FUNDAMENTA
MATHEMATICAE
182 (2004)

Diagonalization in proof complexity

by

Jan Krajíček (Praha)

Abstract. We study diagonalization in the context of implicit proofs of [10]. We prove that at least one of the following three conjectures is true:

- There is a function $f : \{0,1\}^* \to \{0,1\}$ computable in $\mathcal{E}$ that has circuit complexity $2^{\Omega(n)}$.
- $\mathcal{NP} \neq \mathrm{co}\mathcal{NP}$.
- There is no p-optimal propositional proof system.

We note that a variant of the statement (either $\mathcal{NP} \neq \mathrm{co}\mathcal{NP}$ or $\mathcal{NE} \cap \mathrm{co}\mathcal{NE}$ contains a function $2^{\Omega(n)}$ hard on average) seems to have a bearing on the existence of good proof complexity generators. In particular, we prove that if a minor variant of a recent conjecture of Razborov [17, Conjecture 2] is true (stating conditional lower bounds for the Extended Frege proof system EF) then actually unconditional lower bounds would follow for EF.

The only method for demonstrating unprovability of a Π^0_1-sentence in a theory (containing some amount of arithmetic) is diagonalization. One would like to adapt this to a non-uniform setting in which theories are replaced by propositional proof systems and Π^0_1-sentences by tautologies. This fails, at least in the straightforward adaptation. In particular, many strong proof systems do prove their own consistency by polynomial size proofs (cf. [5, Chpt. 14]). However, there is a non-uniform setting in first-order theories where diagonalization gives non-trivial lower bounds.

To explain this we need to fix some notation first. By a theory we shall mean a set of axioms, not the set of its consequences. We shall assume (without loss of generality) that the language of all theories in question is the language of bounded arithmetic theory S^1_2. For a natural number N

2000 *Mathematics Subject Classification*: Primary 68Q15; Secondary 03F20, 68Q17.

Partially supported by grant # A 101 94 01 of the Academy of Sciences of the Czech Republic and by project LN00A056 of The Ministry of Education of the Czech Republic.

The author is also member of the Institute for Theoretical Computer Science of the Charles University. Part of this work was done while he was visiting the Mathematical Institute at Oxford.

the symbol $\underline{N}$ is the dyadic numeral inductively defined by: $\underline{0} := 0$, $\underline{1} := 1$, $\underline{2} := (1+1)$, $\underline{2k} := (\underline{2} \cdot \underline{k})$, and $\underline{2k+1} := (\underline{2k} + 1)$.

Given a theory T definable in the language of S^1_2, let $\mathrm{Prf}_T(y, x)$ be a formula in the language of S^1_2 expressing that y is a T-proof of formula x. There is a canonical formalization of the syntax of logic (terms, formulas, proofs, etc.) and of the notion of provability (see [13] or [5] for details), and its usual properties are provable in S^1_2. Assuming that $T \supseteq S^1_2$, as we shall do, thus allows us to use this formalization in T too.

Define a diagonal formula $A(x)$ satisfying

$$A(x) \equiv \forall y, |y| \le s(x) \to \neg\, \mathrm{Prf}_T(y, \ulcorner A(\dot{x}) \urcorner),$$

where $s(x)$ is any term and $\ulcorner A \urcorner$ is a (canonical) number encoding A such that its length is proportional to the size of A, and $\dot{x}$ is the formalization of dyadic numerals. Then, by the standard argument, we have the following theorem. (The condition that $T \in \mathcal{NP}$ is technical and implies, in particular, that T is definable and that the binary relation $\mathrm{Prf}_T(y, x)$ is in $\mathcal{NP}$ and definable by a Σ^b_1-formula, and that all its true instances have polynomial size proofs in S^1_2; cf. [5, p. 303].)

THEOREM 0.1. *Let $T \supseteq S^1_2$ be a consistent theory such that $T \in \mathcal{NP}$. Then for any $N \ge 1$ the sentence $A(\underline{N})$ is true and provable in T but any T-proof of the sentence must have size at least $s(N)$.*

Note that the length of the formula $A(\underline{N})$ is $O(\log N)$ and hence the lower bound is non-trivial, as long as $s(N) \gg \log(N)$. A foremost example of this reasoning is the Finitistic Gödel theorem of Friedman [3] and Pudlák [13, 14] (the theorem says that any T-proof of a formula $\mathrm{Con}_T(\underline{N}) := \forall y, |y| \le \underline{N} \to \neg\, \mathrm{Prf}(y, \ulcorner 0 = 1 \urcorner)$ expressing the consistency of T with respect to proofs of length at most N, must have size at least $N^{\Omega(1)}$). Other, quantitatively more subtle applications, can be found in bounded arithmetic (cf. [5, Chpt. 10]).

