

# On the Automatizability of Polynomial Calculus

Nicola Galesi · Massimo Lauria

© Springer Science+Business Media, LLC 2009

**Abstract** We prove that Polynomial Calculus and Polynomial Calculus with Resolution are not automatizable, unless  $W[P]$ -hard problems are fixed parameter tractable by one-side error randomized algorithms. This extends to Polynomial Calculus the analogous result obtained for Resolution by Alekhnovich and Razborov (SIAM J. Comput. 38(4):1347–1363, 2008).

**Keywords** Automatizability · Polynomial calculus · Proof complexity · Degree lower bound

## 1 Introduction

Automated theorem proving is one of the most important fields investigated both from a theoretical and an applied point of view. It is widely conjectured (but still far to be proved) that for any proof system we have families of tautologies requiring exponentially long proofs. For several concrete examples (e.g. Resolution) we know such a result (see for Resolution, among many others, [5, 6, 13]). Hence investigating automated theorem proving from a theoretical point of view, we should consider proof searching algorithms for a system  $S$  as “efficient” if on any formula  $F$ , they produce a proof of  $F$  efficiently in the size of the best proof of  $A$  in  $S$ .

*Automatizability* is a property for proof systems introduced by Bonet et al. in [8], which captures the discussion above: a proof system  $S$  is *automatizable* if there exists

---

Research of N. Galesi was supported by La Sapienza research projects: (1) “Algoritmi efficienti su modelli avanzati di comunicazione e di calcolo” and (2) “Limiti di compressione in combinatoria e complessit computazionale”.

Research of M. Lauria was partially founded by the grant #13393 of Templeton Foundation.

N. Galesi · M. Lauria (✉)

Department of Computer Science, Sapienza—Università di Roma, Roma, Italy  
e-mail: lauria.massimo@gmail.com

an algorithm  $\mathcal{A}_S$  such that on any tautology  $F$ ,  $\mathcal{A}_S$  produces a proof of  $F$  in  $S$  in time polynomially bounded in the size of the shortest proof of  $F$  in  $S$ . This definition is interesting since it makes the existence of efficient proof searching algorithms for a specific proof system independent from the existence of hard formulas. Recently the notion of automatizability was improved and tightened for applications. Atserias and Bonet in [4] introduced the notion of *weak automatizability* of a proof system.  $S$  is weakly automatizable if there exists a proof system  $S'$  that efficiently simulates  $S$  and that moreover is automatizable.

In general it appears that the stronger the proof system is the lesser is the chance that the proof system is automatizable. Krajíček and Pudlák [17], proved that Extended Frege systems are not automatizable under widely accepted cryptographic conjectures. They did it by exploiting the connection found by Bonet et al. in [8] that automatizability implies Feasible Interpolation Property [8, 16]. Later [7, 8], extended this result down to bounded-depth Frege but weakening the cryptographic assumption considered.

This line of proving non automatizability was not suitable for systems, like Resolution, known to have the Feasible Interpolation Property. In a major breakthrough Alekhovich and Razborov [2], proved that even Resolution is not automatizable unless some parameterized complexity assumption, believed to be false, does hold.

Polynomial Calculus (shortened as PC) is an algebraic refutational proof system introduced in [9] based on deriving polynomials. In the analysis of the complexity of proofs in PC (see [18] for a survey on algebraic systems) we deal with two parameters: the maximal degree of a polynomial used in the proof, and the number of monomials in the proof (usually referred to as *size*).

It is known (see [9, 18]) that *constant degree* PC is automatizable. In [9] it is shown an algorithm based on the Gröbner Basis which finds a PC proof of a set of polynomials  $\mathcal{P}$  over  $n$  variables in time  $n^{O(d)}$ , where  $d$  is the degree required for any PC proof of  $\mathcal{P}$ .

The previous result does not give automatization of PC with respect to the size, i.e. the number of monomials. This question is also interesting because: (1) there are examples of families of polynomials (e.g. pigeon hole formulas) requiring an exponential number of monomials to be refuted in PC (see [14]); (2) our recent result [12], that there are families of polynomials over  $n$  variables requiring  $\Omega(\sqrt{n})$  degree but refutable with only a polynomial number of monomials. This means that the algorithm in [9] cannot be used to recover efficiently proofs of every set of polynomials efficiently provable in PC.

In this paper we study the question of the automatizability of PC. Since PC is one of the refutational system proved to have the Feasible Interpolation Property (see [20]), to prove non automatizability for PC we follow the approach of Alekhovich and Razborov in [2]. Following their strategy we prove that PC is not efficiently automatizable with respect to size unless  $W[P]$ -hard problems are fixed parameter tractable by one-side error randomized algorithms. This extends to Polynomial Calculus their analogous result for Resolution [2].

The original part of the proof in our result is a new degree lower bound for a formula, invented in [2], encoding the optimization problem known as *minimum monotone circuit value problem*. We refer the reader to Sect. 2 for more details on the formula and the proof strategy.

We prove the degree lower bound following a technique initially invented by Razborov in [21] and used also in the papers [1, 12]. As for the degree lower bound in [12], our degree lower bound extends the technique of Razborov to new examples of formulas.

The paper is organized as follows. In Sect. 2 we give the main definitions and we discuss in detail the proof strategy. Section 3 is devoted to the formulation and the polynomial encoding of the principle used. We put efforts in trying to simplify the formulation of the principle. Section 4 contains the proof of the degree lower bound. Finally in the last Section we combine together the degree lower bound with previous results of [2] to get the non automatizability of PC. We conclude with a final section on Open Problems.

## 2 Preliminaries

### 2.1 Algebraic Proof Systems and Automatizability

We consider multivariate polynomials on variables  $x_1, \dots, x_n$  over a field  $\mathbb{F}$ . Let  $p_1 \dots p_l$  be such polynomials: a *common root* of  $p_1 \dots p_l$  is an assignment  $\vec{\alpha}$  such that  $p_i(\vec{\alpha}) = 0$  for  $1 \leq i \leq l$ .

Consider an algebraic combinations  $\Delta := \sum_{i=1}^l h_i p_i$  for some polynomials  $h_1 \dots h_l$ . Every common root of the initial polynomials is a root of  $\Delta$ , thus if  $\Delta$  is the polynomial 1 then  $p_1 \dots p_l$  have no common roots at all (i.e. they are collectively unsatisfiable). This way of proving unsatisfiability can be adapted to propositional logic by

- Adding  $x_i^2 - x_i$  for any  $1 \leq i \leq n$  to the initial set of polynomials. This forces common roots to be 0-1 assignments.
- Encoding boolean constraints as polynomials over 0-1 assignments.

