The Complexity of Computing Hard Core Predicates

Mikael Goldmann and Mats Näslund

Royal Institute of Technology,
Dept. of Numerical Analysis and Computing Science,
S–100 44 Stockholm, Sweden
e-mail: {migo,matsn}@nada.kth.se

Abstract. We prove that a general family of hard core predicates requires circuits of depth $(1-o(1))\frac{\log n}{\log\log n}$ or super-polynomial size to be realized. This lower bound is essentially tight. For constant depth circuits, an exponential lower bound on the size is obtained. Assuming the existence of one-way functions, we explicitly construct a one-way function $f(x)$ such that for any circuit c from a family of circuits as above, $c(x)$ is almost always predictable from $f(x)$.

Keywords: pseudo-randomness, small-depth circuit, one-way function

1 Introduction

One of the most useful cryptographic primitives is the *pseudo-random generator*. This is a function that deterministically expands a short random seed to a longer, random "looking" string. Blum and Micali [3], and Yao [13], showed a simple way to construct such a generator, using the following method. Assume that f is a permutation and that we have a set of boolean functions C. Choose c at random in C, a random x, and output $f(x), c(x)$, and (the description of) c. Clearly it suffices that $c(x)$ is unpredictable given $f(x)$ and c. Since computing $c(x)$ is no harder than inverting $f(x)$, this in turn implies that $f(x)$ must be a so called *one-way function*. If we indeed have this situation, C is said to be a *family of hard core predicates for* f.

For efficiency reasons, we would like the functions in C to be very simple to compute, so that an essential question is: *How simple can they be?* It seems natural to consider functions such as $c(x) =$ "some bit of x". Although there are examples of certain conjectured one-way functions f for which such simple c are hard core predicates, e.g. see [1,3,7], these constructions are too simple to work in a general setting without any assumptions on f. The reason for this is that a one-way function may depend on a relatively small number of its input bits and output the rest of them unchanged. The knowledge of these latter bits may be enough to deduce the value of such a "hard core". Since the case when no assumptions on f (except that it's one-way) are needed clearly is the most attractive, this is the case that we study in this paper.

Yao observed in [13] that although a one-way function may reveal many bits of the input, it *must* hide at least *some* bits. We may in general not know which these bits are, so a good candidate for a hard core should depend on all (or almost all) of its input bits. The first construction of a hard core predicate for any one-way function is due to Goldreich and Levin, [5], and uses the inner product modulo 2 of x and a random

binary string r. Two more constructions, affine functions in GF[2^n] and $\mathbb{Z}_p$, are due to Näslund, [11,12]. These are functions depending on all bits in x. Though simple, it is not obvious that we cannot use even simpler functions as long as they depend on all bits.

In this paper we shall prove that the existing constructions are basically the simplest possible. To measure the simplicity/complexity we will use the computational model of circuits, i.e. how large/deep a circuit of boolean AND/OR/NOT-gates that is needed to compute the hard core predicate. All the three general constructions mentioned above can be computed by circuits of logarithmic depth, polynomial size, and constant fan-in, that is, NC^1-circuits. So, the next natural step-down in complexity would be to consider AC^0-circuits; circuits of constant depth, polynomial size, and unbounded fan-in. There are numerous results indicating that this class of circuits is not very powerful. For instance, it is known from [10] by Mansour, Nisan, and Tiwari that universal hash functions (in general good candidates for hard core predicates) can not be computed by such simple circuits. A similar negative result on the existence of so called *pseudo-random functions* was given by Linial, Mansour, and Nisan in [9].

The widely used technique for showing computational limitations of small-depth circuits is the application of the Håstad switching lemma, see [6]. This proves to be useful here too since the lemma basically says that knowing some of the inputs to a small-depth circuit is very likely to be enough to deduce the output value of the circuit. This method is probabilistic and will give non-uniform results. However, we show that it is possible to obtain uniform results as well.

The paper is organized as follows. First we give some basic definitions and a proof outline in Section 2. Section 3 describes some tools from the theory of circuit complexity. Although perhaps known as a "folklore theorem", we prove in Section 4 that no family of constant depth, constant fan-in circuits can be a family of hard core predicates. We choose to do this since it illustrates the basic techniques. In Section 5 we then prove that not even polynomial size, constant depth, and unbounded fan-in circuits can be hard core predicates.

2 Preliminaries

If x is a binary string, $|x|$ is the length of x (if S is a set $|S|$ is the cardinality). By $y \in_{\mathcal{D}} S$, we mean a y chosen from S according to the distribution $\mathcal{D}$. Here, $\mathcal{U}$ will denote the uniform distribution on S. For two binary strings x, y, $x \circ y$ denotes the concatenation of the strings. If $x = x_1 x_2 \cdots x_n \in \{0,1\}^n$ and $I \subseteq \{1,2,\ldots,n\}$ let $x_I = x_{i_1} x_{i_2} \cdots x_{i_{|I|}}$, $i_j \in I$, $i_1 < i_2 < \cdots < i_{|I|}$. $x_{\overline{I}}$ is defined analogously by taking the complement of I.

Let $\mathcal{B} = \{b : \{0,1\}^* \mapsto \{0,1\}\}$, $\mathcal{B}_n = \{b : \{0,1\}^n \mapsto \{0,1\}\}$. A *circuit* is a directed acyclic graph having *gates* as vertices. A gate can be of type OR, AND, or NOT and computes the corresponding boolean function of its incoming edges, the incoming edges being outputs of other gates or one of the inputs, x_i, $i = 1,2,\ldots,n$ or the negation of an input $\overline{x_i}$. The *fan-in* of a gate is the number of incoming edges. There is a unique gate the output of which is the output of the whole circuit. The *size* of the circuit is the number of gates. By modifying the circuit (and making it slightly bigger), we can assume that NOT-gates only appear at the inputs and that the circuit is leveled with the gates at level i taking their inputs from gates at level $i-1$ and that all gates at a given

level are of the same type (AND/OR), types alternating from level to level. Hence all inputs x_i are at level 0. The *depth* of the circuit is the number of levels.

A circuit c computing $b \in \mathcal{B}_n$ is said to *depend* on m bits if there is a fixed $I \subseteq \{1,2,\ldots,n\}$, $|I| = m$, so that for all x, $|x| = n$, $c(x)$ is uniquely determined by x_I. Notice that a circuit c can be evaluated on input x by an algorithm whose running time is polynomial in the size of c by simply traversing c's gates.

By NC^0 we mean the set of $b \in \mathcal{B}$ so that for some $c,d,k \in O(1)$, for all n and $x \in \{0,1\}^n$, $b(x)$ is computable by a circuit with size, depth, and fan-in bounded by n^c, d, and k respectively. AC^0 is defined similarly but without the restriction on the fan-in.

