Natural Proofs

Alexander A. Razborov
School of Mathematics
Institute for Advanced Study
Princeton, NJ 08540
and
Steklov Mathematical Institute
Vavilova 42, 117966, GSP-1
Moscow, RUSSIA

Steven Rudich
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15212

November 30, 1996

Abstract

We introduce the notion of natural proof. We argue that the known proofs of lower bounds on the complexity of explicit Boolean functions in non-monotone models fall within our definition of natural. We show based on a hardness assumption that natural proofs can’t prove super-polynomial lower bounds for general circuits. Without the hardness assumption, we are able to show that they can’t prove exponential lower bounds (for general circuits) for the discrete logarithm problem. We show that the weaker class of \( AC^0 \)-natural proofs which is sufficient to prove the parity lower bounds of Furst, Saxe, and Sipser, Yao, and Hastad is inherently incapable of proving the bounds of Razborov and Smolensky. We give some formal evidence that natural proofs are indeed natural by showing that every formal complexity measure which can prove super-polynomial lower bounds for a single function, can do so for almost all functions, which is one of the two requirements of a natural proof in our sense.

*Supported by the grant # 93-6-6 of the Alfred P. Sloan Foundation, by the grant # 93-011-16015 of the Russian Foundation for Fundamental Research, and by an AMS-FSU grant

†Partially supported by NSF grant CCR-9119319
1. Introduction

It is natural to ask what makes lower bound questions such as $P \not\leq PSPACE$, $P \not\leq NP$, and $P \not\leq NC$ so difficult to solve. A non-technical reason for thinking they are difficult might be that some very bright people have tried and failed — but this is hardly satisfactory. A technical reason along the same lines would be provided by a reduction to these questions from another problem known to be really hard such as the Riemann Hypothesis. Perhaps the ultimate demonstration that $P \not\leq NP$ is a hard problem would be to show it to be independent of set theory (ZFC).

Another way to answer this question is to demonstrate that known methods are inherently too weak to solve problems such as $P \not\leq NP$. This approach was taken in Baker, Gill, and Solovay [7], who used oracle separation results for many major complexity classes to argue that relativizing proof techniques could not solve these problems. Since relativizing proof techniques involving diagonalization and simulation were the only available tools at the time of their work, progress along known lines was ruled out.

Because of this, people began to study these problems from the vantage of Boolean circuit complexity, rather than machines. The new goal is to prove a stronger, non-uniform version of $P \not= NP$, namely that SAT (or some other problem in $NP$) does not have polynomial-size circuits. Many new proof techniques have been discovered and successfully applied to prove lower bounds in circuit complexity, as exemplified by [11, 1, 40, 14, 27, 28, 3, 2, 37, 4, 29, 36, 8, 5, 23, 24, 15, 13, 17, 26, 6] among others, although the lower bounds have not come up near the level of $P$ or even $NC$. These techniques are highly combinatorial, and in principle they are not subject to relativization. They exist in a much larger variety than their recursion-theoretic predecessors. Even so, in this paper we give evidence of a general limitation on their ability to resolve $P \not\leq NP$ and other hard problems.

Section 2 introduces and formalizes the notion of a natural proof. We argue that all lower bound proofs known to date against non-monotone Boolean circuits are natural, or can be represented as natural. In Section 3 we present diverse examples of circuit lower bound proofs and show why they are natural in our sense. While Section 5 gives some general theoretical reasons why proofs against circuits tend to be natural. Section 4 gives evidence that "naturalizable" proof techniques cannot prove strong lower bounds on circuit size. In particular, we show modulo a widely believed cryptographic assumption that no natural proof can prove super-polynomial lower bounds for general circuits, and show unconditionally that no natural proof can prove exponential lower bounds on the circuit size of the discrete logarithm problem.
Natural proofs form a hierarchy according to the complexity of the combinatorial property involved in the proof. We show without using any cryptographic assumption that $AC^0$-natural proofs, which are sufficient to prove the parity lower bounds of [11, 40, 14], are inherently incapable of proving the bounds for $AC^0[q]$-circuits of [29, 36, 8].

One application of natural proofs was given in [33]. It was shown there that in certain fragments of Bounded Arithmetic any proof of super-polynomial lower bounds for general circuits would naturalize, i.e., could be recast as a natural proof. Combined with the material contained in Section 4 of this paper, this leads to the independence of such lower bounds from these theories (assuming our cryptographic hardness assumption). See also [19, 34] for interpretations of this approach in terms of the propositional calculus, [10, 25] for further results in this direction, and [35] for an informal survey.

1.1. Notation and definitions

We denote by $F_n$ the set of all Boolean functions in $n$ variables. Most of the time, it will be convenient to think of $f_n \in F_n$ as a binary string of length $2^n$, called the truth-table of $f_n$. $f_n$ is a randomly chosen function from $F_n$, and in general we reserve the bold face in our formulae for random objects.

The notation $AC^k$, $NC^k$ is used in the standard sense to denote non-uniform classes. $AC^0[m]$, $TC^0$ and $P/poly$ are the classes of functions computable by polynomial-size bounded-depth circuits allowing $MOD-m$ gates, bounded-depth circuits allowing threshold gates and unbounded-depth circuits over a complete basis, respectively.

2. Natural proofs

2.1. Natural combinatorial properties

We start by defining what we mean by a “natural combinatorial property”; natural proofs will be those that use a natural combinatorial property.

Formally, by a combinatorial property of Boolean functions we will mean a set of Boolean functions $\{C_n \subseteq F_n \mid n \in \omega\}$. Thus, a Boolean function $f_n$ will possess property $C_n$ if and only if $f_n \in C_n$. (Alternatively, we will sometimes find it convenient to use function notation: $C_n(f_n) = 1$ if $f_n \in C_n$, and $C_n(f_n) = 0$ if $f_n \not\in C_n$.) The combinatorial property $C_n$ is natural if it contains a subset $C^*_n$ with the following two conditions:
Constructivity: The predicate $f_n \in C_n^*$ is in $P$. Thus, $C_n^*$ is computable in time which is polynomial in the truth table of $f_n$;

Largeness: $|C_n^*| \geq 2^{-O(n)} \cdot |F_n|$.

A combinatorial property $C_n$ is useful against $P/poly$ if it satisfies:

Usefulness: The circuit size of any sequence of functions $f_1, f_2, \ldots, f_n, \ldots$, where $f_n \in C_n$, is super-polynomial, i.e., for any constant $k$, for sufficiently large $n$, the circuit size of $f_n$ is greater than $n^k$.

A proof that some function does not have polynomial-sized circuits is natural against $P/poly$ if the proof contains, more or less explicitly, the definition of a natural combinatorial property $C_n$ which is useful against $P/poly$.

Note that the definition of a natural proof, unlike that of a natural combinatorial property, is not precise. This is because while the notion of a property being explicitly defined in a journal paper is perfectly clear to the working mathematician, it is a bit slippery to formalize. This lack of precision will not affect the precision of our general statements about natural proofs (see Section 4) because they will appear only in the form “there exists (no) natural proof...”, and should be understood as equivalent to “there exists (no) natural combinatorial property $C_n$...”

The definitions of natural property and natural proof can be explained much less formally. First, a proof that some explicit function $\{g_n\}$ does not have polynomial-sized circuits must plainly identify some combinatorial property $C_n$ of $g_n$ that is used in the proof. That is, the proof will show that all functions $f_n$ that have this property, including $g_n$ itself, are hard to compute. In other words, $C_n$ is useful. If $\{g_n\} \in NP$, then the proof concludes $P \neq NP$. Our main contention, backed by evidence in the next section, is that current proof techniques would strongly tend to make this $C_n$ large and constructive as defined above. (Or at least these two conditions would hold for some sub-property $C_n^*$ of $C_n$.)

