Electronic Colloquium on Computational Complexity, Report No. 14 (2000)

ECCC

On Learning versus Distinguishing and the Minimal Hardware Complexity of Pseudorandom Function Generators

Matthias Krause and Stefan Lucks*

Theoretische Informatik, Univ. Mannheim, 68131 Mannheim, Germany
e-mail: krause,lucks@informatik.uni-mannheim.de

Abstract. A set F of n-ary Boolean functions is called a pseudorandom function generator (PRFG) if communicating with a randomly chosen secret function from F cannot be efficiently distinguished from communicating with a truly random function. We ask for the minimal hardware complexity of a PRFG. This question is motivated by design aspects of secure secret key cryptosystems, which on the one hand should have very fast hardware implementations, and on the other hand, for security reasons, should behave like PRFGs. By constructing appropriate distinguishing algorithms we show for a wide range of basic nonuniform complexity classes, induced by depth restricted branching programs and several types of constant depth circuits, that they do not contain PRFGs. Observe that in [15] we could show that TC^0_3 seems to contain a PRFG. Moreover, we relate our concept of distinguishability to the learnability of Boolean concept classes. In particular, we show that, if membership queries are forbidden, each efficient distinguishing algorithm can be converted into a weak PAC learning algorithm. Finally, we compare distinguishability with the concept of Natural Proofs and strengthen the main observation of *Razborov* and *Rudich* in [28].

Keywords Cryptography, Pseudorandomness, Boolean Complexity Theory, Learnability, Computational Distinguishability

* Supported by DFG grant Kr 1521/3-1.

ISSN 1433-8092

1 Basic Definitions and Motivation

1.1 Function Generators, Distinguishing Algorithms and Pseudorandomness

Let us fix an input length n and a key length k. A *function generator* is an efficient algorithm, which, for each key $s \in \{0,1\}^k$, provides a Boolean function $f_s : \{0,1\}^n \longrightarrow \{0,1\}$. Clearly, each output bit of a secret key encryption algorithm

$$E : \{0,1\}^n \times \{0,1\}^k \longrightarrow \{0,1\}^m,$$

where n denotes the plaintext block length, k the key length and m the ciphertext block length, can be considered a function generator.

More formally, we define a function generator to be a sequence $F = (F_n)_{n \in \mathbb{N}}$, where, for each n, $F_n = \{f_s, s \in \{0,1\}^{k(n)}\}$ denotes a set of keyed n-ary Boolean functions, and $k(n)$ denotes the key length associated with input length n. Moreover, F is defined by a polynomial time algorithm, which, for all $x \in \{0,1\}^n$ and $s \in \{0,1\}^{k(n)}$ returns $f_s(x)$. (This implies that the key length fulfils $k(n) \in n^{O(1)}$.)

We call a function generator F *pseudorandom* if it is impossible to distinguish efficiently between a truly random function $f \in B_n$ and a function f_s which is randomly chosen from F_n.

For giving the formal definition of pseudorandomness we introduce the notion of an **H-oracle**, where $H \subseteq B_n$. (B_n denotes the set of all 2^{2^n} Boolean function in n variables.) An H-oracle chooses randomly, via the uniform distribution on H, a secret function $h \in H$ and communicates with an enemy cryptanalyst D, which tries to get information about h, via membership queries. (Membership queries means that D submits some input $x \in \{0,1\}^n$ of his choice and the oracle immediately answers by $h(x)$.) We consider an H-oracle to be a truly random source if $H = B_n$, and to be a pseudorandom source, if H is a subset of B_n of size $2^{n^{O(1)}}$). Observe that the truly random source can be considered to work as follows: It answers new membership queries by a fair coinflip, but stores given questions and answers for answering repeated questions consistently.

A **distinguishing algorithm** for a function generator $F = F_n$ is a randomized oracle Turing machine D which gets an input parameter n and which communicates via membership queries with an H-oracle, where either $H = B_n$ (the truly random source) or $H = F_n$ (the pseudorandom source). The aim of D is to find out via membership queries whether $H = B_n$ (in this case, D outputs 0) or $H = F_n$ (in this case, D outputs 1). At the beginning of the computation it is completely unknown to D wether $H = B_n$ or $H = F_n$.

The relevant cost parameters of a distinguishing algorithms are the **worst case running time** $t_D = t_D(n)$ and the **advantage** $\varepsilon_D = \varepsilon_D(n)$, which is defined as

$$\varepsilon_D(n) = Pr[D \text{ outputs } 1 | H = F_n] - Pr[D \text{ outputs } 1 | H = B_n].$$

The probabilities are taken w.r.t. the internal randomization of D and the randomization of the oracle, i.e. the uniform distribution on H. The **ratio** $r_D = r_D(n)$ of a distinguishing algorithm D is defined to be $r_D(n) = t_D(n) \cdot \varepsilon_D^{-1}(n)$. For all $f \in F_n$ let

$$\varepsilon_D(f) = Pr[D \text{ outputs } 1 | H = F_n \wedge f] - Pr[D \text{ outputs } 1 | H = B_n]$$

denote the advantage achieved by D in the pseudorandom case under the condition that the F_n-oracle has chosen the secret function f. Observe that

$$\varepsilon_D(n) = \mathbf{E}_f[\varepsilon_D(f)].$$

The **worst case advantage** of D on F_n is defined to be the value $\min_{f \in F_n} \varepsilon_D(f)$. The product of the inverse of the worst case advantage and the worst case running time is called the **worst case ratio** of D on F_n.

We will call F to be a **pseudorandom function generator** (for short: **PRFG**) if for all distinguishing algorithms D for F it holds that $r_D \in 2^{n^{\Omega(1)}}$. This definition of pseudorandomness is consistent with the definition given by *Goldreich, Goldwasser, Micali* in [9].

Observe that there is a trivial distinguishing algorithm against any function generator F of ratio $O(|F_n| \log(|F_n|))$. (For given n, fix $\tilde{n}$ to be the minimal number such that $2^{\tilde{n}} \geq |F_n|$ and fix an arbitrary set of inputs $X \subseteq \{0,1\}^n$ of size $\tilde{n}$. Accept if the oracle function f coincides with some $f_s \in F_n$ on X. Clearly, the running time of this algorithm is $O(|F_n| \log(|F_n|))$, as $\tilde{n} \in O(\log(|F_n|))$. In the pseudorandom case the probability of acceptance is obviously 1. In the truly random case ($H = B_n$) the probability of acceptance is at most $1/2$, as there are at least $2|F_n|$ possible assignments to X. Consequently, the worst case advantage is at least $1/2$.) This implies that each function generator F can be distinguished with ratio $2^{n^{O(1)}}$.

Let us present another more involved example of a distinguishing algorithm. Suppose we are given a function generator F for which it holds that for all n and for all functions $f_s \in F_n$ there is a variable ordering π such that the uniquely determined minimal ordered binary decision diagram (OBDD) which computes f_s w.r.t. the variable ordering π has size at most $m(n) \in n^{O(1)}$. (For instance, consider F_n to be the following function generator $IP_n = \{ip_\pi;\ \pi \in \mathcal{S}_n\}$, n even, which was discussed in [18]. The function ip_π is defined as

$$ip_\pi(x_1, \ldots, x_n) = x_{\pi(1)}x_{\pi(2)} \oplus x_{\pi(3)}x_{\pi(4)} \oplus \ldots \oplus x_{\pi(n-2)}x_{\pi(n-1)}.$$

Here, $m(n) = 2n$ as ip_π can be computed by a π-OBDD of size 2n.)

Consider the following distinguishing algorithm D for F. D fixes $\tilde{n}$ to be the minimal number for which $2^{\tilde{n}} \geq 2m(n)$, and randomly chooses disjoint subsets X and Y of $\{x_1, \ldots, x_n\}$ fulfilling $|X| = |Y| = \tilde{n}$. D accepts if the following $2^{\tilde{n}} \times 2^{\tilde{n}}$ matrix M has at most $m(n)$ distinct rows. For all assignments c to the variables from X, and all assignments d to the variables from Y, the coefficient $M_{c,d}$ is defined to be the value of the oracle function f on the input obtained by assigning c to the X-variables, d to the Y-variables, and 0 to the remaining variables.

The running time of D is in $O(2^{2\tilde{n}}) \subseteq n^{O(1)}$. In the pseudorandom case we know that the π-OBDD size of f is at most $m(n)$ for some variable ordering π. We say that X and Y are separated by π if $\pi(i) < \pi(j)$ for all $x_i \in X$ and $x_j \in Y$. It is not hard to verify that the probability that the random X and Y are separated by π can be bounded from below by n^{-a} for some constant a. A standard lower bound argument for the size of OBDDs implies that in this case the probability that D accepts is 1. Concerning the truly random case, the probability that a random $2^{\tilde{n}} \times 2^{\tilde{n}}$ matrix has at most $m(n)$ distinct rows is exponentially small in n, as $2^{\tilde{n}} \geq 2m(n)$. We obtain that $\varepsilon_D^{-1}(n) \in n^{O(1)}$ and, thus, the ratio of D is polynomially bounded in n. We obtain

Theorem 1. *Each function generator consisting of functions with polynomially bounded OBDD-size cannot be pseudorandom.*

1.2 Cryptographic Weakness of Nonuniform Computational Models

For any nonuniform computational model $\mathcal{M}$ we denote by $\chi_{\mathcal{M}}$ the corresponding complexity measure for Boolean functions and by $P(\mathcal{M})$ the set of sequences of Boolean functions $G = (g_n)_{n \in \mathbb{N}}$ for which $\chi_{\mathcal{M}}(g_n) \in n^{O(1)}$. A function generator $F = (F_n)_{n \in \mathbb{N}}$ is said to be in $P(\mathcal{M})$ if there is some polynomial bound $m = m(n) \in n^{O(1)}$ such that for all n and $f_s \in F_n$ it holds that $\chi_{\mathcal{M}}(f_s) \leq m(n)$.

We call a computational model $\mathcal{M}$ **cryptographically strong** if $P(\mathcal{M})$ contains a pseudorandom function generator, and **cryptographically weak** otherwise. One aim of this paper is to classify the central nonuniform computational models induced by depth bounded branching programs and constant depth circuits with respect to cryptographic strongness and weakness.

We show cryptographic weakness of particular models $\mathcal{M}$ by constructing *universal* distinguishing algorithms for $\mathcal{M}$ of the following type.

A distinguishing algorithm $D = D(n,m)$, depending on the two input parameters n (input length) and m (complexity parameter) is called a **distinguishing algorithm of polynomially (resp. quasipolynomially) bounded ratio for** $\mathcal{M}$ if for all bounds $m = m(n) \in n^{O(1)}$ and function generators F with $\chi_{\mathcal{M}}$-complexity at most m it holds the following. On all inputs $n, m(n)$, D distinguishes F_n with polynomially, resp. quasipolynomially, bounded ratio. Obviously, any distinguishing algorithm of quasipolynomially bounded ratio proves the model $\mathcal{M}$ to be cryptographically weak.

Observe that in the above example there is described a distinguishing algorithm of polynomially bounded ratio for OBDDs, i.e., OBDDs are a cryptographically weak computational model. Let us present a list of the nonuniform computational models which will be considered in this paper.