An *ensemble of circuits* is a sequence, $\mathfrak{C} = \{\mathfrak{C}_n\}_{n\geq 1}$, where each $\mathfrak{C}_n$ is a probability distribution on circuits computing functions in $\mathcal{B}_n$. If there is a probabilistic polynomial time Turing machine (pptm) that on input 1^n outputs a c according to $\mathfrak{C}_n$, we shall say that we have a *polynomial ensemble of circuits*. An ensemble of *functions*, $\mathfrak{F} = \{\mathfrak{F}_n\}_{n\geq 1}$, is defined analogously, but with each $\mathfrak{F}_n$ being a distribution on functions mapping $\{0,1\}^n \mapsto \{0,1\}^*$. An ensemble of circuits, $\{\mathfrak{C}_n\}_{n\geq 1}$, is said to be $(s(n),d(n),k(n))$-*bounded* if for all n, $\mathfrak{C}_n$ has support only on circuits c with $\mathrm{size}(c) \leq s(n)$, $\mathrm{depth}(c) \leq d(n)$, and fan-in bounded by $k(n)$. If one of the three parameters, e.g. the fan-in, is unbounded we shall omit it and write $(s(n),d(n),\cdot)$-bounded etc.

A function $\xi(n)$ is *negligible* if for every constant $a > 0$ and for every sufficiently large n, $\xi(n) < n^{-a}$. A *one-way function* is a deterministic poly-time computable function f such that for every pptm, M, the probability that $M(f(x)) \in f^{-1}(x)$ is negligible. The probability is taken over $x \in_U \{0,1\}^n$ and M's random choices. Referring to a simple padding argument, we shall assume that all one-way functions are *length-preserving*, $|f(x)| = |x|$.

Let $\mathfrak{C} = \{\mathfrak{C}_n\}_{n\geq 1}$ be a polynomial ensemble of circuits and let f be a one-way function. An $\varepsilon(n)$-*adversary* for $\mathfrak{C}$ is a pptm A such that $\Pr[A(f(x),c) = c(x)] \geq 1/2 + \varepsilon(n)$, the probability taken over $x \in_U \{0,1\}^n$, c chosen according to $\mathfrak{C}_n$, and A's random choices. We call $\mathfrak{C}$ a *hard core predicate for* f if no $\varepsilon(n)$-adversary exists for non-negligible $\varepsilon(n)$. Normally, $\mathfrak{C}_n$ is the uniform distribution on some set of circuits, but we shall here allow other distributions. If $\mathfrak{C}$ is a hard core predicate for *any* one-way function, we simply call $\mathfrak{C}$ a (general) hard core predicate.

2.1 General Proof Outline

Assume that we have a one-way function[1] f of the form $f(x) = g(x_I) \circ x_{\bar{I}}$ where $I \subset \{1,2,\ldots,n\}$ and where g is another one-way function. In other words, f is defined by applying g to a part of x and output the rest of x unchanged. (It may be the case that g itself outputs some bits unchanged, but we shall see that this can only help us.)

Suppose now that we are the adversary A. Given $f(x)$ and a circuit c, we want to compute $c(x)$. How would we go about this? Since we know the bits in $x_{\bar{I}}$, a natural approach would be to try to make a partial evaluation of c using only these bits. If, for instance, we know that one of the inputs to an AND-gate in c is a zero, we can simplify the circuit by deleting this gate and replacing it by the constant zero and so on. If we are able to make enough simplifications from the information in $x_{\bar{I}}$, the circuit

[1] Without this assumption, the notion of hard core predicate is, of course, meaningless.

will be a constant, determined by $x_{\overline{I}}$, and independent of x_I. It is not clear how to do this simplification/evaluation in polynomial time, nor is it clear how to tell *if* the circuit indeed is independent of x_I. However, if it *almost always* is the case that c "collapses" in this way, we can always act as if the value x_I *is* unimportant, substitute an arbitrary value z for x_I, and then evaluate the circuit using $z, x_{\overline{I}}$. In the case where c doesn't depend on x_I, this strategy will give a correct value for $c(x)$. Also, this is only a simple evaluation, and can be done in time polynomial in size(c). (This is polynomial in A's input, $(f(x), c)$, but it is not polynomial in $n = |x|$ unless size(c) is. We point out this difference since we shall include larger circuits later in our study.)

The hardness of inverting f is now reduced to the hardness of inverting g and the length of the argument of g (g's security parameter) is decreased. Hence, we lower the security of f correspondingly. As long as this length reduction is within a polynomial factor though, this is, at least from a theoretical standpoint, of no importance.

If we for the moment accept this idea, there remains one big concern. How should we choose the set I that g is applied to? Surely, we cannot hope that a fixed I will work as it seems likely that we could find a circuit that only uses the bits in x_I that are hidden to us and thus the circuit output would be unpredictable. We should therefore use a random I each time. This randomness must be taken somewhere and there are two ways of doing this; either we "hardwire" the randomness into f and we have a non-uniform construction or, to get uniformity, we "borrow" randomness from x since x is assumed to be random. This second approach can be realized as follows. Writing x as $x = x' \circ x''$ (we shall determine the lengths of x', x'' later), we now interpret (in some way) x' as an encoding of a subset I of the bits in x''. We then compute f as $f(x) = f(x' \circ x'') = g(x''_I) \circ x''_{\overline{I}} \circ x'$. Since all information on I is available in x' which is supplied to the circuit c, we must choose this encoding carefully to avoid c "figuring out" which bits it should use, namely those in I, hidden by g.

These are the main ingredients and the bulk of the paper basically concerns three things. 1. Find an encoding that circumvents the problem just mentioned. 2. Quantify how many bits in x we (the adversary) will need to know (the size of I). 3. Analyze how likely it is that the circuit indeed "collapses" given the bits in I.

3 Random Restrictions

The notion of "knowing" bits in x is formalized by *random restrictions*, introduced in [4].

Definition 1. A *restriction* is a partial assignment to the inputs of a circuit c, assigned inputs are given values in $\{0,1\}$ and the rest are assigned the symbol $*$ to denote that they remain variables. By $R^{(n,p)}$ we mean the set of restrictions assigning $*$ to some pn-subset of $x_1, \ldots, x_n$ and values in $\{0,1\}$ to the other $n(1-p)$ x_is. A *random restriction* in $R^{(n,p)}$ then, assigns $*$ to a random pn-subset of the inputs and values in $\{0,1\}$ independently, and with equal probability, to the other $n(1-p)$ inputs.