In order to understand the definition of large more intuitively, let $N = 2^n$. Largeness requires that $|C_n| \geq \frac{1}{N^k}$ for some fixed $k > 0$, i.e., $f_n$ has a non-negligible chance of having property $C_n$.

Constructivity is a more subtle notion to understand and justify. We take as our basic benchmark of “constructive” that $f_n \in C_n$ be decidable in time $2^{O(n)}$, i.e., polynomial as a function of $2^n$. Now, this is exponential in the number $n$ of variables in $f_n$, and this makes our concept somewhat mysterious, especially since we are going to employ it
for studying computations which are polynomial in \( n! \). The best justification we have is empirical: the vast majority of properties of Boolean functions or \( n \)-vertex graphs (etc.) that one encounters in combinatorics are at worst exponential-time decidable, and, as a matter of fact, known lower bounds proofs operate only with such properties. It also should be noted that even with this loose notion of constructivity we manage to prove in Section 4 strong negative results on the non-existence of natural proofs.

More specifically, consider a commonly-envisioned proof strategy for proving \( P \neq NP \):

- Formulate some mathematical notion of “discrepancy” or “scatter” or “variation” of the values of a Boolean function, or of an associated polytope or other structure. (In our terms, this notion would be formalized as a combinatorial property \( C_n \) that is true of any function with sufficiently high discrepancy.)

- Show by an inductive argument that polynomial-sized circuits can only compute functions of “low” discrepancy. (In our terms, this would mean showing that \( C_n \) is “useful”, because any function with property \( C_n \) can’t be computed by a polynomial-sized circuit.)

- Then show that SAT, or some other function in \( NP \), has “high” discrepancy. (In our terms, this means showing that SAT has property \( C_n \).)

Our main theorem in Section 4 gives evidence that no proof strategy along these lines can ever succeed. We show that any large and constructive \( C_n \) that is useful against \( P/poly \) provides a statistical test that can be used to break any polynomial-time pseudo-random number generator. Specifically, it would violate the fairly widely believed conjecture that there exist pseudo-random generators of hardness \( 2^n \), for some \( \epsilon > 0 \) (e.g., the standard generator based on the discrete logarithm function [9] is believed to be \( 2^{n/2} \)-hard).

What we are saying, subject to the truth of the hard pseudo-random generator conjecture, is this: Any proof that some function \( \{f_n\} \) does not have small circuits must either seize on some very specialized property of \( f_n \), i.e. one shared by only a negligible fraction of functions, or must define a very complicated property \( C_n \), one outside the bounds of most mathematical experience. In our terms, the proof must be unnatural by violating either “largeness” or “constructivity.” In Section 5 we give some solid theoretical evidence for largeness, by showing that any \( C_n \) based on a formal complexity measure must be large. We do not have any similar formal evidence for constructivity, but from experience it is plausible to say that we do not yet understand the mathematics of \( C_n \) outside exponential time (as a function of \( n \)) well enough to use them effectively in a combinatorial style proof.
We make this point in Section 3, where we argue that all known lower bound proofs against non-monotone circuits are natural by our definition.

The best example of a purportedly unnatural argument is a traditional counting argument. The combinatorial property $C_n$ would just be something asserting that $\{f_n\}$ is not in $P/poly$ (e.g., $C_n(f_n) = 1$ exactly when the complexity of $f_n$ is greater than $n^{\log n}$). The proof that $C_n$ is large does not give us the least hint as to how to prove the existence of a large constructive subset $C_n^* \subseteq C_n$. Moreover, a consequence of Theorem 4.1 is that if our pseudo-random generator assumption is true then such $C_n^*$ cannot exist at all! Thus, a counting argument is presumably not a natural argument. This poses no problem for us since counting arguments (closely associated with diagonalization arguments) have yet not proved any lower bounds for explicit functions (except when counting is used for limited purposes, as in [36, 5]. These examples perfectly fit our general framework – see Sections 3.2.1, 3.4.) The question of whether (unlimited) counting or diagonalization arguments are sufficiently powerful to resolve barrier problems in complexity theory predates the combinatorial style lower bounds of the 1980s. Our results have nothing to say – one way or the other – concerning the future promise of diagonalization and counting arguments.

Another exception to our scheme is the list of strong lower bounds proofs against monotone circuit models [2, 3, 4, 17, 26, 27, 28, 37]. Here the issue is not constructivity – the properties used in these proofs are all feasible – but that there appears to be no good formal analogue of the largeness condition. In particular, no one has formulated a workable definition of a “random monotone function.”

All the lower bound proofs surveyed in this paper explicitly state a natural property, and so are natural proofs. In some cases this property is explicit in the original paper, while in others we need to do some work to bring out a natural property $C_n^*$ that yields the same lower bound. We call this latter process naturalizing the original proof. This can be subtle (see e.g. Section 3.2.1 below). Given $C_n$, one must exhibit $C_n^*$ and prove that it has both the constructivity and largeness conditions. The key to doing this seems to lie in carefully analyzing the lower bound proof that used $C_n$. In the case where a researcher intends to build a lower bound proof around some property $C_n$, evaluating $C_n$ for naturalness might be non-trivial. Nonetheless, in light of our framework, such an evaluation could be worthwhile: if it is natural, $C_n$ is not a useful property for solving $P \nsubseteq NP$ and similar questions. Just as a researcher might rule out an approach to lower bounds because it relativizes, he/she might rule out an approach to circuit lower bounds because it “naturalizes”.

6
2.2. Properties which are $\Gamma$-natural against $\Lambda$ with density $\delta_n$  

It is easy and useful to extend the definition of natural proof to a more general, parameterized version. Understanding this more general definition is important to understanding the results as presented in this paper.

Let $\Gamma$ and $\Lambda$ be complexity classes. Call a combinatorial property $C_n$ $\Gamma$-natural with density $\delta_n$ if it contains $C^*_n \subseteq C_n$ with the following two conditions:

Constructivity: The predicate $f_n \in C^*_n$ is computable in $\Gamma$ (recall, $C^*_n$ is a set of truth-tables with $2^n$ bits);

Largeness: $|C^*_n| \geq \delta_n \cdot |F_n|$.

A combinatorial property $C_n$ is useful against $\Lambda$ if it satisfies:

Usefulness: For any sequence of functions $f_n$, where the event $f_n \in C_n$ happens infinitely often, $\{f_n\} \notin \Lambda$.

A lower bound proof that some explicit function is not in $\Lambda$ is called $\Gamma$-natural against $\Lambda$ with density $\delta_n$ if it states a $\Gamma$-natural property $C_n$ which is useful against $\Lambda$ with density $\delta_n$.

The “default” settings of our parameters will be $\Gamma = P$, $\Lambda = P/poly$, and $\delta_n = 2^{-O(n)}$, as in the initial definition. Our main result implies the negative statement that, under our pseudo-randomness assumption, no proof with these parameters can show that SAT does not have polynomial-sized circuits. In fact, as we survey the known lower bound arguments they all remain natural even when the parameters are more restrictively adjusted. We are unaware of a lower bound proof for which we cannot exhibit a $C^*_n$ which is $P$-natural with density close to one. For most known arguments, $\Gamma$ can be restricted to $NC^2$ or lower.