One would like to adapt this to propositional proof complexity. The problem with this is not that we deal with first-order theories (as we know how to pass between them and propositional proof systems) but with the fact that $A(x)$ is not a bounded formula and the instances $A(\underline{N})$ translate into propositional formulas of length at least $s(N)$, which is superpolynomial in the length of $\underline{N}$ in order to get a non-trivial lower bound in Theorem 0.1. An idea suggests itself at this point: to use the concept of implicit proofs from [10]. This is what we investigate in this paper. A variant of the idea of implicit proofs is recalled in Section 1. In Section 2 we derive the theorem mentioned in the abstract. Its variant is derived in Section 3 and linked to proof complexity generators in Section 4.

The paper is self-contained, assuming that the reader has a general background in bounded arithmetic and proof complexity (only basic things are assumed); see [5]. Let us just recall the terminology and some more notation. $\mathrm{Time}(t(n))$ is the class of languages computable in deterministic time $O(t(n))$, $\Sigma_i\,\mathrm{Time}(t(n))$ is the Σ_i-level of the time $O(t(n))$ hierarchy, $\mathrm{NTime}(t(n)) = \Sigma_1\,\mathrm{Time}(t(n))$, and $\mathcal{E} = \mathrm{Time}(2^{O(n)})$. For a function $f : \{0,1\}^* \to \{0,1\}$, $C_f(n)$ is the circuit complexity of computing f on $\{0,1\}^n$, while $H_f(n)$ is its hardness on average in the sense of [12]. A "proof system" tacitly means a "propositional proof system": it is a nondeterministic acceptor of the set of propositional tautologies in the De Morgan language. A proof of a tautology in a proof system is any particular computation of the proof system accepting the formula. A proof system is *p-bounded* iff the proof system accepts all tautologies in a fixed polynomial time, and it is *p-optimal* iff it polynomially simulates other proof systems (cf. [2]). The length of a string w is denoted $|w|$, and a number is identified with the string of its bits (i.e. $|m| \sim \log(m)$).

A prominent proof system is the Extended Frege proof system EF. It is the usual Hilbert style calculus based on a finite number of axiom schemes and inference rules (sound and implicationally complete in a complete language) that has an additional ability (via the so called Extension Rule) to abbreviate large formulas occurring in a proof by new atoms (see [2, 5] for details). Alternatively one can view EF as a Hilbert style calculus operating with circuits rather than only with formulas. The prominence of EF comes (at least in our context) from its relation to theory S^1_2.

Finally let us recall a well-known translation of formulas into propositional formulas, just for the case of $A(x)$. Given $N \geq 1$, there is a propositional 3DNF formula denoted $\|A(x)\|_N$ of size $s(N)^{O(1)}$ that expresses (by the fact of being a tautology) that $A(N)$ is true. It is constructed as in the proof of the $\mathcal{NP}$-completeness of satisfiability as follows. Fix a particular Σ^b_1-formula ($\mathcal{NP}$-definition) Prf_T. The formula $\|A(x)\|_N$ has $s(N)$ atoms for bits of a potential y, $s(N)^{O(1)}$ atoms for bits of a potential witness z to the validity of the $\mathcal{NP}$-statement $\mathrm{Prf}_T(y, \ulcorner A(\underline{N})\urcorner)$, and an additional $s(N)^{O(1)}$ auxiliary atoms. The auxiliary atoms are used for naming bits in the canonical computation of the truth value of $\mathrm{Prf}_T(y, \ulcorner A(\underline{N})\urcorner)$ with witness z. The formula $\|A(x)\|_N$ is built by induction on the logical complexity of A and says that if all local conditions in the computation are satisfied then $\mathrm{Prf}_T(y, \ulcorner A(\underline{N})\urcorner)$ fails. See [5, Chpt. 9] for details of the translation.