The original set of boolean constraints is satisfiable if and only if its polynomial encoding has a common root. We now define some algebraic proof systems for propositional logic, varying in the way  $h_1, \dots, h_l$  are constructed and in the way  $\Delta$  is shown to be equal to 1. Notice that for efficiency reasons  $h_1, \dots, h_l$  and  $\Delta$  may be not explicit exhibited in the actual refutation. An introduction to algebraic proof systems is in the survey [18].

We denote the *degree* of a polynomial  $p$  as  $\deg(p)$ , and the number of monomials in  $p$  as  $S(p)$ . In the rest of the section we assume  $p_1 \dots p_l$  to be polynomials with no common roots among 0-1 assignments.

*Nullstellensatz (HN)*: A proof in Nullstellensatz is a set of polynomials  $g_1 \dots g_l, h_1 \dots h_n$  such that

$$\sum_i g_i p_i + \sum_i h_i (x_i^2 - x_i) = 1$$

The *degree* of a HN proof is  $\max_i \{\deg(g_i p_i)\}$ .

*Polynomial Calculus (PC)*: A proof in Polynomial Calculus is a set of polynomials  $g_1 \dots g_m$  such that  $g_m = 1$  and any  $g_i$  is either an axiom  $x_j^2 - x_j$  for some  $j \in [n]$ ,

or a  $p_j$  for  $j \in [l]$ , or a polynomial derived according one of the following inference rules:

$$\frac{g_a \quad g_b}{\alpha g_a + \beta g_b} \qquad \frac{g_a}{x_j \cdot g_a}$$

where  $1 \leq a, b < i$ ,  $\alpha, \beta \in \mathbb{F}$  and  $j \in [n]$ .

The *degree* of a PC proof is  $\max_i \{\deg(g_i)\}$ , and the *size* of a PC proof is  $\sum_i S(g_i)$ .

*Polynomial Calculus with Resolution* (PCR): it is an extension of PC. Polynomials are allowed to use additional variables  $\bar{x}_1 \dots \bar{x}_n$ . A proof is similar to a PC proof with the addition that a line can be also be an axiom of the form  $1 - x_j - \bar{x}_j$ .

As in PC, the *degree* of a PCR proof is  $\max_i \{\deg(g_i)\}$ , and the *size* of a PCR proof is  $\sum_i S(g_i)$ .

Given a set of polynomials  $P = \{p_1 \dots p_l\}$  we define as  $\deg_X(P)$  and  $S_X(P)$  the minimum degree and size achievable in a proof of  $P$  by the proof system  $X$ . It easy to see that the following relations hold for any  $P$

$$\deg_{\text{HN}}(P) \leq \deg_{\text{PC}}(P) = \deg_{\text{PCR}}(P)$$

$$S_{\text{HN}}(P) \leq S_{\text{PC}}(P) \leq S_{\text{PCR}}(P)$$

We say a that proof system  $X$  is *automatizable* (*quasi-automatizable*) if there is an algorithm which given  $P$  outputs a valid  $X$  proof of  $P$  in time polynomial (resp. quasi-polynomial) with respect to  $S_X(P)$ .

## 2.2 Notions from Commutative Algebra

Given a field  $\mathbb{F}$ , we consider polynomials over  $\mathbb{F}[x_1, \dots, x_n]$ . Given a set  $E = \{f_1, \dots, f_n\}$  of polynomials, by  $\text{Span}(E)$  we denote the ideal generated by  $E$ , that is the set  $\{\sum_i (f_i \cdot h_i) \mid h_i \in \mathbb{F}[x_1, \dots, x_n]\}$ . We say that a set of polynomials  $f_1, \dots, f_n$  *semantically implies* a polynomial  $g$  if any assignment that satisfies  $f_i = 0$  for all  $i \in [n]$ , also satisfies  $g = 0$ . We write  $f_1, \dots, f_n \models g$  or  $E \models g$ .

We define a notion of residue of polynomials with respect to an ideal. We consider the *grlex* order  $<_{\mathbb{P}}$  on monomials as given in [10]. In particular *grlex* is defined as follows:  $1 <_{\mathbb{P}} x_1 <_{\mathbb{P}} x_2 <_{\mathbb{P}} \dots <_{\mathbb{P}} x_n$ . For any two products of variables  $m, m'$  and a variable  $x$  hold the following two properties: (a) if  $m <_{\mathbb{P}} m'$  then  $xm <_{\mathbb{P}} xm'$ ; (b)  $m <_{\mathbb{P}} xm$ . This order is lexicographically extended to polynomials, and 0 is the smallest of them.

Notice that *grlex* is not a total order, thus there could be incomparables  $q, q' \in \text{Span}(E)$ . This can happen if and only if the underlining sets of monomials are equal but have different coefficients. In that case there exists a linear combination of  $q$  and  $q'$  which is strictly smaller than both, and which is in  $\text{Span}(E)$ . Thus a minimum element in  $\text{Span}(E)$  always exists.

Given a polynomial  $q$ , we define  $R_E(q)$  as the minimal, with respect to  $<_{\mathbb{P}}$ , polynomial  $p$  such that  $q - p \in \text{Span}(E)$

$$R_E(q) = \min\{p \in \mathbb{F}[x_1, \dots, x_n] : q - p \in \text{Span}(E)\}$$

In the following sections we use some properties of the operator  $R_E$  which can be easily derived from the definition:

**Lemma 1** *Let  $E$  be a set of polynomials and let  $p$  and  $q$  be two polynomials. Then:*

- $R_E(p) \leq_{\mathbb{P}} p$ ;
- if  $p - q \in \text{Span}(E)$ , then  $R_E(p) = R_E(q)$ ;
- $R_E$  is a linear operator;
- $R_E(pq) = R_E(p \cdot R_E(q))$ .

We shall consider polynomials on the field  $\mathbb{F}$  defined on the domain  $\{0, 1\}^n$ . More explicitly we consider elements of the ring  $\mathbb{F}[x_1, \dots, x_n]/\{x_i^2 - x_i\}_{i \in [n]}$ . Such polynomials are the base for all algebraic proof systems we consider.

### 2.3 Proof Strategy

We show that if we could automatize one of the proof systems we just introduced, then we could efficiently solve the following optimization problem:

*Minimum Monotone Circuit Satisfying Assignment (MMCSA)*

*Instance:* A monotone circuit over  $\wedge, \vee$  in  $n$  variables.

*Solution:* An input  $a$  such that  $C(a) = 1$ .

*Objective function:*  $w(a)$ , the Hamming weight of  $a$ .

In [2] Alekhovich and Razborov use the automatization of Resolution as a primitive to efficiently solve MMCSA. The idea can be summarized in three independent steps.