- Depth k circuits of unbounded fan-in over AND, OR, and NOT-gates. AC^0_k denotes the corresponding complexity class of problems computable within polynomial size, $AC^0 = \bigcup_k AC^0_k$.
- Depth k circuits of unbounded fan-in over AND, OR, NOT and MOD_m gates. MOD_m is defined by $\mathrm{MOD}_m(x_1,\ldots,x_n)=1$ if and only if $x_1 + \ldots + x_n \not\equiv 0 \mod m$.) The corresponding complexity class of problems computable within polynomial size is usually denoted by $AC^0_k[m]$.
- Depth k unbounded fan-in circuits with unweighted and weighted threshold gates. Unweighted treshold gates $T^n_{\geq r}$, resp. $T^n_{\leq r}$, are defined by the relations

$$T^n_{\geq r}(x_1,\ldots,x_n) = 1 \iff x_1 + \ldots + x_n \geq r$$

  and $T^n_{\leq r}(x_1,\ldots,x_n) = 1 \iff x_1 + \ldots + x_n \leq r$. A weighted treshold gate $T^{\boldsymbol{a}}_{\geq r}$, where $\boldsymbol{a} \in \mathbb{Z}^n$, is defined by the relation

$$T^{\boldsymbol{a}}_{\geq r}(x_1,\ldots,x_n) = 1 \iff a_1x_1 + \ldots + a_nx_n \geq r.$$

  LT_k denotes the complexity class corresponding to depth k polynomial size weighted threshold circuits, and TC^0_k denotes the class corresponding to depth k polynomial size unweighted threshold circuits.
- Boolean Circuits, this is the unrestricted case of fanin-2 circuits over $\{\vee, \wedge, \neg\}$. The corresponding complexity class is denoted by $P/poly$.
- Several types of branching programs, alternatively called binary decision diagrams (BDDs). A branching program is a directed acyclic graph $G = (V, E)$ with one source. Each sink is labeled by a Boolean constant and each inner node by a Boolean variable. Inner nodes have two outgoing edges one labeled by 0 and the other by 1. A BDD represents a Boolean function $f \in B_n$ in the following way. The input a activates, for x_i-nodes, the outgoing a_i-edge. Then $f(a)$ is equal to the label of the sink reached by the unique activated path starting at the source. The relevant cost measure of BDDs, the size, is the number of nodes. The corresponding complexity class $\mathcal{P}_{\mathcal{M}}$ is denoted by $L/poly$.
- Read-once BDDs (sometimes called Free BDDs (FBDDs)). Here, at each path in the program each variable must not occur more than once.
- Ordered binary decision diagrams (OBDDs), where each computational path has to respect the same variable ordering. An OBDD which respects a fixed variable ordering π is called a π-OBDD.

– Syntactic Read-k-BDDs, for which at each path each variable is forbidden to occur more than k times.

.

1.3 Motivation and Structure of the Paper

There is a good deal of applications of pseudorandom function generators, e.g. as building blocks for block ciphers [20, 21], for remotely keyed encryption schemes [22, 4], for message authentication [3], and others. One intention of this paper is to shed more light on the tradeoff phenomenon between hardware complexity and security for cryptographic primitives. The question is which hardware complexity a secret key encryption algorithm must have for being suited to serve as a encryption/decryption mechanism for a secure cryptosystem.

There are two conflicting requirements to modern cryptosystems. The first requirement, clearly, is security. Modern security standards for practically relevant cryptosystems require that no output bit of the underlying secret key encryption algorithm can be efficiently distinguished from a truly random source. This means, each output bit should behave like a pseudorandom function generator. This is because each statistical test which distinguishes an output bit from a truly random source and which is significantly more efficient than exhaustive search could give a promising approach for breaking the cipher. Suppose, for instance, that some cryptanalyst could present an algorithm which, with significant probability, distinguishes an output bit of a proposed cipher of key length, say 64, on the basis of at most 2^{40} plaintext/ciphertext pairs. Then this would immediately rule out this cipher for serious practical applications.

The second requirement is that it should be possible to perform encryption/decryption very quickly. This suggests the choice of a secret key encryption mechanisms which have small complexity with respect to computational models of low expressive power, like depth restricted branching programs or bounded depth circuits.

The remaining part of the paper is organized as follows. In section 2 we prove cryptographic weakness for a number of fundamental computational models like read-k BDDs, and several types of constant depth circuits over AND-, OR-, MOD_m-, and threshold gates. (Theorems 2, 3, 4, 5) We do this by constructing corresponding distinguishing algorithms. The message of these results, adressed to practicians, is the following: Let $\mathcal{M}$ denote a nonuniform computational model for which there is an efficient distinguishing algorithm $D = D(n, m)$. Further suppose that the $\chi_{\mathcal{M}}$-complexity of a secret key encryption algorithm E is significantly smaller than the average $\chi_{\mathcal{M}}$-complexity of a random Boolean operation. Then E cannot be considered to be secure as D provides an efficient distinguishing attack against E.

In section 3 we relate the concept of distinguishing pseudorandom functions of low complexity from truly random functions to the concept of learning unknown function from a given set of low complexity functions. In terms of cryptanalysis, we compare the task of distinguishing output bits of a given secret key cipher from a truly random source and the task of gaining real information about the secret key. We prove the not very surprising fact that any learning algorithm can be converted into a distinguishing algorithm (Theorem 6). Further, and this is one of our main results, we prove that if membership queries are forbidden then efficient distinguishing algorithms can be simulated by efficient learning algorithms (Theorem 7).

In section 4 we relate the concept of distinguishability to the concept of Natural Proofs (Theorem 8) developed in [28] and strengthen an observation of *Razborov*

and *Rudich* on the connection of the cryptographical strongness of a model $\mathcal{M}$ and provability of lower bounds on $\chi_{\mathcal{M}}$. Moreover, we adress some open problems.

The proof of Theorems 3, being quite long and technical, has been separated from the paper. It can be found in the Appendix. Furthermore, the Appendix contains some usefull probability estimations which will be used at several places in the remaining sections

2 Cryptographic Strongness and Weakness for Particular Models

Before proving the nonexistence of PRFGs within several complexity classes let us ask for upper bounds for pseudorandom function generators. *Goldreich, Goldwasser*, and *Micali* [9] provide a construction of pseudorandom function generators based on (generic) pseudorandom bit generators. Using this method, *Naor* and *Reingold* [23], [24] construct the following function generator $F = (F_n)_{n\in\mathbb{N}}$, where F_n consists of functions of type $f_{(P,Q,g,\boldsymbol{a})}$. P is an n-bit prime, Q is a prime divisor of $P-1$, g an element of order q in $\mathbb{Z}_P^*$, and $\boldsymbol{a} = (a_0, a_1, \ldots, a_n)$ a sequence of $n+1$ elements of $\mathbb{Z}_Q$. The function $f_{(P,Q,g,\boldsymbol{a})}$ is defined by

$$f_{(P,Q,g,\boldsymbol{a})}(x) := (g^{a_0})^{\prod_{1\le i\le n} x_i a_i} \bmod P.$$

This function generator behaves pseudorandomly under the condition that the so called **Decisional Diffie Hellman Conjecture** is true [23]. They showed further that this function generator belongs to TC^0_5. In [15] we showed that *Naor* and *Reingold*s PRFG has even depth 3 unweighted threshold circuits of quasipolynomial size. Moreover, if a well believed number theoretic assumption is true, these circuits can be proved to have polynomial size, i.e., depth 3 unweighted threshold circuits seem to be cryptographically strong. In contrast to this, depth 2 unweighted threshold circuits can be proved cryptographically weak [15].

In this section, we show the following results.

Theorem 2. *There is a distinguishing algorithm of polynomially bounded ratio for weighted threshold-MOD_2 circuits, i.e., threshold-MOD_2 circuits are cryptographically weak.*

Proof. We use the following result of *Bruck* [7]. For all $f \in B_n$ the minimal number of MOD_2-nodes in a weighted threshold-MOD_2-circuits computing f can be bounded from below by $\|f\|_{\max}^{-1}$. Hereby, $\|f\|_{\max}$ denotes the maximal absolute value of a spectral coefficient of f, i.e.

$$\|f\|_{\max} = \max\{|(\hat{f}, \hat{l}_\alpha)|;\ \alpha \in \{0,1\}^n\},$$

where $\hat{f} : \{0,1\}^n \longrightarrow \{1,-1\}$ is defined by $\hat{f}(x) = (-1)^{f(x)}$, l_α is defined as $l_\alpha(x_1, \ldots, x_n) = \bigoplus_{i,\alpha_i=1} x_i$, and, for functions $f, g : \{0,1\}^n \longrightarrow \{1,-1\}$, the scalar product (f,g) is defined as $(f,g) = |X|^{-1} \sum_{x\in X} f(x)g(x)$.

Observe the relation $(f,g) = Pr[f = g] - Pr[f \neq g]$, and that the set $\{\hat{l}_\alpha;\ \alpha \in \{0,1\}^n\}$ form an orthonormal basis w.r.t. $(\cdot,\cdot)$.

Fix an arbitrary polynomially bounded function $m = m(n) \in n^{O(1)}$. Given the input parameters n and $m(n)$, the distinguishing algorithm D works as follows. Let f denote the function chosen by the oracle.

1.) Set $m' = m + 1$ and compute the minimal number $\tilde{n}$ such that for $\tilde{N} = 2^{\tilde{n}}$ it holds that $\tilde{N} \ge 6m'^2 \ln(m')$.

2.) Output 1 iff $\|g\|_{\max} \geq 1/m'$, where g denotes the subfunction of the oracle function f obtained by assigning 0 to each varaiable $x_{\tilde{n}+1}, \ldots, x_n$.

Observe that $\tilde{n} \in O(\log(n))$, that the running time of step 2 is $O(\tilde{N}^2) \in n^{O(1)}$, and that D needs $\tilde{N}$ queries to the oracle.

In the pseudorandom case, by the above mentioned result of [7], the probability that D accepts is one.

Consider now the truly random case and observe that

$$Pr[\|g\|_{\max} > 1/m'] \leq 2\tilde{N} Pr\left[\sum_{i=1}^{\tilde{N}} X_i > \tilde{N}/m'\right],$$

where X_i denote mutually independent random variables taking values 1 and -1 with $Pr[X_i = 1] = Pr[X_i = -1] = 1/2$. By relation (12) (*Hoeffdings* Inequality, see Appendix, subsection 5.1), it holds that

$$Pr[\|g\|_{\max} > 1/m'] \leq 2\tilde{N} e^{-(\tilde{N}^2/(2m'^2\tilde{N}))} \leq 12m'^2 \ln(m') e^{-3\ln(m')} \leq 12\ln(m')/m'.$$

Observe that the last term is smaller than $1/4$ for $m' \geq e^5$. We obtain a polynomial distinguishing ratio for D. □

Theorem 3. *For all primes p and all constant depth bounds d there is a distinguishing algorithm of quasipolynomially bounded ratio for depth d circuits over $\{AND, OR, MOD_p\}$.*

The proof is quite lengthy and can be found in the Appendix, subsection 5.2. As MOD_{p^k} belongs to $AC_2^0[p]$ [30], the proof for prime powers follows immediately.

Theorem 4. *For all $k \geq 1$ it holds that there is a distinguishing algorithm of quasipolynomially bounded ratio for nondeterministic syntactic read–k times branching programs.*

Proof. The first exponential lower bounds on read k branching programs were independently proved in [6] and [27]. See also [13] for further nice applications of the method. We use these methods for designing the desired distinguishing algorithm D. Let us fix an arbitrary natural constant $k \geq 1$, a polynomial bound $m = m(n) \in n^{O(1)}$, an input length n and a Boolean function $f \in B_n$ computable by a nondeterministic syntactic read–k times branching program of size $m(n)$. Let us denote $X_n = \{x_1, \ldots, x_n\}$.

In [13] *Jukna* shows the existence of a number $W \in m^{O(1)} = n^{O(1)}$ and a constant $\gamma \in (0,1)$ such that f can be written as

$$f = \bigvee_{i=1}^{W} f_i, \tag{1}$$

where for all i, $1 \leq i \leq W$, it holds that there is a partition $X_n = U_i \cup V_i \cup W_i$ of pairwise disjoint subsets U_i, V_i, W_i of X_n such that

$$f_i(X_n) = g_i(U_i, V_i) \wedge h_i(V_i, W_i),$$

where $|U_i| \geq \gamma n$ and $|W_i| \geq \gamma n$.