1. Implicit proofs of implicit formulas. The idea of implicit proofs is to represent a proof in a proof system, a binary string, not by the string itself but by a circuit that computes any individual bit of the proof knowing only its position but not the whole proof. A proof can be, in principle, exponentially larger than a circuit representing it. Hence a proof system operating with such circuits instead of directly with proofs can be quite powerful. This concept has been proposed and studied in [10]. In this paper we borrow the general idea of representing long proofs (and, in fact, also long formulas) by small circuits but otherwise we do not use anything from [10]. Let us now proceed formally.

Let w be a 0-1 string of length 2^k. Identify $i < 2^k$ with vectors $i = (i_1, \ldots, i_k) \in \{0,1\}^k$ ordered lexicographically. We say that a circuit $C(x_1, \ldots, x_k)$ *represents* w if $C(i_1, \ldots, i_k) = w_i$ (the ith bit of w) for all $i < 2^k$. Similarly, if W is a 0-1 $2^k \times 2^k$ matrix then we say that a circuit $D(x_1, \ldots, x_k, y_1, \ldots, y_k)$ *represents* W if $D(i,j) = W_{i,j}$ for all $i, j < 2^k$.

Let M be a non-deterministic polynomial time machine. For any input w of length 2^l and represented by a circuit $C(z_1, \ldots, z_l)$, for suitable $k = O(l)$ (given by the time of M) and any $2^k \times 2^k$ matrix W represented by a circuit $D(x_1, \ldots, x_k, y_1, \ldots, y_k)$ we can write down a propositional formula $\sigma^M_{C,D}$ expressing that W is an accepting computation of M (given by W listing in its rows all instantaneous descriptions of the computation) on input w. Moreover, the size of $\sigma^M_{C,D}$ is only proportional to the sizes of C and D. Such a formula can be constructed as follows.

The formula has atoms $x_1, \ldots, x_k, y_1, \ldots, y_k, z_1, \ldots, z_l$ for inputs of D and C respectively. By using these atoms any local condition imposed on W by M (there are a finite number of them) can be written by a circuit of size proportional to the sizes of C and D (as a local condition speaks only about a constant number of positions in W). This means that the circuit computes constantly value 1 (for all evaluations of the atoms, i.e. for all positions in W) iff all local conditions are satisfied on the whole of W. By using additional auxiliary atoms for values of subcircuits of C and D this can be expressed by a formula (as in the proof of the $\mathcal{NP}$-completeness of satisfiability). To summarize: $\sigma^M_{C,D}$ is a tautology iff W is indeed an accepting computation of M on w.

We will need one additional formula, this time first-order. Assume that a circuit $C(x_1, \ldots, x_k)$ represents a 3DNF formula φ_C (of size at most 2^k). Let $\ulcorner C \urcorner$ be its number code. Then there is a formula $\mathrm{BigTaut}(x)$ such that $\mathrm{BigTaut}(\ulcorner C \urcorner)$ formalizes that φ_C is a tautology. This can be expressed without going through φ_C: For any truth assignment y (of length bounded by a suitable term in C) there is a term in the 3DNF formula represented by C (this quantifies over inputs of C and not over the whole big formula) satisfied by y.

We formulate the following lemma for the diagonal formula $A(x)$ only, but it holds for any formula of a similar logical form.

LEMMA 1.1. *There is a polynomial time function $g(x)$ such that, for any $N \geq 1$, $g(N)$ is (the number code of) a circuit representing (in the sense as* *above) the formula* $\|A(x)\|_N$. *Moreover, the valid implication*

$$\mathrm{BigTaut}(g(x)) \to A(x)$$

is provable in theory S^1_2.

Proof. The existence of such a function g follows from the high uniformity (for a fixed formula $A(x)$) of the construction of $\|A(x)\|_N$ as recalled at the end of the introductory section. In fact, there is a polynomial time function $g^*_A(x,y)$ such that $g^*_A(N,i)$ computes the ith bit of $\|A(x)\|_N$ ($g(N)$ is then the circuit computing $g^*_A(N,y)$ for y bounded by a suitable term $N^{O(1)}$). The function g^*_A is defined by induction on the logical complexity of A from analogous functions g^*_B for subformulas B of A. The claim for the base case of atomic formulas follows as the local conditions are the same across the whole computation (of the values of terms in the atomic formula) and the algorithm computing g^*_B just needs to know a position in the computation tableaux (i.e. the index i) in order to use the appropriate atoms while writing down the particular local conditions in the propositional translation.