1. Given a monotone circuit  $C$ , build an unsatisfiable CNF  $\phi := F(C, w, r)$  and prove that the size of the shortest proof of unsatisfiability for  $\phi$  is strongly related to the size of minimum satisfying assignment of the circuit. Here  $w$  is a guess for the minimum value of the objective function. To amplify the degree complexity of  $\phi$  and to apply the random restriction method we use error correcting codes in the construction: the rates of such codes depends on  $r$ .
2. Assuming automatizability of the proof systems, use the automatization algorithm to find a proof of unsatisfiability for  $\phi$  of approximately small size. This gives an approximation of the minimum assignment size.
3. Apply (randomized) gap amplification procedures to improve the approximation factor up to an error smaller than one, thus obtaining an exact value.

The first step depends on the proof system. Using a slight modification of the formula built in [2] we will prove the first step for algebraic proof systems. We will see that the other two steps are essentially independent from the proof system. Formally we prove the following two theorems.

**Theorem 1** *If any of HN, PC and PCR is automatizable, then for a fixed  $\epsilon > 0$  there exists an algorithm  $\Phi$  working on monotone circuits  $C$  which runs in time  $\exp(w(C)^{O(1)})|C|^{O(1)}$  and approximates the value of  $w(C)$  to within a factor  $(1 + \epsilon)$ .*

**Theorem 2** *If HN, PC or PCR are automatizable then  $\text{MMCSA} \in \text{co-FPR}$ .*

MMCSA is believed to be an hard optimization problem, thus the previous theorems suggest the impossibility of efficient automatization. To support the hardness of MMCSA we briefly introduce the framework of *parameterized complexity* (for more details we suggest [11]): we consider languages which are subsets of  $\{0, 1\}^* \times \mathbb{N}$ . Take as an example the pairs  $(G, k)$  where  $G$  is a graph with a vertex cover of size  $k$ . A problem is said to be *fixed parameter tractable* if the membership of  $(I, k)$  in the language can be decided in time  $f(k) \cdot |I|^{O(1)}$  where  $f$  is an arbitrary monotone increasing function. This framework models the fact that if  $k$  is very small then the computation is still feasible (given that  $f$  is reasonable). Notice that the previous parameterized version of vertex cover is fixed parameter tractable. We can also consider fixed parameter tractable reductions between languages. Consider the following hierarchy of complexity classes:  $W[1] \subseteq W[2] \subseteq \dots \subseteq W[i] \subseteq \dots \subseteq W[\text{SAT}] \subseteq W[P]$ .  $W[P]$  consists in the languages fixed parameter reducible to the problem of deciding if a monotone circuit has a satisfying assignment of weight  $k$ .  $W[\text{SAT}]$  is similar but the circuit is restricted to be a formula. For all constant  $i$ ,  $W[i]$  is the subset of  $W[\text{SAT}]$  where formulas have depth at most  $i$ . By definition of  $W[P]$  it is easy to see that MMCSA is  $W[P]$ -hard with respect to fixed parameter tractable reductions. It is widely believed that this hierarchy is strict, so there are very few hopes that MMCSA could have a feasible algorithm.

*FPR* is a hybrid class of *RP* and *FPT* introduced by [2], defined as follows:

**Definition 1** The class *FPR* of parameterized problems, consists, of all languages  $L \subseteq \{0, 1\}^* \times \mathbb{N}$ , for which there exists a probabilistic algorithm  $\Phi$ , a constant  $c$  and a recursive function  $f : \mathbb{N} \rightarrow \mathbb{N}$  such that:

1. the running time of  $\Phi(\langle x, k \rangle)$  is at most  $f(k) \cdot |x|^c$ ;
2. If  $\langle x, k \rangle \in L$ , then  $\Pr[\Phi(\langle x, k \rangle) = 1] \geq 1/2$ ;
3. If  $\langle x, k \rangle \notin L$ , then  $\Pr[\Phi(\langle x, k \rangle) = 1] = 0$ .

To get Theorem 1, we prove the following reduction as in [2].

**Lemma 2** Let  $C$  be a monotone circuit, and  $w$  an integer parameter. Assume that  $r = \Theta(w)$ :

1. Any PCR refutation of  $F(C, w, r)$  has size at least

$$2^{\Omega(w \cdot \min\{w(C), w\})}$$

2. If  $w(C) \leq w$  then there is a HN proof of  $F(C, w, r)$  of size

$$|C| \cdot 2^{O(w \cdot w(C))}$$

In turn the lemma is proved using our main and new result in the paper, which is a degree lower bound for  $F(C, w, r)$ .

**Theorem 3** Any PCR refutation of  $F(C, w, r)$  requires degree at least  $r \cdot \min\{w(C) - 1, w\}$ .

### 3 The Principle

Let  $C$  be a monotone circuit on  $n$  inputs of size polynomial in  $n$ . Let  $w(C)$  be the minimum hamming weight of an assignment satisfying  $C$  and let  $w$  be a parameter whose intended meaning is to guess a value for the size of  $w(C)$ . Let  $r$  be a parameter used to amplify the degree hardness of the principle,  $r$  is intended to be  $\Theta(w)$ .

A combinatorial object called *Paley matrix* is used for the construction. Such matrix has the property that any projection on a small set of columns consists of all possible binary strings with an almost fair frequency. For our purpose it is sufficient that any set of  $w$  columns has a full 1 row and any set of  $w$  rows has a full 0 column. We will show a construction for a matrix  $A$  of size  $2^{\Theta(w)}$ .

Let  $q > 2^{4w}$  be an arbitrary prime, and consider a  $q \times q$  matrix where cell  $(i, j)$  contains 1 iff  $i - j$  is a quadratic residue modulo  $q$ , or 0 otherwise.<sup>1</sup> It is well known [3, 15] that any projection on  $w$  rows or columns contains every string in  $\{0, 1\}^w$ .

From  $C$ ,  $w$ ,  $r$  and  $A$ , we are going to define a set of equations  $F(C, w, r)$ . The principle encodes the property that there exists a set of  $n$  columns of the matrix  $A$  that contains no satisfying assignment for the circuit  $C$ . This claim is unsatisfiable if  $w(C) < w$ . This is because whatever the selection of  $n$  columns is, the  $w$  columns corresponding to the 1 coordinates of such assignment will contain a full 1s string which satisfies the circuit  $C$ .

The  $n$  columns are selected by  $n$  partial functions called *generators*. These functions will be defined in the following.

For technical reason, widely explained in [2], we add complexity to the principle: instead of feeding one input to a single circuit, we feed it to several copies of the same circuit, and we select one of those copies by a *partial* function called *activator*. In the principle we will express the fact that only the gates of the active circuits have to perform the computation correctly.

The principle claims that there exists a choice for the ‘generators and activators’ inputs such that: (a) values for such inputs are defined and (b) no circuit outputs 1.

