J. Cryptology (2001) 14: 177–195
DOI: 10.1007/s00145-001-0007-6

Journal of
CRYPTOLOGY

Complexity Bounds on General Hard-Core Predicates[*]

Mikael Goldmann
Numerical Analysis and Computer Science,
Royal Institute of Technology,
Stockholm, Sweden
migo@nada.kth.se

Mats Näslund
Ericsson Research,
Stockholm, Sweden
mats.naslund@era-t.ericsson.se

Alexander Russell
Department of Computer Science and Engineering,
University of Connecticut,
Storrs, CT 06269, U.S.A.
acr@cse.uconn.edu

Communicated by Oded Goldreich

Received April 2000 and revised September 2000
Online publication 9 April 2001

Abstract. A Boolean function b is a hard-core predicate for a one-way function f if b is polynomial-time computable but $b(x)$ is difficult to predict from $f(x)$. A general family of hard-core predicates is a family of functions containing a hard-core predicate for any one-way function. A seminal result of Goldreich and Levin asserts that the family of parity functions is a general family of hard-core predicates. We show that no general family of hard-core predicates can consist of functions with $O(n^{1-\varepsilon})$ average sensitivity, for any $\varepsilon > 0$. As a result, such families cannot consist of

- functions in AC^0,
- monotone functions,
- functions computed by generalized threshold gates, or
- symmetric d-threshold functions, for $d = O(n^{1/2-\varepsilon})$ and $\varepsilon > 0$.

[*] The results in this article were originally presented at CRYPTO 1997 [6] and STACS 2000 [7]. Part of this research by the first author was done while visiting McGill University. The research of the second author was done at the Royal Institute of Technology, Stockholm. The third author was supported by NSF NYI Grant No. CCR-9457799 and a David and Lucile Packard Fellowship for Science and Engineering. Part of this research was done while he was a postdoctoral fellow at the University of Texas at Austin and the University of California, Berkeley, and part done while visiting McGill University.

The above bound on sensitivity is obtained by (lower) bounding the high-order terms of the Fourier transform. We also explore lower bounds on the size of small-depth circuits implied by the above bound on the average sensitivity.

Key words. Cryptography, One-way function, Hard-core predicate, Pseudorandom generator.

1. Introduction

A basic assumption on which much of modern (theoretical) cryptography rests is the existence of *one-way functions*. In general, such functions may have quite pathological structure, and the development of useful cryptographic primitives from general one-way functions (often with additional properties) is one of the triumphs of modern cryptography. One of the more troubling ways that a one-way function may be unsatisfactory is that it may "leak" information about x into $f(x)$: in particular, it may be possible to compute nearly all of x from $f(x)$ in polynomial time. The problem of showing that $f(x)$ hides *at least one bit* of information about x is the *hard-core predicate* problem, identified in the seminal work of Blum and Micali [3]. Specifically, they show how to construct a pseudorandom bit generator from a specific permutation (exponentiation modulo a large prime) under the assumption that computation of discrete logarithms is difficult. The generator starts with a seed x_0 and then iteratively computes $x_{i+1} = g^{x_i} \bmod p$, where g is a generator of $\mathbb{Z}_p^*$. In every iteration the generator outputs a pseudorandom bit $b(x_i)$, which is, more or less, the most significant bit of x_i. The proof of security of the Blum–Micali generator relies on the fact that if computing discrete logarithms is hard, then it is difficult to predict $b(x)$ given $g^x \bmod p$—the predicate b is a *hard-core predicate* for modular exponentiation. As they point out, their construction can be applied to any given one-way permutation and a hard-core predicate for that permutation.

At this point it was natural to ask if hard-core predicates exist for all one-way permutations. An affirmative, and quite satisfactory, answer was given by Goldreich and Levin [9] who demonstrated that *every one-way function has a hard-core predicate*.[1] Specifically, they show that for any one-way function f, there is a polynomial-time predicate b_f so that $b_f(x)$ is difficult to compute from $f(x)$. A hard-core predicate, though a basic primitive, has remarkable potency.

- If f is a permutation, a hard-core predicate immediately gives rise to a pseudorandom generator (by the Blum–Micali construction).
- If f is a permutation, a hard-core predicate immediately gives rise to a secure bit-commitment scheme (see [2]).
- If f is a one-way trapdoor permutation, a hard-core predicate for f immediately gives rise to a semantically secure encryption scheme (see [10] and [27]).
- The Goldreich–Levin construction of a hard-core predicate for any one-way function is an important ingredient in the proof that the existence of one-way functions implies the existence of pseudorandom generators [18], [13], [14].

[1] Specifically, a nonuniform hard-core predicate can be found for any one-way function f. If one can tolerate a minor alteration to the one-way function f, a uniform predicate can always be found.

Considering their importance (and the result of Goldreich and Levin mentioned above), it is natural to wonder how simple such predicates can be. For specific conjectured one-way functions such as RSA or discrete exponentiation, extremely simple hard-cores can be found, e.g., [1], [3], [17], and [16], but each of these proofs relies heavily on the specific structure of the relevant one-way function. In general, one seems to need relatively complex predicates.

In this article we demonstrate that, in general, hard-core predicates must be quite rich, having a nonnegligible portion of their Fourier transform concentrated on high-degree coefficients. From this follows a number of richness conditions for such predicates: general hard-core predicates

- cannot have small average sensitivity (specifically, they cannot have average sensitivity $O(n^{1-\varepsilon})$ for any $\varepsilon > 0$),
- cannot be computed in AC^0,
- cannot be monotone,
- cannot be computed by generalized threshold functions, and
- cannot be computed by symmetric d-threshold functions, with $d = O(n^{1/2-\varepsilon})$ for any $\varepsilon > 0$.

It is interesting to note that these results parallel those for universal hash functions obtained by Mansour et al. in [24]. See also [20] for related work on the complexity of pseudorandom generators.

Section 2 defines the notions of one-way function and hard-core predicate. Section 3 briefly erects the framework of Fourier analysis for Boolean functions. Sections 4 and 5 are devoted to proving the main theorem and discussing some applications. Section 6 refines the result for AC^0 functions by providing strong lower bounds on the size of constant-depth circuits computing hard-core predicates.

Preliminary versions of the results in this article were originally presented in [6] and [7]. The results in Section 6 of this article were originally obtained in [6] using Håstad's switching lemma [12] rather than the bounds on sensitivity used here.