The distinguishing algorithm D gets n and m as input parameters and computes at first internal parameters s, q, Q, and r which will be specified later. Then D proceeds as follows.

Step 1: Test whether $Pr[f(x) = 1] \geq \frac{1}{3}$ by asking $f(x^j)$ for n randomly chosen inputs $x^1, \ldots, x^n$ and stop with output 1 if

$$\sum_{j=1}^{n} f(x^j) \ < \frac{5n}{12}.$$

Step 2: Choose randomly disjoint subsets U, W from X_n with $|U| = |W| = q$, and a $\{0,1\}$-assignment b of $V = X \setminus (U \cup W)$. Let $Q = 2^q$.

Step 3: Choose random assignments $a^1, \ldots, a^r$ of U. Stop with output 1 if

$$\sum_{c=0}^{Q-1} F(c) \ \geq \ \frac{Q}{6W},$$

where for all $\{0,1\}$-assignments c of W the value $F(c) \in \{0,1\}$ is defined as

$$F(c) = 1 \quad \text{iff} \quad f(a^1, b, c) = \ldots = f(a^r, b, c) = 1.$$

Output 0 otherwise.

We show that for $q = \lfloor \log_2(W^2 n) \rfloor$ and $r = \lfloor \log_2(12W) \rfloor$ it holds that $\varepsilon_D(f) \geq \epsilon(n,m)$, where $\epsilon(n,m)^{-1} \in m^{O(\log(m))} = n^{\log^{O(1)} n}$.

In the truly random case, by inequality (13) from subsection 5.1, it follows that the probability that D stops accepting in step 1 is not greater than

$$e^{-2(1/144)n} \ = \ e^{-(1/72)n}. \tag{2}$$

Now look at inequality (15) and let $p = 2^{-r} \leq 1/12W$. Observe that $Q \geq W^2 n$. It follows that the probability that D stops with output 1 in step 3 is not greater than

$$e^{-2(1/6W-p)^2 Q} \ \leq \ e^{-2(1/144W^2)Q} \ \leq \ e^{-(1/72)n}. \tag{3}$$

Consider now the pseudorandom case and suppose that the oracle has chosen the secret function f.

Case 1: It holds $Pr[f(x) = 1] \ < \ \frac{1}{3}$. Then the probability that D stops accepting in Step 1 is at least $1 - e^{-n/72}$. This follows as above from inequality (15) by setting $p = \frac{1}{3}$.

Case2: It holds $Pr[f(x) = 1] \geq \frac{1}{3}$. We estimate the probability that D stops accepting in Step 3.

Observe at first that by (1) there is a number j, $1 \leq j \leq W$, such that

$$Pr[f_j(x) = 1] \ > \ \frac{1}{3W}. \tag{4}$$

As q is polylogarithmically bounded in n we have that $q < (\gamma/2)n$ for n large enough. Then it follows from relation (16) in subsection 5.1 that the probability for the event (E1) that $U \subseteq U_j$ and $W \subseteq W_j$ is at least $P_1 := (\gamma/2)^{2q}$. Observe further that the probability for event (E2) that we choose in Step 2 an assignment b of V fulfilling

$$Pr_{a,c}\left[f_j^b(a,c) = 1\right] > 1/6W$$

is at least $P_2 := 1/6W$. (Otherwise we would have that

$$Pr_x\left[f_j(x) = 1\right] < 1/6W + (1 - 1/6W)1/6W = (1/6W)(2 - 1/6W) < 1/3W$$

which contradicts (4).) Observe that under condition (E_1) for all assignments a of U and c of W it holds that

$$f_j(a,b,c) = g_j^b(a) \wedge h_j^b(c) \quad \text{and} \quad f_j(a,b,c) = 1 \Longrightarrow f(a,b,c) = 1.$$

By (4) we obtain that

$$Pr_a[g_j^b(a) = 1] \geq 1/6W \quad \text{and} \quad Pr_c[h_j^b(c) = 1] \geq 1/6W. \tag{5}$$

Consequently, under the condition (E1) and (E2), the probability for the event (E3) that for all i, $1 \leq i \leq r$, it holds that $g_j^b(a^i) = 1$, is at least $P_3 := (1/12W)^r$. (This follows from (16).)

From (5) we obtain that under condition (E1), (E2), and (E3) it holds that

$$Pr_c[f(a^1, b, c) = \ldots = f(a^r, b, c) = 1] \geq Pr_c[f_k(a^1, b, c) = \ldots = f_k(a^r, b, c) = 1]$$

$$\geq Pr_c[h_k^b(c) = 1] \ \geq \ 1/6W.$$

It follows that the probability that D stops accepting in Step 3 is at least

$$P_1 P_2 P_3 \ = \ (\gamma/2)^{2q}(1/6W)(1/12W)^r.$$

It is easy to check (looking at (2) and (3)) that the advantage of D is at least

$$\epsilon(n, m) = P_1 P_2 P_3 - 2e^{-n/72}$$

and that $\epsilon(n, m)^{-1} \in m^{O(\log(m))}$. □

Theorem 5. *For all $k \geq 1$ it holds that there is a distinguishing algorithm of quasipolynomially bounded ratio for depth $k + 1$ circuits consisting of k levels of AND and OR gates connected with one weighted threshold gate as output gate.*

Proof. Let us call an unbounded fanin depth k circuit Σ_k-circuit, resp. Π_k-circuit, if the circuit consists of k inner levels, which contain either only AND-gates, or only OR-gates, and if the top gate is an OR-gate, resp. an AND-gate.

We use the fact that for each Boolean function f with polynomial size weighted threshold-Π_k, or with polynomial size weighted threshold-Σ_k circuits the following holds. With high probability, a random subfunction of f can be written as the sign of a real polynomial with polylogarithmically bounded degree. We consider the set $\{0, 1, *\}^n$ of partial assignments to the set of variables $\{x_1, \ldots, x_n\}$ with respect to the probability distribution $R(p)$ which is defined by

$$Pr[\rho] = \Pi_{i=1}^n Pr[\rho_i],$$

where $Pr[\rho_i = *] = p$, and $Pr[\rho_i = 0] = Pr[\rho_i = 1] = (1 - p)/2$.

We exhibit the *Switching Lemma* [11] saying that for all $f \in B_n$, $p \in (0, 1)$ and $s, t \leq n$ it holds the following. If f has a Σ_2- (resp. Π_2-circuit) of bottom fan-in $\leq t$ than the probability that f^ρ has a Π_2-circuit (resp. Σ_2-circuit) of bottom fan-in $\leq s$ is at least $1 - \alpha^s$, where the partial assignment ρ is distributed according to $R(p)$ and the value α can be estimated by $\alpha < 5pt$ (see [31] pp. 325-331 for a nice presentation of the proof).

Moreover, it is shown in [19] that if f has a Σ_2-circuit of bottom fan-in $\leq t$ and a Π_2-circuit of bottom fan-in $\leq s$ then f has a decision tree of depth st, and, consequently, can be computed exactly by a real polynomial of degree st.

Let us fix a polynomial bound $m = m(n) \in n^{O(1)}$ and suppose that $f \in B_n$ can be computed by a threshold-Σ_k circuit S, where each level of the circuit consists of at most $m(n)$ nodes. The case of threshold-Π_k circuits can be treated in a similar way. Fix $s \in O(\log(n))$ to be the smallest number for which $2^s \geq m(n)$. The gates at level 1 of S can be seen as Σ_2- (resp. Π_2-) circuits of bottom fanin $1 \leq s$. Fix an appropriate probability p, which will be specified later, and consider partial assignments ρ of $\{x_1, \ldots, x_n\}$ to be distributed according to $R(p)$. Observe that a standard probability estimation shows that the probability that f^ρ depends on at

least pn variables is at least $1/3$. Consequently, the probability that each bottom gate of S can be replaced by an equivalent Π_2- (resp. Σ_2-) circuits of bottom fanin s is at least

$$1 - 2/3 - 2^s \alpha^s < 1/3 - (10ps)^s.$$

We fix a number r in such a way that for $p = 2^{-r}$ holds $(10ps)^s \leq 1/6$. Observe that $p^{-1} \in O(\log(n))$.

It follows that the probability that f^ρ depends on at least pn variables and has threshold-Σ_k of width $m(n)$ and bottom fanin s is at least $1/6$. This argument can be iteratively applied to f^ρ. It turns out that for ρ distributed according to $R(p^k)$, the probability that f^ρ depends on at least $p^k n$ variables and has voting polynomial degree s^2 is at least $(1/6)^k$. Observe that this implies that f^ρ has threshold-MOD_2 circuits of size

$$\phi(n,s) = \sum_{i=0}^{s^2} \binom{n}{i} \in n^{O(\log^2 n)},$$

i.e., we can apply the distinguishing algorithm for threshold-MOD_2 circuits. Let $m' = \phi(n,s) + 1$ and $\tilde{n}$ and $\tilde{N}$ be defined as above in the proof of Theorem 2. We suppose that n, s are large enough such that $12\ln(m')/m' < (1/6)^{k+1}$ and $p^k n > \tilde{n}$.

The distinguishing algorithm for weighted threshold-Σ_k- and weighted threshold-Π_k- circuits of width $m(n)$ works as follows. Choose randomly a partial assignment ρ of $\{x_1, \ldots, x_n\}$, where ρ is distributed according to $R(p^k)$ and test whether f^ρ has weighted threshold-MOD_2 circuits of size $\phi(n,s)$ with the algorithm of Theorem 2. The choice of the internal parameters p, s, m' and $\tilde{n}$ yields that the advantage is at least $(1/6)^k - (1/6)^{k+1}$ and that the running time is quasipolynomially bounded in n. □

Observe that in all these cases, the ideas for the distinguishing algorithms for the particular models $\mathcal{M}$ can be derived from known lower methods for the corresponding complexity measures $\chi_{\mathcal{M}}$. In a certain sense, the distinguishing algorithms can be considered as the algorithmical versions of the corresponding lower bound methods. See section 4 for a discussion about the existence of distinguishing algorithms versus lower bound proofs.

3 Distinguishing versus Learning

In this section, we describe the relation between distinguishability of function generators and the learnability of Boolean concept classes. (A Boolean concept class is simply a set of Boolean functions of the same input length. Observe that each function generator F can be identified with a sequence of Boolean concept classes in a straightforward way.)

In our setting, the definition of a learning algorithm L for a sequence of concept classes $C = (C_n)_{n \in \mathbb{N}}$, $C_n \subseteq B_n$, refers to the following scenario. It consists of a randomized oracle Turing machine L which knows C, gets the input parameter n and communicates with a C_n-oracle via membership queries. L outputs the standard encoding of a Boolean circuit computing a Boolean function $h \in B_n$, called the hypothesis.

For given numbers $\epsilon(n), \delta(n) \in (0,1)$, L is called an $(\epsilon(n), \delta(n))$-learner for C if for all $n \in \mathbb{N}$ the probability (w.r.t. the internal randomization of L) that L outputs an $(1/2 + \epsilon(n))$-approximator of f (i.e., a function h fulfilling $Pr_x[f(x) = g(x)] \geq 1/2 + \epsilon(n)$) is at least $\delta(n)$. The product of the worst case running time of L and $(\epsilon(n)\delta(n))^{-1}$ is called the learning ratio r_L of L.