It is clear that even in this version of the principle  $w(C) < w$  implies that for any input any  $w$  generators produce a full 1 row. This row is fed to a set of  $r$  copies of the circuit  $C$ . One of them is active and has to propagate the computation through the gates. Thus that circuit copy outputs 1, which is in contradiction with the claim no active circuit outputs 1.

*Circuit family* Let  $q = 2^{\Theta(w)}$  the size of the matrix. We consider an array of  $q \times r$  copies of  $C$ , indexed as  $C_{ik}$ .

Let  $s$  be a parameter we specify later and which should be intended to be  $\Theta(w + \log q)$  which will be  $\Theta(w + r)$ .

*Activators* For all  $i \in [q]$  we consider a possibly partial function  $A_i : \{0, 1\}^s \mapsto [r]$  which selects one of the circuit among  $C_{i1} \cdots C_{ir}$ . Only for gates in the selected circuit we ensure to propagate correctly through the circuit. The other ones are relieved from the computation.

<sup>1</sup>Note that in literature the original Paley Matrix is defined in a different fashion with 0 values on diagonal, 1 for quadratic residues and  $-1$  for non residues. But this is not an issue here.

*Generators* For  $j \in [n]$   $G_j : \{0, 1\}^s \rightarrow \{0, 1\}^q$  generates a column vector of  $A$ . Thus  $\{G_1 \cdots G_n\}$  define a  $q \times n$  matrix. For all  $i \in [q]$  the  $i^{\text{th}}$  row of such matrix is fed to  $C_{i1} \cdots C_{ir}$  as input.

*Generator and Activators with  $r$ -surjectivity and parameter  $s$*  As in [2] (see there for an extensive discussion about this requirement) we need both activators and generators to be  $r$ -surjective functions on domain  $\{0, 1\}^s$ , for a suitable  $s$ . Here is a construction (see also [2], proof Lemma 3.1(ii)), that allows us to define surjective functions. Consider a general binary surjective function  $g : \{0, 1\}^l \mapsto S$  and a decoding function  $D : \{0, 1\}^{\alpha l} \mapsto \{0, 1\}^l$  of a code with distance  $\geq 2l$  (constructions exist [22] such that  $\alpha$  is a universal constant). Then  $g \cdot D : \{0, 1\}^{\alpha l} \mapsto S$  is  $l$ -surjective on  $S$ . To see this consider the restricted bits as “error” bits. Any message can be decoded from the proper codeword with those  $l$  bits changed.

The output of a generator function  $G_j$  is completely specified by the choice of a column in the Paley matrix (hence  $\Theta(w)$  bit), the output of the activator  $A_i$  is specified by the choice of a number in  $[r]$ . Thus  $\Theta(w + r)$  input bits are enough to define both surjective generators and selectors. Then by composition with a  $2r$  distance code we obtain the desired  $r$ -surjective functions on  $\{0, 1\}^s$  with  $s = \Theta(w + r)$ .

Because of the properties of Paley matrix, generators and activators the following fact holds.

**Fact 1** For any monotone circuit  $C$ , and any  $w > w(C)$ , at least one of  $\{C_{ik}\}_{ik}$  outputs 1 whatever the assignment for activators and generators is.

### 3.1 Propositional Encoding

We now describe the polynomials which encode the negation of the principle discussed so far. Describing the elements of the principle we will be consistent with the following notation.

*Notation reference*

$C$  the monotone circuit.

$n$  input variables in the circuit  $C$ .

$v$  index refers to a the gates of  $C$ . It goes from 1 to  $|C|$ . We assume first  $1..n$  indexes correspond to input gates. And  $|C|$  corresponds to the output gate.

$w(C)$  is the minimum hamming weight of an assignment satisfying  $C$ .

$q = 2^{\Theta(w)}$  a prime number, size of the Paley matrix.

$r$  is a parameter.

$i$  index used to denote a row in the Paley matrix, a selector function and one of the inputs fed to the circuits. It goes from 1 to  $q$ .

$j$  index used to denote one input variable of  $C$  and one of the generator functions. It goes from 1 to  $n$ .

$k$  index used to denote one of the possible outcomes of the selector functions. It goes from 1 to  $r$ .

$C_{ik}$   $k$ th copy of  $C$ , fed with the  $i$ th row generated by generator functions.

$A_i$   $i$ th activator function which selects among circuits  $C_{i1}, \dots, C_{ir}$ .

$G_j$   $j$ th generator function which chooses the  $j$ th columns to feed as  $j$ th input of the circuits.

$s$  length of the binary input of activators and generators, it is  $\Theta(w + r)$ .

$x_j(\alpha), y_i(\beta)$  For a vector  $\alpha, \beta \in \{0, 1\}^s$  we denote as  $x_j(\alpha)$  and  $y_i(\beta)$  the characteristic polynomials of  $\alpha$  and  $\beta$  respectively on variables  $x_{j1}, \dots, x_{js}$  and  $y_{i1}, \dots, y_{is}$ .

For example  $x_j(001\dots)$  is  $\bar{x}_{j1}\bar{x}_{j2}x_{j3}\dots$ .

Polynomials which encode the principle are expressed in the following set of variables.

*Variables*

For all  $i, j, k$ , and gate  $v$  as above:

- $x_{j1} \dots x_{js}$  are the variables representing the input of generator  $G_j$ .
- $y_{i1} \dots y_{is}$  are the variables representing the input of activator  $A_i$ .
- $z_{ik}^v$  represents the value of gate  $v$  in circuit  $C_{ik}$ .

We now give the encoding, dividing polynomial equations in several sets. For any index  $i \in [q]$  we encode in an equation set  $F_i$  all equations relevant to row  $i$ . Indexes  $i, j, k, v$  and  $\alpha, \beta$  apply properly as described above.  $F_i$  are shown below:

$$x_j(\alpha)y_i(\beta)\bar{z}_{i,k}^j = 0 \quad \text{iff the } i\text{th bit of } G_j(\alpha) \text{ is } 1 \text{ and } A_i(\beta) = k \tag{1}$$

$$y_i(\beta) = 0 \quad \text{when } \beta \notin \text{dom}(A_i) \tag{2}$$

$$y_i(\beta)z_{i,k}^A z_{i,k}^B \bar{z}_{i,k}^v = 0 \quad \text{gate } v \leftarrow A \wedge B \text{ and } A_i(\beta) = k \tag{3}$$

$$y_i(\beta)z_{i,k}^A \bar{z}_{i,k}^v = 0, \quad y_i(\beta)z_{i,k}^B \bar{z}_{i,k}^v = 0 \quad \text{gate } v \leftarrow A \vee B, \text{ and } A_i(\beta) = k \tag{4}$$

$$y_i(\beta)z_{i,k}^{|C|} = 0 \quad A_i(\beta) = k \tag{5}$$

Equation (1) says that if the  $i$ th bit of the column generated by  $G_j$  is 1 and the active circuit for  $i$ th row is  $k$ , then the  $j$ th input of  $C_{ik}$  must be on. Equation (2) forces the  $y_i$  variables to encode a value in the domain of the activator function. This is necessary because activators are partial functions. Equations (3) and (4) force the active circuit to compute the gates correctly. The equation (5) claims that the output of the active circuit is zero.