The validity of the implication in the second part of the lemma is clear from the definition of the propositional translation. To see that it is also provable in S^1_2, reason as follows. Assume that y is a witness for the failure of $A(x)$: $|y| \le x \wedge \mathrm{Prf}_T(y, \ulcorner A(\dot{x})\urcorner)$. Then there is a truth assignment $f(x,y)$, computed from x and y by a polynomial time function f, to atoms of $\|A(x)\|_x$ that violates the formula (cf. [5, Sec. 9.3] for an analogous statement and a detailed discussion). We claim that S^1_2 proves (from the assumption that y witnesses the failure of $A(x)$) that "$f(x,y)$ does not satisfy any term in the 3DNF formula represented by $g(x)$" (and hence $\neg\,\mathrm{BigTaut}(g(x))$). This is showed again by induction on logical complexity of A, the base case being that of atomic formulas. For them the claim is proved by induction on the size of circuits computing the terms in the atomic formula, i.e. by length induction on parameter y. ∎

Note that since g is polynomial time computable, the circuit $g(N)$ has size at most $\log(N)^{O(1)}$.

2. A theorem

THEOREM 2.1. *At least one of the following three statements is true*:

(i) *There is a function* $f : \{0,1\}^* \to \{0,1\}$ *computable in* $\mathcal{E}$ *that has circuit complexity* $2^{\Omega(n)}$.

(ii) $\mathcal{NP} \ne \mathrm{co}\mathcal{NP}$.

(iii) *There is no p-optimal propositional proof system.*

Proof. Before we start with the actual proof let us recall a well-known technical fact (see e.g. [5, p. 303]) that will be used several times in the proof (in steps 2, 9 and 11): Any true Σ_1^b-sentence has a polynomial size proof in S_2^1. This is proved by induction on the logical complexity of the sentence.

Let us begin with the proof proper. We shall assume that all three statements are false and derive a contradiction via Theorem 0.1.

1. Let P be a proof system witnessing that both (ii) and (iii) fail; then P is p-bounded and also p-optimal. Assume without loss of generality that P contains EF. Let $\mathrm{Prf}_P(u,v)$ be a Σ_1^b-formula formalizing that "u is a P-proof of formula v".

2. Define a theory T to be S_2^1 augmented by an extra axiom, a form of reflection principle:

$$\forall x, y, u, \quad \mathrm{Prf}_P(u, \ulcorner \sigma^P_{x,y} \urcorner) \to \mathrm{BigTaut}(x).$$

Note that the antecedent of the implication is a Σ_1^b-formula because the (code of the) formula $\sigma^P_{x,y}$ is polynomial time computable from x, y (for fixed P, cf. Section 1). This implies, in particular, that all true instances of the formula have polynomial size proofs in S_2^1.

3. Let $A(x)$ be the diagonal formula from the introduction, with the term $s(x)$ being simply x. Consider the propositional formula $\|A(x)\|_N$. The formula is a tautology as $A(N)$ is true. The size of the formula is $N^{O(1)}$ but by Lemma 1.1 there is a circuit $C_N := g(N)$ of size $n^{O(1)}$, $n := \log(N)$, representing the formula (in the sense of Section 1).

4. The set of all formulas $\|A(x)\|_N$, $N \geq 1$, is polynomial time decidable and hence we can use it as axioms in some proof system. By the hypothesis that P is p-optimal there is a (deterministic) polynomial time algorithm M computing from the string $\|A(x)\|_N$ a P-proof of $\|A(x)\|_N$.

5. The output of M is a particular accepting computation of P, i.e. a $2^{O(n)} \times 2^{O(n)}$ matrix W^N encoding the computation. As M runs in deterministic polynomial time, $W^N_{i,j}$ as a function of $i, j \in \{0,1\}^{O(n)}$ is in $\mathcal{E}$.

6. Assuming that also statement (i) fails, there exists a circuit $D(i,j)$ in two $O(n)$ variables and of size $2^{\delta \cdot n}$ that represents W^N, for arbitrarily small $\delta > 0$. We shall choose a particular δ in step 12.

