_n(\alpha,i)$ is true. Moreover, ψ_n can be computed uniformly in polynomial time in n. Next, arithmetize ψ_n in the folklore way (see, for example, [18]): first, compute in polynomial time a polynomial $q(x,\alpha,i), x = (x_1,\ldots,x_m)$, that gives the truth value of the quantifier free formula $\phi(x,\alpha,i)$, then replace quantifiers of the form $\exists x_j q(...,x_j,...)$ by $1 - \prod_{x_j=0}^{1}(1 - q(...,x_j,...))$ (this guarantees the result to stay in $\{0,1\}$) and quantifiers of form $\forall x_j q(...,x_j,...)$ by $\prod_{x_j=0}^{1} q(...,x_j,...)$. This gives uniformly a binary polynomial formula $G_2(n,\alpha,i)$ computing $a(n,\alpha,i)$.

Step 4: A binary formula for the sign $(-1)^{a(n,\alpha,0)}$ of a monomial x^α is given as $-2 \cdot G_2(n,\alpha,0) + 1$.

Putting everything together, a binary polynomial formula representing $f_n(x_1,\ldots,x_{p(n)})$ results from two further exponential sums, both expressed in our scheme via polynomially many applications of construction rule iii). Identifying as before $i = (i_1,\ldots,i_{p(n)}), \alpha_j = (\alpha_{j1},\ldots,\alpha_{jp(n)})$ and $\alpha = (\alpha_1,\ldots,\alpha_{p(n)})$ and recalling that $G_1(0) = 0$ this binary polynomial formula is

$$\sum_{j=1}^{p(n)} \sum_{\alpha_{j1}=0}^{1} \ldots \sum_{\alpha_{jp(n)}=0}^{1} \left[(-2G_2(n,\alpha,0)+1) \cdot \left(\sum_{i_1=0}^{1} \ldots \sum_{i_{p(n)}=0}^{1} G_1(i) \cdot G_2(n,\alpha,i) \right) \cdot G_\alpha(x_1,\ldots,x_{p(n)}) \right]. \quad \square$$

The theorem now can easily be applied to prove, maybe a bit surprisingly, that the classical technique by Shamir leads relatively far when designing interactive protocols also in the real number framework. More precisely, we have

▶ **Theorem 11.** *It holds* UniformVPSPACE$^0 \subseteq$ IP$_\mathbb{R}$ *in the following sense: Let* $\{f_n\}_n$ *be a family in* UniformVPSPACE^0 *such that* f_n *depends on* $u(n)$ *variables. Then there exists a real interactive protocol for the language* $\{(n,x,y) \in \mathbb{N} \times \mathbb{R}^{u(n)} \times \mathbb{R} \mid f_n(x) = y\}$.

Proof. The proof is an immediate application of Theorem 10 and the original proof of IP=PSPACE in [18]. Given an instance (n,x,y) the verifier first computes in polynomial time the binary polynomial formula obtained at the end of the proof of Theorem 10 representing $f_n(x)$. Note that it involves real numbers resulting from the input values x_j, has polynomial length and contains a polynomial number of operators of the form $\sum_{t=0}^{1}$ and $\prod_{t=0}^{1}$. This is the decisive observation; it implies that the technique used in Shamir's proof to verify interactively an equation $f_n(x) = y$ can be applied in our setting as well without major modifications: Once again, as briefly outlined in the introduction, the verification of $f_n(x) = y$ can be done by eliminating one after the other the leftmost of the operators. The fact that we deal with binary polynomial formulas of polynomial size guarantees that the univariate polynomials

obtained with Shamir's construction have polynomially bounded degree. Therefore, the protocol runs in polynomial time.

□

4 Applications

In view of the difficulties described in the introduction when trying to design an interactive proof for problems in $\mathrm{PAR}_{\mathbb{R}}$ directly, an idea is to study oracle algorithms in the BSS model. More precisely, algorithms that are of interest use as information from an oracle different function evaluations. If f is a member of a family of functions such that for an argument x and a value y the equality $f(x) = y$ can be verified by an interactive protocol, then the outcome of a polynomial time BSS oracle computation having access to an oracle for values of f can be verified interactively as well; for each oracle query the verifier performs an interactive proof with the prover asking the latter to provide proofs of the correct oracle answers. Those are verified by the verifier. If it detects an error in any of the claimed oracle answers it rejects.