2. One-Way Functions and Hard-Core Predicates

A function $f\colon \{0,1\}^* \longrightarrow \{0,1\}^*$ is *length-preserving* if $f(\{0,1\}^n) \subset \{0,1\}^n$ for all n. We write $f^{(n)}$ for f restricted to inputs of length n. For convenience, we restrict our attention to length-preserving one-way functions. This restriction does not affect the generality of the discussion: any one-way function can be modified so as to be length-preserving by a standard padding argument.

Definition 1. A (length-preserving) function $f\colon \{0,1\}^* \longrightarrow \{0,1\}^*$ is a *one-way function* if f is computable in polynomial time, and for all functions $A\colon \{0,1\}^* \longrightarrow \{0,1\}^*$ computable by polynomial-size circuits and for all $k > 0$,

$$\Pr\left[f(A(f^{(n)}(x))) = f^{(n)}(x)\right] = O(n^{-k}),$$

this probability taken uniformly over all $x \in \{0,1\}^n$.

In this cryptographic setting we consider A to be a polynomially bounded adversary attempting to invert the function f.

As discussed in the Introduction, a *hard-core predicate* for a one-way function f is a polynomial-time predicate b for which the value $b(x)$ is difficult to predict from $f(x)$. For reasons which will become clear later, it is convenient for us to express Boolean functions as functions taking values in the set $\{\pm 1\}$.

Definition 2. The Boolean function $b\colon \{0,1\}^* \longrightarrow \{\pm 1\}$ is a *hard-core predicate* for a length-preserving one-way function f if b is computable in polynomial time and for all functions $A\colon \{0,1\}^* \longrightarrow \{\pm 1\}$, computable by polynomial-size circuits, and for all $k > 0$,

$$\mathsf{Pr}\left[A(f^{(n)}(x)) = b^{(n)}(x)\right] = \tfrac{1}{2} + O(n^{-k}),$$

this probability taken uniformly over all $x \in \{0,1\}^n$.

Clearly, it would be desirable to have a predicate that is hard-core for any one-way function, but this is too much to hope for. Assuming that one-way functions exist, it is easy to see that for any predicate b one can construct a one-way function f for which $b(x)$ is easy to compute given $f(x)$. However, Goldreich and Levin showed that there is a fixed predicate b that is a hard core predicate for a padded version of any one-way function.

Theorem 1 [9]. *Let f be a one-way function, and define $g_f(x, w) = f(x) \circ w$ where $\circ$ denotes concatenation, and $|x| = |w|$. Then $b_{\mathrm{GL}}(x, w) = (-1)^{\sum_{i=1}^{|x|} x_i w_i}$ is a hard-core predicate for g_f.*

In Section 5 we discuss the concept of a *general family of hard-core predicates*, a family of functions containing a hard-core predicate for any one-way function. We also discuss there issues of uniformity for such predicates.

For a more detailed discussion of one-way functions, hard-core predicates, and their uses in modern cryptography, see [8] and [23].

3. Fourier Analysis of Boolean Functions

Let $L(\mathbb{Z}_2^n) = \left\{f\colon \mathbb{Z}_2^n \to \mathbb{R}\right\}$ denote the set of real-valued functions on $\mathbb{Z}_2^n = \{0,1\}^n$. Though our interest is in Boolean functions, it is temporarily convenient to consider this richer space. $L(\mathbb{Z}_2^n)$ is a vector space over $\mathbb{R}$ of dimension 2^n, and has a natural inner product: for $f, g \in L(\mathbb{Z}_2^n)$, we define

$$\langle f, g\rangle = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x) g(x).$$

For a subset $\alpha \subset \{1, \ldots, n\}$, we define the function $\chi_\alpha\colon \{0,1\}^n \to \mathbb{R}$ so that $\chi_\alpha(x) = \prod_{a \in \alpha} (-1)^{x_a}$. These functions χ_α are the *characters* of $\mathbb{Z}_2^n = \{0,1\}^n$. Among their many wonderful properties is the fact that *the characters form an orthonormal basis for $L(\mathbb{Z}_2^n)$*:

Proposition 2.

1. $\forall \alpha \subset [n]$,
$$\sum_{x \in \{0,1\}^n} \chi_\alpha(x) = \begin{cases} 2^n & \textit{if } \alpha = \emptyset, \\ 0 & \textit{otherwise}, \end{cases}$$
2. $\forall \alpha, \beta \subset [n]$, $\chi_\alpha(x)\chi_\beta(x) = \chi_{\alpha \oplus \beta}(x)$, *where* $\alpha \oplus \beta$ *denotes the symmetric difference of* α *and* β, *and*
3. $\forall \alpha, \beta \subset [n]$,
$$\langle \chi_\alpha, \chi_\beta \rangle = \begin{cases} 1 & \textit{if } \alpha = \beta, \\ 0 & \textit{otherwise}. \end{cases}$$

Considering item 3, the characters $\{\chi_\alpha \mid \alpha \subset [n]\}$ are orthogonal and have unit length. Since there are 2^n characters, they span $L(\mathbb{Z}_2^n)$, as promised. Any function $f\colon \{0,1\}^n \to \mathbb{R}$ may then be written in terms of this basis:

$$f = \sum_{\alpha \subset [n]} \widehat{f}_\alpha \chi_\alpha,$$

where $\widehat{f}_\alpha = \langle f, \chi_\alpha \rangle$ is the projection of f onto χ_α. These coefficients $\widehat{f}_\alpha$, $\alpha \subset [n]$, are the *Fourier coefficients* of f, and, as we have observed above, uniquely determine the function f.

Given the above, it is easy to establish the *Plancherel* equality:

Proposition 3. *Let* $f \in L(\mathbb{Z}_2^n)$. *Then* $\|f\|_2^2 = \sum_\alpha \widehat{f}_\alpha^2$, *where* $\|f\|_2^2 = \langle f, f \rangle = (1/2^n) \sum_{x \in \{0,1\}^n} f(x)^2$.

As always, $\widehat{f}_\emptyset = \mathsf{Exp}[f]$ and, when f is Boolean,

$$\sum_\alpha \widehat{f}_\alpha^2 = \|f\|_2^2 = 1.$$

It is interesting to note that if we let an n-bit string w encode a set $\alpha(w) \subset \{1, \ldots, n\}$ in the natural way, then the Goldreich–Levin predicate $b_{\mathrm{GL}}(x, w) = \chi_{\alpha(w)}(x)$.