7. Take an instance of the reflection principle by substituting for x and y the codes of D and C_N respectively:

$$\forall u, \quad \mathrm{Prf}_P(u, \ulcorner \sigma^P_{\ulcorner C_N \urcorner, \ulcorner D \urcorner} \urcorner) \to \mathrm{BigTaut}(\ulcorner C_N \urcorner).$$

8. By Section 1 the size of $\sigma^P_{C_N,D}$ is polynomial in the sizes of C_N and D, i.e. it is $2^{O(\delta \cdot n)}$. Now we use the hypothesis that P is also p-bounded. Hence there is a P-proof e of $\sigma^P_{C_N,D}$ of size $2^{O(\delta \cdot n)}$. Note that the constants implicit in the O-notation are fixed and independent of δ. Substituting $\ulcorner e \urcorner$ for u in

the formula in step 7 we get

$$\mathrm{Prf}_P(\ulcorner e\urcorner, \ulcorner\sigma^P_{\ulcorner C_N\urcorner,\ulcorner D\urcorner}\urcorner) \to \mathrm{BigTaut}(\ulcorner C_N\urcorner).$$

9. The antecedent of the formula in step 8 is a true Σ^b_1-sentence of size $2^{O(\delta\cdot n)}$ and has a proof in S^1_2 (and hence in T) of polynomial size, i.e. of size $2^{O(\delta\cdot n)}$.

10. Applying modus ponens to the formulas in steps 8 and 9 we get a proof of size $2^{O(\delta\cdot n)}$ of the sentence $\mathrm{BigTaut}(\ulcorner C_N\urcorner)$.

11. We claim that the implication

$$\mathrm{BigTaut}(\ulcorner C_N\urcorner) \to A(\underline{N})$$

has an S^1_2-proof of size $n^{O(1)}$. This is because it can be obtained from an instance of a universal implication (provable in S^1_2 by Lemma 1.1)

$$\mathrm{BigTaut}(g(x)) \to A(x)$$

by substituting for x the numeral $\underline{N}$, and by proving $\ulcorner C_N\urcorner = g(\underline{N})$: That is a true Σ^b_1-sentence of size $n^{O(1)}$ and has a polynomial size proof in S^1_2.

12. Putting steps 10 and 11 together we get a size $2^{O(\delta\cdot n)}$ proof in T of $A(\underline{N})$. Taking $\delta > 0$ so small that $2^{O(\delta\cdot n)} < N$ (this can be done as the O-constant is independent of δ) we get a contradiction with Theorem 0.1. ∎

Let us remark that instead of using $\mathcal{E}$ and circuit size $2^{\Omega(n)}$ in (i) we could have used $\mathrm{Time}(t(n))$ and circuit size $t(n)^{\Omega(1)}$, as long as there is no polynomial upper bound for $t(n)$. This follows by a padding argument or by a simple change to the proof above: Use the diagonal formula for the term $s(x) := t(|x|)$ instead of $s(x) := x$ in step 3.

3. A variant of the theorem. It is not difficult to see that the property of a string to be the truth table of a function on $\{0,1\}^n$ with $2^{\delta\cdot n}$ circuit complexity, or even $2^{\delta\cdot n}$ hard on average, is in the polynomial time hierarchy $\mathcal{PH}$. Taking the lexicographically first such strings (at least one exists of each length 2^n, $n \gg 0$, by a simple counting) we see that there is such an f computable in $\mathcal{E}^{\mathcal{PH}}$.

If $\mathcal{NP} = \mathrm{co}\mathcal{NP}$ then such an f is in $\mathcal{E}^{\mathcal{NP}\cap\mathrm{co}\mathcal{NP}} = \mathcal{NE}\cap\mathrm{co}\mathcal{NE}$. If, in addition, $\mathcal{E} = \mathcal{NE}$ then such an f is in $\mathcal{E}$.

This simple (apparently folklore) argument yields the following theorem [1]. We shall use only part (ii) in Section 4 but we state also part (i) as it is actually a weaker version (if hardness on average is replaced by circuit size) of Theorem 2.1: It is known, by [11], that the non-existence of a

[1] This argument has been pointed out to me by E. Jeřábek, and has also been noted by V. Kabanets, and replaces my original proof: A simple modification of the proof of Theorem 2.1 shows that $\mathcal{NP} = \mathrm{co}\mathcal{NP}$ implies that $\mathcal{NE}\cap\mathrm{co}\mathcal{NE}$ contains a function with exponential circuit complexity which was then turned into a function with exponential hardness on average by the construction from [4].

p-optimal proof system implies that $\mathrm{Time}(t(n)) \neq \mathrm{NTime}(t(n))$ (as long as $t(n) \le 2^{n^{O(1)}}$) and also $\mathcal{E} \neq \mathcal{NE}$ (the opposite implication is unknown but Verbitsky [18] constructed a relativized world where it does not hold).