Observe that equivalence queries can also be simulated probably almost correct in this model. (An equivalence query is defined by submitting the encoding of a hypothesis h to an oracle which knows the function to be learnt. The oracle answers

"o.k." if $h = f$, or returns a counterexample x with $h(x) \neq f(x)$.) A simulates equivalence queries as follows. It randomly chooses inputs $x^1, \ldots, x^t \in \{0,1\}^n$ for some appropriate t and tests for all $i, 1 \leq i \leq t$, whether $h(x^i) = f(x^i)$. Either one gets a counterexample, or, with high probability, h is a good approximation for f. Consequently, the usual learning models for Boolean concept classes like PAC learning with respect to the uniform distribution and exact learning with membership and equivalence queries are captured by our model. Observe the following connection between learnability and distinguishability.

Theorem 6. *Each $(\epsilon(n), \delta(n))$-learner L for a sequence of concept classes $C = (C_n)_{n \in \mathbb{N}}$ can be transformed into a distinguishing algorithm D for C with a distinguishing ratio $r_D \in r_L^{O(1)}$.*

Proof. D communicates with an H-oracle, where $H \in \{B_n, C_n\}$, which has been chosen a secret function f. D chooses an appropriate number $s \in O(\delta^{-1}(n))$ and does s mutually independent trials of L. One gets s encodings of hypotheses $h^1, \ldots, h^s$. If there is an output which is not a standard encoding of a fully specified circuit over n input variables, then D decides that $H = B_n$ and outputs 0. The next step of D is to fix an appropriate number $t \in O(\epsilon^{-1}(n))$ and to randomly choose t mutually independent inputs $x^1, \ldots, x^t \in \{0,1\}^n$. D accepts if there is some $i, 1 \leq i \leq s$, such that for at least $t(1+\epsilon)/2$ of the inputs $x^j, 1 \leq j \leq t$, it holds that $h^i(x^j) = f(x^j)$. Using standard probability estimations like *Hoeffding's* inequality (see Appendix, subsection 5.1) it is quite straightforward to verify that D has the required properties. □

Observe that it is easy to construct concept classes which can be efficiently distinguished but not learned. Take for instance the set of all P/poly-functions with the additional property that the image of the constant-0 vector is one. We can distinguish this set with advantage 1/2 by asking for the value of the constant-0 vector. However, the (quasi)polynomial learnability of this set would imply that unrestricted Boolean circuits are efficiently learnable, which is a contradiction to fundamental cryptographic hardness assumptions (see [14], [28]).

Given a computational model $\mathcal{M}$ we say that $P(\mathcal{M})$-functions are polynomially, resp. quasipolynomially, learnable iff there is an universal learning algorithm learning every sequence of concept classes from $P(\mathcal{M})$ with polynomially, resp. quasipolynomially bounded ratio. By Theorem 6 it follows that if $P(\mathcal{M})$-functions are quasipolynomially learnable then we have a distinguishing algorithm of quasipolynomially bounded ratio for $\mathcal{M}$, i.e., $\mathcal{M}$ is cryptographically weak. There are known polynomially bounded learning algorithms for depth 2 circuits over AND,OR,¬ [12], for OBDDs [2] and quasipolynomially bounded learning algorithms for depth k circuits over AND,OR,¬ for all constants $k \geq 3$ [19].

Observe that cryptographic limitations of learning were already detected by *Kearns* and *Valiant* in [14]. It is shown there that efficient learnability of TC_3^0-functions would imply a contradiction to widely believed cryptographic hardness assumptions like the security of RSA or *Rabin*'s cryptosystem. This is because the corresponding cryptographic primitives $x^y \mod z$ and $x^2 \mod z$ belong to TC_3^0, see [29]. Note that for all models $\mathcal{M}$ for which we have shown cryptographical weakness in the previous section it holds that it is unknown whether $P(\mathcal{M})$- functions are efficiently learnable.

Let us now consider the case of **passive** learning and distinguishing. Here, the randomized algorithm is not allowed to ask membership queries. The oracle, on demand, provides pairs $(x, f(x))$, where x is randomly chosen w.r.t. the uniform distribution on the input set for f. Observe that, in the terminology of practical cryptanalysis, **active distinguishing** corresponds to a so called **Chosen Plaintext Attack** and **passive distinguishing** to a so called **Known Plaintext Attack**.

It is one of the main results of this paper that in the passive case efficient distinguishing algorithms can be simulated by efficient learning algorithms.

Theorem 7. *Each passive distinguishing algorithm D of (worst case) ratio r_D for a sequence of concept classes $C = (C_n)_{n\in\mathbb{N}}$ can be transformed into a passive learning algorithm L for C with learning ratio $r_D^{O(1)}$.*

Proof. Fix an input length n and a function $f \in C_n$. Denote by t and ε the worst case time, resp. worst case advantage of D on input length n.

For each Boolean function $h : \{0,1\}^n \longrightarrow \{0,1\}$ denote by $adv(h)$ the advantage which h achieves w.r.t. f, i.e.,

$$adv(h) = Pr[h(x) = f(x)] - Pr[h(x) \neq f(x)],$$

where the probabilities are taken w.r.t. to the uniform distribution on $\{0,1\}^n$. Observe that for all $\gamma \in [-1,1]$, h is an $(\frac{1}{2}+\gamma)$-approximator of f iff $adv(h) = 2\gamma$.

For a subset $X \subseteq \{0,1\}^n$ we denote by $adv(h,X)$ the advantage which h achieves w.r.t. f on X, i.e.

$$adv(h,X) = Pr[X](Pr[h(x) = f(x)|x \in X] - Pr[h(x) \neq f(x)|x \in X]).$$

Observe that for all partitions of $\{0,1\}^n$ into disjoint subsets $X^0,\ldots,X^P$ it holds that

$$adv(h) = \sum_{u=0}^{P} adv(h,X^u). \tag{6}$$

Suppose that, during each computation, C sends $l \leq t$ requests to the oracle. We call, for $0 \leq k \leq l$, a sample $(x^1,y_1),\ldots,(x^l,y_l)$ (f,k)-distributed if the x^j are uniformly and independently chosen from $\{0,1\}^n$, if for all $j \leq k$ it holds that $y_j = f(x^j)$, and if for all $j > k$ it holds that the $y_j = f'(x^j)$ for some random function $f' \in B_n$. (The last condition means that all different inputs occuring in $\{x^j,\ k+1 \leq j \leq l\}$ get their y-value via an unbiased coin flip, but for all $j \neq j' \in \{k+1\ldots l\}$ it must hold that if $x^j = x^{j'}$ then $y_j = y_{j'}$.)

We denote by $Pr(f,k)$ the probability that D outputs 1 on a (f,k)-distributed sample.

Observe that by definition $Pr(f,l) - Pr(f,0) \geq \varepsilon$. Consequently, there is some $k,\ 1 \leq k \leq l$, such that

$$Pr(f,k) - Pr(f,k-1) \geq \frac{\varepsilon}{l} \geq r_D(n)^{-1}. \tag{7}$$

Let us denote by P the minimal natural number for which $\varepsilon(n)/l(n) \leq 1/P$.

For all $k,\ 1 \leq k \leq l$, inputs x and $b \in \{0,1\}$, a a random sample $(x^1,y_1),\ldots,(x^l,y_l)$ is said to be (f,k,x,b)-distributed if for all $j, 1 \leq j \leq l,\ j \neq k$, the x^j are uniformly and independently chosen from $\{0,1\}^n$, if for all $j < k$ it holds that $y_j = f(x^j)$, if for all $j > k$ it holds that the $y_j = f'(x^j)$ for some random function $f' \in B_n$, and if $x^k = x$ and $y_k = b$.

For $b \in \{0,1\}$ denote by $p^k(x,b)$ the probability that D accepts an (f,k,x,b)-distributed sample and let $d^k(x) = p^k(x,f(x)) - p^k(x,\neg f(x))$. Observe that

$$Pr(f,k) - Pr(f,k-1)$$

$$= \mathbf{E}_x[p^k(x,f(x)) - \frac{1}{2}(p^k(x,f(x)) + p^k(x,\neg f(x)))] = \frac{1}{2}\mathbf{E}_x[d^k(x)].$$

We call a number $k,\ 1 \leq k \leq l(n)$, to be "good" if

$$\mathbf{E}_x[d^k(x)] \ \geq \ \frac{2}{P}. \tag{8}$$

Observe that relation (7) ensures the existence of "good" numbers k.

For all k, $1 \leq k \leq l$, define the Boolean function $h^k : \{0,1\}^n \longrightarrow \{0,1\}$ as

$$h^k(x) = 1 \quad \text{iff} \quad p^k(x,1) > p^k(x,0).$$

Observe that, in general, even under the condition that k is "good" the function h^k is not a good approximator of f which can be illustrated with the following example. Suppose that d^k is distributed as follows: It holds $d^k(x) = 4/P$ with probability $1/2$ and $d^k(x) = -(2/P)$ with probability $1/2$. It is easy to see that $adv(h^k) = 0$.

The idea for constructing the hypothesis is to find an appropriate subset X^* of $\{0,1\}^n$ such that, if k is good, $adv(h^k, X^*)$ is sufficiently large, and to define the hypothesis H to coincide with h^k on X^*, and to be constant b, for a random constant $b \in \{0,1\}$, outside of X^*. As on each subset of $\{0,1\}^n$ either the constant 0 or the constant 1 achieves nonnegative advantage we obtain that the advantage of H is not smaller than $adv(h^k, X^*)$ with probability at least $1/2$.

It remains to construct an appropriate set X^*. For all $X \subseteq \{0,1\}^n$ let

$$q(X) = Pr[x \in X],$$

$$q^k_{=}(X) = Pr[p^k(x, f(x)) > p^k(x, \neg f(x)) | x \in X],$$

$$q^k_{\neq}(X) = Pr[p^k(x, f(x)) < p^k(x, \neg f(x)) | x \in X],$$

where the probability is always taken w.r.t. the uniform distribution on $\{0,1\}^n$. Observe that if $p^k(x, f(x)) \neq p^k(x, \neg f(x))$ for all $x \in X$ then

$$adv(h^k, X) = q(X)(q^k_{=}(X) - q^k_{\neq}(X)).$$

For $k \in \{1, \ldots, l(n)\}$, a partition $\Pi = (X_0, X_1, \ldots, X_P)$ of the set $\{0,1\}^n$ into $P+1$ disjoint subsets is called a k-partition if for all $u = 0 \ldots P$ it holds that for all $x \in X_u$

$$\left| |d^k(x)| - \frac{u}{P} \right| \ \leq \ \frac{1}{P}.$$

A standard example for a k-partition is given by

$$X_u = \{x \in \{0,1\}^n, \ \frac{u}{P} \leq |d^k(x)| < \frac{u+1}{P}\}.$$

Observe that for each k-partition it holds

$$\mathbf{E}_x[d^k(x)] \ = \ \sum_{u=0}^{P} q(X_u) \mathbf{E}_x[d^k(x) | x \in X_u].$$

Further remember that for all $u = 0 \ldots P$

$$|\mathbf{E}_x[d^k(x) | x \in X_u] - \mathbf{E}_x[\frac{u}{P} | x \in X_u]| \ \leq \ \frac{1}{P}.$$

It holds $\mathbf{E}_x[\frac{u}{P} | x \in X_u] = \frac{u}{P}(q^k_{=}(X_u) - q^k_{\neq}(X_u))$. Consequently,

$$\frac{1}{P} \ \geq \ |\mathbf{E}_x[d^k(x)] - \sum_{u=1}^{P} q(X_u) \frac{u}{P} (q^k_{=}(X_u) - q^k_{\neq}(X_u))|$$

$$= |\mathbf{E}_x[d^k(x)] - \frac{1}{P} \sum_{u=1}^{P} u \cdot adv(h^k, X^u)|$$

It follows that if k is "good" then

$$\frac{1}{P}\sum_{u=1}^{P} u \cdot adv(h^k, X^u)| \ \geq \ \frac{2}{P} - \frac{1}{P} = \frac{1}{P}.$$

Consequently, there is some u, $0 \leq u \leq P$, with $adv(h^k, X^u) \geq \frac{1}{P^2}$. Let us call such a number u to be "good" with respect to Π.