For a circuit c and a restriction ρ, $c\restriction_\rho$ denotes the circuit computing the restricted function of the remaining $*$ after ρ is applied. For $\rho \in R^{(n,p)}$, let $*(\rho)$ be the np-subset (of indices) that is assigned $*$ by ρ. Since $*(\rho)$ and the assignment to the other bits in x uniquely determines ρ, we can by setting $I = *(\rho)$ specify ρ by the notation $[I; x_{\overline{I}}]$.

3.1 Encoding Restrictions as Integers

Consider restrictions in $R^{(n,p)}$. We will encode these as integers. For a $\rho = [*(\rho); z]$, it is trivial to encode z as a binary string, so the only possible problem is how to encode the set $*(\rho)$. We show how to do this. Note that there are $\binom{n}{np}$ possibilities for $*(\rho)$.

Lemma 2. *Let $u(n) = \binom{n}{np}$. For every $v \geq 0$ there is a polynomial time computable surjective function*

$$Q_v : \{0,1\}^{\lceil \log u(n) \rceil + v} \mapsto J = \{I \mid I \subset \{1,2,\ldots,n\}, |I| = np\}$$

such that for $h \in_U \{0,1\}^{\lceil \log u(n) \rceil + v}$, for every $I \in J$:

$$\left(1 - \frac{1}{2^v}\right) \frac{1}{u(n)} \leq \Pr_h[Q_v(h) = I] \leq \left(1 + \frac{1}{2^v}\right) \frac{1}{u(n)}.$$

The proof is straightforward and therefore omitted. The main idea is to interpret the integer $q = h \bmod u(n)$ as "the lexicographically qth np-subset".

As noted, there could still be a problem with how we perform the encoding, since the circuits could gain information on the restriction. We will take care of this when computing the value h that Q_v is applied to and we return to this later. In the remainder of this paper we abuse notation slightly and refer to the value h as coding a restriction rather than $Q_v(h)$. As long as h is uniformly distributed in $\{0,1\}^{\lceil \log u(n) \rceil + v}$, this is by the Lemma above basically the same thing.

4 There are no Hard Core Predicates in NC^0

The situation for NC^0 circuits is quite simple. Since such circuits have fan-in bounded by $k \in O(1)$, they can depend on only $k^{\text{depth}(c)} \in O(1)$ of the inputs x_i. The proofs in this case are simple combinatorial arguments.

Proposition 3. *Let c be a circuit of depth d, fan-in k, $d,k \in O(1)$, computing some function $b \in \mathcal{B}_n$ and let $\rho \in_U R^{(n,p)}$, $p \leq k^{-d}$. Then*

$$\Pr_{*(\rho)}[c \restriction_\rho \text{ is a constant function}] \geq 1 - k^d p.$$

Furthermore, if this is the case then given ρ we can deterministically in time polynomial in $\text{size}(c)$ decide what this constant is.

Proof. Since c can depend on at most k^d of its inputs, the probability that c depends on an input x_i such that $i \in *(\rho)$ is at most $k^d \frac{np}{n}$.

If c in this way collapses under ρ, we can simply evaluate the circuit by assigning an arbitrary (even fixed) value to the x_is for $i \in *(\rho)$ since c does not depend on these. □

4.1 Non-uniform Case

Theorem 4. *For any $\delta \in (0,1)$ there is a polynomial ensemble of one-way functions $\mathfrak{F}(\delta) = \{\mathfrak{F}_n\}_{n\geq 1}$ and a deterministic polynomial time algorithm A such that for all constants d,k, for any $(\cdot,d,k)$-bounded ensemble of circuits $\{\mathfrak{C}_n\}_{n\geq 1}$, for every c supported by $\mathfrak{C}_n$, and for all $x \in \{0,1\}^n$,*

$$\Pr_{f\in\mathfrak{F}_n}[A(f(x),c) = c(x)] \geq 1 - O\left(n^{-(1-\delta)}\right).$$

Proof. Let g be a one-way function and let $\mathfrak{F}_n$ be the uniform distribution on the following set of one-way functions:

$$\{f_I(x) = g(x_I) \circ x_{\overline{I}} \circ I \mid I \subseteq \{1,2,\ldots,n\}, |I| = n^\delta\}.$$

For any c chosen according to $\mathfrak{C}_n$ and any x, having a value of the form $f_I(x)$ for random I, corresponds in a natural way to having a restriction $\rho = [I; x_{\overline{I}}]$ in $R^{(n,n^{-(1-\delta)})}$ on the input of c. The result now follows directly from Proposition 3. □

By standard probabilistic arguments we get the Corollary below.

Corollary 5. *For all constants d,k,δ, $\delta \in (0,1)$, for any $(\cdot,d,k)$-bounded ensemble of circuits $\{\mathfrak{C}_n\}_{n\geq 1}$, there is a non-uniform one-way function f and a deterministic polynomial time algorithm A so that for all $x \in \{0,1\}^n$,*

$$\Pr_{c\in\mathfrak{C}_n}[A(f(x),c) = c(x)] \geq 1 - O\left(n^{-(1-\delta)}\right).$$

4.2 Uniform Case

In the construction of the one-way functions $\{f_I\}$ in the previous subsection we used extra randomness when selecting I. To get a uniform result we must somehow eliminate this. As mentioned in the outline, we cannot use a fixed subset I.

The idea is that since x, the argument of $f(x)$, is supposed to be a random string, we will "borrow" a few random bits from x itself to "point out" which subset I to use when computing $g(x_I)$ (and thus also which subset to output unaffected). We will therefore need a mapping from, say the first $l(n)$ bits of x to the set of all n^δ-subsets of $\{l(n)+1, l(n)+2, \ldots, n\}$. If we split x as $x = x' \circ x''$, we would like to interpret x' as an encoding of a random subset of the bits in x''. But we know how to do this from Lemma 2. We need roughly $l(n) = \lceil \log \binom{n}{n^\delta} \rceil \leq n^\delta \log n$ bits to encode all n^δ-subsets. To be more precise we should have $l(n) = |x'| = \lceil \log \binom{|x''|}{n^\delta} \rceil$, and since $|x''| = |x| - |x'| = n - l(n)$, $l(n)$ should in fact satisfy the equation $l(n) = \lceil \log \binom{n-l(n)}{n^\delta} \rceil$. Instead of solving this equation we can cheat slightly and simply choose $l(n)$ "large enough". This also means that we will use a ν-value greater than zero when referring to Lemma 2 and this will give a more uniform distribution on the restrictions.

We must also be slightly careful, since the subset we are to compute g on is now supplied to the circuit c via x', and c might use that information to correlate itself to that subset and maybe even become dependent on some of the bits hidden by g. To avoid this we will use a slightly more elaborate encoding of the n^δ-subsets as described in the proof below.