Our full negative result (strengthened by an observation of Razborov [33]) is that, under our pseudo-randomness assumption, no property with $\Gamma$=quasi-polynomial-sized circuits, $\Lambda = P/poly$, and $\delta_n = 2^{-O(n)}$ can exist. Thus, our negative result rules out proofs with much more inclusive parameters than currently known circuit lower bounds.

3. Examples of naturalizing arguments

3.1. $AC^0$ lower bounds for parity: $AC^0$-natural

One of the first combinatorial arguments to give people hope and direction in lower bound research was [11] where it was shown that $PARITY \notin AC^0$ (independently this result,
using somewhat different machinery, was discovered in [1]). Substantial technical improvements to their bounds were subsequently given by [40, 14]. All these proofs are $AC^0$-natural.

The $C_n$ used by these arguments simply says that there does not exist a restriction of the variables with the appropriate number of unassigned variables which forces $f_n$ to be a constant function. The “appropriate” number of unassigned variables is different in [11, 40, 14] and determines the bounds obtained.

All three papers argue explicitly that $C_n(f_n) = 1$ implies that $\{f_n\} \not\subseteq AC^0$, in other words, that $C_n$ is useful against $AC^0$. $C_n$ is a natural property. In fact, we can choose $C_n^* = C_n$.

A simple counting argument shows that $C_n^*$ is true of a random function ($C_n^*$ has the largeness condition).

$C_n^*$ is in $AC^0[n]$ ($C_n^*$ has constructivity). Indeed, suppose $k$ is the “appropriate” number of unassigned variables. Given the truth table for $f_n$ as input, we compute $C_n^*(f_n)$ as follows. List all $\binom{n}{k} 2^{n-k} = 2^{O(n)}$ restrictions of $n - k$ variables. For each one there is a circuit of depth 2 and size $2^{O(n)}$ which outputs a 1 iff that restriction does not leave $f_n$ a constant function. Output the AND of all these circuits. The resulting circuit has depth 3 and is polynomial-sized in $2^n$.

3.2. $AC^0[q]$ lower bounds: $NC^2$-natural

In this subsection we look at the proofs from [29, 36, 8] of lower bounds on the size of $AC^0[q]$-circuits, $q$ being a power of a prime. The naturalness of these proofs is especially transparent in the framework of [29]. Namely, we have a $GF[2]$-linear mapping $M$ from $F_n$ to a matrix space, and we simply take $C_n^*$ to be the set of all $f_n \in F_n$ for which $\text{rank}(M(f_n))$ is large. After reviewing the argument in Section 3.2.1 below, it will be an exercise for the reader to show that $C_n^*(f_n) = 1$ for at least $1/2$ fraction of all $f_n \in F_n$. Since computing the rank is in $NC^2$, we see that the proof is $NC^2$-natural. Smolensky’s proof [36] is analyzed below.

We will show in Section 4 that there is no $AC^0$-natural proof against $AC^0[2]$. Along with the previous subsection, this gives the insight that [29, 36, 8] had to require arguments from a stronger class than those of [11, 40, 14].

3.2.1. Smolensky’s proof: a non-trivial example of naturalization

The argument given in Smolensky [36] is a perfect example of a natural circuit lower bound proof, but this is not immediately obvious. We will outline a special case of his argument:
a proof that parity does not have small $AC^0[3]$ circuits.

First, we recall the notion of polynomial approximation of a Boolean function. Think of the Boolean value TRUE as corresponding to the field element $-1$ and the Boolean value FALSE as corresponding to the field element 1. Let $f$ be a Boolean function and $p$ be a polynomial over $\mathbb{Z}_3$ where $f$ and $p$ have an identical set of variable names. Any assignment $A$ to $f$ can be viewed as an assignment to $p$; in the case $p(A)$ and $f(A)$ evaluate to corresponding values we consider them equal on this assignment. Otherwise, we consider them to differ. The better $p$ approximates $f$, the fewer assignments on which they differ. Since we will only be interested in the values polynomials take on $\{-1, 1\}$ (Boolean) assignments, we will consider polynomials to be multi-linear by default (no variable gets raised to a power greater than one).

**Proof outline:** Smolensky’s proof has two main pieces. (1) Any function computed by a “small” $AC^0[3]$ circuit can be “reasonably” approximated by a “low” degree polynomial over $\mathbb{Z}_3$. (2) The parity function in $n$ variables can’t be “reasonably” approximated by a “low” degree polynomial over $\mathbb{Z}_3$. The proof of (1) is not important here and is omitted. (2) is proved by contradiction. Suppose there were a “low” degree (degree $d$) polynomial $p$ which agrees with the polynomial $x_1x_2x_3\cdots x_n$ (the parity function) on all but a “small” number of Boolean assignments. Let $W$ be the set of Boolean assignments on which they differ. Let $N = 2^n$. Let $w$ be the size of the set $W$. We will assume that $n$ is odd and use $l_1$ and $l_2$ to denote polynomials of degree less than $n/2$. Every multi-linear polynomial $q$ can be written in the form $x_1\cdots x_nl_1 + l_2$. This means that, ignoring the inputs in $W$, every $\mathbb{Z}_3$-valued function on $\{-1, 1\}^n \setminus W$ (and there are $3^{N-w}$ of them) can be represented in the form $pl_1 + l_2$. This representation has degree $(n-1)/2 + d$ which by a counting argument can’t represent as many as $3^{N-w}$ functions. Contradiction.

This proof might seem to be exploiting a very particular fact about how the parity function is expressed as a polynomial; it is not obvious how this same proof would apply to a large fraction of functions. Even worse, the proof refers to a seemingly non-constructive counting argument. However, the proof technique is by its nature applicable to many functions, and counting Boolean functions eventually boils down to counting dimensions of certain linear spaces which already are feasible in our sense.

There is one choice of $C_n$ clear from the proof: $C_n(f_n) = 1$ if $f_n$ can’t be reasonably approximated by a low degree polynomial over $\mathbb{Z}_3$ (for the appropriate definitions of reasonable and low). Part (1) of Smolensky’s argument proves that $C_n$ is useful against $AC^0[3]$. Why is $C_n$ natural? To see it we have to make a choice of $C_n^*$. The simple choice is $C_n^* = C_n$. It is fairly obvious that $C_n^*$ satisfies the largeness condition. But what about $P/poly$-constructivity? It is not at all clear that there is a polynomial-size circuit which can determine if a function (given by its truth-table) can be
approximated by a low-degree polynomial over $\mathbb{Z}_3$. This remains an open problem. 

Thus we sink deeper into the proof and try to put

$$C_n^*(f_n) = 1 \text{ if every polynomial } q \text{ can be written in the form } \bar{f}_n l_1 + l_2,$$

(1)

where $\bar{f}_n$ is the unique multi-linear polynomial representing $f_n$. Then we have constructivity.

In order to see this, denote by $L$ the vector space of all polynomials of degree less than $n/2$, and by $T$ the complementary vector space of all (multi-linear) polynomials without monomials of degree less than $n/2$. The whole polynomial space is then represented as the direct sum $L \oplus T$ and also, since $n$ is odd, we have $\dim(L) = \dim(T) = N/2$. Now, $C_n^*(f_n) = 1$ iff the linear mapping $\pi_{\bar{f}_n} : L \to T$ taking $l \in L$ to the projection of $\bar{f}_n l \in L \oplus T$ onto $T$ is one-to-one (the reader can check his understanding at this point by verifying that the parity function has this property). Thus checking that $C_n^*(f_n) = 1$ amounts to checking that a matrix easily computable from $f_n$ is non-singular which can be done in $NC^2$.