Clearly, the computation of $h^k(x)$ and the test whether $x \in X_u$ is based on computing good estimations for $d^k(x)$. This will be done by computing the following values. For a parameter $U \in \mathbb{N}$, which will be specified later, let a random string R consist of U mutually independently chosen $(f, k, x, 0)$-distributed samples $S_1^0, \ldots, S_U^0$ and of U mutually independently chosen $(f, k, x, 1)$-distributed samples $S_1^1, \ldots, S_U^1$. Let the random values $p^k(x, 0, R)$, $p^k(x, 1, R)$, $\delta^k(x, R)$ and $h^k(x, R) \in \{0, 1\}$ be defined as

$$p^k(x, 0, R) = \frac{1}{U}\sum_{i=1}^{U} D(S_i^0),$$

$$p^k(x, 1, R) = \frac{1}{U}\sum_{i=1}^{U} D(S_i^1),$$

$$\delta^k(x, R) = |p^k(x, 0, R) - p^k(x, 1, R)|$$

$$h^k(x, R) = 1 \quad \text{iff} \quad p^k(x, 1, R) > p^k(x, 0, R).$$

Observe that by inequalities (15) it follows that for all inputs x, all $b \in \{0, 1\}$ and all natural W the probability that $p^k(x, b, R)$ differs by more than $1/W$ from $p^k(x, b)$ is at most $2e^{-2U/(W^2)}$. On the other hand, if both values differ by at most $1/2W$ then $\delta^k(x, R)$ and $|d_k(x)|$ differ by at most $1/W$. Consequently, for all natural U, W it holds that

$$Pr\left[\left||d^k(x)| - \delta^k(x, R)\right| > \frac{1}{W}\right] \ < \ 2(2e^{-2(1/(2W))^2 U}) \ = \ 4e^{-U/(2W^2)}. \qquad (9)$$

If we decide about $x \in X_u$ via the random value $\delta^k(x, R)$ then the following problem could arise. If all $d^k(x)$-values of the elements in X_u are concentrated exponentially (in P) close to u/P or to $(u+1)/P$ then for all polynomially bounded U the test gives the wrong answer with probability nearly $1/2$. In order to prevent this situation it has to be guaranteed that most of the $d^k(x)$-values of the $x \in X_u$ belong to

$$[\frac{u}{P} + \frac{1}{W}, \frac{u+1}{P} - \frac{1}{W})$$

for some appropriate polynomially bounded W, and then to test whether

$$\delta^k(x, R) \in [\frac{u}{P} + \frac{1}{2W}, \frac{u+1}{P} - \frac{1}{2W}).$$

The only way we found to guarantee this is to choose the k-partition Π randomly from an appropriate polynomially bounded family of k-partitions.

We fix an appropriate positive natural number S which will be specified later. For all $s \in \{1, \ldots, S\}$ we define the partition $\Pi^s = \{X_0^s, \ldots, X_P^s\}$ as

$$X_u^s = \{x \in \{0, 1\}^n, \ a_u^s \leq |d^k(x)| < a_{u+1}^s\},$$

where $a_0^s = 0$, $a_{P+1}^s = 1$ and $a_u^s = \frac{u-1}{P} + \frac{s}{SP}$ for $u = 1, \ldots, P$. Observe that for all s Π^s is a k-partition.

Further let the error set E^s contain all those elements x for which

$$a_u^s - \frac{1}{2PS} \leq |d^k(x)| < a_u^s + \frac{1}{2PS}$$

for some $u \in \{0, \ldots, P+1\}$. For all $u = 0 \ldots P$, let Y_u^s be defined as $Y_u^s = X_u^s \setminus E^s$.

We say that s is "good" with respect to k if $q(E^s) \leq 1/S$. The fact that we can cover $\{0,1\}^n$ by at most S disjoint E^s sets guarantees the existence of "good" numbers s.

Let $S = 1/(4P^2)$. It can be straightforwardly shown that if k is good, if s is good with respect to k, and if u is good with respect to k and Π^s then

$$adv(h^k, Y_u^s) \;\geq\; \frac{1}{P^2} - \frac{1}{4P^2} = \frac{3}{4P^2}. \tag{10}$$

The learning algorithm is defined as follows. (The parameter U will be specified below, x denotes an input.)

1 choose a random $k \in \{0, \ldots, l\}$
2 choose a random $s \in \{0, \ldots, S\}$
3 choose a random $u \in \{0, \ldots, P\}$
4 choose a random $b \in \{0,1\}$
5 choose a random assignment R as defined above
6 output a Boolean circuit for the n-nary Boolean function assigning to each $x \in \{0,1\}^n$ the value $H^k(s,u,b,x,R) \in \{0,1\}$ which is defined by the following rule:
7 **if** $a_u^s + \frac{1}{4PS} < \delta^k(x,R) \leq a_{u+1}^s - \frac{1}{4PS}$
8 **then** $H^k(s,u,b,x,R) = h^k(x,R)$
9 **else** $H^k(s,u,b,x,R) = b$

Let us suppose that we haven chosen a good k, an s which is good w.r.t. k, and an u which is good w.r.t. k and Π^s. This happens with probability at least $\Omega(P^{-4})$. For estimating the advantage of the hypothesis we define a random error function $e : \{0,1\}^n \longrightarrow \{0,1\}$ as follows. Let $e(x) = 1$ if $x \notin X_u^s$ but the test in line 7 yields true, or if $x \in X_u^s$ but the test in line 7 yields false.

Observe that for $x \notin X_u^s \setminus Y_u^s$ the probability, taken with respect to R, that $e(x) = 1$ is, by (9), bounded by $4e^{-\frac{U}{2(4PS)^2}} = 4e^{-\frac{U}{512P^6}}$. Now choose U to be the minimal natural number for which

$$4e^{-\frac{U}{512P^6}} \;<\; 2^{-(n+1)} \tag{11}$$

and observe that $U \in O(nP^6)$, i.e., $H^k(s,u,b,x,R)$ can be computed by a deterministic algorithm of running time $O(n(r_D(n))^6 t(n))$.

Observe that due to (11) the probability, taken over all R, that $e(x) = 0$ for **all** x outside $X_u^s \setminus Y_u^s$ is at least $1/2$. We will call such an R to be good with respect to k, s and u.

It follows that with probability $1/4$, taken over R and b, $h = H^k(s,u,b,\cdot,R)$ achieves a nonnegative advantage outside X_u^s and advantage $\frac{3}{4P^2}$ on Y_u^s. As the advantage of h on $X_u^s \setminus Y_u^s$ can decrease this value by at most $|X_u^s \setminus Y_u^s| \leq 1/S = 1/(4P^2)$ we obtain that h achieves an advantage of at least $1/(2P^2)$. Consequently, the learning ratio of the above algorithm is bounded by $O(nr_D^1 2(n))$. □

4 Final Discussion: Distinguishing Algorithms, Lower Bound Methods, Open Problems

Let us relate the notion of pseudorandomness underlying our concept of distinguishability to a stronger definition of pseudorandomness occuring in the literature, e.g. in [26] and [28]. Let Γ be a complexity class and $T = (T_n) \in \Gamma$ be a sequence of Boolean functions for which the input length of T_n is N=2^n. T is called an efficient Γ-test against a function generator $F = (F_n)_{n\in\mathbb{N}}$ if for all n

$$|Pr_f[T_n(f) = 1] - Pr_s[T_n(f_{n,s}) = 1]| \geq p^{-1}(N)$$

for a polynomially bounded function $p : \mathbb{N} \longrightarrow \mathbb{N}$. Hereby, functions $f \in B_n$ are considered strings of length $N = 2^n$. The probability on the left side is taken w.r.t. the uniform distribution on B_n (the truly random case), the probability on the right side is taken w.r.t. the uniform distribution on F_n (the pseudorandom case).

A function generator F is called **pseudorandom w.r.t. Γ-tests** if there is no efficient Γ-test against F. A computational model $\mathcal{M}$ is said to be **cryptographically strong w.r.t. Γ-tests** if there is a function generator in $P(\mathcal{M})$ which is pseudorandom w.r.t. Γ-tests, and **cryptographically weak w.r.t. Γ-tests** otherwise. *Nisan* constructed in [26] a function generator in $AC^0[2]$ which is pseudorandom w.r.t. AC^0-tests, i.e., $AC^0[2]$ is cryptographically strong w.r.t. AC^0-tests (but, due to Theorem 3, cryptographically weak w.r.t. distinguishing algorithms).

Observe that, in contrast to our concept of pseudorandomness which refers to distinguishing algorithms, the existence of an efficient Γ-test for a given PRFG does not give any feasible attack against the corresponding cipher because the whole function table has to be processed.

In [28], at hand of a set of representative examples, *Razborov* and *Rudich* give convincing empirical evidence that for all nonuniform computational models $\mathcal{M}$ it holds the following. If there is known an effective lower bound method for $\mathcal{M}$ (i.e., a method which allows to prove superpolynomial lower bounds on the $\chi_{\mathcal{M}}$-complexity of some explicitly defined sequences of Boolean functions) then $\mathcal{M}$ is cryptographically weak w.r.t. Γ-tests for some $\Gamma \subseteq P/poly$.

In particular, on the one hand each effective lower bound method for $\mathcal{M}$ can be transformed into a so-called *Γ-natural proof* for $P(\mathcal{M})$ for some $\Gamma \subseteq P/poly$ (the somewhat technical definition of natural proofs is omitted here). On the other hand (and this is the property of natural proofs which is important in our context) each Γ-natural proof for $P(\mathcal{M})$ yields an efficient Γ-test against any function generator in $P(M)$. Consequently, a $P/poly$-natural proof against $P/poly$ would imply the nonexistence of function generators which are pseudorandom w.r.t. $P/poly$-tests. But this implies the nonexistence of pseudorandom bit generators [28], contradicting widely believed cryptographic hardness assumptions. We strengthen the observation of [28] in the following sense.

Theorem 8. *If there is a distinguishing algorithm D of ratio $2^{O(n)}$ for a function generator $F = (F_n)_{n\in\mathbb{N}}$ then there is an efficient $P/poly$–test against F.*

Proof. Let $R(n)$ denote the set of random seeds used by D on input length n. For all $r \in R(n)$ and $f \in B_n$ denote by $D(n,r)(f) \in \{0,1\}$ the output of D on oracle function f under the condition that the internal random bits of D take values according to r. As D is $2^{O(n)}$ time bounded, by a standard simulation result of Turing machines by Boolean circuits it is straightforward to derive that for all $r \in R(n)$ the Boolean function $D(n,r) : B_n \longrightarrow \{0,1\}$ has Boolean circuits of size $2^{O(n)} = (2^n)^{O(1)}$.

Observe further that the advantage $\epsilon(n)$ of D can be written as

$$\mathbf{E}_r \left[Pr_s[D(n,r)(f_s) = 1] - Pr_f[D(n,r)(f) = 1] \right],$$

where the probability on the left side is taken w.r.t. the uniform distribution on F_n and the probability on the right side is taken w.r.t. the uniform distribution on B_n. Consequently, for all n there is a random seed $r^*(n)$ such that

$$|Pr_s[D(n,r^*(n))(f_s) = 1] - Pr_f[D(n,r^*(n))(f) = 1]| \geq \epsilon(n).$$

As $\epsilon(n)^{-1} \in 2^{O(n)} = (2^n)^{O(1)}$ it follows that $D(n,r^*(n))$ define effective $P/poly$-tests for F. □

Consequently, each function generator which belongs to a cryptographically weak complexity class has efficient $P/poly$-tests.