A prominent theme in the study of (continuous) Fourier analysis is *local-global duality*:

> . . . the speed of convergence of a Fourier series improves with the smoothness of f. This reflects the fact that *local* features of f (such as smoothness) are reflected in *global* features of $\widehat{f}$ (such as rapid decay at $n = \pm\infty$). *This local-global duality is one of the major themes of Fourier series and integrals,. . .*
>
> Dym and McKean [5, p. 31]

This very same duality (between smoothness of f and rapid decay of $\widehat{f}$) is central for our study. In our framework, a natural measure of *smoothness* for a Boolean function f is its *average sensitivity*:

Definition 3. The *average sensitivity* of a Boolean function $f : \{0,1\}^n \to \{\pm 1\}$ is the quantity

$$\overline{\mathsf{S}}(f) = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} \sum_{i=1}^{n} \frac{|f(x) - f(x \oplus e_i)|}{2},$$

where $e_i \in \{0,1\}^n$ denotes the vector containing a single 1 at position i and $\oplus$ denotes coordinatewise sum modulo 2. (The $\frac{1}{2}$ factor appearing in the last term here reflects the choice of $\{\pm 1\}$ as the range for Boolean functions.)

We look at some examples. The average sensitivity for the n-input parity function is n, since for any input x, flipping any of the n input bits will change the parity. The n-input OR function has average sensitivity $2n2^{-n}$: if the input is all 0, then flipping any bit changes the value of the function (for this input the inner sum is equal to n), for any of the n inputs with weight 1, flipping the single 1 bit will change the value of the function (for each of these n inputs the inner sum is equal to 1), and for all inputs with weight at least 2, the inner sum is equal to 0.

Observe that average sensitivity is proportional to the likelihood that a random pair of neighboring points take on different values: "smooth" functions, where neighboring points are likely to agree, evidently have small average sensitivity.

The connection between average sensitivity (smoothness) and rapid decay of the Fourier transform is given by the following equality, due to Kahn et al. [19]:

$$\overline{\mathsf{S}}(f) = \sum_{\alpha \subset [n]} |\alpha| \, \widehat{f}_\alpha^{\,2}. \tag{1}$$

Considering the above equality, and recalling that $\|f\|_2 = 1$ for a Boolean function f, the average sensitivity of f is exactly determined by the distribution of this unit mass among the terms $\widehat{f}_\alpha^{\,2}$. This is a manifestation of the local-global duality principle mentioned above: functions having their Fourier transform concentrated on small coefficients (those for which $|\alpha|$ is small) have small average sensitivity and, as such, are smooth. In this case, we opt to *define* our notion of smoothness in terms of the Fourier transform as follows:

Definition 4. We say that a function $f\colon \{0,1\}^n \longrightarrow \{\pm 1\}$ is (t, δ)-smooth iff

$$\sum_{|\alpha| > t} \widehat{f}_\alpha^{\,2} \le \delta.$$

A function $g\colon \{0,1\}^* \longrightarrow \{\pm 1\}$ is $(t(n), \delta(n))$-smooth iff there exists $n_0 > 0$, so that for all $n \ge n_0$, $g^{(n)}$ is $(t(n), \delta(n))$-smooth.

A final word on notation: we frequently study functions $f(x, y)$ that take two strings x, y as input. Using I_x and I_y as the (disjoint) index sets for x and y, respectively, it is convenient to index the Fourier coefficients of f with two sets α, β, where $\alpha \subset I_x$ and $\beta \subset I_y$:

$$f(x, y) = \sum_{\substack{\alpha \subset I_x \\ \beta \subset I_y}} \widehat{f}_{\alpha,\beta} \chi_{\alpha \cup \beta}(x, y) = \sum_{\substack{\alpha \subset I_x \\ \beta \subset I_y}} \widehat{f}_{\alpha,\beta} \chi_\alpha(x) \chi_\beta(y),$$

where $\chi_{\alpha \cup \beta}(x, y) = \chi_\alpha(x)\chi_\beta(y)$ since $\alpha \cap \beta = \emptyset$.

4. Main Result

Before proceeding we discuss the relevance of sensitivity to the problem of finding hard core predicates.

4.1. *Motivation and Proof Outline*

First, notice that simple (and natural) candidates such as individual bits of x, e.g., $b(x) = \mathsf{lsb}(x)$ (the least significant bit) will in general fail to be hard-core predicates. A simple counterexample can be found by taking any one-way function g and defining $f(x_{n-1} \cdots x_0) = g(x_{n-1} \cdots x_1) \circ x_0$. Though f is a one-way function, computing $\mathsf{lsb}(x)$ from $f(x)$ is trivial. As mentioned in the Introduction one-way functions may, in general, "leak" substantial information about the input. The proof of our main result capitalizes on this. One can think of the proof as a construction of a one-way function f which leaks so much information about the input x, that for any sufficiently smooth function b, $b(x)$ is often determined by the information f leaks.

Now, to consider a somewhat more complicated candidate than $\mathsf{lsb}(x)$, suppose that f is a one-way function of the form $f(x) = g(x_J) \circ x_{\overline{J}}$ where $J \subset [n]$, $|J| = k$, and where x_J denotes $x_{i_1} x_{i_2} \cdots x_{i_k}$, $i_j \in J$, $i_1 < i_2 < \cdots < i_k$. In other words, f applies the one-way function g to a part of x and outputs the rest of x unchanged. 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. As long as this length reduction is within a polynomial factor, f will be a one-way function. We briefly take the perspective of the adversary: given $f(x)$, we should like to predict some function $b(x)$. We know a part of the input to b, i.e. $x_{\overline{J}}$, but not all of it. Now, if the sensitivity of b is low and the unknown part of x is small, then it is likely that the unknown bits, x_J, are irrelevant to the output, $b(x)$. In particular, substituting random values for these bits together with the known bits is likely to produce the correct value, $b(x)$.

Our plan is to construct a function f as above, which requires choice of an appropriate set J. Clearly, we cannot hope that a fixed J will work since we could then also find a function b that only depends on the bits in x_J that are hidden to us; such a b could then very well be unpredictable. This suggests that we use a *random* J. Choosing a single random J (for all x of a given length, say) and defining f in terms of this J also fails: first, this would lead to a nonuniform function f which is undesirable, but more importantly, it would lead to a considerably weaker result than we are aiming for. We wish to prove that "there exists a one-way function f such that every smooth predicate b fails to be a hard-core predicate for f." However, picking a random fixed J would only (naively) show that "for every smooth predicate b there is a one-way function f such that b fails to be a hard-core predicate for f" (which we already knew). Selecting a random J *for every input* x alleviates this difficulty. This randomness must be taken from somewhere and we do this by "borrowing" randomness from x itself since x is assumed to be random. This approach can be realized as follows. Writing the n input bits as $x \circ y$ (where $|x| = |y| = n/2$), we now interpret (in some way) y as an encoding of a subset $J = J(y)$ of the bits in x. We then compute f as $f(x, y) = g(x_{J(y)}) \circ x_{\overline{J(y)}} \circ y$.