For so chosen $C_n^*$ the largeness condition also looks plausible. But we have no easy proof of it.

We turn around this difficulty by trying to extend the definition of (1) as much as we can (so that we’ll have more functions satisfying it) while preserving its spirit (so that constructivity will also be preserved) and keeping the lower bound provided by it. A short examination shows that the definition

$$C_n^*(f_n) = 1 \text{ iff } \dim(\bar{f}_n L + L) \geq N (1/2 + \epsilon)$$

(2)

which for $\epsilon = 1/2$ is the same as (1), is actually as good as (1) itself for arbitrary fixed $\epsilon > 0$. Indeed, (2) implies that at least $3^{N(1/2+\epsilon)}$ functions on $\{-1,1\}^n \setminus W$ can be represented by a degree $(n - 1)/2 + d$ polynomial, and the same counting argument still works.

But if we define $C_n^*$ as in (2) with $\epsilon = 1/4$, we also have largeness. This immediately follows from the fact that for every $f_n \in F_n$ either $C_n^*(f_n) = 1$ or $C_n^*(x_1 \oplus \cdots \oplus x_n \oplus f_n) = 1$ (cf. the proof of Theorem 5.2 a) below).

To show this fact, note that if $\dim(\bar{f}_n L + L) \geq 3N/4$ then $C_n^*(f_n) = 1$. Otherwise we have

$$\dim \left( (x_1 \cdots x_n \bar{f}_n L + L)/L \right) =$$

$$\dim \left( (x_1 \cdots x_n L + \bar{f}_n L)/\bar{f}_n L \right) \geq$$

\[1\] $C_n^*$ can be further adjusted to be a property of density close to one, as opposed to $1/2$. 

10
\[
\dim \left( (x_1 \cdots x_n L + \tilde{f}_n L + L) / (\tilde{f}_n L + L) \right) = \\
\dim \left( (T + L) / (\tilde{f}_n L + L) \right) \geq N/4
\]

(the first equality here comes from the observation that \((\tilde{f}_n)^2 = 1\) and thus multiplying by \(\tilde{f}_n\) defines an automorphism of \(L \oplus T\)). This gives us \(C^*_n(x_1 \oplus \ldots \oplus x_n \oplus f_n) = 1\).

So, \(C_n\) is an \(NC^2\)-natural property.

Smolensky’s proof is the most difficult example of naturalization we have encountered in our analysis. On the other hand, it perfectly illustrates the general empirical idea of “adjusting” \(C_n\) in both directions in order to come up with a natural \(C^*_n\).

### 3.3. Perceptron lower bounds for parity: \(P\)-natural

In [6], it is shown that a small constant-depth circuit (over \(\{\land, \lor, \neg\}\)) which is allowed a single majority gate can’t approximate the parity function. The authors did this by first showing tight lower bound on the degree of a perceptron required to approximate parity to within a given \(\epsilon\). Their argument is natural.

Some definitions from [6]. A real polynomial \(p\) strongly represents a Boolean function\(^2\) \(f\) just in case \(\text{sgn}(p(x)) = f(x)\) for all input vectors \(x\); such a polynomial is also called a perceptron to compute \(f\). Let \(p\) weakly represent \(f\) just in case \(p\) is not the constant zero function on \(\{-1, 1\}^n\), and \(\text{sgn}(p(x)) = f(x)\) for all \(x\) where \(p(x)\) is nonzero. The weak degree, \(d_w(f)\), is defined as the least \(k\) for which there exists a non-zero degree \(k\) polynomial which weakly represents \(f\).

A natural \(C_n\) stated in the paper is that \(f_n\) can’t be well approximated by the sign of a low degree polynomial. It is explicitly shown that any \(f_n\) with property \(C_n\) can’t be approximated by a small, constant-depth circuit with one majority gate, i.e., \(C_n\) has usefulness. To see that \(C_n\) is natural one must exhibit a proper subset \(C^*_n\).

Let \(C^*_n(f_n) = 1\) if \(d_w(f_n)\) is greater than the appropriate threshold. [6] explicitly showed that \(C^*_n(f_n) = 1\) implies that a polynomial must have appropriately high degree to approximate \(f_n\) with its sign, i.e., \(C^*_n(f_n) = 1\) implies that \(C_n(f_n) = 1\). \(d_w\) is computable in polynomial-time using linear programming. This shows that \(C^*_n\) has constructivity. Since the linear programming seems essential it is doubtful that anything substantially more constructive than \(C^*_n\) could be found in the above argument, e.g., an \(NC\)-natural property for example.

\(^2\)In this section we, similarly to 3.2.1, represent Boolean functions as mappings from \(\{-1, 1\}^n\) to \(\{-1, 1\}\), and \(fg\) stands for the point-wise product, which is the same as \(f \oplus g\) in the \(\{0, 1\}\)-notation.
To argue that \( C_n^* \) has the largeness property, we can show the following improvement of an \( \Omega(n/ \log n) \) lower bound from [6]:

**Theorem 3.1.** For a uniformly chosen \( f_n \in F_n \), \( P[d_\omega(f_n) \geq n/20] > 1 - 2^{-2^{\Omega(n)}}. \)

**Proof.** We use the following well-known facts:

**Proposition 3.2.** Let \( a_1, \ldots, a_N \in \mathbb{R} \). Then there exist \( a'_1, \ldots, a'_N \in \mathbb{Z} \) such that \( |a'_i| \leq \exp(O(N \log N)) (1 \leq i \leq N) \), and for every \( x_i \in \{-1, 1\}^N \),

\[
\text{sgn} \left( \sum_{i=1}^{N} a_i x_i \right) = \text{sgn} \left( \sum_{i=1}^{N} a'_i x_i \right).
\]

**Proposition 3.3.** Every integer polynomial \( p(x_1, \ldots, x_n) \) of degree \( d \) which is not an identically zero on \( \{-1, 1\}^n \), differs from zero on at least \( 2^{n-d} \) points from \( \{-1, 1\}^n \).

The proof of Proposition 3.2 can be found e.g. in [21]; Proposition 3.3 is folklore.

Let \( f_n \in F_n \). If \( f_n \) is weakly represented by a polynomial \( p \) of degree at most \( n/20 \), we firstly apply Proposition 3.2 to the vector of coefficients of \( p \). The length \( N \) of this vector is \( \sum_{i=0}^{n/20} \binom{n}{i} \leq 2^{n(\mathcal{H}(1/20)+o(1))} \), where \( \mathcal{H}(\epsilon) \) is the entropy function. We find that \( p \) can be replaced by a polynomial \( p' \) with integer coefficients whose bit size is at most \( O \left( N^2 \log N \right) \leq 2^{n(2\mathcal{H}(1/20)+o(1))}. \)

\( f_n \) can be uniquely retrieved from the pair \((p', f'_n)\), where \( f'_n \) is the list of values of \( f_n \) on zeros of \( p' \) (arranged, say, in the lexicographic order). From Proposition 3.3 we know that the bit size of \( f'_n \) is at most \( 2^n - 2^{19/20n} \), thus the bit size of the pair \((p', f'_n)\) is at most \( 2^n - 2^{19/20n} + 2^{n^2(\mathcal{H}(1/20)+o(1))} \). Since \( 2 \cdot \mathcal{H}(1/20) < \frac{19}{20} \), the proof is completed by the standard counting argument. \( \square \)

### 3.4. Lower bounds on formula size: AC\(^0\)-natural

Andreev [5] gives a promising lower bound for the formula size of an explicit function. His bound was subsequently improved in [23, 24]. Finally, Håstad [15] gave a nearly optimal lower bound (almost \( n^3 \)) of the formula size for Andreev’s function.