Theorem 6. *For any $\delta \in (0,1)$ there is a one-way function f and a deterministic polynomial time algorithm A such that for all constants d,k, for any $(\cdot,d,k)$-bounded ensemble of circuits $\{\mathfrak{C}_n\}_{n\geq 1}$, for all sufficiently large n, for every c supported by $\mathfrak{C}_n$,*

$$\Pr_{x\in_U\{0,1\}^n}[A(f(x),c) = c(x)] \geq 1 - O\left(n^{-(1-\delta)}\right).$$

Consequently, there are no hard core predicates in NC^0.

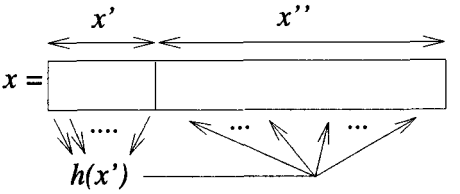
Proof: Let $0 < \tau < 1-\delta$, let g be a one-way function and write $x = x' \circ x''$ where x' is the first $l(n) = n^\tau n^\delta \log n$ bits of x and x'' the last $n - l(n)$ bits. Define

$$h^{(i)}(x') = \sum_{j=(i-1)n^\tau+1}^{in^\tau} x'_j \bmod 2$$

(the exclusive-or over the ith n^τ-bit segment of x') and set

$$h(x') = h^{(1)}(x') \circ h^{(2)}(x') \circ \cdots \circ h^{(n^\delta \log n)}(x').$$

Observe that $h(x')$ is an $n^\delta \log n$-bit string where, for sufficiently large n, each individual bit is totally random to the circuits we are considering. This holds since each bit is an exclusive-or of n^τ bits, and the circuits we study can depend on no more than $k^d \in O(1)$ bits. Hence, if x and therefore x' is random, any such circuit is completely uncorrelated with $h(x')$. Now use $h = h(x')$ as described in Lemma 2 to encode an n^δ-subset of x''. (To simplify notation, we abuse it slightly by writing $h(x')$ rather than $Q_v(h(x'))$.) The encoding can be viewed as in the figure above. Now define the one-way function



$$f(x) = f(x' \circ x'') = g(x''_{h(x')}) \circ x''_{\overline{h(x')}} \circ x'.$$

For any c from $\mathfrak{C}_n$, given $f(x)$, $c(x)$ is now equivalent to a circuit $c'(x'') = c(x' \circ x'')$ upon which we have an (almost) random restriction $\rho = [h(x'); x''_{\overline{h(x')}}]$ in $R^{(n-l(n),p)}$, $p = n^\delta/(n-l(n)) = (n^{1-\delta} - n^\tau \log n)^{-1}$. The distribution on the restrictions is not *exactly* the uniform distribution, but by Lemma 2, no "bad" restriction (one that does not force c' to a constant) is chosen by more than twice the probability it is chosen by the uniform distribution. (And this can also be made arbitrarily close to uniform distribution by Lemma 2.) Hence, by Proposition 3 the output of this circuit is completely determined by ρ with probability at least $1 - 2k^d(n^{1-\delta} - n^\tau \log n)^{-1}$, and if so, we can determine the output in polynomial time. □

5 There are no Hard Core Predicates in AC^0

For AC^0 we cannot use the same simple counting arguments since these circuits may very well depend on all n bits in x. We therefore need some more powerful tools from the theory of circuit complexity.

5.1 The Switching Lemma

The Håstad switching lemma, see [6], quantifies how much and how likely it is that a circuit is simplified under a restriction. Similar results are known from [4] and [14]. We have the following version of the switching lemma, derived from [2].

Lemma 7 (The Switching Lemma). *Let G be an AND-gate whose inputs are OR-gates all of fan-in at most r and let $\rho \in_U R^{(n,p)}$, where $p \leq 1/7$. Then the probability that $G\restriction_\rho$ can be written as an OR of ANDs, each AND having fan-in strictly less than t is at least $1-(7pr)^t$.*

A dual lemma holds by replacing AND by OR and vice versa. We can prove the following powerful result.

Lemma 8. *For any $\delta \in (0,1)$, for all sufficiently large n the following holds. If c is a circuit of depth $d(n) \leq \frac{\log n}{\log\log n}$ and size $s(n) \leq n^{-(1-\delta)/d(n)} 2^{\frac{1}{14}n^{(1-\delta)/d(n)}}$, computing some function in $\mathcal{B}_n$, then with $p = n^{-(1-\delta)}$,*

$$\Pr_{\rho \in_U R^{(n,p)}}[c\restriction_\rho \text{ is a constant function}] \geq 1 - 8n^{-(1-\delta)/d(n)}.$$

If this is the case, we can given ρ find this constant in time polynomial in size(c).

Proof. Let c be a circuit as mentioned in the lemma. We will choose $\rho \in R^{(n,p)}$ in $d(n)$ steps, each step consisting of picking a random restriction ρ_i on the remaining unset inputs. Our ρ will be the composition of all these restrictions. Let $t = \frac{1}{14}n^{(1-\delta)/d(n)}$.

The first step is needed to get fan-in at most t at level 1. Assume it consists of AND-gates. For the purposes of the switching lemma, we view these gates as ANDs of ORs where each OR has fan-in 1 (a variable x_i or its negation). We pick a random restriction from $R^{(n,p_0)}$, where $p_0 = 1/14$. By the Switching Lemma we know that each AND of fan-in-1 ORs can be replaced by an OR of ANDs of fan-in t with probability at least $1-(7p_0)^t = 1-2^{-t}$. The OR can now be "collapsed" into the level above, as that level contains OR-gates.

In steps 2 through $d(n)-1$ we reduce the circuit by applying the switching lemma to the bottom two levels of the circuit, switch ANDs of ORs to ORs of ANDs (or vice versa) and collapsing adjacent levels of OR-gates (or AND-gates) maintaining the bound t on the bottom fan-in. This is done as follows.

Let $p_1 = (14t)^{-1}$, and let $n_i = p_0(p_1)^{i-1}n$. At step i we pick a random restriction $\rho_i \in_U R^{(n_{i-1},p_1)}$, where the domain of ρ_i is the input variables that have not been set by ρ_1 through ρ_{i-1}. Notice that after step i there are n_i variables that remain unset.

For every AND of ORs (or OR of ANDs) that we consider, the probability that the restriction doesn't allow us to switch is at most $(7p_1t)^t = 2^{-t}$. Over steps 1 through $d(n)-1$ we invoke the Switching Lemma once for each gate in the circuit (except the top gate), and each time the probability of failure is at most 2^{-t}. So with probability at least $1-s(n)2^{-t}$ the entire circuit has been collapsed to a single AND of ORs of fan-in $\leq t$ (or to an OR of ANDs of fan-in $\leq t$), and there are still $n_{d(n)-1} = p_0(p_1)^{d(n)-2}n$ variables unset.