This encoding could potentially introduce a problem. Since all information on J is available in y (which is also supplied to the candidate hard-core b), this encoding must be

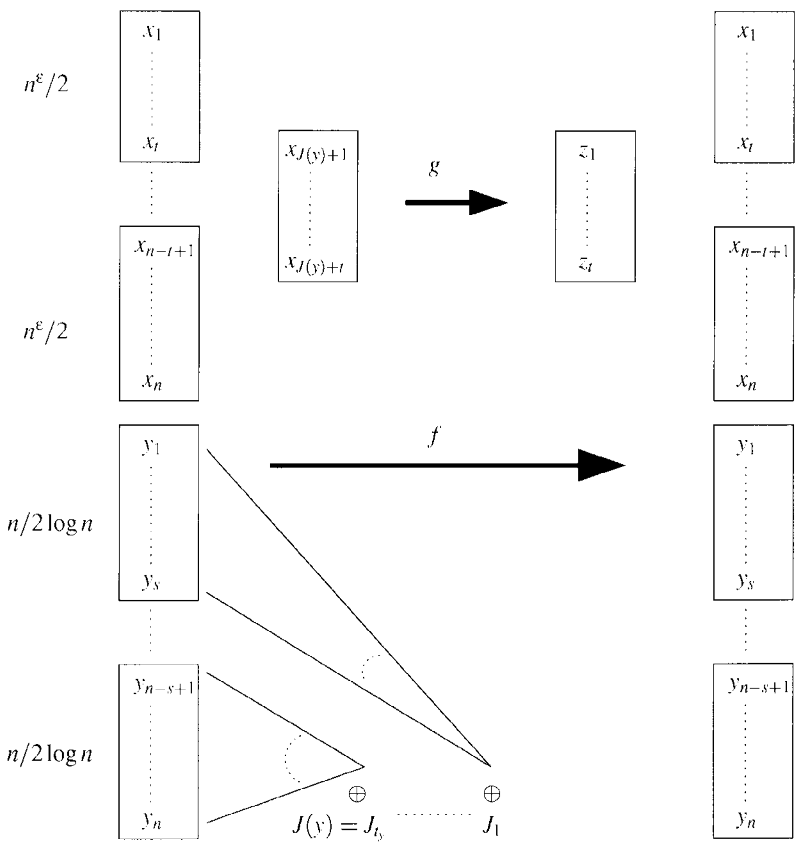

Fig. 1. Construction of f.

rich enough to avoid b "figuring out" which bits it should use, namely those in J, hidden by g. To this end, we use the fact that we are assuming that b has low sensitivity and define J in terms of highly sensitive (but still polynomial-time computable) functions, namely XORs, see Fig. 1.

4.2. *The Spectral Bound*

We can now begin working toward our main result which asserts that if one-way functions exist, then there are one-way functions for which every hard-core predicate is highly nonsmooth. In Section 5 we explore the consequences of this theorem for general hard-core predicates.

Theorem 4. *If there exists a one-way function, then for every $\varepsilon > 0$ there is a one-way function f_ε such that no $(\gamma\, n^{1-\varepsilon}, \delta)$-smooth Boolean function b: $\{0,1\}^* \longrightarrow \{\pm 1\}$ can be a hard-core predicate for f_ε if $\gamma + \delta < \frac{1}{16}$.*

As is shown in Section 5, this implies the bound on sensitivity claimed in the Introduction.

Proof. Let g: $\{0,1\}^* \to \{0,1\}^*$ be a (length-preserving) one-way function. Fix an arbitrary constant $\varepsilon > 0$. We describe below a one-way function $f_\varepsilon = f(x, y)$ of n variables so that if

$$b^{(n)}(x,y) = \sum_{\alpha,\beta} \widehat{b^{(n)}}_{\alpha,\beta}\chi_\alpha(x)\chi_\beta(y)$$

and $\sum_{|\alpha|+|\beta|>\gamma\, n^{1-\varepsilon}} (\widehat{b^{(n)}}_{\alpha,\beta})^2 \le \delta$, then, given $f(x, y)$, one can guess $b(x, y)$ with high probability.

Assume that n is even and define $f(x, y)$ where $|x| = |y| = n/2$ as follows. For an element $w \in \{0,1\}^k$ and a subset $S = \{s_1, \ldots, s_l\} \subset \{1, \ldots, k\}$, let $w_S = w_{s_1} \cdots w_{s_l}$, where $s_1 < \cdots < s_l$. The input x is divided into $t_x \overset{\text{def}}{=} n^{1-\varepsilon}$ blocks each consisting of $n^\varepsilon/2$ bits. Similarly, y is divided into $t_y \overset{\text{def}}{=} \log n^{1-\varepsilon}$ blocks of $n/(2\log n^{1-\varepsilon})$ bits. For simplicity, we ignore issues of integrality for these quantities. Then the value $f(x, y)$ is computed as follows. Write $y = y_{B_1} \cdots y_{B_{t_y}}$, where y_{B_i} is the ith block of y. Let $J_i = \bigoplus_{k\in B_i} y_k$ be the parity of the bits in y_{B_i}, and interpret the result $J_1, \ldots, J_{\log n^{1-\varepsilon}}$ as a binary coded integer $J(y) \in \{0, \ldots, n^{1-\varepsilon} - 1\}$. We define the set

$$\mathcal{J}(y) = \left\{ J(y)\frac{n^\varepsilon}{2}, J(y)\frac{n^\varepsilon}{2} + 1, \ldots, [J(y)+1]\frac{n^\varepsilon}{2} - 1 \right\};$$

these are precisely the indices of the $J(y)$th block of x. Finally, define $f(x, y) = (z, y)$ where $z_i = x_i$ when $i \notin \mathcal{J}(y)$ and $z_{\mathcal{J}(y)} = g(x_{\mathcal{J}(y)})$. Clearly, f is a one-way function, since inverting f in polynomial time implies inversion of g on n-bit inputs in time polynomial in $(2n)^{1/\varepsilon}$.