Andreev’s function is a Boolean function \( A_{2n} \) on \( 2n \) bits: \( a_1, a_2, \ldots, a_n; b_1, b_2, \ldots, b_n \). The \( a \)'s are partitioned into \( \log n \) groups of size \( n/\log n \) each. Let \( h_j \) be the parity of the
bits in the $j$th group. The bits $h_1, h_2, \ldots, h_{\log n}$ index a number $i$ from 1 to $n$. The value of the function $A_{2n}$ is the bit $b_i$.

All these proofs work by using a shrinkage factor $T$ which was successively improved in the last three papers until $T = \Omega(n^2)$. ($\Omega$ is the “soft Omega” notation which is like $\Omega$ but ignores multiplicative factors of $(\log n)^k$ for constant $k$.)

The meaning of $T$ is that when a formula is hit by a random restriction it is almost certain to shrink by a factor of $T$. Thus, to prove a formula lower bound, just show that a formula must have size $s$ after being hit by a random restriction. It follows that the original formula had size around $sn^2$.

The natural property $C_{2n}$ is that there is a restriction of $b$'s such that any of its extensions leaving at least one unrestricted variable in each group of $a$’s induces a formula of size $\Omega(n/\log n)$. This property is useful since a random restriction leaving $(\log n)^2$ unrestricted variables leaves at least one such variable in each group: for some fixing of $b$’s, a random restriction to the $a$’s will shrink the formula to $\Omega(n/\log n)$. Obviously, $A_{2n}$ has $C_{2n}$ (simply restrict $b$’s so that they will encode the most complex function in $\log n$ variables) which implies that it must have formula complexity at least $\Omega(n^2)$.

We can choose $C_{2n}^\star = C_{2n}$. The fact that $C_{2n}^\star$ has largeness is easy to prove. Constructivity is also easy if we observe that there are only $2^{O(n)}$ formulas of size less than $n/\log n$.

3.5. Lower bounds against depth-2 threshold circuits: $TC^0$-natural

Hajnal et al. [13] show that the MOD-2 inner-product function requires depth-2 threshold circuits of exponential size. Any Boolean function can be viewed as a Boolean matrix by dividing the inputs into two equal sets with the left half indexing the rows and the right half indexing the columns. Seen in this way the inner-product function is a Hadamard matrix. Their proof shows that any matrix with low discrepancy can’t be computed by small depth-2 threshold circuits. Choose $C_n$ to be true of all functions whose matrices have low discrepancy. Their main lemma shows that any Hadamard matrix has low discrepancy. The same argument shows that any matrix which is almost Hadamard in the sense that the dot product of any two rows or any two columns is small also has the low discrepancy property. Thus, the $C_n^\star$ suggested by their proof is to check that the function viewed as a matrix is almost Hadamard, for the appropriate definition of almost. It is possible to define “almost” so as to guarantee that $C_n^\star$ has largeness and preserves usefulness. Constructivity: For each of the $2^{O(n)}$ dot products, feed the binary AND’s into a threshold gate; feed the
outputs of the threshold gates into a large fan-in AND. This is in $TC^0$.


It was shown in [30] that any switching-and-rectifier network (in particular, any nonde-

terministic branching program) for a large variety of symmetric functions must have size

$\Omega(n\alpha(n))$, where $\alpha(n)$ is a function which slowly grows to infinity. A similar result was

proven in [18] for $\oplus$-branching programs.

The proofs are based upon a purely combinatorial characterization of the network size

in terms of particular instances of the MINIMUM COVER problem. Let $C_n$ be the set of

those functions $f_n$ for which the size $\tau(f_n)$ of the minimal solution to the corresponding

instance is $\Omega(n\alpha(n))$.

The key lemma in these proofs says that if $f_n$ outputs a 1 on any input with $s(n)$ ones, and

outputs a 0 on any input with $s(n) - d(n)$ ones, then $\tau(f_n) \geq \Omega(n\alpha(n))$ ($s(n)$ and

d($n$) are functions which slowly grow to infinity; $s(n) \gg d(n)$).

Denote this property by $A_n$. It obviously violates the largeness condition. We circum-

vent this by letting $C^*_n$ be the set of those functions for which any restriction $\rho$ assign-

ing $n/2$ variables to zero can be extended to another restriction $\rho'$ by assigning to zero

$(n/2 - \log \log n)$ additional variables in such a way that the induced function has $A_{\log \log n}$.

To see $C^*_n \subseteq C_n$, recall from [30, 18] that every covering set $\delta_{i,c}(A)$ has its associated

variable $x_i$ such that restricting this variable to 0 kills $\delta_{i,c}(A)$. Now, for any collection of

$o(n\alpha(n))$ covering sets we simply assign $n/2$ most frequently represented $x_i$'s to 0, and this

leaves us with a collection in which every variable corresponds to at most $o(\alpha(n))$ sets.

Hence, for every extension $\rho'$ of this restriction, the size of the resulting collection will be

$o(\log \log n \cdot \alpha(n))$. Thus, by the above lemma, this collection (and hence the original one)
does not cover all the points from the universe ($\alpha(n)$ and $\alpha(\log \log n)$ differ by at most 1).

$C^*_n$ is in $AC^0$ (cf. Section 3.1).

To see the largeness condition, note that for every $\rho$ we can choose $n^{3/2}$ extensions

$\rho'_1, \ldots, \rho'_{n^{3/2}}$ so that the sets of variables unassigned by every two different $\rho'_i, \rho'_j$ from this

list have at most one variable in common. The event “$f_n$ restricted by $\rho'_i$ has $A_{\log \log n}$”

depends only on those inputs that have either $s(n)$ or $s(n) - d(n)$ ones, and, moreover, all

these ones correspond to variables not assigned by $\rho'_i$. Since $d(n) > 1$ and $s(n) - d(n) > 1$, our

assumption on $\rho'_1, \ldots, \rho'_{n^{3/2}}$ implies that these sets of inputs are pairwise disjoint (when

$i$ ranges over $\{1, \ldots, n^{3/2}\}$). Hence, the events “$f_n$ restricted by $\rho'_i$ has $A_{\log \log n}$” are

independent, and we can apply the standard counting argument.

14
4. Inherent limitations of natural proofs

In this section, we argue that natural proofs for lower bounds are almost self-defeating. The idea is that a natural proof that some function $f$ is not in $P/poly$ has an associated algorithm. But just as the proof must distinguish $f$ from a pseudo-random function in $P/poly$ (one being hard the other not), the associated algorithm must be able to tell the difference between the two. Thus, the algorithm can be used to break a pseudo-random generator. This is self-defeating in the sense that a natural proof that hardness exists would have as an automatic by-product an algorithm to solve a “hard” problem.

For a pseudo-random generator $G_k: \{0,1\}^k \to \{0,1\}^{2k}$ define its hardness $H(G_k)$ as the minimal $S$ for which there exists a circuit $C$ of size $\leq S$ such that

$$|P[C(G_k(x)) = 1] - P[C(y) = 1]| \geq \frac{1}{S}$$

(cf. [9]). Here, as usual, $x$ is taken at random from $\{0,1\}^k$, and $y$ is taken at random from $\{0,1\}^{2k}$.