Finally, in step $d(n)$ we do as follows. Assume we have been successful in steps 1 through $d(n)-1$, and that we are left with an AND of ORs, where each OR has fan-in at most t. Let $p_2 = (14t^2)^{-1}$ and pick $\rho_d \in_{\mathcal{U}} R^{(n_{d(n)-1}, p_2)}$. By the Switching Lemma, the probability that the AND of ORs can be written as an OR of ANDs, each AND of fan-in strictly less than 1 (and must thus be a constant) is at least $1-(7p_2t) = 1-(2t)^{-1}$.

The probability that *all* the ρ_i are successful is at least $1 - s(n)2^{-t} - (2t)^{-1} \geq 1 - 8n^{-(1-\delta)/d(n)}$. Notice also that $p = p_{d(n)} = p_0(p_1)^{d(n)-2}p_2 = n^{-(1-\delta)}$.

Finally, to find the constant we substitute arbitrary values for x_i, $i \in *(\rho)$, and evaluate the circuit like before. □

Notice that for the circuit depths covered by the Lemma, the failure probability, $8n^{-(1-\delta)/d(n)} \in o(1)$. Hence, almost surely, for such circuits, $c\lceil_\rho$ *will* be a constant.

5.2 Non-uniform Case

Theorem 9. *For any $\delta \in (0,1)$ there is a polynomial ensemble of one-way functions $\{\mathfrak{F}_n\}_{n\geq 1}$, and a deterministic polynomial time algorithm A for which the following hold. For any $(s(n),d(n),\cdot)$-bounded ensemble of circuits $\{\mathfrak{C}_n\}_{n\geq 1}$, where $d(n) \leq \frac{\log n}{\log\log n}$ and $s(n) \leq n^{-(1-\delta)/d(n)}2^{\frac{1}{14}n^{(1-\delta)/d(n)}}$, for all sufficiently large n, for every c supported by $\mathfrak{C}_n$,*

$$\Pr[A(f(x),c) = c(x)] = 1 - O\left(n^{-(1-\delta)/d(n)}\right),$$

the probability taken over $x \in_{\mathcal{U}} \{0,1\}^n$, and f chosen according to $\mathfrak{F}_n$.

Proof. Let $p = n^{-(1-\delta)}$, assume that g is a one-way function and let $\mathfrak{F}_n$ be the uniform distribution on the one-way functions

$$\{f_I(x) = g(x_I) \circ x_{\overline{I}} \circ I \mid I \subset \{1,2,\ldots,n\}, |I| = np\}.$$

Note that random for random x and I, $f_I(x)$ corresponds to the random restriction $\rho = [I; x_{\overline{I}}] \in R^{(n,p)}$ on the input of c.

The result now follows, since for any c chosen from $\mathfrak{C}_n$, the probability that $c\lceil_\rho$ is a constant is by Lemma 8 at least $1 - 8n^{-(1-\delta)/d(n)}$ and this constant can be found in polynomial time using $f_I(x)$. □

Again, by standard "averaging" arguments we have as an immediate Corollary:

Corollary 10. *Let $\delta \in (0,1)$ and let $\{\mathfrak{C}_n\}_{n\geq 1}$ be an $(s(n),d(n),\cdot)$-bounded ensemble of circuits where $d(n) \leq \frac{\log n}{\log\log n}$ and $s(n) \leq n^{-(1-\delta)/d(n)}2^{\frac{1}{14}n^{(1-\delta)/d(n)}}$. Then, there is a non-uniform one-way function f and deterministic polynomial time algorithm A such that for all sufficiently large n,*

$$\Pr[A(f(x),c) = c(x)] = 1 - O\left(n^{-(1-\delta)/d(n)}\right),$$

the probability taken over $x \in_{\mathcal{U}} \{0,1\}^n$, and c chosen according to $\mathfrak{C}_n$.

5.3 Uniform Case

In the bounded fan-in case we could derandomize our proofs by encoding the restriction as a part of x, the argument to the one-way function. We had to choose this encoding so that the circuit was completely uncorrelated with the restriction and this could be done by observing that bounded fan-in circuits cannot "see" all bits in x. For unbounded fan-in circuits however, the situation is more difficult, since theoretically at least, the circuit can have full information on the restriction. We will still use the same principal encoding of the restrictions, but we have to be more careful in the analysis.

We will now consider restrictions in $R^{(n,n^{-(1-\varepsilon)})}$, i.e. leaving n^ε $*$ for some $\varepsilon > 0$. We will encode them lexicographically like before. Let $x = x' \circ x''$ where x' is the first $L(n) = n^\alpha n^\varepsilon \log n$ bits in x and x'' is the $n - L(n)$ last bits. The constants ε and α will be determined later. Now let

$$H_\alpha^{(i)}(x') = \sum_{j=(i-1)n^\alpha+1}^{in^\alpha} x'_j \mod 2,$$

i.e. the XOR over the ith n^α-bit segment of x', and let

$$H_{\alpha,\varepsilon}(x') = H_\alpha^{(1)}(x') \circ H_\alpha^{(2)}(x') \circ \cdots \circ H_\alpha^{(n^\varepsilon \log n)}(x')$$

which we by setting $h = H_{\alpha,\varepsilon}(x')$ like in Lemma 2 interpret as an encoding of a restriction on the bits in x''. (We simplify, writing $H_{\alpha,\varepsilon}(x')$ instead of $Q_v(H_{\alpha,\varepsilon}(x'))$.)

We now get what in a natural way corresponds to restrictions on $x = x' \circ x''$ of the form $\rho = [H_{\alpha,\varepsilon}(x'); x' \circ x''_{\overline{H_{\alpha,\varepsilon}(x')}}]$. What we would like to do is to analyze the probability that a circuit collapses when subjected to such a restriction. However, we now clearly do not have the uniform distribution on restrictions, in particular we have all $*$ concentrated to the x''-part of x. The simple combinatorial arguments applicable to NC^0-circuits cannot be applied here. We must make a closer analysis of the induced distribution on the restrictions to be able to apply the switching lemma.

Below we describe three distributions $\mathcal{D}_1(\alpha)$, $\mathcal{D}_2(\alpha)$, and $\mathcal{D}_3(p,\alpha)$ on $R^{(n,n^{-(1-\varepsilon)})}$. The plan is to show that for suitable choices of p, α, ε,

(A) A random restriction from $\mathcal{D}_3(p,\alpha)$ will collapse our circuits with probability close to 1. (This will be Lemma 13.)
(B) $\mathcal{D}_3(p,\alpha)$ is equal to $\mathcal{D}_2(\alpha)$. (See Lemma 14.)
(C) $\mathcal{D}_2(\alpha)$ is very close to $\mathcal{D}_1(\alpha)$. (Lemma 15.)