Now, let b: $\{0,1\}^* \to \{\pm 1\}$ be a $(\gamma\, n^{1-\varepsilon}, \delta)$-smooth Boolean predicate. Our goal is to show that b cannot be a hard-core predicate for f. For the remainder of this section, fix the input length to n, an integer large enough so that $b^{(n)}$ is $(\gamma\, n^{1-\varepsilon}, \delta)$-smooth and $4\gamma < n^\varepsilon/\log n^{1-\varepsilon}$. To simplify notation, for the remainder of this section we write b rather than $b^{(n)}$.

The following lemma then implies the theorem.

Lemma 5. *If $b(x, y)$ is $(\gamma\, n^{1-\varepsilon}, \delta)$-smooth, then there is a probabilistic polynomial-time algorithm A_b such that*

$$\Pr[A_b(f(x,y)) = b(x,y)] \ge 1 - 8(\gamma + \delta),$$

this probability is taken over uniformly random choice of x and y and the coin tosses of A_b.

Remark 1. Hard-core predicates are defined with respect to adversaries that are polynomial-size circuits. However, since probabilistic polynomial-time algorithms are less powerful than polynomial-size circuits, the above lemma is sufficient.

Describing A_b is easy. Given $f(x, y)$, y, and hence $\mathcal{J}(y)$, is known. Also, all of x except $x_{\mathcal{J}(y)}$ is known. Form x' by letting $x'_i = x_i$ when $i \notin \mathcal{J}(y)$, and picking $x'_{\mathcal{J}(y)}$ uniformly at random. Finally, let $A_b(f(x, y)) = b(x', y)$. The guess is correct when $b(x, y) = b(x', y)$. Note that x' and y are independent and uniformly distributed. However, x and x' are of course highly dependent.

By the above discussion, Lemma 5 follows from Lemma 6, below.

Lemma 6. *If $b(x, y)$ is $(\gamma n^{1-\varepsilon}, \delta)$-smooth, and x, x', y are generated as described above, then*

$$\Pr[b(x, y) = b(x', y)] \geq 1 - 8(\gamma + \delta).$$

Let Z be the indicator function

$$Z(A, B) = \begin{cases} 1 & \text{if } A \cap B \neq \emptyset, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

and define

$$\begin{aligned} e(x, y) &= \sum_{|\alpha|+|\beta| \leq \gamma n^{1-\varepsilon}} \widehat{b}_{\alpha,\beta} \chi_\alpha(x) \chi_\beta(y)(1 - Z(\alpha, \mathcal{J}(y))), \\ h(x, y) &= \sum_{|\alpha|+|\beta| \leq \gamma n^{1-\varepsilon}} \widehat{b}_{\alpha,\beta} \chi_\alpha(x) \chi_\beta(y) Z(\alpha, \mathcal{J}(y)), \quad \text{and} \\ r(x, y) &= \sum_{|\alpha|+|\beta| > \gamma n^{1-\varepsilon}} \widehat{b}_{\alpha,\beta} \chi_\alpha(x) \chi_\beta(y). \end{aligned}$$

Now, $b(x, y) = e(x, y) + h(x, y) + r(x, y)$, and note that $e(x, y)$ depends only on inputs *exposed* by $f(x, y)$ (i.e., it does not depend on $x_{\mathcal{J}(y)}$), whereas each term in $h(x, y)$ depends on some *hidden* bits (i.e., bits in $x_{\mathcal{J}(y)}$).

Observe that when x, x', y are generated according to the above procedure, $e(x, y) = e(x', y)$. We will prove that for random (x, y), with high probability both $|r(x, y)|$ and $|h(x, y)|$ are small. Since (x', y) has the same distribution as (x, y) these bounds hold for $\left|r(x'y)\right|$ and $\left|h(x', y)\right|$ as well. This is sufficient to prove that with high probability $b(x, y) = b(x', y)$.

The contributions of $r(x, y)$ and $h(x, y)$ are bounded by the following two lemmas.

Lemma 7. *If $b(x, y)$ is $(\gamma n^{1-\varepsilon}, \delta)$-smooth, and x, y are uniformly distributed, then*

$$\Pr\left[|r(x, y)| \geq \lambda\right] \leq \lambda^{-2}\delta.$$

Proof. The probability is bounded with the Chebychev inequality: for a real-valued random variable X,

$$\Pr\left[|X - \mathrm{Exp}[X]| \geq \lambda\right] \leq \lambda^{-2}\,\mathrm{Var}[X]. \tag{2}$$

Observe that the expectation of each term in $r(x, y)$ is 0 since either $|\alpha| > 0$ or $|\beta| > 0$; hence $\mathrm{Exp}[r(x, y)] = 0$ and

$$\Pr\left[|r(x, y)| \geq \lambda\right] \leq \lambda^{-2}\,\mathrm{Var}[r(x, y)]\,.$$

As the terms appearing in $r(x, y)$ are pairwise independent, the variance of the sum is equal to the sum of the variances. Then

$$\mathrm{Var}[\widehat{b}_{\alpha,\beta} \chi_\alpha(x) \chi_\beta(y)] = (\widehat{b}_{\alpha,\beta})^2\,\mathrm{Var}[\chi_{\alpha\cup\beta}(x, y)]$$

and, since $\alpha \cup \beta \neq \emptyset$ for the terms in this sum, $\mathsf{Var}[\chi_{\alpha\cup\beta}(x, y)] = 1$ so that

$$\mathsf{Var}[r(x, y)] = \sum_{|\alpha|+|\beta|>\gamma\, n^{1-\varepsilon}} (\widehat{b}_{\alpha,\beta})^2 \leq \delta,$$

as desired. □

Lemma 8. *If* $4\gamma < n^{\varepsilon}/\log n^{1-\varepsilon}$, *then* $\mathsf{Pr}\,[|h(x, y)| \geq \lambda] \leq \lambda^{-2}\gamma$.

The proof of Lemma 8 is slightly technical, so we first see that Lemmas 7 and 8 together imply that with high probability $b(x, y) = b(x', y)$.

Proof. [Proof of Lemma 6] Since $|b(x, y)| = \left|b(x', y)\right| = 1$ it is enough to show that with high probability

$$\left|b(x, y) - b(x', y)\right| < 2.$$

Considering that $e(x, y) = e(x', y)$, applying the triangle inequality we have

$$\left|b(x, y) - b(x', y)\right| \leq |r(x, y)| + \left|r(x', y)\right| + |h(x, y)| + \left|h(x', y)\right|.$$

Therefore,

$$\begin{aligned}\mathsf{Pr}\left[\left|b(x, y) - b(x', y)\right| \geq 2\right] &\leq \mathsf{Pr}\left[|r(x, y)| \geq \tfrac{1}{2}\right] + \mathsf{Pr}\left[\left|r(x', y)\right| \geq \tfrac{1}{2}\right]\\ &\quad + \mathsf{Pr}\left[|h(x, y)| \geq \tfrac{1}{2}\right] + \mathsf{Pr}\left[\left|h(x', y)\right| \geq \tfrac{1}{2}\right]\\ &\leq 8\delta + 8\gamma.\end{aligned}$$

The last inequality follows by two applications of Lemma 7 and two applications of Lemma 8 (with $\lambda = \frac{1}{2}$). □

As mentioned above, Theorem 4 follows from Lemma 6. □

It remains to prove Lemma 8.