On the other hand, the efficient distinguishing algorithms for particular complexity classes given in section 2 can be summarized by the following observation.

For each nonuniform computational model $\mathcal{M}$ it seems to hold the following. If there is an effective lower bound method for $\chi_{\mathcal{M}}$ then $\mathcal{M}$ is not only cryptographically weak w.r.t. Γ-tests for some complexity class $\Gamma \subseteq P/poly$ (as it was observed in [28]), but $\mathcal{M}$ is even cryptographically weak w.r.t. distinguishing algorithms of quasipolynomially (sometimes even polynomially) bounded ratio.

At the moment, we know only one computational model $\mathcal{M}$, namely threshold-MOD_p circuits, p an odd prime power, for which we know an effective lower bound method (see [16]) but no efficient distinguishing algorithm. We adress this as an open problem.

A deeper investigation of the relations between distinguishability of function generators, learnability of Boolean concept classes and provability of lower bounds for nonuniform complexity measures is, by our oppinion, an interesting challenge for further research.

Is it possible to construct efficient distinguishing algorithms for models like constant depth circuits over AND-, OR-, MOD_m-gates, m a composite number, or for weighted depth 2 threshold circuits? Both models could neither be proved to be cryptographically weak, nor strong. It is open to prove superpolynomial lower bounds for these models. Is it easier to find distinguishing algorithms than to find effective lower bound methods for these models?

Another question is wether it is possible to find efficient learning algorithms for all cryptographically weak computational models? Theorem 4 gives some evidence that this might be not the case. An efficient learning algorithm for OBDDs would yield a good heuristic for the variable ordering problem for OBDDs. However, this contradicts the main (negative) result of [18] on approximations by OBDDs.

We further hope that the simulation of passive distinguishing algorithms by passive learning algorithms helps to clarify the existence of efficient passive learning algorithms for other computational models like depth 2 AND,OR-circuits, unweighted depth 2 threshold circuits, and others.

References

1. N. Alon, J. Spencer, P. Erdös. The probabilistic method. Wiley & Sons 1992.
2. F. Bergadano, A. Beimel, N. Bshouty, E. Kushilevitz, S. Varicchio. On the application of multiplicity automata in learning. FOCS '96, pp. 349-358.
3. M. Bellare, S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. Crypto '89, Springer LNCS, pp. 194–211, 1990.

4. M. Blaze, J. Feigenbaum, M. Naor. A Formal Treatment of Remotely Keyed Encryption. Eurocrypt '98, Springer LNCS, 1998.
5. M. Blum, W. Evans, P. Gemmell, S. Kannan, M. Naor. Checking the correctness of memories. Algorithmica, pp. 225–244, 1994.
6. A. Borodin, A. Razborov, R. Smolensky. On lower bounds for read k times branching programs. J. Computational Complexity 3, 1993, 1-13.
7. J. Bruck. Harmonic Analysis of polynomial threshold functions. SIAM Journal of Discrete Mathematics. 3:22, 1990, pp. 168-177.
8. O. Goldreich, S. Goldwasser, S. Micali. On the cryptographic applications of random functions, Crypto '84, Springer LNCS 196, 276–288.
9. O. Goldreich, S. Goldwasser, S. Micali. How to construct random functions. J. of the ACM, vol 33, pp. 792–807, 1986.
10. A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, G. Turan. Threshold circuits of bounded depth. FOCS'87, pp. 99-110.
11. J. Hastad. Almost optimal lower bounds for small depth circuits. STOC'86, pp. 6-20.
12. J. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution, FOCS'94, pp. 42-53.
13. S. Jukna. A note on read-k time branching programs. Theoretical Informatics and Applications 29(1), 1995, 75-83.
14. M. Kearns, L. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. J. of the ACM, vol. 41(1), 1994, pp. 67-95.
15. M. Krause, S. Lucks. Secure Pseudorandom Function Generators can have small threshold circuits of depth 3 but not of depth 2. Technical Report. Submitted.
16. M. Krause, P. Pudlak. On the computational power of depth-2 circuits with threshold and modulo gates. J. Theoretical Computer Science 174, 1997, pp. 137-156. Prel. version in STOC'94, pp. 49-59.
17. M. Krause, P. Pudlak. Computing Boolean functions by polynomials and threshold circuits. J. Comput. complex. 7 (1998), pp. 346-370. Prel. version in FOCS'95, pp. 682-691.
18. M. Krause, P. Savicky, I. Wegener. Approximation by OBDDs, and the variable ordering problem. Lect. Notes Comp. Science 1644, Proc. of ICALP'99, pp. 493-502.
19. N. Linial, Y. Mansour, N. Nisan. Constant depth circuits, Fourier transform, and learnability. J. of the ACM, vol. 40(3), 1993, pp. 607-620. Prel. version in FOCS'89, pp. 574-579.
20. M. Luby, C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. SIAM J. Computing, Vol. 17, No. 2, pp. 373–386, 1988.
21. S. Lucks. Faster Luby-Rackoff Ciphers. Fast Software Encryption 1996, Springer LNCS 1039, 189–203, 1996.
22. S. Lucks. On the Security of Remotely Keyed Encryption. Fast Software Encryption 1997, Springer LNCS 1267, 219–229, 1997.
23. M. Naor, O. Reingold. Syntesizers and their application to the parallel construction of pseudo-random functions. Proc. 36th IEEE Symp. on Foundations of Computer Science, pp. 170–181, 1995.
24. M. Naor, O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. Preliminary Version. Proc. 38th IEEE Symp. on Foundations of Computer Science, 1997.
25. M. Naor, O. Reingold. On the construction of pseudo-random permutations: Luby-Rackoff revisited. J. of Cryptology, Vol. 12, No 1, 29–66, 1999.
26. N. Nisan. Pseudorandom bits for constant depth circuits. J. Combinatorica 11 (1), 63-70, 1991.
27. E. Okolshnikova. On lower bounds for branching programs. Siberian Advances in Mathematics 3(1), 1993, 152-166.
28. A. Razborov, S. Rudich. Natural Proofs. J. of Computer and System Science, vol. 55(1), 1997, pp. 24-35. Prel. version STOC '94, pp. 204-213.
29. K. Siu, J. Bruck, T. Kailath, T. Hofmeister. Depth efficient neural networks for division and related problems. IEEE Trans. of Inform. Theory, vol. 39, 1993, pp. 946-956
30. R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. STOC'87, pp. 77-82.
31. I. Wegener. The complexity of Boolean functions. John Wiley & Sons, 1987.

5 Appendix

5.1 Usefull Probability Estimations

In the following, we use the following inequalities (see, e.g., [ASE92] Appendix A) known as *Höffding Bounds*. The first says that for all $a > 0$

$$Pr[\sum_{i=1}^{n} X_i \ > \ a] \ < \ e^{-a^2/2n}, \tag{12}$$

where X_i, $1 \leq i \leq n$, are mutually independent random variables defined by $Pr[X_i = 1] = Pr[X_i = -1] = \frac{1}{2}$.

We obtain as a consequence that for Y_i, $1 \leq i \leq n$, mutually independent random variables defined by $Pr[Y_i = 1] = Pr[Y_i = 0] = \frac{1}{2}$, and $\epsilon \in (0, 1/2)$ it holds that.

$$Pr[\sum_{i=1}^{n} Y_i \ < \ \epsilon n] \ < \ e^{-1/2(1/2-\epsilon)^2 n}. \tag{13}$$

In order to see this let $X_i = 1 - 2Y_i$. Observe that X_i, $1 \leq i \leq n$, are mutually independent as well as $Pr[X_i = 1] = Pr[X_i = -1] = \frac{1}{2}$ and

$$Pr[\sum_{i=1}^{n} Y_i \ < \ \epsilon n] \ = \ Pr[\sum_{i=1}^{n} 1/2(1 - X_i) \ < \ \epsilon n]$$

$$= Pr[\sum_{i=1}^{n} X_i \ > \ (1/2 - \epsilon)n] \ < \ e^{-1/2(1/2-\epsilon)^2 n}.$$

The second is that for all $p \in (0, 1)$

$$Pr[\sum_{i=1}^{n} X_i \ > \ a] \ < \ e^{-2a^2/n}, \tag{14}$$

where X_i, $1 \leq i \leq n$, are mutually independent random variables defined by $Pr[X_i = 1 - p] = p$ and $Pr[X_i = -p] = 1 - p$.

We obtain as a consequence that for all $p, q \in (0, 1)$, $p < q$, and all Z_i, $1 \leq i \leq n$, mutually independent random variables defined by $Pr[Z_i = 1] = p$, and $Pr[Z_i = 0] = 1 - p$ it holds

$$Pr[\sum_{i=1}^{n} Z_i \ > \ qn] \ < \ e^{-2(q-p)^2 n}. \tag{15}$$

In order to see this let $X_i = Z_i - p$. Observe that X_i, $1 \leq i \leq n$, are mutually independent as well as $Pr[X_i = 1 - p] = p$ and $Pr[X_i = -p] = 1 - p$. and that

$$Pr[\sum_{i=1}^{n} Z_i \ > \ qn] \ = \ Pr[\sum_{i=1}^{n} X_i + p \ > \ qn]$$

$$= Pr[\sum_{i=1}^{n} X_i \ > \ (q - p)n] \ < \ e^{-2(q-p)^2 n}.$$

For the third relation let X be a finite set and U be a subset of X. Further let U' denote a random subset of X of cardinality $|U'| < (|U|/2)$. Then

$$Pr[U' \subseteq U] \ \geq \ (|U|/(2|X|))^{|U'|}. \tag{16}$$

Observe that

$$Pr[U' \subseteq U] \;=\; \Pi_{i=0}^{|U'|-1} \frac{|U|-i}{|X|-i},$$

and that for all positive real numbers a, b, c with $a \geq b > 2c$ it holds

$$\frac{b-c}{a-c} \;>\; \frac{b}{2a}.$$

5.2 The Cryptographic Weakness of Constant Depth Circuits over $\{AND, OR, MOD_p\}$, p prime

We start with some preliminaries: Let K denote an arbitrary field and $B = \{a, b\}$ an arbitrary two-element subset of K. Observe that each function $h : B^n \longrightarrow K$ has a unique representation as an n–variate multilinear polynomial over K. Let us denote by $\deg_K(h)$ the degree of this representation, i.e., the maximal length of a monomial occuring with nonzero coefficient in this representation. Fix a Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$. The unique function $\hat{f} : B^n \longrightarrow B$, which is obtained from f by replacing all occurences of 0 by a and of 1 by b is said to be the (a,b)-variant of f. Now fix another two elements $a' \neq b'$ of K and denote by g the (a',b')-variant of f. Observe that for all $(y_1, \ldots, y_n) \in \{a', b'\}^n$ the relation

$$g(y_1, \ldots, y_n) = \frac{a'-b'}{a-b} \hat{f}(x_1, \ldots, x_n) + \frac{ab'-a'b}{a-b} \tag{17}$$

holds, where $x_i = \frac{a-b}{a'-b'} y_i + \frac{a'b-ab'}{a'-b'} \in \{a, b\}$ for all $i = 1, \ldots, n$. As this transformation is linear it follows that for all two elements $a \neq b \in K$ it holds that the K-degree of the (a,b)-variant of f is the same. We denote this value by $\deg_K(f)$.

For $K = \mathbf{F}_r$, $r = p^k$ prime power, we use the denotation $\deg_r(f)$. If the context is clear and some field K is fixed we identify Boolean functions with their $(0_K, 1_K)$-variants. We start now with the proof of Theorem 3.