The distribution $\mathcal{D}_1(\alpha)$ will be the one we actually have, and $\mathcal{D}_3(p,\alpha)$ is the one we will analyze. Where will this lead? Like before, we shall construct a one way function,

$$f(x) = f(x' \circ x'') = g(x''_{H_{\alpha,\varepsilon}(x')}) \circ x''_{\overline{H_{\alpha,\varepsilon}(x')}} \circ x'$$

(where again, g is another one-way function). For $x \in_U \{0,1\}^n$, $f(x)$ will correspond in a natural way to a random restriction $\rho \in_{\mathcal{D}_1(\alpha)} R^{(n,n^{-(1-\varepsilon)})}$. Hence, using our previous strategy for evaluating circuits under restrictions (substituting arbitrary values for

unknown inputs), we have an algorithm A such that by (A), (B), and (C) above,

$$\begin{aligned}\Pr_x[A(f(x),c) = c(x)] &\geq \Pr_{\mathcal{D}_1(\alpha)}[c\restriction_\rho \text{ is a constant}] \\ &\approx \Pr_{\mathcal{D}_3(p,\alpha)}[c\restriction_\rho \text{ is a constant}] \quad \text{(by (B),(C))} \\ &= 1 - o(1) \quad \text{(by (A))}.\end{aligned}$$

With this program in mind, we now define the distributions.

DISTRIBUTION $\mathcal{D}_1(\alpha)$

1. Choose x' uniformly at random in $\{0,1\}^{L(n)}$, and set $I = H_{\alpha,\varepsilon}(x')$.
2. Assign 0/1 with equal probability to the bits in x''_I.
3. Assign $*$ to all of $x''_{\overline{I}}$.

Let $\rho = [I; x' \circ x''_{\overline{I}}] \in R^{(n,n^{-(1-\varepsilon)})}$ be the induced restriction on x.

DISTRIBUTION $\mathcal{D}_2(\alpha)$

1. Choose I, a random n^ε-subset of the bits in x'' and assign $*$ to the bits in x''_I.
2. Assign 0/1 with equal probability to the bits in $x''_{\overline{I}}$
3. Choose x' uniformly at random in $H^{-1}_{\alpha,\varepsilon}(I)$.

Let $\rho = [I; x' \circ x''_{\overline{I}}] \in R^{(n,n^{-(1-\varepsilon)})}$ be the induced restriction on x.

The difference between $\mathcal{D}_1(\alpha)$ and $\mathcal{D}_2(\alpha)$ is that in $\mathcal{D}_1(\alpha)$ we choose the argument of $H_{\alpha,\varepsilon}$ and compute I from this, in $\mathcal{D}_2(\alpha)$ we reverse the procedure. Intuitively, since the "hash-function" $H_{\alpha,\varepsilon}$ is well behaved, it should not matter too much in which order we do these operations.

Finally we define a last distribution, the one that we will analyze. This last distribution is constructed in four steps where we first apply a random restriction from $R^{(n,p)}$ to all of x, i.e. *both* to x' *and* x''. For $\rho \in R^{(n,p)}$, write $\rho = \rho' \circ \rho''$ where ρ' and ρ'' are the parts of ρ assigning values to x' and x'' respectively.

DISTRIBUTION $\mathcal{D}_3(p,\alpha)$

1. Fix some bits in x', x'' by choosing a random $\rho = \rho' \circ \rho'' \in_{\mathcal{U}} R^{(n,p)}$, such that
   (a) $|*(\rho'')| \geq n^\varepsilon$ and such that
   (b) for $i = 1, \ldots, n^\varepsilon \log n$, $H^{(i)}_\alpha\restriction_\rho = *$. (I.e. each $H^{(i)}_\alpha(x')$ is undetermined by the restriction ρ' on x'.)
2. Choose I, a random n^ε-subset of $*(\rho'')$ and let these remain as $*$ in x''.
3. Assign 0/1 with equal probability to the remaining $*$ in $x''_{\overline{I}}$.
4. Assign 0/1 with equal probability to the remaining $*$ in x', but assert that $x' \in H^{-1}_{\alpha,\varepsilon}(I)$.

Let $\rho = [I; x' \circ x''_{\overline{I}}] \in R^{(n,n^{-(1-\varepsilon)})}$ be the induced restriction on x.

Let us give some motivation for studying this distribution. In step 1, we will fix some of the bits in x', but by condition 1b, they are not many enough to determine any of the $H^{(i)}_\alpha$ components of $H_{\alpha,\varepsilon}$ at all. To be able to later fix x' so that $H_{\alpha,\varepsilon}(x')$ can take any n^ε-subset of the indices in x'' as a value, we need at least n^ε $*$ in the x''-part and this is asserted by condition 1a. Therefore, after step 1, all possibilities for I (determined

in step 2) are still at this stage equally likely. The point is now that *if* our circuit c has collapsed after step 1, it will have done so without having any chance of obtaining information concerning I. Later steps, 2, 3 and 4, will only decrease the number of $*$ and thus c will remain collapsed. There is some hope to apply Lemma 8 to the restriction we have at step 1 *if* we can show that it "almost" random. Finally, steps 2, 3, and 4 will assert that the final restriction is consistent with distribution $\mathcal{D}_1(\alpha)$.

We start by showing that the restriction obtained after step 1 above is "just as good" as a uniformly distributed restriction. We first need two preparatory propositions.

Proposition 11. *Let* $p = n^{-(1-\delta)}$, $\rho \in R^{(n,p)}$. *For any* $J \subset \{1,2,\ldots,n\}$, $|J| = n^{\alpha}$ *with* $1-\delta < \alpha < 1$, *then*

$$\Pr_{\rho \in_U R^{(n,p)}}[|*(\rho) \cap J| = 0] \leq e^{-pn^{\alpha}}$$

for all sufficiently large n.

We omit the elementary proof.

Proposition 12. *If* $I \subset \{1,2,\ldots,n\}$, $|I| = n - n^{\alpha+\varepsilon}\log n$, $0 < \varepsilon < \delta < 1$, $\alpha+\varepsilon < 1$, *and* $p = n^{-(1-\delta)}$, *then for all sufficiently large n,*

$$\Pr_{\rho \in_U R^{(n,p)}}[|*(\rho) \cap I| < n^{\varepsilon}] \leq e^{-\frac{1}{2}(n^{\delta-\varepsilon}-1)}$$

Proof. The proposition follows from simple combinatorial arguments, approximating the binomial coefficients involved by Stirling's formula. □

Lemma 13. *Assume* $0 < \varepsilon < \delta < 1$, $\alpha \in (1-\delta, 1-\varepsilon)$, *and let c be a circuit of depth* $d(n) \leq \frac{\log n}{\log\log n}$ *and size* $s(n) \leq n^{-(1-\delta)/d(n)} 2^{\frac{1}{14}n^{(1-\delta)/d(n)}}$. *Then with* $p = n^{-(1-\delta)}$, *for all sufficiently large n,*

$$\Pr_{\rho \in_{\mathcal{D}_3(p,\alpha)} R^{(n,n^{-(1-\varepsilon)})}}[c\restriction_\rho \text{ is a constant function}] \geq 1 - O\left(n^{-(1-\delta)/d(n)}\right),$$

and if so, given ρ, *this constant can be found in time polynomial in* $s(n)$.