**Theorem 4.1.** There is no lower bound proof which is $P/poly$-natural against $P/poly$, unless $H(G_k) \leq 2^{k \cdot (1)}$ for every pseudo-random generator $G_k: \{0,1\}^k \to \{0,1\}^{2k}$ in $P/poly$.

In particular, if $2^n$-hard functions exist then there is no $P/poly$-natural proof (against $P/poly$).

**Proof.** For the sake of contradiction, suppose that such a lower bound proof exists and $C_n$ is associated $P/poly$-natural combinatorial property. Let $C^*_n \subseteq C_n$ satisfy the constructivity and largeness conditions. W.L.o.g. we may assume from the very beginning that $C^*_n = C_n$.

We use a slightly modified construction from [12]. Let $G_k: \{0,1\}^k \to \{0,1\}^{2k}$ be a polynomial time computable pseudo-random generator, and $\varepsilon > 0$ be an arbitrary constant. Set $n = \lceil k \varepsilon \rceil$. We use $G: \{0,1\}^k \to \{0,1\}^{2k}$ for constructing a pseudo-random function generator $f: \{0,1\}^k \to F_n$ in the same way as in [12]. Namely, let $G_0, G_1: \{0,1\}^k \to \{0,1\}^k$ be the first and the last $k$ bits of $G$, respectively. For a string $y \in \{0,1\}^n$ we define $G_y: \{0,1\}^k \to \{0,1\}^k$ by $G_y \equiv G_y \circ G_{y_{n-1}} \circ \cdots \circ G_1$, and for $x \in \{0,1\}^k$ let $f(x)(y)$ be the first bit of $G_y(x)$.

Note that $f(x)(y)$ is computable by poly-size circuits, hence (from the definition of a proof natural against $P/poly$) the function $f(x) \in F_n$ is not in $C_n$ for any fixed $x \in \{0,1\}^k$ and any sufficiently large $k$. In other words, $C_n$ has empty intersection with

15
\[ \{ f(x) \mid x \in \{0,1\}^k \}, \text{ and this disjointness implies that } C_n \text{ provides a statistical test for } f(x), \text{ with} \]

\[ |P[C_n(f_n) = 1] - P[C_n(f(x)) = 1]| \geq 2^{-O(n)}. \] (3)

Note that this test is computable by circuits of size \(2^O(n)\).

Constructing from this a statistical test for strings in our case is even simpler than in [12]. Namely, we arrange all internal nodes of the binary tree \( T \) of height \( n \):

\[ v_1, v_2, \ldots, v_{2^n-1} \]

in such a way that if \( v_i \) is a son of \( v_j \) then \( i < j \). Let \( T_i \) be the union of subtrees of \( T \) made by \( \{v_1, \ldots, v_i\} \) along with all leaves. For a leaf \( y \) of \( T \) let \( v_i(y) \) be the root of the subtree in \( T_i \) containing \( y \). Let \( G_i(y) = G_{y_n} \circ \cdots \circ G_{y_{n-h(i,y)+1}} \), where \( h(i,y) \) is the distance between \( v_i(y) \) and \( y \). Finally, define the random collection \( f_{i,n} \) by letting \( f_{i,n}(y) \) be the first bit of \( G_i(y)(x_{v_i(y)}) \), where \( x_v \) are taken from \( \{0,1\}^k \) uniformly and independently for all roots \( v \) of trees from \( T_i \).

Since \( f_{0,n} \) is \( f_n \), and \( f_{2^n-1,n} \) is \( f(x) \), we have from (3) that for some \( i \),

\[ |P[C_n(f_{i,n}) = 1] - P[C_n(f_{i+1,n}) = 1]| \geq 2^{-O(n)}. \]

Fix \( x_v \) for all roots \( v \) of subtrees in \( T_{i+1} \) other than \( v_{i+1} \) so that the bias \( 2^{-O(n)} \) is preserved. Then we have a statistical test for strings distinguishing between \( G (x_{v_{i+1}}) \) and \( (x_{v'}, x_{v''}) \), where \( v', v'' \) are the two sons of \( v_{i+1} \). Thus \( H(G_k) \leq 2^{O(n)} \leq 2^{O(k^\epsilon)} \). As \( \epsilon \) was arbitrary, the result follows.

The assumption that \( 2^{n^\epsilon} \)-hard functions exist is quite plausible. For example, despite many advances in computational number theory, multiplication seems to provide a basis for a family of such functions (known factoring algorithms are sufficiently exponential).

Based upon lower bounds for the parity function, Nisan [22] constructed a very strong generator secure against \( AC^0 \)-attack. In fact, an easy analysis of his generator in terms of its own complexity gives the following:

**Theorem 4.2.** For any integer \( d \), there exists a family \( G_{n,s} \subseteq F_n \), where \( s \) is a seed of size polynomial in \( n \) such that \( G_{n,s} \in AC^0[2] \) and \( G_{n,s} \) looks random for \( 2^{2^{O(n)}} \)-size depth-\( d \) circuits, i.e., for any polynomial-size (in \( 2^n \)) depth \( d \) circuit family \( C_n : F_n \rightarrow \{0,1\} \),

\[ |P[C_n(f_n) = 1] - P[C_n(G_{n,s}) = 1]| < 2^{-\omega(n)}. \] (4)

Here \( s \) is a random seed of the appropriate size.
Theorem 4.3. There is no lower bound proof which is AC⁰-natural against AC⁰[2].

Proof. Assume, on the contrary, that such a proof exists, and that Cₙ has the same meaning as in the proof of Theorem 4.1. Let d be the depth of a size 2^{O(n)} circuit to compute Cₙ. Let Gₙ,s be the generator which is pseudo-random against depth-d 2^{O(n)}-sized circuits from Theorem 4.2. From the definition of a proof natural against AC⁰[2], for sufficiently large n, Cₙ(Gₙ,s) = 0. Now, (4) immediately contradicts the largeness condition.

In fact, it is clear from the above proofs that whenever a complexity class Λ contains pseudo-random function generators that are sufficiently secure against Γ-attack, then there is no Γ-natural proof against Λ. E.g., it is easy to see that Theorems 4.1, 4.3 are still valid for the larger class of Γ-natural proofs, where Γ consists of languages computable by quasi-polynomial-sized circuits. This observation is of little importance for the examples of natural proofs given in this paper. However, it is useful in the context of proofs feasible in the logical sense [33], where quasi-polynomial limitations on the complexity arise more often. Formally, we define P/qpoly as the class of non-uniform, quasi-polynomial size circuits, i.e., size n^{O(log)}.

Theorem 4.4. There is no lower bound proof which is P/qpoly-natural against P/poly unless H(Gₖ) ≤ 2^{k(1)} for every pseudo-random generator Gₖ : {0, 1}^k → {0, 1}^{2k} in P/poly.