In order to obtain an interactive proof for interesting real complexity classes we can therefore consider such oracle computations. A typical classical example along this line is the computation of the permanent polynomial. In [13] an interactive protocol for verifying the value of a permanent of a 0-1-matrix was given (before Shamir's result was known). Together with Toda's theorem that the polynomial hierarchy PH is included in $P^{\#P}$ and the $\#P$-completeness of the permanent computation this implies the existence of an interactive protocol for all problems in the polynomial hierarchy. The protocol for the permanent, as for example described in [1], works as well for real matrices in the BSS model. This implies that real problems that can be decided by a polynomial time BSS algorithm having access to an oracle computing the permanent of real number matrices, i.e., all problems in class $\mathrm{P}_{\mathbb{R}}^{Perm}$, belong to $\mathrm{IP}_{\mathbb{R}}$. However, it is not known whether the permanent plays a similar role for real counting problems as it does in the Turing model. This is an active field of research. Basu and Zell [3] have given a real analogue of Toda's theorem. Instead of the permanent in this approach the computation of so-called Betti numbers of semi-algebraic sets plays a crucial role. The latter express certain topological properties of semi-algebraic sets. But they seem to be even more difficult to handle than permanent computations. And for the permanent itself a real variant of Toda's theorem is currently not known to hold.

In our context, Theorem 11 along the above lines has interesting consequences due to the strong relation the class $\mathrm{PAR}_{\mathbb{R}}$ has to $\mathrm{UniformVPSPACE}^0$. The main result of [12] is a transfer result which roughly states that if families in $\mathrm{UniformVPSPACE}^0$ can be evaluated efficiently, then there is a collapse of $\mathrm{PAR}_{\mathbb{R}}$ to $\mathrm{P}_{\mathbb{R}}$. On the way to prove this result the authors show a result most interesting for us; it witnesses the strength of oracle algorithms that query function evaluations of members of families in $\mathrm{UniformVPSPACE}^0$. We first state this result more precisely, starting with the following definition.

▶ **Definition 12** ([12]). A polynomial-time algorithm with $\mathrm{UniformVPSPACE}^0$-tests is a family $\{f_n(x_1, \ldots, x_{u(n)})\}_n \in \mathrm{UniformVPSPACE}^0$ together with a uniform family $\{C_n\}_n$ of constant-free algebraic circuits of polynomial size. The circuits in addition to their usual gates have special oracle gates of indegree $u(n)$. Those gates on input $x \in \mathbb{R}^{u(n)}$ output 1 if $f_n(x) \leq 0$ and 0 otherwise.

▶ **Theorem 13** ([12]). *For each $A \in \mathrm{PAR}_{\mathbb{R}}^0$ there is a polynomial-time algorithm with $\mathrm{UniformVPSPACE}^0$-tests deciding A.*

Given the remark following Lemma 8 the theorem holds analogously for all problems in $\mathrm{PAR}_{\mathbb{R}}$. Together with Theorem 11 we can now prove our main result.

▶ **Theorem 14.** $\mathrm{PAR}_{\mathbb{R}} \subsetneq \mathrm{IP}_{\mathbb{R}}$

Proof. Let $A \in \mathrm{PAR}_{\mathbb{R}}$. Theorem 13 and the subsequent remark imply that there exists a family $\{f_n\}_n \in \mathrm{UniformVPSPACE}^0$ such that membership in A can be decided by a polynomial time BSS algorithm that has access to an oracle answering questions of the form: is $f_n(x) \leq 0$ for certain arguments x computed during the algorithm. Now each time such an oracle question is posed the verifier asks the prover for a $y \leq 0$ (or $y > 0$, respectively, if the answer should be $f_n(x) > 0$). Then, it applies the algorithm behind the proof of Theorem 11 to verify the result and to continue with the correct oracle answer. If no error occurs the given input is accepted to belong to A, otherwise it is rejected. Given the arguments at the beginning of this section the statement follows. □

Applying the same line of arguments and picking up the above discussion it also follows that $\mathrm{P}_{\mathbb{R}}^{Res} \in \mathrm{IP}_{\mathbb{R}}$, where $Res = \{Res_n\}_n$ denotes the family of resultant polynomials of $n+1$ homogeneous polynomials in $n+1$ variables. This follows from [12] because there it is shown that $\mathrm{Res} \in \mathrm{UniformVPSPACE}^0$.

Problem 1. How large is the class $\mathrm{P}_{\mathbb{R}}^{\mathrm{UniformVPSPACE}^0}$?

In this paper we have derived a first significant lower bound for the class $\mathrm{IP}_{\mathbb{R}}$. Summarizing the results already mentioned the current picture is $\mathrm{PAR}_{\mathbb{R}} \subsetneq \mathrm{IP}_{\mathbb{R}} \subseteq \mathrm{MA}\exists\mathbb{R} \subseteq \mathrm{PAT}_{\mathbb{R}}$. There are some further immediate questions resulting from our lower bound. Given Shamir's characterization of classical IP it follows that IP is closed under complementation. However, without Shamir's result there seems no obvious way to prove this. Thus, in the real number setting we currently do not know whether the analogue statement holds.

Problem 2. Is it true that $\mathrm{IP}_{\mathbb{R}} = \text{co-}\mathrm{IP}_{\mathbb{R}}$?