Proof of Lemma 8. As before, we use the Chebychev inequality,

$$\mathsf{Pr}\,[|h(x, y)| \geq \lambda] \leq \lambda^{-2}\,\mathsf{Var}[h(x, y)] = \mathsf{Exp}[(h(x, y))^2] - \mathsf{Exp}[h(x, y)]^2.$$

By linearity of expectation and independence of x and y,

$$\mathsf{Exp}[h(x, y)] = \sum_{|\alpha|+|\beta|\leq\gamma\, n^{1-\varepsilon}} \widehat{b}_{\alpha,\beta}\,\mathsf{Exp}[\chi_{\alpha}(x)]\,\mathsf{Exp}[Z(\alpha, \mathcal{J}(y))\chi_{\beta}(y)].$$

Now, $Z(\emptyset, \mathcal{J}(y)) = 0$ for all y, and when $\alpha \neq \emptyset$ then $\mathsf{Exp}[\chi_{\alpha}(x)] = 0$; hence each term in $h(x, y)$ has expectation 0 and $\mathsf{Exp}[h(x, y)] = 0$, and we have

$$\begin{aligned}\mathsf{Var}[h(x, y)] &= \mathsf{Exp}[(h(x, y))^2]\\ &= \sum_{\substack{|\alpha|+|\beta|\leq\gamma\, n^{1-\varepsilon}\\ |\alpha'|+|\beta'|\leq\gamma\, n^{1-\varepsilon}}} \widehat{b}_{\alpha,\beta}\widehat{b}_{\alpha',\beta'}\,\mathsf{Exp}[\chi_{\alpha}(x)\chi_{\alpha'}(x)]\\ &\qquad \cdot \mathsf{Exp}[\chi_{\beta}(y)\chi_{\beta'}(y)Z(\alpha, \mathcal{J}(y))Z(\alpha', \mathcal{J}(y))].\end{aligned}$$

As $\mathsf{Exp}[\chi_\alpha(x)\chi_{\alpha'}(x)] = \delta_{\alpha,\alpha'}$, we have

$$\mathsf{Var}[h(x,y)] = \sum_{\substack{|\alpha|+|\beta|\le\gamma\, n^{1-\varepsilon} \\ |\alpha|+|\beta'|\le\gamma\, n^{1-\varepsilon}}} \widehat{b}_{\alpha,\beta}\widehat{b}_{\alpha,\beta'}\,\mathsf{Exp}[\chi_{\beta\,\oplus\,\beta'}(y) Z(\alpha, \mathcal{J}(y))].$$

Accept, for the moment, the following claim.

Claim 9. *For any* β, β' *with* $\max(|\beta|, |\beta'|) \le \gamma\, n^{1-\varepsilon}$, $\chi_{\beta\,\oplus\,\beta'}(y)$ *and* $Z(\alpha, \mathcal{J}(y))$ *are independent*.

This allows us to estimate the variance:

$$\begin{aligned}
\mathsf{Var}[h(x,y)] &= \sum_{\substack{|\alpha|+|\beta|\le\gamma\, n^{1-\varepsilon} \\ |\alpha|+|\beta'|\le\gamma\, n^{1-\varepsilon}}} \widehat{b}_{\alpha,\beta}\widehat{b}_{\alpha,\beta'}\,\mathsf{Exp}[\chi_{\beta\,\oplus\,\beta'}(y)]\,\mathsf{Exp}[Z(\alpha, \mathcal{J}(y))] \\
&= \sum_{\substack{|\alpha|+|\beta|\le\gamma\, n^{1-\varepsilon} \\ |\alpha|+|\beta'|\le\gamma\, n^{1-\varepsilon}}} \widehat{b}_{\alpha,\beta}\widehat{b}_{\alpha,\beta'}\delta_{\beta,\beta'}\,\mathsf{Exp}[Z(\alpha, \mathcal{J}(y))] \\
&= \sum_{|\alpha|+|\beta|\le\gamma\, n^{1-\varepsilon}} (\widehat{b}_{\alpha,\beta})^2\,\mathsf{Exp}[Z(\alpha, \mathcal{J}(y))] \\
&\le \sum_{|\alpha|+|\beta|\le\gamma\, n^{1-\varepsilon}} (\widehat{b}_{\alpha,\beta})^2 \frac{\gamma\, n^{1-\varepsilon}}{n^{1-\varepsilon}} \\
&\le \gamma\ .
\end{aligned}$$

We have used the Plancherel equality to conclude that $\sum_{|\alpha|+|\beta|\le n^{1-\varepsilon}} (\widehat{b}_{\alpha,\beta})^2 \le 1$. Also, $\mathsf{Exp}[Z(\alpha, \mathcal{J}(y))]$ is exactly the probability that the fixed set α intersects the randomly chosen block indicated by y; this probability is at most $|\alpha|$ divided by the number of blocks.

It remains to prove the claim. We need to show that $\chi_{\beta\,\oplus\,\beta'}(y)$ and $Z(\alpha, \mathcal{J}(y))$ are independent. However, this follows from the fact that each "bit" in $J(y)$ is the parity of $n/(2\log n^{1-\varepsilon})$ bits, whereas $\chi_{\beta\,\oplus\,\beta'}(y)$ depends on the parity of $|\beta \oplus \beta'| \le 2\gamma n^{1-\varepsilon}$ bits. By the assumption stated in the lemma, $2\gamma n^{1-\varepsilon} < n/(2\log n^{1-\varepsilon})$. Therefore, even when all the bits in $\beta \oplus \beta'$ are fixed, the bits of $J(y)$ are still uniformly and independently distributed. □

In the next section we explore the ramifications of Theorem 4 for general families of hard-core predicates.