THEOREM 3.1. (i) *At least one of the following three statements is true*:

(a) *There is a function* $f : \{0,1\}^* \to \{0,1\}$ *computable in* $\mathcal{E}$ *that has* $2^{\Omega(n)}$ *hardness on average.*
(b) $\mathcal{NP} \neq \mathrm{co}\mathcal{NP}$.
(c) $\mathcal{E} \neq \mathcal{NE}$.

(ii) *At least one of the following two statements is true*:

(a) *There is a function* $f : \{0,1\}^* \to \{0,1\}$ *computable in* $\mathcal{NE} \cap \mathrm{co}\mathcal{NE}$ *that has* $2^{\Omega(n)}$ *hardness on average.*
(b) $\mathcal{NP} \neq \mathrm{co}\mathcal{NP}$.

4. Proof complexity generators. By a *proof complexity generator* we mean a map $g : \{0,1\}^n \to \{0,1\}^m$, $m = m(n)$ and $m > n$, whose bits can be computed in $\mathrm{NTime}(m(n)^{O(1)}) \cap \mathrm{coNTime}(m(n)^{O(1)})$. This assumption about the computability of the bits is the weakest one allowing us to write down, for any $b \in \{0,1\}^m$, a size $m^{O(1)}$ propositional formula $\tau_b(g)$ that is a tautology iff $b \notin \mathrm{Rng}(g)$ (see below). (The notation $\tau_b(g)$ is somewhat misleading as the formula depends on a particular definition of g and not only on g, but we will ignore this here: The lower bounds conjectured later should hold for all $\mathrm{NTime}(m(n)^{O(1)}) \cap \mathrm{coNTime}(m(n)^{O(1)})$-definitions.)

A generator is good if for a strong proof system P and with high probability in choosing a random $b \in \{0,1\}^m$, the formula $\tau_b(g)$ requires very long (in particular, superpolynomial) P-proofs. The quality of the generator is measured by the strength of P and by the probability that b yields a hard τ-formula. At present it is not ruled out that some generator g works for all P and all b. Following [17] we shall say that a generator g *is hard for* P if all $\tau_b(g)$, for all b's, require superpolynomial size P-proofs (cf. [17]).

The τ-formulas have been defined in [6] and independently in [1], and their theory has made first steps in [7, 16, 8, 17]. I shall not describe the development of the ideas and known lower bound results; this can be found in the introductions to [8] or [17]. Instead I shall briefly describe one motivation and why we speak about "generators".

Proving lower bounds for strong propositional proof systems appears hard. In fact, we do not know any such lower bounds. A factor contributing to this is that it is actually not easy to come up with sensible tautologies that would be good candidates for requiring long proofs even in strong systems (cf. [8] for a detailed discussion). The τ-formulas seem to be candidates worth studying in this context.

The word "generator" is used because some of the usual pseudo-random number generators seem to be good candidates. In particular, a good proof complexity generator must behave as a hitting set generator with respect to the $\mathcal{NP}$/poly-test (cf. [8]).

Just as the existence of good pseudo-random generators can be proved under some computational hardness assumptions (cf. [12, 4]) we may also try to reduce the existence of good proof complexity generators to a suitable computational hardness assumption. This is discussed in [9] in a broader perspective, and in the introduction to [17].

The most studied map in this context is the classic Nisan–Wigderson generator (cf. [12]). This has been proposed as possibly a good proof complexity generator in [1] and taken up in [8], although the motivations (and, more importantly, the choice of parameters in the construction and the formalization of the notion of *hardness* of the generators (2)) are different. Let us first recall the definition of NW-generators (and fix the notation in the process).

Let A be an $m \times n$ 0-1 matrix with l ones per row. Set $J_i(A) := \{j \le n \mid A_{ij} = 1\}$. Let $f : \{0,1\}^l \to \{0,1\}$ be a boolean function. Then $\mathrm{NW}_{A,f} : \{0,1\}^n \to \{0,1\}^m$ is the NW-generator based on A and f: the ith bit of output is computed by f from the bits of the input that belong to $J_i(A)$.