Notice that this principle is slightly different from the one in [2]. They use additional variables to indicate circuit activation. We choose not to use them because they would cause trouble in some technical steps of the following proofs.

$F_i$  specifies in a truth table fashion how gates of a circuit in a row  $i$  behave. When such circuit is activated the input of generators causes some bits to be fed in it. The principle also claims such circuit outputs 0. The principle consists in the conjunction of  $F_i$  for  $i \in [q]$ .

**Definition 2** We call  $F(C, w, r)$  the principle described above as  $\bigcup_i F_i$ , where  $C$  is a monotone circuit,  $q = 2^{\Theta(w)}$  is a prime, and  $r$  is the surjectivity parameter of generators and activators.

**Fact 2** *The equations of principle  $F(C, w, r)$  can be produced in  $|C|2^{O(w+r)}$  time and space. If  $w(C) \leq w$  then  $F(C, w, r)$  is unsatisfiable.*

*Proof* Remember  $s = O(w + r)$ . For any  $\alpha, \beta$  two strings of  $s$  bits, any  $i \in [q]$ , any  $j \in [n]$  there is at most one  $k$  such that  $x_j(\alpha)y_i(\beta)\bar{z}_{i,k}^j = 0$  is in the principle. Then for any  $\beta$  and  $i$  there is at most one  $k$  for which there are gate propagation equations. Otherwise there is one single clause  $y_i(\beta) = 0$  if  $\beta$  is not in the domain of  $A_j$ . In total we have  $n2^{O(s)} + |C|2^{O(s)}$  equations. Notice that if we have dual variable for encoding negations then all equations can be encoded as monomials. Otherwise equations will be encoded as polynomials exponentially long in the degree of the equation. Biggest degree appearing is  $2s$  thus the size of the formula  $F(C, w, r)$  is again  $n2^{O(s)} + |C|2^{O(s)}$ . The obvious output strategy satisfies the resource bound. Unsatisfiability comes as a restatement of Fact 1.  $\square$

#### 4 Degree Lower Bounds for $F(C, w, r)$

In this section we prove that the formula  $F(C, w, r)$  requires a high degree to be refuted. We need a tailored degree measure for this purpose.

**Definition 3** For a monomial  $t$  consider the three sets

$$\begin{aligned} X_t &:= \{(j, l) : x_{jl} \in t\} \\ Y_t &:= \{(i, l) : y_{il} \in t\} \\ Z_t &:= \{(i, k) : \text{there is a } v \text{ for which } z_{ik}^v \in t\} \end{aligned}$$

And we define the *index-degree* of  $t$  as

$$\text{ideg}(t) = |X_t| + |Y_t| + |Z_t|$$

The index-degree of a polynomial is the biggest index-degree among its monomials.

**Theorem 4** *Any PCR refutation of  $F(C, w, r)$  contains a polynomial of index-degree at least  $r \cdot \min\{w(C) - 1, w\}$ .*

From now on we define  $m := \min\{w(C) - 1, w\}$ . The index-degree lower bound relies on the construction of an operator  $K$  over multivariate polynomials such that

1.  $K$  is a linear operator.
2.  $K(p) = 0$  for any  $p$  in  $F(C, w, r)$ .
3. If  $\text{ideg}(t) < rm$  then  $K(xt) = K(xK(t))$  holds for any variable  $x$ .
4.  $K(1) \neq 0$ .

*Proof of Theorem 4* Assume a proof of index-degree less than  $rm$  exists: each line of such proof is either an equation in  $F(C, w, r)$ , or a sum of previous lines, or the product of a previous line with a variable where index-degree stays below  $rm$ . Then

properties (1), (2), (3) imply that  $K$  maps to 0 every line in the proof. This contradicts property (4) which claims  $K$  can not map last line to 0.  $\square$

We now show such operator  $K$ . Assume  $I$  is a set of row indexes contained in  $[q]$  and consider the set of polynomials  $\mathcal{I}$  containing  $F_i$  for  $i \in I$  and also containing all PCR axioms. We denote as  $R_I(p)$  the residue of polynomial  $p$  modulo the ideal generated by  $\mathcal{I}$ . More concretely

$$R_I(p) = \operatorname{argmin}_q \left\{ p - q = \sum_{s \in \mathcal{I}} h_s s \right\}$$

for some  $h_s$  multivariate polynomials. We write  $I \vdash p$  if  $p$  is in the ideal generated by  $\mathcal{I}$ .

**Definition 4** (Function  $I$  and operator  $K$ ) Fix a monomial  $t$ : we can write  $t = t_1 t_2 \cdots t_q t'$  where each  $t_i$  contains only variables indexed by  $i \in [q]$  and  $t'$  contains only  $x_{jI}$  variables.

$I(t)$  is the set of  $i \in [q]$  such that  $\operatorname{ideg}(t_i) \geq r$ .  $K(t)$  is equal to  $R_{I(t)}(t)$ . On a formal polynomial  $p = \sum_i c_i t_i$  we define  $K(p) := \sum_i c_i K(t_i)$ .

We now check that  $K$  satisfies properties (1)–(4). (1) comes from the definition. (2) If a premise  $p$  is an axiom then any of its terms is reduced with respect to an ideal which contains  $p$  itself. Any premise  $p$  in  $F_i$  is a monomial  $t$  which contains more than  $r$  variables indexed by  $i$ . Thus such  $p$  is in the ideal  $I(t)$ . This implies  $p$  is reduced to 0. (4) is true because  $I(1)$  contains a set of polynomials with a common 0-1 solution.

To prove (3) we need the following results about ideals:

**Lemma 3** For any polynomial  $p$  and ideal  $I$  generated by  $q_1, q_2, \dots, q_m$ , all variables appearing in  $R_I(p)$  also occur in  $p, q_1, q_2, \dots, q_m$ .