5. Applications to General Families of Hard-Core Predicates

In this section and the next, we adopt a stronger notion of efficient computation with respect to hard-core predicates. Specifically, we broaden our consideration to include all predicates for which there is a family of polynomial-size Boolean circuits $\{C_n \mid n > 0\}$ so that $b(w) = C_n(w)$ for all $w \in \{0,1\}^n$. As we have adopted the convention that Boolean functions take values in the set $\{\pm 1\}$, we must demand the same of our circuits.

For concreteness, then, we treat all Boolean circuits as though the map taking 0 to 1 and 1 to -1 is implicitly applied to their outputs.

The reason for considering this *nonuniform* notion of hard-core predicate is that rather than focusing on the behavior of a single predicate, we wish to explore families of predicates guaranteed to contain a hard-core predicate for any one-way function f. Such families are called *general hard-core predicates*. Typically, as in the Goldreich–Levin construction, a randomly chosen member of the family of predicates is likely to be a hard-core predicate for f, and folding this "random choice" into the definition of b (naively) requires nonuniformity. This does not greatly affect the results in the previous section. The only difference is that the algorithm A_b requires a (polynomial-size) circuit for the predicate b so that it can evaluate b on an input x, y.

Definition 5. A family $\mathcal{B} \subset \{\pm 1\}^{\{0,1\}^*}$ is called a *general family of hard-core predicates* if for every one-way function f there is a (nonuniform) polynomial-time computable predicate $b \in \mathcal{B}$, such that b is a hard-core predicate for f.

It is a consequence of the proof of Theorem 1 that the collection of functions

$$\mathcal{B}_{\mathrm{GL}} = \left\{ p\colon \{0,1\}^* \to \{\pm 1\} \,\middle|\, \forall n,\ p^{(n)} = \chi_{\alpha^{(n)}},\ \text{for some } \alpha^{(n)} \subset \{1, \ldots, n\} \right\}$$

is a general family of hard-core predicates. For completeness, we outline a proof below.

Proof Sketch. For a family $A = (\alpha^{(1)}, \alpha^{(2)}, \ldots)$, with each $\alpha^{(n)} \subset [n]$, define b_A: $\{0,1\}^* \to \{\pm 1\}$ to be the predicate $b_A(w_1, \ldots, w_n) = \chi_{\alpha^{(n)}}(w)$. Our goal is to show that for any one-way function f, there is a family A for which the function b_A is a hard-core predicate for f. To simplify notation, we only consider the case where f is a permutation.

For a predicate b and an input length n, the *hardness* of b is defined as $H_f(b, n) = s$ where s is the largest integer such that every circuit C of size $< s$ has $\Pr_{x \in \{0,1\}^n}[C(f(x)) = b(x)] < \frac{1}{2} + s^{-1}$. Since f is a permutation, $H_f(b, n)$ is always finite. It is easy to see that b is a hard-core predicate for f if and only if for all c and all sufficiently large n it holds that $H_f(b, n) > n^c$.

Let $A = (\alpha^{(1)}, \alpha^{(2)}, \ldots)$, where $\alpha^{(n)} \subset [n]$ is chosen to maximize $H_f(\chi_{\alpha^{(n)}}, n)$. Then b_A is a hard-core predicate. Assume to the contrary that there is a $c > 0$ so that for infinitely many n, $H_f(b_A, n) \le n^c$. We show that in this case, one can construct a polynomial-size circuit family that inverts f with nonnegligible probability on infinitely many input lengths n; this contradicts that f is one-way.

Implicit in the proof of Theorem 1 (due to [9]) is a construction of a (uniform) polynomial-time oracle machine M^B which has the property that

$$\Pr_{\substack{x \in \{0,1\}^n \\ \alpha \subset [n]}}[B(f(x), \alpha) = \chi_\alpha(x)] > \frac{1}{2} + \frac{1}{n^{c_0}} \quad \Longrightarrow \quad \Pr[M^B(f(x)) = x] > \frac{1}{n^{c_1}}, \tag{3}$$

where c_0 is any positive constant and c_1 depends on c_0. The latter probability is taken over the choice of x and the coin tosses of M. Furthermore, the α_i for which M queries $B(f(x), \alpha_i)$ *depend only on the coin tosses of* M; they are independent of $f(x)$ and the answers of the oracle B.

We consider an arbitrary n such that $H_f(b_A, n) \leq n^c$, and let $C[\alpha]$ be a circuit of size n^c such that $\Pr[C[\alpha](f(x)) = \chi_\alpha(x)] \geq \frac{1}{2} + n^{-c}$. Let the oracle B be defined by $B(f(x), \alpha) = C[\alpha](f(x))$. Then $\Pr[M^B(f(x)) = x] \geq n^{-c'}$ for some c'. Fix the random bits of M in such a way that it inverts f on at least a fraction $n^{-c'}$ of the inputs. This fixes the α_i for which the oracle is queried, and we can replace the oracle by the circuits $C[\alpha_i]$, giving us a polynomial-size circuit. This construction can be repeated for the infinite collection of integers n for which $H_f(b_A, n) \leq n^c$, as desired. □

By being more careful, one can prove a stronger, but still nonuniform, result. Specifically, we show that if A is constructed by picking $\alpha^{(n)} \subset [n]$ uniformly at random for each n, then with probability 1, b_A is a hard-core predicate.

Proof Sketch. The basic structure of the proof is analogous to that of the above existence proof. First, note that for any constants $c, d > 0$, if $\Pr_{\alpha \subset [n]}[H_f(\chi_\alpha, n) \leq n^c] \geq n^{-d}$, then we can repeat the above construction of the oracle B, this time letting $C[\alpha]$ be the circuit of size n^c that maximizes $\Pr_{x \in \{0,1\}^n}[C[\alpha](f(x)) = \chi_\alpha(x)]$. Then

$$\Pr_{x,\alpha}[B(f(x), \alpha) = \chi_\alpha(x)] \geq \tfrac{1}{2} + \Pr_\alpha[H_f(\chi_\alpha, n) \leq n^c] \cdot n^{-c} \geq \tfrac{1}{2} + n^{-cd}$$

and as before we may construct a polynomial-size circuit that inverts f.