4.1. Natural proofs are not applicable to the discrete logarithm problem

It is possible (though we are unaware of any such examples) that a lower bound proof for restricted models might be natural, but cannot be applied to any explicit function. In other words, the proof might simply argue that many functions are complex without providing us with any explicit examples of such functions. Given our hardness assumption, no natural proof can prove lower bounds against P/poly whether or not the proof makes explicit what the hard function is. Avi Wigderson has pointed out that if we restrict ourselves to certain explicit functions, we can prove unconditional results in the style of Theorem 4.1. A good example of such a function is the discrete logarithm. The key point is that the discrete logarithm is known to be hard on average if and only if it is hard in the worst case. In this section, we show that there is no natural proof that the discrete logarithm requires exponential-sized circuits.
Recall from [9] that for a prime \( p \) and a generator \( g \) for \( \mathbb{Z}_p^* \), the predicate \( B_{p,g}(x) \) on \( \mathbb{Z}_p^* \) is defined to be 1 if \( \log_g x \leq (p-1)/2 \) and 0 otherwise. \( B_{p,g}(x) \) was shown in [9] to be a hard bit of the discrete logarithm problem. We consider \( B_{p,g}(x) \) as a Boolean function in \( \lceil \log p \rceil \) variables (extended by, say, zeros on those inputs \( x \) which do not represent an integer in the range \([1, p-1]\)). Our principal goal in this section is to show that no \( P/poly \)-natural proof against “sufficiently large” Boolean circuits can be applied to \( B_{p,g}(x) \).

To explain the meaning of “sufficiently large”, we need a couple of technical definitions. For an integer-valued function \( t(n) \), let \( \text{SIZE}(t(n)) \) be the complexity class consisting of all functions \( \{f_n\} \) which have circuit size \( O(t(n)) \). Let

\[
t^{-1}(n) \equiv \max \{ x \mid t(x) \leq n \}.
\]

We say that \( t(n) \) is half-exponential if it is non-decreasing and

\[
t^{-1}(n^C) \leq o(\log t(n))
\]

for every \( C > 0 \). The meaning of this definition is that, roughly speaking, the second iteration of \( t(n) \) should grow faster than the exponent. For example, \( t(n) = 2^{n^2} \) is half-exponential, whereas \( t(n) = 2^{(\log n)^C} \) is not.

**Theorem 4.5.** Let \( t(n) \) be an arbitrary half-exponential function. Then there is no combinatorial property \( C_n \) useful against \( \text{SIZE}(t(n)) \) and satisfying \( P/poly \)-constructivity and largeness conditions such that \( \bigcup_{n \in \omega} C_n \) contains infinitely many functions of the form \( B_{p,g}(x) \).

**Proof.** Assume the contrary, and let \( \{B_{p_n,g_n}\} \) be an infinite sequence contained in \( \bigcup_{n \in \omega} C_n \) such that \( \lceil \log p_1 \rceil < \lceil \log p_2 \rceil < \ldots \) Let \( k_n \equiv \lceil \log p_n \rceil \). Applying the usefulness condition to the sequence \( f_n \) obtained from \( \{B_{p_n,g_n}\} \) by letting \( f_n \equiv 0 \) for those \( n \) which are not of the form \( \lceil \log p_n \rceil \), we will find in \( \{B_{p_n,g_n}\} \) an infinite subsequence where all functions have the circuit size at least \( t(k_n) \). W.l.o.g, we may assume that this is the case for our original sequence.

Let \( G_\nu : \{0,1\}^{2k_\nu} \rightarrow \{0,1\}^{4k_\nu} \) be the standard pseudo-random generator from [9] based upon \( \{B_{p_\nu,g_\nu}\} \). It is easy to check that the proof of [9, Theorem 3] actually extends to showing that the circuit size of \( \{B_{p_\nu,g_\nu}\} \) is polynomial in \( H(G_\nu) + k_\nu \). Thus, we have

\[
t(k_\nu) \leq (H(G_\nu) + k_\nu)^{O(1)}.
\]

Now we convert \( G_\nu \) into the pseudo-random function generator \( f_\nu : \{0,1\}^{2k_\nu} \rightarrow F_{n_\nu} \) as in the proof of Theorem 4.1, where \( n_\nu \) will be specified a little bit later. There exists a
fixed constant $C > 0$ such that for almost all $\nu$, $f_\nu(x)(y)$ is computable by circuits of size $(k_\nu + n_\nu)^C$. Let $n_\nu \equiv t^{-1}(k_\nu^{C+1}) + 1.$

(5) implies that $t(k_\nu) \geq k_\nu^{C+1}$ for almost all $\nu$, since otherwise we would have $k_\nu \leq t^{-1}(k_\nu^{C+1}) \leq \log t(k_\nu) \leq (C + 1) \log k_\nu.$ Hence $n_\nu \leq k_\nu$. Now we have that for almost all $\nu$ every function in the image of the generator $f_\nu$ has circuit size at most $(k_\nu + n_\nu)^C \leq (2k_\nu)^C \leq k_\nu^{C+1} \leq t(n_\nu).$ Applying the usefulness condition again, we find that for almost all $\nu$, the image of the generator $f_\nu$ has the empty intersection with $C_n$. Arguing as in the proof of Theorem 4.1, we get from this

$$H(G_\nu) \leq 2^{O(n_\nu)}. \quad (7)$$

Finally note that $C_n \neq \emptyset$ for almost all $n$ (from largeness) and, thus,

$$t(n) \leq 2^n \quad (8)$$

(again, for almost all $n$.)

The required contradiction is now obtained simply by combining the inequalities (5) (with $n := k_\nu$, $C := C + 1$), (6), (7), (8):

$$n_\nu = t^{-1}(k_\nu^{C+1}) + 1 \leq o(\log t(k_\nu)) \leq o(\log H(G_\nu) + \log k_\nu) \leq o(n_\nu) + o(\log k_\nu) \leq o(n_\nu).$$

\[ \blacksquare \]

**Corollary 4.6.** There is no combinatorial property $C_n$ useful against $\cap_{c > 0} \text{SIZE} \left(2^n\right)$ and satisfying $P/\text{poly}$-constructivity and largeness conditions such that $\cup_{n \in \omega} C_n$ contains infinitely many functions of the form $B_{p,q}(x)$.

**Proof.** $\cap_{c > 0} \text{SIZE} \left(2^n\right) \supseteq \text{SIZE} \left(2^{2\sqrt{\log n}}\right)$, and $t(n) = 2^{2\sqrt{\log n}}$ is half-exponential. \[ \blacksquare \]

It is easy to see that the above proof is actually valid for an arbitrary collection $\{f_{p,q}\}$ of functions poly-time nonuniformly Turing reducible to the corresponding discrete logarithm problem in place of $\{B_{p,q}\}$.

5. **One property of formal complexity measures**

A formal complexity measure (see e.g. [38, Section 8.8], [31]) is an integer-valued function $\mu$ on $F_n$ such that $\mu(f) \leq 1$ for $f \in \{\neg x_1, \ldots, \neg x_n, x_1, \ldots, x_n\}$ and $\mu(f \circ g) \leq \mu(f) +$
\( \mu(g) \) for all \( f, g \in F_n \) and \( \ast \in \{ \land, \lor \} \). The meaning of this definition is that for every formal complexity measure \( \mu \), \( \mu(f) \) provides a lower bound on the formula size of \( f \), and actually many known lower bounds, both for monotone and non-monotone formulae, can be viewed from this perspective. See the above-cited sources for examples. Also, for any approximation model \( \mathcal{M} \) (see [39, 32] for most general definitions), we have \( \rho(f \ast g, \mathcal{M}) \leq \rho(f, \mathcal{M}) + \rho(g, \mathcal{M}) + 1 \), hence \( \rho(f, \mathcal{M}) + 1 \) is a formal complexity measure.

In this section we show that any formal complexity measure \( \mu \) which takes a large value at a single function, must take large values almost everywhere. In particular, every combinatorial property based on such a measure automatically satisfies the largeness condition in the definition of natural property.