Assume that f is in $\mathrm{NTime}(t(n)) \cap \mathrm{coNTime}(t(n))$. Given particular $\mathrm{NTime}(t(n))$-definitions α: $\exists v, \alpha_\varepsilon(u,v)$ ($|u| = l$, $|v| \le t(n)$ and α_ε p-time) of $f(u) = \varepsilon$, for $\varepsilon = 0,1$, the τ-formulas are defined by

$$\tau_b^\alpha := \bigvee_{i \le m} \neg \alpha_{b_i}(x \downarrow J_i(A), v^i),$$

where $b = (b_1, \ldots, b_m) \in \{0,1\}^m$, x is an n-tuple of variables and v^i are disjoint $t(n)$-tuples of variables. Clearly $\tau_b \in \mathrm{TAUT}$ iff $b \notin \mathrm{Rng}(\mathrm{NW}_{A,f})$. Note that the size of τ_b^α is $t(n)^{O(1)} \cdot m(n)$, and hence we get a size $m^{O(1)}$ formula as long as $t(n) \le m(n)^{O(1)}$.

An idea, formulated in [1] in general terms and then quite specifically in [17], is that $\mathrm{NW}_{A,f}$ forms a good proof complexity generator, as long as A has suitable combinatorial properties (being an (l,d) combinatorial design in the sense of [12]: $J_i(A)$'s have size l and the intersection of any two different rows has size $\le d$) and as long as f is computationally hard. Specifically, Razborov [17] has made the following conjecture.

Conjecture 4.1 (A. A. Razborov [17, Conjecture 2]). *Any* NW-*generator based on a matrix A which is a combinatorial design with the same*

(2) I shall describe neither the set-up from [8] nor the conjectured hardness in terms of pseudo-surjectivity here as this is not relevant to the topic of this paper.

parameters as in [12] *and on any function* f *in* $\mathcal{NP} \cap \mathrm{co}\mathcal{NP}$ *that is hard on average for* $\mathcal{P}/\mathrm{poly}$, *is hard for* EF.

Let us interpret the specifications. The parameters in the main construction of combinatorial designs in [12, L.2.5] satisfy: $d = \log(m)$, $\log(m) \leq l \leq m$ and $n = O(l^2)$ ([3]). Writing this dually:

$$m = 2^{\varepsilon \cdot n^{1/2}}, \quad l = \varepsilon \cdot n^{1/2}, \tag{1}$$

where $\varepsilon > 0$ is a constant (determined by the O-constant in the expression $n = O(l^2)$). We are taking the maximal value allowed for m by [12, L.2.5] as that is the chief case in [12], and it also links with [16, 17] studying "function" generators.

The phrase "hard on average for $\mathcal{P}/\mathrm{poly}$" presumably means that the hardness of f in the sense of [12] is exponential, as needed in [12]:

$$H_f(l) \geq 2^{\Omega(l)}. \tag{2}$$

The conjecture requires that $f \in \mathcal{NP} \cap \mathrm{co}\mathcal{NP}$. We shall relax this condition to

$$f \in \mathrm{NTime}(2^{O(l)}) \cap \mathrm{coNTime}(2^{O(l)}). \tag{3}$$

By (1) we have $2^{O(l)} = m^{O(1)}$ so (3) means that f is in $\mathrm{NTime}(m^{O(1)})) \cap \mathrm{coNTime}(m^{O(1)})$. By the earlier discussion this means that the size of the τ-formula will still be $m^{O(1)}$. Thus this modification of the original formulation of the conjecture seems quite harmless, and moreover, very much in the spirit of [12] allowing one to compute the output bits of the generator in time $m^{O(1)}$ rather than just $n^{O(1)}$.

Nevertheless, (3) is a modification of the original specifications and so we do not want to talk about Conjecture 4.1 in the next theorem. For this reason let us formulate the following *Statement* R:

R Let g be an NW-generator based on an $m \times n$ matrix A that is an $(l, \log(m))$ combinatorial design and on any function f such that the constraints in (1), (2) and (3) are satisfied. Then g is hard for EF.

THEOREM 4.2. *Assume that Statement* R *is true. Then* EF *is not p-bounded.*

Proof. Assume that EF is p-bounded. We shall arrive at a contradiction with Statement R. If EF is p-bounded then, in particular, $\mathcal{NP} = \mathrm{co}\mathcal{NP}$.