Proof. By Lemma 8, if we for the moment consider all of $R^{(n,p)}$ (i.e. regardless of whether or not it passes the constraints in step 1 in the definition of $\mathcal{D}_3(p,\alpha)$), we have

$$\Pr_{\rho \in_U R^{(n,p)}}[c\restriction_\rho \text{ is a constant function}] \geq 1 - 8n^{-(1-\delta)/d(n)}.$$

Let us call a restriction in $R^{(n,p)}$ *bad* if it does not satisfy the additional constraints in step 1. Write $\rho = \rho' \circ \rho''$ as defined above. First we see that the probability that at least one of the $H_\alpha^{(i)}$s (there are $n^{\varepsilon}\log n$ of them) becomes determined by ρ' (a violation of constraint 1b) is for large n by Proposition 11 bounded by

$$(n^{\varepsilon}\log n)e^{-pn^{\alpha}} = (n^{\varepsilon}\log n)e^{-n^{\alpha-(1-\delta)}}$$

which is negligible since $\alpha > 1-\delta$.

Furthermore, the probability that we get fewer than n^ε $*$ in ρ'' (violation of constraint 1a) is by Proposition 12 bounded by $e^{-\frac{1}{2}(n^{\delta-\varepsilon}-1)}$ since $\alpha+\varepsilon<1$. Thus, for large n

$$\Pr_{\rho\in_U R^{(n,p)}}[\rho \text{ is bad}] \leq (n^\varepsilon \log n)e^{-n^{\alpha-(1-\delta)}} + e^{-\frac{1}{2}(n^{\delta-\varepsilon}-1)}.$$

The lemma now follows since

$$\Pr_{\rho\in_{\mathcal{D}_3(p,\alpha)}R^{(n,n^{-(1-\varepsilon)})}}[c\restriction_\rho \text{ is a constant}] \geq 1 - 8n^{-(1-\delta)/d(n)} - \Pr_{\rho\in_U R^{(n,p)}}[\rho \text{ is bad}].$$

This is actually the probability the circuit has collapsed already after step 1 in forming $\mathcal{D}_3(p,\alpha)$, but as noted, the circuit will then surely remain collapsed. □

Next, notice that the distributions $\mathcal{D}_3(p,\alpha), \mathcal{D}_2(\alpha)$ are the same and furthermore, that $\mathcal{D}_1(\alpha), \mathcal{D}_2(\alpha)$ are close.

Lemma 14. *With $p = n^{-(1-\delta)}$, $0<\varepsilon<\delta<1$, $\alpha\in(1-\delta,1-\varepsilon)$, the two distributions $\mathcal{D}_3(p,\alpha)$, $\mathcal{D}_2(\alpha)$ on $R^{(n,n^{-(1-\varepsilon)})}$ are equal.*

Proof. We prove that for both distributions: (*i*) The location of $*$ in ρ'', i.e. the set I, have the same distribution. (*ii*) Non-$*$ in ρ'' are uniformly distributed in $\{0,1\}$. (*iii*) The ρ'-part is a uniformly distributed value consistent with $H_{\alpha,\varepsilon}(x') = I$. This will establish the claim.

(*i*) In $\mathcal{D}_2(\alpha)$ we first choose a random n^ε-subset as $*(\rho'')$ so each such is chosen with probability $\binom{|x''|}{n^\varepsilon}^{-1}$. In $\mathcal{D}_3(p,\alpha)$ after step 1 in forming the distribution, let R_1 be the random variable corresponding to $*(\rho'')$ at this stage and let R_2 similarly be the value of $*(\rho'')$ after step 2. Since the final $*(\rho'')$ is obtained by first choosing R_1 at random and then R_2 as a subset of R_1, we have by symmetry that for any n^ε-subset I:

$$\Pr_{\rho\in_{\mathcal{D}_3(p,\alpha)}R^{(n,n^{-(1-\varepsilon)})}}[*(\rho'') = I] = \binom{|x''|}{n^\varepsilon}^{-1} \Pr[|R_1| \geq n^\varepsilon].$$

But in $\mathcal{D}_3(p,\alpha)$ we discard precisely those ρ for which $|R_1| < n^\varepsilon$, and thus each n^ε-set is chosen with the same probability as by $\mathcal{D}_2(\alpha)$.

(*ii*) Note that there is no difference in the distribution on non-$*$ in ρ'' in the two distributions since they are in both cases assigned 0/1 with equal probability.

(*iii*) Lastly, when assigning 0/1 to ρ' in $\mathcal{D}_2(\alpha)$, we choose at random a $x'\in H^{-1}_{\alpha,\varepsilon}(I)$. By the XOR-construction of each $H^{(i)}_\alpha$, we can in each n^α-bit segment of x' choose any set of $n^\alpha-1$ indices uniformly at random in $\{0,1\}$ and the last bit in each block will have to be assigned a unique value determined by the condition $H_{\alpha,\varepsilon}(x') = I$. In $\mathcal{D}_3(p,\alpha)$ we also choose I before choosing x'. Since we in $\mathcal{D}_3(p,\alpha)$ by constraint 1b have at least one $*$ in each n^α-bit segment of ρ' and all $*$ were selected uniformly at random, we can there also fix all but one of the remaining $*$ as 0/1 at random and the last bit is determined uniquely by $H_{\alpha,\varepsilon}(x') = I$. □

Lemma 15. *For any fixed* $\rho_0 \in R^{(n,n^{-(1-\varepsilon)})}$,

$$2^{-1} \Pr_{\mathcal{D}_2(\alpha)}[\rho = \rho_0] \leq \Pr_{\mathcal{D}_1(\alpha)}[\rho = \rho_0] \leq 2 \Pr_{\mathcal{D}_2(\alpha)}[\rho = \rho_0].$$

Proof. Follows immediately from Lemma 2. □

Corollary 16. *Let* $0 < \varepsilon < \delta < 1$, $\alpha \in (1-\delta, 1-\varepsilon)$, *and let* c *be a circuit of depth* $d(n) \leq \frac{\log n}{\log\log n}$ *and size* $s(n) \leq n^{-(1-\delta)/d(n)} 2^{\frac{1}{14} n^{(1-\delta)/d(n)}}$. *Then, for sufficiently large* n,

$$\Pr_{\rho \in_{\mathcal{D}_1(\alpha)} R^{(n,n^{-(1-\varepsilon)})}}[c\restriction_\rho \text{ is a constant function}] \geq 1 - O\left(n^{-(1-\delta)/d(n)}\right),$$

and if so, this constant can from ρ *be computed in time polynomial in* $s(n)$.