Of course, we are still missing a characterization of $\mathrm{IP}_{\mathbb{R}}$. The work in [8] gives rise to conjecture $\mathrm{MA}\exists\mathbb{R} \subsetneq \mathrm{PAT}_{\mathbb{R}}$ which would imply that $\mathrm{IP}_{\mathbb{R}}$ is neither characterized by $\mathrm{PAR}_{\mathbb{R}}$ nor by $\mathrm{PAT}_{\mathbb{R}}$. Comparing our results with the different discrete characterizations of IP there seems to be only one more natural class left as a candidate, namely the class $\mathrm{PSPACE}_{\mathbb{R}}$ of problems being decidable by an algorithm using both exponential time and polynomially many registers. Note that requiring both conditions at the same time makes the coding argument from [15] not working. As mentioned above it is known that $\mathrm{PAR}_{\mathbb{R}} \subsetneq \mathrm{PSPACE}_{\mathbb{R}} \subseteq \mathrm{MA}\exists\mathbb{R} \subseteq \mathrm{PAT}_{\mathbb{R}}$. For establishing $\mathrm{PSPACE}_{\mathbb{R}}$ as a lower bound for $\mathrm{IP}_{\mathbb{R}}$ using the above techniques we should first get a similar result to Theorem 11 for a class like $\mathrm{UniformVPSPACE}^0$ such that using this class in oracle computations will cover $\mathrm{PSPACE}_{\mathbb{R}}$. We do not not know whether $\mathrm{UniformVPSPACE}^0$ itself or another similar class satisfies this. The upper bound $\mathrm{MA}\exists\mathbb{R}$ should then also be replaced by $\mathrm{PSPACE}_{\mathbb{R}}$.

Problem 3. What is the relation between $\mathrm{PSPACE}_{\mathbb{R}}$ and $\mathrm{IP}_{\mathbb{R}}$?

References

1 S. Arora, B. Barak: Computational Complexity: A Modern Approach. Cambridge University Press, 2009.

2 M. Baartse, K. Meer: Some results on interactive proofs for real computations. Extended abstract in: Proc. 11th conference Computability in Europe CiE 2015, Bucharest, A. Beckmann, V. Mitrana, M. Soskova (eds.), Springer LNCS 9136, 107–116, 2015.

3 S. Basu, T. Zell: Polynomial hierarchy, Betti numbers, and a real analogue of Toda's theorem. Foundations of Computational Mathematics, 10(4), 429–454, 2010.

4 L. Blum, F. Cucker, M. Shub, S. Smale: Complexity and Real Computation. Springer, 1998.

5 L. Blum, M. Shub, S. Smale: On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. Bull. Amer. Math. Soc., vol. 21, 1–46, 1989.

6 O. Chapuis, P. Koiran: Saturation and Stability in the Theory of Computation over the Reals. Ann. Pure Appl. Logic 99 (1-3), 1–49, 1999.

7 F. Cucker: On the complexity of quantifier elimination: The structural approach. The Computer Journal, vol. 36 No. 5, 400–408, 1993.

8 F. Cucker, I. Briquel: A note on parallel and alternating time. Journal of Complexity, vol. 23, 594–602, 2007.

9 S. Goldwasser: Interactive Proof Systems. In: Computational Complexity Theory, J. Hartmanis (ed.), Proc. of Symposia in Applied Mathematics, Vol. 38, 108–128, 1989.

10 S. Goldwasser, M. Sipser: Private coins versus public coins in interactive proof systems. In Proc. of the 18th Symposium on Theory of Computing STOC, 59–68, 1986.

11 S. Ivanov, M. de Rougemont: Interactive Protocols on the reals. Computational Complexity 8, 330–345, 1999.

12 P. Koiran, S. Perifel: VPSPACE and a transfer theorem over the reals. Computational Complexity 18 (4), 551–575, 2009.

13 C. Lund, L. Fortnow, H. Karloff, N. Nisan: Algebraic methods for interactive proof systems. Journal of the ACM 39 (4), 859–868, 1992.

14 G. Malod: Succinct Algebraic Branching Programs Characterizing Non-Uniform Complexity Classes. Extended abstract in: Proc. 18th International Symposium on Fundamentals of Computation Theory FCT 2011, Oslo, Lecture Notes in Computer Science 6914, 205–216, 2011.

15 C. Michaux: Une remarque à propos des machines sur $\mathbb{R}$ introduites par Blum, Shub et Smale. C.R. Acad. Sci. Paris, t. 309, série I, 435–437, 1989.

16 B. Poizat: Â la recherche de la définition de la complexité d'espace pour le calcul des polynômes à la manière de Valiant. Journal of Symbolic Logic, 73:4, 1179–1201, 2008.

17 J. Renegar: On the computational Complexity and Geometry of the first-order Theory of the Reals , I - III. Journal of Symbolic Computation, 13, 255–352, 1992.

18 A. Shamir: IP = PSPACE. Journal of the ACM, vol. 39(4), 869–877, 1992.

19 L.J. Stockmeyer, A.R. Meyer: Word problems requiring exponential time. In: Proceedings STOC, ACM, 1–9, 1973.