By Theorem 3.1(ii) there is a function f in $\mathcal{NE} \cap \mathrm{co}\mathcal{NE}$ that is exponentially hard on average. Having such an f we may apply Statement R

([3]) There is also a construction specific for the application to $\mathcal{BPP}$ with parameters satisfying $l = c \cdot \log(m)$ and $n = O(c^2 \cdot \log(m))$, where c is a constant (cf. [12, L.2.6]). That is not good in proof complexity as a trivial proof of the τ-formulas going through all possible seeds would have a polynomial size.

(suitable matrices A are constructed in [12]) to conclude that EF is not p-bounded. That is a contradiction. ■

Note that if we postulate smaller m then although the size of the formula might not be polynomial in m any more it will still be superpolynomially smaller than $2^{O(n)}$, i.e. than a lower bound to the size of the trivial proof going through all seeds.

An optimist may conclude that one only has to prove a conditional statement in order to prove that EF is not p-bounded. A pessimist may conclude that if the conclusion of R holds even without its hypothesis then the hypothesis is irrelevant. I think an interesting modification of R may be obtained by stating it for one particular function f, e.g. the hard bit of the discrete logarithm, and maybe claiming the lower bound for $\tau_b(g)$ for a random b with high probability, rather than for all b's.

Acknowledgements. I thank E. Jeřábek and P. Pudlák for discussions and suggestions. In particular, P. Pudlák suggested that some variant of Theorem 2.1 should be relevant to proof complexity generators. I thank the referee for valuable comments.

References

[1] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson, *Pseudorandom generators in propositional proof complexity*, in: Electronic Colloquium on Computational Complexity, Rep. No. 23, 2000. Extended abstract in: Proc. 41st Annual Sympos. on Foundation of Computer Science, 2000, 43–53.

[2] S. A. Cook and A. R. Reckhow, *The relative efficiency of propositional proof systems*, J. Symbolic Logic 44 (1979), 36–50.

[3] H. Friedman, *On the consistency, completeness, and correctness problems*, unpublished preprint, 1979.

[4] R. Impagliazzo and A. Wigderson, *$P = BPP$ unless E has sub-exponential circuits: derandomizing the XOR lemma*, in: Proc. 29th Annual ACM Symposium on Theory of Computing, 1997, 220–229.

[5] J. Krajíček, *Bounded Arithmetic, Propositional Logic, and Complexity Theory*, Encyclopedia Math. Appl. 60, Cambridge Univ. Press, 1995.

[6] —, *On the weak pigeonhole principle*, Fund. Math. 170 (2001), 123–140.

[7] —, *Tautologies from pseudo-random generators*, Bull. Symbolic Logic 7 (2001), 197–212.

[8] —, *Dual weak pigeonhole principle, pseudo-surjective functions, and provability of circuit lower bounds*, J. Symbolic Logic 69 (2004), 265–286.

[9] —, *Hardness assumptions in the foundations of theoretical computer science*, Arch. Math. Logic, to appear.

[10] —, *Implicit proofs*, J. Symbolic Logic 69 (2004), 387–397.

[11] J. Krajíček and P. Pudlák, *Propositional proof systems, the consistency of first order theories and the complexity of computations*, ibid. 54 (1989), 1063–1079.

[12] N. Nisan and A. Wigderson, *Hardness vs. randomness*, J. Comput. System Sci. 49 (1994), 149–167.

[13] P. Pudlák, *On the length of finitistic consistency statements in first order theories*, in: Logic Colloquium 84, North-Holland, 1986, 165–196.

[14] —, *Improved bounds to the length of proofs of finitistic consistency statements*, in: Contemp. Math. 65, Amer. Math. Soc., 1987, 309–331.

[15] A. A. Razborov, *Unprovability of lower bounds on the circuit size in certain fragments of bounded arithmetic*, Izv. Ross. Akad. Nauk Ser. Mat. 59 (1995), 201–224 (in Russian).

[16] —, *Resolution lower bounds for perfect matching principles*, in: Proc. 17th IEEE Conf. on Computational Complexity, 2002, 29–38.

[17] —, *Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution*, preprint, 2003.

[18] O. V. Verbitsky, *Optimal algorithms for* co-NP *sets and the problem* EXP $=^?$ NEXP, Mat. Zametki 50 (1991), no. 2, 37–46 (in Russian).

Mathematical Institute
Academy of Sciences
Žitná 25
CZ 115 67 Praha 1, Czech Republic
E-mail: krajicek@math.cas.cz

Received 6 January 2004;
in revised form 6 June 2004