Theorem 3 *For all primes p and all constant depth bounds d there is a distinguishing algorithm of quasipolynomially bounded ratio for depth d circuits over AND, OR, MOD_p-gates.*

Let us fix a prime p and a depth bound d. The proof of the Theorem is based on the following result of *Smolensky* [30]:

Lemma 1. *Let $f, g_1, \ldots, g_k \in B_n$ be given such that $f = \bigvee_{i=1}^k g_i$. Then for all $r < n$ there is a $\mathbf{F}_p$-polynomial $q = q(g_1, \ldots, g_m)$ of degree at most $(p-1)r$ such that $Pr_x[f(x) \neq q(g_1(x), \ldots, g_m(x))] \leq 2^{-r}$. The same statement holds if $f = \bigwedge_{i=1}^k g_i$.*

It is quite straightforward to derive

Corollary 1. *If $f \in B_n$ can be computed by a depth d AND,OR,MOD_p-circuit of size m then for each r, $p \leq r < n$, there is a function $\tilde{f} : \{0,1\}^n \longrightarrow \mathbf{F}_p$ such that $\deg_p(f) \leq ((p-1)r)^d$ and $Pr_x[f(x) \neq \tilde{f}(x)] \leq ((m^d - 1)/(m-1))2^{-r}$.*

Proof. The approximating function $\tilde{f}$ is obtained by replacing all AND- and OR-gates by $\mathbf{F}_p$-polynomials which approximate the gate with parameter r as in Lemma 1. Taking into account that the $\mathbf{F}_p$-degree of MOD_p is p-1 and that the indegree of each AND- and OR-gate is bounded by m it is easy to see that the degree of $\tilde{f}$ is bounded by $\delta_d(m)$ and the error probability is bounded by $E_d(m)$, where $\delta_d(m)$ and $E_d(m)$ are defined via the recursion $\delta_1(m) = (p-1)r$, $E_1(m) = 2^{-r}$, $\delta_d(m) =$

$(p-1)r\delta_{d-1}(m)$ and $E_d(m) = mE_{d-1}(m) + E_1(m)$. Evaluating this recursion gives the claim. □

Consequently, distinguishing $AC^0_d[p]$-functions from truly random functions can be reduced to testing that a given sample is induced by a function which can be well approximated by a low degree polynomial over $\mathbf{F}_p$. If $p \neq 2$ the idea for such a test can be derived from *Razborov's* and *Rudich's* Natural Proof against $AC^0[3]$ [28]: Let us fix some odd number n. In the following, we do all arithmetic operations with respect to the field $\mathbf{F}_p$. For all Boolean functions $f : \{0,1\}^n \longrightarrow \{0,1\}$ we denote by $\hat{f}$ the (1,-1)-variant of f. As the characteristic of $\mathbf{F}_p$ is odd we have $1 \neq -1$.

Let us denote by V the $\mathbf{F}_p$-vector space of all functions $h : \{1,-1\}^n \longrightarrow \mathbf{F}_p$. It holds $\dim_p(V) = N := 2^n$. We denote further by L the subspace of all $h \in V$ with $\deg_p(h) < n/2$. As n is odd we have $\dim_p(L) = N/2$. The complexity parameter $D_p(\hat{f})$ which is essential for us is defined as

$$D_p(f) = \dim_p(L + \hat{f}L),$$

where $\hat{f}L$ denotes the subspace of functions which can be written as $\hat{f} \cdot h,\ h \in L$, where $\cdot$ denotes argumentwise multiplication. (Observe that the set of functions $\hat{f} : \{1,-1\}^n \longrightarrow \{1,-1\}$ is closed under argumentwise multiplication.)

Observe the following properties of the parameter D:

(i) If f coincides with a function $g : \{0,1\}^n \longrightarrow \mathbf{F}_p$ of degree $P \leq \gamma\sqrt{n},\ \gamma \in (0,1)$, outside a fixed input set $E \subseteq \{0,1\}^n$ then $D_p(f) \leq (1/2 + \gamma)N + |E|$.
In order to see this observe at first that there is a function $\hat{g} : \{1,-1\}^n \longrightarrow \mathbf{F}_p$ with degree P which coincides with $\hat{f}$ outside a fixed input set $E' \subseteq \{1,-1\}^n$, where $|E| = |E'|$.
Consequently, outside of E' all functions in $L + \hat{f}L$ coincide with a function of degree smaller than $n/2 + P$. Hence,

$$D_p(f) \leq \sum_{k=0}^{n/2+P} \binom{n}{k} + |E| \ \leq\ N(1/2 + P/\sqrt{n}) + |E|.$$

(The last calculation is a consequence of Stirling's Formula which gives that $\binom{n}{\lfloor n/2 \rfloor} \leq 2^n/\sqrt{n}$.)

(ii) For the parity function $\pi = x_1 \oplus \ldots \oplus x_n$ it holds that $D_p(\pi) = N$. This follows from the well-known fact that $\hat{\pi} = y_1 y_2 \ldots y_n$. Consequently, (over $\{1,-1\}^n$) for each monomial m of degree larger than $n/2$ there is a monomial m' of degree smaller than $n/2$ such that $m = \hat{\pi}m'$.

(iii) For all Boolean functions f it holds that $D_p(f) + D_p(\pi \oplus f) \geq 3/2N$. In order to see this observe that

$$D_p(\pi \oplus f) - N/2 \ =\ \dim_p(L + \hat{\pi}\hat{f}L/L) \ =$$

$$\dim_p(\hat{f}L + \hat{\pi}L/\hat{f}L) \ \leq\ \dim_p(\hat{f}L + \hat{\pi}L + L/(\hat{f}L + L)) \ =$$

$$\dim_p(V/(L + \hat{f}L)) \ =\ N - D_p(f).$$

The statement follows directly. As a consequence of (3) we obtain:

(iv) The amount of Boolean functions $f : \{0,1\}^n \longrightarrow \{0,1\}$ with $D_p(f) \geq 3/4N$ is at least 50%.

(v) In order to evaluate $D_p(f)$, one has to compute the $\mathbf{F}_p$-rank of an $N \times N$-matrix, i.e., it can be done in time $N^{O(1)}$.

We describe now the distinguishing algorithm D for $\{AND, OR, MOD_p\}$-circuits, where $p \neq 2$. Fix a polynomial $m = m(n) \in n^{O(1)}$. Given input parameters n and $m = m(n)$, D at first computes the minimal number r and the minimal odd number $\tilde{n}$ such that

$$64m^{d-1} < 2^r \quad \text{and} \quad (p-1)^d r^d < (1/8)\sqrt{\tilde{n}}.$$

Observe that $r \in O(\log(n))$, $n \in O(\log^{2d}(n))$ and let $\tilde{N} = 2^{\tilde{n}}$.

Then D chooses randomly an 0,1-assignment c to the set of variables $\{x_{\tilde{n}+1}, \ldots, x_n\}$ and accepts if $D_p(f^c) < (3/4)\tilde{N}$.

Observe that by (v), this computation can be done using $\tilde{N} := 2^{\tilde{n}}$ oracle queries in time $\tilde{N}^{O(1)} = \exp(\log^{O(1)} n)$.

In the truly random case, by (iv), the probability that A outputs 1 is at most $1/2$.

Now consider the pseudorandom case and denote by f the secret function chosen by the oracle. By Corollary 1, there is a function $\tilde{f} : \{0,1\}^n \longrightarrow \mathbf{F}_p$ such that $\deg_p(\tilde{f}) \leq ((p-1)r)^d$ such that the probability that f differs from $\tilde{f}$ is bounded by $((m^d - 1)/(m-1))2^{-r}$.

Observe that for at least 75% of the 0,1-assignments c to the variables $\{x_{\tilde{n}+1}, \ldots, x_n\}$ it holds that the probability that f^c differs from $\tilde{f}^c$ is bounded by

$$4((m^d - 1)/(m-1))2^{-r} < 8m^{d-1}2^{-r}. \tag{18}$$

This implies that f^c differs from $\tilde{f}^c$ on a set E of less than $8m^{d-1}2^{\tilde{n}-r} \leq (1/8)2^{\tilde{n}}$ inputs, i.e., by (i) and as $\deg_p(\tilde{f}) < (1/8)\sqrt{\tilde{n}}$ we obtain

$$D_p(\tilde{f}) < (1/2 + 1/8)\tilde{N} + (1/8)\tilde{N} = (3/4)\tilde{N}.$$

Consequently, the probability that D accepts is at least $3/4$. It follows directly that D distinguishes $AC^0_d[p]$-functions from truly random functions with quasipolynomially bounded ratio.

Now let us consider the case $p = 2$. Clearly, if a given Boolean function f coincides outside a set E with a function g with $\deg_2(g) = d$, then for all fields K of characteristic 2 and all $a \neq b \in K$ it holds that the (a,b)-variant of f coincides with a function $\hat{g}$ of K-degree d outside a set $\hat{E}$ with $|E| = |\hat{E}|$.

The problem is that $1 = -1$ holds for fields of characteristic 2.

We choose the field $K = \mathbf{F}_4 = \{0, 1, z, z+1\}$. Observe the relation $z^2 = z + 1$ and the fact that $k^3 = 1$ for all $k \in \{1, z, z+1\}$. For a Boolean function f we denote by $\hat{f}$ the (1,z)-variant of f. As above, we fix an odd n, denote $N = 2^n$, denote by V the N-dimensional K-vector space of all functions from $\{1, z\}^n$ into K, and by L the $N/2$-dimensional subspace of all functions of K-degree smaller than $n/2$.

Further let for all functions $h : \{1, z\}^n \longrightarrow \{1, z, z+1\}$

$$D_2(h) = \dim_K(L + \hat{f}L).$$

For Boolean functions $f : \{0,1\}^n \longrightarrow \{0,1\}$ let $D_2(f) := D_2(\hat{f})$. Observe that property (i) of D_p holds in the same way for D_2. Consider further the function $\rho : \{1, z\}^n \longrightarrow \{1, z, z+1\}$ defined by

$$\rho(y_1, \ldots, y_n) = y_1 y_2 \ldots y_n.$$

Observe now the following properties of D_2:

(I) It holds that $\dim_K(L + \rho^2 L) = N$. In order to prove this it is sufficient to show that each monomial m of length larger than n/2 belongs to $\rho^2 L$. We can obviously find a monomial m' of length smaller n/2 such that $m^2 = \rho^2 m'$. On the other hand, using the fact that on $\{1, z\}$

$$y_i^2 = (z+1)y_i + z$$

it can be seen that $m^2 = (z+1)^t m + h$, where t denotes the length of m and h a function of degree smaller than t. Induction on the length of m yields the proof.

(II) The amount of functions $h : \{1, z\}^n \longrightarrow \{1, z, z+1\}$ for which $D_2(h) \geq (3/4)N$ is at least 50%. For proving this observe that for all $h : \{1, z\}^n \longrightarrow \{1, z, z+1\}$

$$D_2(\rho^2 h) - N/2 = dim_K(L + \rho^2 hL/L) = dim_K(h^2 L + \rho^2 L/h^2 L)$$

$$\geq dim_K(h^2 L + \rho^2 L + L/(h^2 L + L)) \;=\; N - D_2(h^2),$$

i.e., $D_2(\rho^2 h) + D_2(h^2) \geq (3/2)N$. As squaring and multiplication with ρ^2 are bijective mappings over the set of functions $h : \{1, z\}^n \longrightarrow \{1, z, z+1\}$ the claim follows.

In other words, if we take a truly random function $h : \{1, z\}^n \longrightarrow \{1, z, z+1\}$ then $D_2(h) \geq (3/4)N$ with significant probability. Unfortunately, we can not show this for the (1,z)-variants of random *Boolean* functions which would be necessary for our distinguishing algorithm. This is because we do not see any way for applying the above distinguishing algorithm straightforwardly in the case $p = 2$. The only way-out we see in the moment is to use the following (allmost complexity preserving) transformation of functions $f \in B_n$ into functions which map into $\{1, z, z+1\}$. We describe the transformation in a more general form which could also be usefull in other similar situations.