*Proof* Let  $x$  be a variable occurring in  $R_I(p)$  and not in  $p$  nor in any generator  $q_i$ . By definition  $p - R_I(p) = \sum_i h_i q_i$  for some polynomials  $h_i$  thus by setting  $x$  to 0 we obtain  $p - R_I(p) \upharpoonright_{x=0} = \sum_i (h_i \upharpoonright_{x=0}) q_i$  where  $R_I(p) \upharpoonright_{x=0}$  is strictly smaller than  $R_I(p)$ . This contradicts the fact  $R_I(p)$  is the minimum according to  $<_{\mathbb{P}}$ .  $\square$

**Corollary 1** Let  $m_1, m_2$  two monomials, if  $t$  is a monomial in  $m_1 \cdot R_{I(m_2)}(m_2)$  then  $I(t) \subseteq I(m_1 m_2)$ .

*Proof* If  $i' \in I(t)/I(m_1 m_2)$  then variables indexed by  $i'$  in  $t$  are more than  $r$ . No such variable is contained in  $F_i$  for  $i' \neq i$ . Thus by Lemma 3 any term in  $R_{I(m_2)}(m_2)$  does not contain more of such variables than  $m_2$  itself contains. Such  $i'$ -indexed variables are then contained in the union of the variable of  $m_1$  and  $m_2$ . Thus  $i' \in I(m_1 m_2)$  but this contradicts the assumption.  $\square$

Next lemma is the heart of the argument: it shows how a small index-degree derivation has local behavior. The set of premises needed in the derivation is a subset of the one given by the operator  $I$ .

**Lemma 4** *Let  $t$  be a monomial of index-degree less than  $rm$  and  $I(t) \subseteq I$  with  $|I| \leq m$ . Then  $R_{I(t)}(t) = R_I(t)$ .*

*Proof* We will show an assignment  $\rho$  such that  $t \upharpoonright_\rho = t$  and  $I \upharpoonright_\rho \subseteq I(t)$ . This is sufficient since

$$I \upharpoonright_\rho \vdash t \upharpoonright_\rho - R_I(t) \upharpoonright_\rho$$

By properties of  $\rho$  we get  $I(t) \vdash t - R_I(t) \upharpoonright_\rho$ . This means  $R_I(t) \upharpoonright_\rho$  is bigger than  $R_{I(t)}(t)$  in the order among polynomials. It is also smaller than  $R_I(t)$  because a partial assignment can't increase the order. Notice that we also have  $R_I(t)$  smaller than  $R_{I(t)}(t)$  because  $I(t)$  is a subset of  $I$  and residue is monotone decreasing with respect to the subscript set.

We now consider  $J$  the set in indexes  $j \in [n]$  such that  $t$  contains less than  $r$  variables among  $x_{j1}, \dots, x_{js}$ . Thus  $|J| \geq n - m$ .

Notice that because of  $r$ -surjectivity of generators we have that for any  $j \in J$  and any vector  $v = v_1 \dots v_q$  in the image of  $G_j$  there is a boolean partial assignment  $\alpha_j$  on " $x_j$ " variables such that no variable in  $t$  is assigned and  $G_j(\alpha) = v$ . We choose a  $v$  such that for any  $i \in I$  we have  $v_i = 0$ . Such choice is possible because  $|I| \leq m$  and  $v$  is a column in a Paley matrix of appropriate size. We add  $\alpha_j$  for  $j \in J$  in  $\rho$ . Such partial assignment does not restrict  $t$ , and set to 0 all equations  $x_j(\alpha)y_i(\beta)z_{ik}^j = 0$  for any  $j \in J, i \in I, k \in [q]$ . Other equations are left untouched.

We now consider a row  $i_0$  in  $I/I(t)$  and  $t_0$  the monomial containing all variables in  $t$  indexed by  $i_0$ . We extend  $\rho$  to satisfy all remaining equations in  $F_{i_0}$ . To achieve such result we notice there is at least one circuit copy  $C_{i_0k}$  such that no variables in  $t$  correspond to a gate of such a circuit, otherwise it would be  $\text{iddeg}(t_0) \geq r$  and  $i_0$  would be in  $I(t)$ . For the same reason we also know in  $t$  there are less than  $r$  variables among  $y_{i_01} \dots y_{i_0s}$ . Both observation together imply there is a partial assignment on  $y_{i_0l}$  variables not contained in  $t$  such that  $y_{i_0}(\beta) = 0$  for all  $\beta$  with  $A_{i_0}(\beta) \neq k$ . So far all equations in  $F_{i_0}$  are satisfied with the exception of the ones corresponding to circuit  $C_{i_0k}$ . We set  $z_{i_0k}^j$  to 0 when  $j \in J$  and 1 otherwise. Then we propagate values among the circuit equations accordingly. We remark that being  $|J| < m \leq w(C)$  we have 0 at the output gate. This satisfies all clauses in  $F_{i_0k}$  without touching  $t$ . We continue to extend  $\rho$  in this way for all  $i \in I/I(t)$ . The resulting assignment satisfies the requested properties. Thus the lemma is proved.  $\square$

**Lemma 5** *If the index-degree of a monomial  $t$  is less than  $rm$  then  $K(xt) = K(xK(t))$ .*

*Proof* Consider a monomial  $t$  of index-degree less than  $rm$ . We will prove that both  $K(xt)$  and  $K(xK(t))$  are equal to  $R_{I(xt)}(xK(t))$ . Consider the following chain of equations

$$K(xt) = R_{I(xt)}(xt) \tag{6}$$

$$= R_{I(xt)}(xR_{I(xt)}(t)) \tag{7}$$

$$= R_{I(xt)}(xR_I(t)) \tag{8}$$

$$= R_{I(xt)}(xK(t)) \tag{9}$$

The equation (6) is the definition; (7) because  $R_I$  is an homomorphism on the ring of multivariate polynomials; (8) holds because of Lemma 4; (9) holds because of the definition of  $K$ . Let us denote  $xK(t)$  as  $\sum_i \alpha_i t_i$  in the next chain of equations.

$$K(xK(t)) = K\left(\sum_i \alpha_i t_i\right) \tag{10}$$

$$= \sum_i \alpha_i K(t_i) \tag{11}$$

$$= \sum_i \alpha_i R_{I(t_i)}(t_i) \tag{12}$$

$$= \sum_i \alpha_i R_{I(t)}(t_i) \tag{13}$$

$$= R_{I(xt)}\left(\sum_i \alpha_i t_i\right) \tag{14}$$

$$= R_{I(xt)}(xK(t)) \tag{15}$$

The first lines holds because the notation just introduced; (11) by linearity of  $K$ ; (12) by definition of  $K$ ; (13) holds because any  $x t_i$  is a monomial in  $x R_{I(t)}(t)$ . We now use Corollary 1 to claim  $I(x t_i)$  is a subset of  $I(xt)$ , which has size less than  $m$ . Lemma 4 finally implies the equation. By using linearity we get (14) and by reverting the change of notation we conclude the proof with (15).  $\square$

### 5 Main Result

In this section we prove a result similar to Lemma 3.1 in [2] for the systems HN and PCR. The result obtained in Sect. 3 of [2] depends on Resolution system, while the self-improvement technique developed in Sect. 4 of [2] refers to MMCSA amplification and is independent from the proof system adopted.