More specifically, we have the following:

**Theorem 5.1.** Let \( \mu \) be a formal complexity measure on \( F_n \), and \( \mu(f) = t \) for some \( f \in F_n \). Then:

a) for at least 1/4 fraction of all functions \( g \in F_n \), \( \mu(g) \geq t/4 \);

b) for any \( \epsilon = \epsilon(n) \) we have that for at least \( (1 - \epsilon) \) fraction of \( g \in F_n \),

\[
\mu(g) \geq \Omega \left( \frac{t}{n + \log \frac{1}{\epsilon^2}} \right) - n.
\]

In fact, the main argument used in the proof of this theorem is valid for arbitrary Boolean algebras, and we formulate it as a separate result since this might be of independent interest.

**Theorem 5.2.** Let \( B \) be a finite Boolean algebra with \( N \) atoms and \( S \subseteq B \).

a) if \( |S| > \frac{3}{4} |B| \) then every element of \( B \) can be represented in the form

\[
(s_1 \land s_2) \lor (s_3 \land s_4); \ s_i \in S \ (1 \leq i \leq 4); \tag{9}
\]

b) if \( S \) contains all atoms and coatoms of \( B \) then every element of \( B \) can be represented in the form

\[
\bigvee_{i=1}^{\ell} \bigwedge_{j=1}^{\ell} s_{ij}, \tag{10}
\]

where \( s_{ij} \in S \) and \( \ell \leq O \left( \log \frac{N |B|}{|S|} \right) \).
Proof of Theorem 5.1 from Theorem 5.2. Let $S = \{ g \mid \mu(g) < t/4 \}$ for part a), and $S = \left\{ g \mid \mu(g) \leq \delta \cdot \frac{t}{(n + \log \frac{1}{\delta})} \right\}$, where $\delta$ is a sufficiently small constant, for part b). Note that in part b) we may assume that $\delta \cdot \frac{t}{(n + \log \frac{1}{\delta})} \geq n + 1$ since otherwise there is nothing to prove. Since $\mu(\bigwedge_{i=1}^{n} p_i) \leq n$ and $\mu(\bigvee_{i=1}^{n} p_i) \leq n$, where $p_i$ is either $x_i$ or $\neg x_i$, this implies that $S$ contains all atoms and coatoms of $F_n$, the latter being viewed as a Boolean algebra.

Now, if $|S| > \frac{1}{4} |B|$ in part a) or $|S| \geq \epsilon |B|$ in part b), then we would apply Theorem 5.2 and represent $f$ in the form (9), (10) respectively. This representation in both cases would imply the bound $\mu(f) < t$, the contradiction.■

Now we prove Theorem 5.2. Denote by $b$ a randomly chosen element of $B$.

Proof of Theorem 5.2 a). Fix $b_0 \in B$ and consider the representation

$$b_0 = (b \land (\neg b \oplus b_0)) \lor (\neg b \land (b \oplus b_0)).$$

As all four random variables $b, (\neg b \oplus b_0), (\neg b, (b \oplus b_0)$ are uniformly distributed on $B$ and $|S| > \frac{3}{4} |B|$, for at least one particular choice $b$ of $b$ we have $b, (\neg b \oplus b_0), (\neg b, (b \oplus b_0) \in S$.■

For proving part b) of Theorem 5.2 we need the following

Lemma 5.3. Let $B$ be a finite Boolean algebra with $N$ atoms and $S \subseteq B$. Then there exists a subset $S_0 \subseteq S$ of cardinality $O(\log N)$ such that $\land S_0$ contains at most $O \left( \frac{\log |B|}{|S|} \right)$ atoms.

Proof of Lemma 5.3. Let us call an atom a good if $P[a \leq s] \leq 2/3$ and bad otherwise. Here $s$ is picked at random from $S$.

Now, the standard entropy-counting argument gives us that there are at most

$$O \left( \log \frac{|B|}{|S|} \right)$$

bad atoms. An equally standard argument implies that if we take a random subset $S_0 \subseteq S$ of cardinality $C \log N$, the constant $C$ being sufficiently large, then for any good atom $a$, $P[a \leq \land S_0] < N^{-1}$. Hence, for at least one particular choice $S_0$ of $S_0$, $\land S_0$ contains only bad atoms, and the lemma follows.■

Proof of Theorem 5.2 b). Denote $\frac{|S|}{|B|}$ by $\epsilon$. Once again, fix $b_0 \in B$. Let us call $c \leq b_0$ good if $P[b \in S \mid b \land b_0 = c] \geq \epsilon$ and bad otherwise. Note that $b \land b_0$ is uniformly
distributed on the Boolean algebra $B_0 = \{ c \mid c \leq b_0 \}$. Hence

$$\mathbb{P}[c \text{ is good}] \geq \frac{e}{2},$$

(11)

where $c$ is chosen from $B_0$ at random.

Now, fix a good $c \in B_0$. The set $B(c) = \{ b \in B \mid b \land b_0 = c \}$ is a Boolean algebra. Applying Lemma 5.3 to this algebra and to $S := S \cap B(c)$, we come up with $S_0 \subseteq S$ of cardinality $O(\log N)$ such that $c \leq \land S_0$ and $(\land S_0 \setminus c)$ has at most $O\left(\log \frac{1}{c}\right)$ atoms. We extend $S_0$ by including to it the corresponding coatoms and find that every good $c \in B_0$ can be represented in the form $\land_{j=1}^{\ell} s_j$, $s_j \in S$, $\ell \leq O\left(\log \frac{N}{c}\right)$.

Next we apply the dual version of Lemma 5.3 to the Boolean algebra $B_0$ and $S := \{ c \in B_0 \mid c \text{ is good} \}$. In view of (11), the same argument as above yields that $b_0 = \lor_{i=1}^{\ell} c_i$, where $c_i$ are either good or atoms. The statement follows.$\blacksquare$

6. Conclusion

We do not conclude that researchers should give up on proving serious lower bounds. Quite the contrary, by classifying a large number of techniques that are unable to do the job we hope to focus research in a more fruitful direction. Pessimism will only be warranted if a long period of time passes without the discovery of a non-naturalizing lower bound proof.

As long as we use natural proofs we have to cope with a duality: any lower bound proof must implicitly argue a proportionately strong upper bound. In particular, we have shown that a natural proof against complexity class $\Lambda$ implicitly shows that $\Lambda$ does not contain strong pseudo-random function generators. In fact, the proof gives an algorithm to break any such generator. Seen this way, even a natural proof against $NC^1$ (or $TC^0$) becomes difficult or impossible. In [16] it is argued based on the hardness of subset sum that a pseudo-random function should exist in $TC^0 \subseteq NC^1$. Consider the plausible conjecture that there exists a (pseudo-random) function $f \in NC^1$ (or $TC^0$) such that $G_{n,s}(x) = f(s\#x)$ is a pseudo-random function generator. A natural proof that $P \neq NC^1$ or $P \neq TC^0$ would give an algorithm to break it. Thus, we see that working on lower bounds using natural methods is like breaking a secret code determined by the class we are working against!

With this duality in mind, it is no coincidence that the technical lemmas of [14, 36, 29] yield much of the machinery for the learning result of [20].

22
7. Acknowledgements

We would like to thank Oded Goldreich, Russell Impagliazzo, Mauricio Karchmer, Silvio Micali, Robert Solovay, and Avi Wigderson for very helpful discussions. We are also indebted to both anonymous referees of the journal version of this paper for many useful comments and remarks.

References


[9] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-


achusetts Institute of Technology, 1986.


[16] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as

[17] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-

[18] M. Karchmer and A. Wigderson. On span programs. In *Proceedings of the 8th Struc-