Generating random functions into $\{1, \ldots, k\}$, $k > 2$

We describe here an operator $T_{n,k,m}$, where n, k, m are positiv natural numbers fulfilling $m < n$ and $k < 2^n$, which assigns to each Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ a k-nary function $T_{n,k,m}(f) : \{0,1\}^m \longrightarrow \{0,1\}$ such that the following holds:

– If f has low complexity w.r.t. to a large number of relevant nonuniform complexity measures then $T_{n,k,m}(f)$ has, too.

– If f is a random Boolean function then, for s large enough, $T_{n,k,m}$ looks "sufficiently random". The construction is based on the following technical

Lemma 2. *For each n and $k \leq 2^n$, and each partition $\pi = (s_1, \ldots, s_k)$ of 2^n, i.e., the s_i are positive natural numbers fulfilling $s_1 + \ldots + s_k = 2^n$, there is a function $h_\pi : \{0,1\}^n \longrightarrow \{0,1\}$ with the following properties:*

(a) For all i, $1 \leq i \leq k$, it holds $|h_\pi^{-1}(i)| = s_i$.

(b) h has a Boolean decision tree with at most $(k-1)n + 1$ leafs.

Proof. A decision tree for a function $h : \{0,1\}^n \longrightarrow \{1, \ldots, k\}$ is a usual Boolean decision tree for which the leafs are labelled by $1, \ldots, k$. The computation mode is straightforward. We identify partitions $2^n = s_1 + \ldots + s_k$ by multisets $\pi = (s_1, \ldots, s_k)$. For each n and $k \leq 2^n$, and each partition $\pi = (s_1, \ldots, s_k)$ we define the corresponding function h_π by giving a decision tree D_π^n for h_π of the appropriate size (=number of leafs). We do this by induction.

Clearly, for $k = 1$ this tree consists of a single leaf labelled by "1". The size is 1 and matches the statement of the lemma.

If $n = 1$ and $k = 2$ (partition 2=1+1) this tree consists of one inner node labelled by x_1 and two leafs labelled "1" and "2".

If $k = 2$ and $n > 1$ and $\pi = (s, s')$, $s + s' = 2^n$, then the tree D^n_π can be (inductively) constructed as follows: Let $t = \max\{s, s'\}$ and observe that $t \geq 2^{n-1}$. D^n_π consists of a source labelled by x_n, one successor is a leaf, the other successor is $D^{n-1}_{(t-2^{n-1}, 2^n - t)}$. It follows easily by induction that the size of $D^n_{(s,s')}$ is at most $n + 1$.

Now let us fix arbitrary $n > 1$, $k > 2$, and a partition $\pi = (s_1, \ldots, s_k)$ of 2^n. Let us fix the uniquely defined l, $1 \leq l \leq k$, for which $s_1 + \ldots + s_{l-1} \leq 2^{n-1}$ and $s_1 + \ldots + s_l > 2^{n-1}$.

Let $s'_l = 2^{n-1} - (s_1 + \ldots + s_{l-1})$, $s"_l = s_l - s'_l$, $\pi' = (s_1, \ldots, s_{l-1}, s'_l)$, and $\pi" = (s"_l, s_{l+1} \ldots, s_k)$. Observe that both π' and $\pi"$ are partitions of 2^{n-1}.

D^n_π can be defined as a source labelled by x_n, the 0-successor of the source is $D^{n-1}_{\pi'}$, the 1-successor is a copy of $D^{n-1}_{\pi"}$ for which the leafs are labelled by $l, l+1, \ldots, k$ instead of $1, 2, \ldots, (k-l)+1$. By induction hypothesis the size of D^n_π is at most

$$(l-1)(n-1) + 1 + (k-l)(n-1) + 1 = (k-1)n + 3 - k \leq (k-1)n + 1.$$

□

We identify each function $h_\pi : \{0,1\}^n \longrightarrow \{1, \ldots, k\}$ with k Boolean functions $h^1_\pi, \ldots, h^k_\pi$ defined by

$$h^j_\pi(x) = 1 \iff h_\pi(x) = j.$$

We call $h^1, \ldots, h^k$ the *characteristic Boolean functions* of h. Observe

Corollary 2. *For all positive natural numbers n and k with $k \leq 2^n$, all partitions π of 2^n of length k, and all j, $1 \leq j \leq k$, it holds that the Boolean functions h^j_π, $1 \leq j \leq k$, can be written as the sum of S_j monomials with $S_1 + \ldots + S_k \leq (k-1)n + 1$.*

Proof. Take the monomials for h^j_π corresponding to the paths in D^n_π leading to leafs with label "j". □

Now, for all positive natural numbers n and k with $k \leq 2^n$ fix the *balanced* partition π of 2^n consisting of r times $\lceil 2^n/k \rceil$ and $k - r$ times $\lfloor 2^n/k \rfloor$, where $r = 2^n \mod k$. Denote by $h^1_{n,k}, \ldots, h^k_{n,k}$ the characteristic Boolean functions corresponding to π.

Fix a further positive natural number $m < n$, and let $S = 2^{n-m}$. We now define the operator $T_{n,k,m}$. For all Boolean functions $f : \{0,1\}^n \longrightarrow \{0,1\}$ let $T_{n,k,m}(f) : \{0,1\}^m \longrightarrow \{0,1\}$ be defined

$$T_{n,k,m}(f)(x_1, \ldots, x_m) = \sum_{j=1}^{k} j h^j_{S,k}(y_1, \ldots, y_S),$$

with $y_j = f(x_1, \ldots, x_m, b^{(j)})$, where $b^{(1)}, \ldots, b^{(S)}$ denote the S possible 0,1-assignments of $x_{n-m+1}, \ldots, x_n$ in the canonical order.

Now denote by $B_{m,k}$ the set of all functions $h : \{0,1\}^m \longrightarrow \{1, \ldots, k\}$. In the following lemma we estimate how much the distribution induced by $T_{n,k,m}(f)$ on $B_{m,k}$ deviates from the uniform distribution on $B_{m,k}$.

Lemma 3. *Fix an arbitrary subset E of $B_{m,k}$ and denote by p the probability of the event E w.r.t. the uniform distribution over $B_{n,k}$, and with $\tilde{p}$ the probability of the event E w.r.t. the distribution which is induced via $T_{n,k,m}(f)$ by uniformly distributed random Boolean functions $f : \{0,1\}^n \longrightarrow \{0,1\}$. Then*

$$|p - \tilde{p}| \leq pk2^{m-S}(1 + k2^{-S})^{2^m}.$$

Corollary 3. *If n, m are choosen in such a way that for $S = 2^{n-m}$ it holds that $2^S > ak2^m$ for some $a \geq 1$, then*

$$|p - \tilde{p}| \leq (p/a)e^{1/a}.$$

Proof. Let us denote $M = 2^m$. Observe that for all $x \in \{0,1\}^m$ and all j, $1 \leq j \leq k$, the probability that $h(x) = j$, where h denotes a random function distributed according to $T_{n,k,m}(f)$, is in $(1/k - 2^{-S}, 1/k + 2^{-S})$. Consequently,

$$|p - \tilde{p}| \leq pk^M(1/k + 2^{-S})^M - p = p\left(1 + k2^{-S}\right)^M - 1 = pMk2^{-S}(1+z)^{M-1}$$

for some $z \in (1, 1 + k2^{-S})$. Hence, $|p - \tilde{p}| \leq pMk2^{-S}(1 + k2^{-S})^M$.

The Corollary follows by applying the well known inequality $(1 + (x/N))^N \leq e^x$ for all $x > 0$, which yields $(p/a)(1 + (1/aM))^M \leq (p/a)e^{1/a}$. □

The distinguishing algorithm for $p = 2$

For all $d \geq 2$, a distinguishing algorithm D for depth d circuits over $\{AND, OR, MOD_2\}$ can be designed as follows. Given input parameters n and $m(n) \in n^{O(1)}$, D fixes parameters r and $\tilde{n}$ as the minimal natural numbers fulfilling

$$192m^{2(d+1)} < 2^r \quad \text{and} \quad r^{d+2} < (1/8)\sqrt{\tilde{n}}.$$

Observe that $192 = 24 \cdot 8$, $r \in O(\log(n))$, and $\tilde{n} \in O(\log^{2(d+2)} n)$, and let $\tilde{N} = 2^{\tilde{n}}$.

At next, D computes a parameter $s \in O(\log\log(n))$ such that for $S = 2^s$ it holds that

$$2^S \geq 12\tilde{N} \quad \text{and} \quad S(m+2) + 1 \leq m^2.$$

This is always possible for n, m large enough.

Then D chooses randomly a 0,1-assignment c to the variables $x_{\tilde{n}+1}, \ldots, x_{n-s}$.

D accepts iff $D_2(h^c) < (3/4)\tilde{N}$, where h denotes the (1,z,z+1)-variant of $T_{n,3,n-s}(f)$. Observe that the evaluation of one value of h needs S oracle queries and evaluations of $h^1_{S,3}, h^2_{S,3}$ and $h^3_{S,3}$, i.e. the running time of the algorithm is bounded by $(\tilde{N}S)^{O(1)}$ which is quasipolynomially bounded in n.

In the truly random case, h^c is a random function from $\{1, z\}^{\tilde{n}}$ into $\{1, z, z+1\}$ which is distributed according to that distribution on $B_{\tilde{n},3}$ which is induced by the uniform distribution on $B_{\tilde{n}+s,2}$ via $T_{\tilde{n}+s,3,\tilde{n}}$.

Remember that by (II) the probability that $D_2(h) \geq (3/4)\tilde{N}$ is at least $1/2$ w.r.t. the uniform distribution on $B_{\tilde{n},3}$. Consequently, by Corollary 3, and as $2^S > 4 \cdot 3 \cdot \tilde{N}$ we obtain that the probability that A accepts is at most

$$1/2 + (1/4)e^{1/4} < 11/16.$$

Now consider the pseudorandom case and denote by f the secret function fixed by the oracle. Observe that for all $u = 1, 2, 3$ the functions $h^u : \{0,1\}^n \longrightarrow \{0,1\}$ defined by

$$h^u(x) = h^u_{S,3}(y_1, \ldots, y_S)$$

with $y_j = f(x, b^j)$, where $b^{(1)}, \ldots, b^{(S)}$ denote the S possible assignments of $x_{n-s+1}, \ldots, x_n$ in the canonical order, can be computed by AND,OR,MOD_2-circuits of depth $d+2$ and size $Sm + 2S + 1 = S(m+2) + 1 \leq m^2$. (see Corollary 2.)

Consequently, for the given r, there is a degree r^{d+2} polynomial g for which the probability that h differs from g is at most

$$3(m^{2(d+2)} - 1)/(m^2 - 1)2^{-r} \leq 6m^{2(d+1)}2^{-r},$$

for m large enough.

Hence, for an amount of at least 75% of all 0,1-assignments c to the variables $x_{\tilde{n}+1}, \ldots, x_{n-s}$ it holds that the error probability of h^c w.r.t. g^c is at least $24m^{2(d+1)}2^{-r}$, i.e. h^c and g^c differ with respect to at most

$$24m^{2(d+1)}2^{\tilde{n}-r} < (1/8)2^{\tilde{n}}$$

inputs. Suppose that we have choosen such a c. Then, as the degree of g^c is smaller than $(1/8)\sqrt{n}$, we get by (i) that $D_2(h^c) < (3/4)\tilde{N}$, i.e., D accepts with probability $3/4 > 11/16$. We obtain quasipolynomial distinguishing ratio.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'