**Lemma 6** *Let  $C$  be a monotone circuit, and  $w$  an integer parameter. Assume  $r = \Theta(w)$ :*

1. Any PCR refutation of  $F(C, w, r)$  has size at least

$$2^{\Omega(w \cdot \min\{w(C), w\})}$$

2. If  $w(C) \leq w$  then there is a HN proof of  $F(C, w, r)$  of size

$$|C| \cdot 2^{O(w \cdot w(C))}$$

*Proof* (1) *Lower bound.* The strategy here follows [2]: we deduce a degree lower bound on the PCR refutation of  $F(C, w, r)$  and then we use a random restriction argument to deduce the size lower bound.

The restriction: for each input set of the generators and activators we restrict uniformly independently at random a set of  $r/2$  of the  $s$  variables. For each  $i \in [q]$  we also choose independently  $r/2$  circuit copies of the  $r$  available and we restrict randomly all the gates of such copies. This restriction (up to index reordering) is essentially subsumed by  $F(C, w, r/2)$ . For any restricted variable we fix the corresponding dual variable to the appropriate value.

Fix  $d := \frac{r}{4} \cdot \min\{w(C) - 1, w\}$ . We show that any monomial with index-degree bigger than  $d$  is set to zero with probability at least  $1 - 2^{-\Omega(d)}$ . Fix a polynomial  $t$  of index degree at least  $d$ . We know  $t = t_1 \cdots t_d t'$  where  $t_i$  is either a power of a generator variable, an activator variable or a non empty product of variables corresponding to a particular circuit  $C_{ik}$ . We can assume variables in  $t_i$ s to be disjoint. We want to estimate the probability that  $t_i$  is set to 0 by the random restriction, assuming  $t_1 \dots t_{i-1}$  haven't been. Consider the case  $t_i$  is a generator or an activator variable:  $s$  is the number of such variables for each generator and activator. With at least  $r/2s$  probability  $t_i$  is chosen among the restricted variables. Then with probability at least  $r/4s$  the monomial is set to zero. Notice that  $r/4s$  is a constant by construction. In the case  $t_i$  is a product of variables of  $C_{ik}$  for some  $i$  and  $k$  then such circuit is chosen to be restricted with probability at least  $1/2$  because no previous one has been, and the product is restricted to zero with at least probability  $1/4$ . The probability of the monomial not to be set to zero is then at most  $c^d$  for some  $c > 1$ .

For a partial assignment  $\rho$  distributed as described  $\Pi \upharpoonright_\rho$  is a proof of  $F(C, w, r) \upharpoonright_\rho$  and of  $F(C, w, r/2)$ . Assuming that  $\Pi$  is of size smaller than  $c^d$  then by union bound there is a restriction  $\rho$  such that  $\Pi \upharpoonright_\rho$  is a proof of degree less than  $d$  for  $F(C, w, r/2)$ . This is in contradiction with the index-degree lower bound proved in Sect. 4.

(2) *Upper bound.* In the hypothesis the principle is unsatisfiability because of Fact 2. In this case a tree-like refutation of size  $|C|2^{O(w \cdot w(C))}$  for  $F(C, w, r)$  exists as it is shown in [2]. Such proof can be simulated in PC and PCR easily. For PC the absence of dual variables leads to manipulate big representations of polynomials, but the asymptotic complexity of the proof stays the same. For completeness we also show a proof in HN.

We now assume wlog the first  $1 \dots w(C)$  inputs correspond to the minimum satisfying assignment.

We have to prove there are multiples of premises which sum up to 1. Notice that by definition of characteristic functions we have  $1 = \sum_{\alpha \in \{0,1\}^s} x_j(\alpha)$  for any  $j \in [n]$  and also  $1 = \sum_{\beta \in \{0,1\}^s} y_i(\beta)$  for any  $i \in [q]$ . Then we get  $1 = \sum_{\alpha_1 \dots \alpha_{w(C)} \beta} x_1(\alpha_1) \cdots x_{w(C)}(\alpha_{w(C)}) y_i(\beta)$  for any  $i$ , in particular we fix  $i := i(\alpha_1, \dots, \alpha_{w(C)})$  to be such that the  $i$ th row is the one containing a satisfying assignment generated by  $\alpha_1, \dots, \alpha_{w(C)}$ . This immediately implies there is a value  $k$  for which  $C_{ik}$  outputs 1. Fix  $p_0 := x_1(\alpha_1) \cdots x_{w(C)}(\alpha_{w(C)}) y_i(\beta)$  be one of the polynomials in the sum, and let be  $k$  the corresponding activated circuit.

We now show that  $p_0$  can be written as sum of premises: consider the propagation of the satisfying assignment through  $C_{ik}$  (from now on we drop the  $ik$  indexes for sake of notation). There is a minimal sequence of gates  $z^1 \dots z^m$  in the circuit such that  $z^m$  is the output gate,  $z^1 \dots z^{w(C)}$  are the input gates activated by generators, for any AND gate both input gates are predecessors in the sequence, for any OR gate at least one of its predecessor is also a predecessor in the sequence. We denote

$p_l := p_0 z^1 \cdots z^l$ . We prove by backward induction on  $l$  that  $p_l$  is provable in Hilbert Nullstellensatz.

Base case:  $p_m$  is a multiple of  $y_i(\beta)z_{ik}^m$  which is a premise.

Induction step: assuming  $p_l$  is provable. By minimality the gate  $z^l$  is activated by some predecessor(s) in the sequence. Then  $p_{l-1} = p_{l-1}(1 - z^l - \bar{z}^l) + p_{l-1}\bar{z}^l + p_{l-1}z^l$ . The first part comes from boolean axioms, the second part is a multiple of  $y_i(\beta)z_{ik}^A z_{ik}^l$  (respectively  $y_i(\beta)z_{ik}^A z_{ik}^B z_{ik}^l$ ) if the gate is an OR (respectively an AND), the third part comes from inductive hypothesis.

Then  $p_0$  can be proved in  $|C|^{O(1)}$ . The number of such polynomials to prove are  $2^{s \cdot w(C) + s}$ .