Proof. With $p = n^{-(1-\delta)}$, the result follows immediately from lemmas 13, 14, and 15. □

Theorem 17. *For any* $\delta \in (0,1)$ *there is a one-way function* f *and a deterministic polynomial time algorithm* A *for which the following hold. For all sufficiently large* n, *for any* $(s(n), d(n), \cdot)$*-bounded ensemble of circuits* $\{\mathcal{C}_n\}_{n\geq 1}$, *where* $d(n) \leq \frac{\log n}{\log\log n}$, $s(n) \leq n^{-(1-\delta)/d(n)} 2^{\frac{1}{14} n^{(1-\delta)/d(n)}}$, *for every* c *supported by* $\mathcal{C}_n$,

$$\Pr_{x \in_U \{0,1\}^n}[A(f(x), c) = c(x)] = 1 - O\left(n^{-(1-\delta)/d(n)}\right).$$

Proof. Choose $\varepsilon \in (0, \delta)$, $\alpha \in (1-\delta, 1-\varepsilon)$, let g be a one-way function and define the one-way function

$$f(x) = f(x' \circ x'') = g(x''_{H_{\alpha,\varepsilon}(x')}) \circ x''_{\overline{H_{\alpha,\varepsilon}(x')}} \circ x'.$$

For any $c \in \mathcal{C}_n$, for random x, $f(x)$ gives a restriction $\rho = [H_{\alpha,\varepsilon}(x'); x' \circ x''_{\overline{H_{\alpha,\varepsilon}(x')}}] \in R^{(n,n^{-(1-\varepsilon)})}$ on c, and this ρ is chosen according to the distribution $\mathcal{D}_1(\alpha)$. It follows from Corollary 16 that $c\restriction_\rho$ will be a constant with probability $1 - O(n^{-(1-\delta)/d(n)})$, and if this is the case, we can use ρ (i.e. $f(x)$) to determine what this constant is in time polynomial in size(c). □

We immediately get the following corollary.

Corollary 18. *An* $(s(n), d(n), \cdot)$*-bounded ensemble of circuits, computing a (general) hard core predicate, requires depth* $d(n) \geq \frac{\log n}{(1+\varepsilon)\log\log n}$ *for every* $\varepsilon > 0$, *or otherwise, requires size* $s(n) \geq 2^{\omega(\log n)}$. *If* $d(n)$ *is a constant, size* $s(n) \geq 2^{n^{\Omega(1)}}$ *is required. In particular, there are no hard core predicates computable in* AC^0.

The existing constructions of hard core predicates such as [5] can be computed by polynomial size circuits of depth $\frac{\log n}{\log\log n}$ and hence, the lower bound in the Corollary is essentially tight.

6 Summary and Open Problems

This paper does not rule out the possibility of generating pseudo-random sequences in AC^0, but it does tell us that "generic" constructions based on an arbitrary one-way function and a hard core predicate does not work. For instance, the construction in [8] *could* still be a pseudo-random generator since it is based on a *particular* conjectured one-way function.

We have found an essentially tight lower bound on the complexity of computing (general) hard core predicates. Note that with respect to AC^0 circuits, the results obtained are uniform in a very strong sense: We have a fixed one-way function that has no hard core predicates computable by any circuit family of constant depth d and size n^r regardless of d, r and the distribution on the circuits.

Together with existing constructions of hard core predicates, we now have a good characterization of them with respect to computational complexity. The next step would therefore be to give, if possible, a more functional characterization of them.

Acknowledgment. We would like to thank Johan Håstad for fruitful discussions and suggestions. We also thank Alex Russell and Jean-Pierre Seifert.

References

1. W. Alexi, B. Chor, O. Goldreich, and C. P. Schnorr: *RSA and Rabin Functions: Certain Parts Are as Hard as the Whole.* SIAM J. on Computing **17** (1988), no 2, pp. 194–209.
2. P. Beame: *A Switching Lemma Primer.* Manuscript, 1994.
3. M. Blum and S. Micali: *How to Generate Cryptographically Strong Sequences of Pseudo-random Bits.* SIAM J. on Computing **13** (1986), no 4, pp. 850–864.
4. M. Furst, J. Saxe, and M. Sipser: *Parity, Circuits, and the Polynomial Time Hierarchy.* Proc. 22nd Symposium on Foundations of Computer Science, IEEE, 1981, pp. 260–270.
5. O. Goldreich and L. A. Levin: *A Hard Core Predicate for all One Way Functions.* Proc. 21st Symposium on Theory of Computing, ACM, 1989, pp. 25–32.
6. J. Håstad: *Computational Limitations of Small-Depth Circuits.* ACM doctoral dissertation award, 1986. MIT Press 1987.
7. J. Håstad, A. W. Schrift, and A. Shamir: *The Discrete Logarithm Modulo a Composite Hides $O(n)$ Bits.* J. of Computer and System Sciences **47** (1993), pp. 376–403.
8. R. Impagliazzo and M. Naor: *Efficient Cryptographic Schemes Provably as Secure as Subset Sum.* J. of Cryptology **9** (1996), no 4, pp. 199–216.
9. N. Linial, Y. Mansour, and N. Nisan: *Constant Depth Circuits, Fourier Transform, and Learnability.* J. of the ACM **40** (1993), no 3, pp. 607–620.
10. Y. Mansour, N. Nisan, and P. Tiwari: *The Computational Complexity of Universal Hashing.* Theoretical Computer Science **107** (1993), pp. 121–133.
11. M. Näslund: *Universal Hash Functions & Hard Core Bits.* Proc. Eurocrypt 1995, LNCS 921, Springer Verlag, pp. 356–366.
12. M. Näslund: *All Bits in $ax+b$ mod p are Hard.* Proc. Crypto 1996, LNCS 1109, Springer Verlag, pp. 114–128.
13. A. C. Yao: *Theory and Applications of Trapdoor Functions.* Proc. 23rd Symposium on Foundations of Computer Science, IEEE, 1982, pp. 80–91.
14. A. C. Yao: *Separating the Polynomial-Time Hierarchy by Oracles.* Proc. 26th Symposium on Foundations of Computer Science, IEEE, 1985, pp. 1–10.