To prove that the sum of characteristic functions is 1 it is sufficient an extensive use of boolean axioms of dual variables. This leads to a proof of size  $|C|^{O(1)} \cdot 2^{s \cdot w(C) + s} + 2^{O(s \cdot w(C) + s)}$ . By using the fact that  $s = \Theta(r + w)$  we get the final claims.  $\square$

Lemma 3.1 of [2] can be now be rephrased for PCR and HN, as follows

**Lemma 7** *There exists a polynomial time computable function  $\tau$  which maps any pair  $\langle C, 1^m \rangle$ , where  $C$  is a monotone circuit and  $m$  is an integer into an unsatisfiable CNF  $\tau(C, m)$  such that:*

- *There is a HN proof of  $\tau(C, m)$  of size  $|C|m^{O(\min(w(C), \log m))}$ .*
- *Any PCR refutations of  $\tau(C, m)$  has size at least  $m^{\Omega(\min(w(C), \log m))}$ .*

*Proof* Follow Lemma 3.1 of [2]. Set  $w = \log m/4$  and  $r = \lceil \log m \rceil$  and  $s = \alpha \lceil \log m \rceil$  (see discussion about  $r$ -surjectivity), exactly as Lemma 3.1 of [2]. The formula  $\tau(C, m)$  is  $F(C, w, r) \wedge \tau_m$  where  $\tau_m$  is the pigeon hole principle  $PHP_n^{n+1}$  where  $n = \log^2 m$  and axioms are extended with  $p_{i,j} + \bar{p}_{i,j} - 1$  for  $(i, j) \in [n + 1] \times [n]$ . The  $p_{ij}$  variables of pigeon hole principle are disjoint from  $F(C, w, r)$  ones. The claim follows by Lemma 6 (notice that  $r = \Theta(w)$  and  $s = \Theta(w + r)$ ). Notice that on the  $PHP_n^{n+1}$ , HN polynomially simulates treelike Resolution and that Feasible Interpolation and the Weak Feasible Disjunction properties hold for Polynomial Calculus (see [19, 20]).  $\square$

Theorem 2.5 and Theorem 2.7 in [2] can be rephrased for HN and PCR proof system as follows.

**Theorem 5** *If any of HN, PC and PCR is automatizable then for a fixed  $\epsilon > 0$  there exists an algorithm  $\Phi$  working on monotone circuits  $C$  which runs in time  $\exp(w(C)^{O(1)})|C|^{O(1)}$  and approximates the value of  $w(C)$  to within a factor  $(1 + \epsilon)$ .*

**Theorem 6** *If HN, PC or PCR are automatizable then  $MMCSA \in co - FPR$ .*

*Proof* (Theorems 5 and 6) Refer to Lemma 4.1 in [2]. Because of our lower and upper bounds we can use the same proof for HN, PC or PCR instead of Resolution. Then the proof of both theorems follow in the same way as in [2].  $\square$

## 6 Open Problems

The construction is complex and probabilistic. Would be nice to derandomize it and/or simplify it. A new proof would also help to solve the following open problem: is it possible to find a proof of size *quasi-polynomial* in the size of the shortest proof? It is conjectured that Resolution, PC and PCR cannot be quasi-automatized. The construction used in this paper and in [2] also proves non automatizability for tree-like resolution, which is quasi-automatizable. Thus it is very unlikely that a modification of this construction could be used for non quasi-automatizability.

## References

1. Alekhovich, M., Razborov, A.A.: Lower bounds for polynomial calculus: Non-binomial case. In: 42nd Annual Symposium on Foundations of Computer Science, pp. 190–199 (2001)
2. Alekhovich, M., Razborov, A.A.: Resolution is not automatizable unless  $W[P]$  is tractable. *SIAM J. Comput.* **38**(4), 1347–1363 (2008)
3. Alon, N.: Tools from higher algebra. In: *Handbook of Combinatorics*, vol. 2, pp. 1749–1783. MIT Press, Cambridge (1995)
4. Atserias, A., Bonet, M.L.: On the automatizability of resolution and related propositional proof systems. *Inf. Comput.* **189**(2), 182–201 (2004)
5. Beame, P., Pitassi, T.: Simplified and improved resolution lower bounds. In: 37th Annual Symposium on Foundations of Computer Science, pp. 274–282. IEEE Press, New York (1996)
6. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow—resolution made simple. In: *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pp. 517–526 (1999)
7. Bonet, M.L., Domingo, C., Gavaldà, R., Maciel, A., Pitassi, T.: Non-automatizability of bounded-depth frege proofs. *Comput. Complex.* **13**(1–2), 47–68 (2004)
8. Bonet, M.L., Pitassi, T., Raz, R.: On interpolation and automatization for frege systems. *SIAM J. Comput.* **29**(6), 1939–1967 (2000)
9. Clegg, M., Edmonds, J., Impagliazzo, R.: Using the Groebner basis algorithm to find proofs of unsatisfiability. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pp. 174–183 (1996)
10. Cox, D., Little, J., O’Shea, D.: *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, 3rd edn. Springer, New York (2007)
11. Downey, R., Fellows, M.: *Parameterized Complexity*. Springer, New York (1999)
12. Galesi, N., Lauria, M.: Degree lower bounds for a graph ordering principle. Submitted. See <http://www.dsi.uniroma1.it/~galesi/publications.html>
13. Haken, A.: The intractability of resolution. *Theor. Comput. Sci.* **39**, 297–308 (1985)
14. Impagliazzo, R., Pudlák, P., Sgall, J.: Lower bounds for the polynomial calculus and the Gröbner basis algorithm. *Comput. Complex.* **8**(2), 127–144 (1999)
15. Jukna, S.: *Extremal Combinatorics: with Applications in Computer Science*. Springer, New York (2001)
16. Krajčec, J.: Interpolation and approximate semantic derivations. *Math. Log. Q.* **48**(4), 602–606 (2002)
17. Krajčec, J., Pudlák, P.: Some consequences of cryptographical conjectures for  $S_2^1$  and ef. In: Leivant, D. (ed.) *LCC. Lecture Notes in Computer Science*, vol. 960, pp. 210–220. Springer, Berlin (1994)
18. Pitassi, T.: Algebraic propositional proof systems. In: Immerman, N., Kolaitis, P.G. (eds.) *Descriptive Complexity and Finite Models. DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 31, pp. 215–244. Am. Math. Soc., Providence (1996)
19. Pudlák, P.: On reducibility and symmetry of disjoint np-pairs. *Theor. Comput. Sci.* **295**, 626–638 (2003)
20. Pudlák, P., Sgall, J.: Algebraic models of computation and interpolation for algebraic proof systems. *DIMACS Ser. Theor. Comput. Sci.* **39**, 279–296 (1998)
21. Razborov, A.A.: Lower bounds for the polynomial calculus. *Comput. Complex.* **7**(4), 291–324 (1998)
22. van Lint, J.H.: *Introduction to Coding Theory*, 3rd edn. Graduate Texts in Mathematics. Springer, New York (1998)