Now, let $A = (\alpha^{(1)}, \alpha^{(2)}, \ldots)$ be the random variable determined by independent (and uniform) selection of each $\alpha^{(n)}$ in $\{0, 1\}^n$. For any $c \in \mathbb{N}$, let $E_{c,n}$ be the event $H_f(\chi_{\alpha^{(n)}}, n) \leq n^c$. By the above discussion, there is a constant $n_0 > 0$ so that $\Pr_{\alpha^{(n)}}[E_{c,n}] < n^{-2}$ for all $n > n_0$. Thus $\sum_n \Pr_{\alpha^{(n)}}[E_{c,n}] \leq N_0 + \sum_n n^{-2} < \infty$. Now, by the Borel–Cantelli lemma (see, for example, Section 8.3 of [4]),

$$\Pr_A[\exists n_c \forall n > n_c, \overline{E_{c,n}}] = 1.$$

As the union of countably many sets of zero measure has zero measure, we conclude that

$$\Pr_A[\forall c \in \mathbb{N}, \exists n_c \forall n > n_c, \overline{E_{c,n}}] = 1.$$

Thus, with probability 1, b_A is a hard-core predicate. □

One consequence of the theorem of the last section is that general families of hard-core predicates cannot be smooth:

Corollary 10. *If $\mathcal{B}$ is a general family of hard-core predicates, then, for every $\varepsilon > 0$, it must contain a function which is not $(n^{1-\varepsilon}, \frac{1}{17})$-smooth.*

The close connection between smoothness and average sensitivity implies the following.

Corollary 11. *If $\mathcal{B}$ is a general family of hard-core predicates then, for every $\varepsilon > 0$, $\mathcal{B}$ must contain a function with average sensitivity greater than $n^{1-\varepsilon}$ for all sufficiently large n.*

Proof. The lower bound on the sensitivity follows from (1) coupled with the lower bound on smoothness. Specifically, let b be a Boolean function and assume that $\overline{\mathsf{S}}(b^{(n)}) \leq n^{1-2\varepsilon}$ for some fixed $\varepsilon > 0$. Then

$$n^{1-2\varepsilon} \geq \overline{\mathsf{S}}(b^{(n)}) \geq \sum_{|\alpha|>n^{1-\varepsilon}} |\alpha| \, (\widehat{b^{(n)}}_\alpha)^2 \geq n^{1-\varepsilon} \sum_{|\alpha|>n^{1-\varepsilon}} (\widehat{b^{(n)}}_\alpha)^2$$

and thus $\sum_{|\alpha|>n^{1-\varepsilon}} (\widehat{b^{(n)}}_\alpha)^2 < n^{-\varepsilon}$ so that $b^{(n)}$ is $(n^{1-\varepsilon}, n^{-\varepsilon})$-smooth. The corollary follows. □

A celebrated theorem of Linial et al. shows that functions in AC^0 are smooth:

Theorem 12 [22]. *Let $f\colon \{0,1\}^* \to \{\pm 1\}$ be a Boolean function with polynomial-size constant depth circuits. Then f is $(\log^{O(1)} n, o(1))$-smooth.*

An immediate corollary is that general hard-core predicates cannot be computed in AC^0:

Corollary 13. *If $\mathcal{B}$ is a general family of hard-core predicates, then it must contain a function which is not in* AC^0.

In Section 6 we derive strong bounds on the size of small-depth circuits that compute general hard-core predicates.

It is interesting to note the folklore theorem [21] which asserts that any monotone function f has small average sensitivity:

Lemma 14. *Let f be a monotone Boolean function, then $\overline{\mathsf{S}}(f) = O(\sqrt{n})$.*

Proof. Focus on $f^{(n)}\colon \{0,1\}^n \to \{-1,1\}$. For an element $x \in \{0,1\}^n$ let $\mathbf{wt}(x) = \sum_i x_i$ denote the weight of x and let $N(x) = \{x \oplus e_i \mid i = 1, \ldots, n\}$ denote the set of neighbors of x. A *minterm* of f is an element $x \in \{0,1\}^n$ so that for all $y \in N(x)$ with $\mathbf{wt}(y) < \mathbf{wt}(x)$, $f(y) < f(x)$. Similarly, a *maxterm* of f is an element $x \in \{0,1\}^n$ so that for all $y \in N(x)$ with $\mathbf{wt}(y) > \mathbf{wt}(x)$, $f(y) > f(x)$. Consider the following two rules:

- Rule A. Let x be a minterm of f with $\mathbf{wt}(x) < n/2$. Transform f into f', the Boolean function equal to f on all points except for x, where $f'(x) = -1$.
- Rule B. Let x be a maxterm of f with $\mathbf{wt}(x) > n/2$. Transform f into f', the Boolean function equal to f on all points except for x, where $f'(x) = 1$.

Observe that these transformations preserve monotonicity and, furthermore, the application of either rule to a function f results in a function f' with $\overline{\mathsf{S}}(f) < \overline{\mathsf{S}}(f')$. Repeated application of these rules (in any order) results in a monotone function g for which neither rule is applicable; hence

$$M^n_{>} \leq g \leq M^n_{\geq},$$

where $M^n_>(x) = 1 \Leftrightarrow \mathbf{wt}(x) > n/2$ and $M^n_\geq(x) = 1 \Leftrightarrow \mathbf{wt}(x) \geq n/2$. For such functions g, $\overline{\mathsf{S}}(g) = O(\sqrt{n})$, so that $\overline{\mathsf{S}}(f) \leq \overline{\mathsf{S}}(g) = O(\sqrt{n})$, which establishes the lemma. $\square$

Clearly, the same bound holds for any generalized monotone function (a generalized monotone function is obtained by negating some of the inputs to a monotone function).

In light of the above, the following is immediate:

Corollary 15. *If $\mathcal{B}$ is a general family of hard-core predicates, then it must contain a nonmonotone function.*

A Boolean function $f\colon \{0,1\}^n \to \{\pm 1\}$ is a *d-threshold function* if there exists a real multivariate polynomial $p \in \mathbb{R}[x_1, \dots, x_n]$ of total degree d or less so that $\forall (x_1, \dots, x_n) \in \{0,1\}^n$,

$$f(x_1, \dots, x_n) = \operatorname{sign} p(x_1, \dots, x_n).$$

When $d = 1$ such functions are *generalized threshold functions* and are generalized monotone functions; their average sensitivity is addressed in Lemma 14 above.

In general, it has been shown by Gotsman and Linial [11] that d-threshold functions are $(d, 1 - \varepsilon_d)$-smooth, for a constant $\varepsilon_d > 0$ independent of n. Though this is not strong enough for our application, they show that under the added assumption that f is *symmetric*, one has

$$\overline{\mathsf{S}}(f) \leq 2^{-n+1} \sum_{k=0}^{d-1} \binom{n}{[(n-k)/2]} \left(n - \left[\frac{n-k}{2} \right] \right),$$

where $[x]$ is the integer part of x. Observe that when $d = O(n^{1/2-\varepsilon})$, this quantity is $O(n^{1-\varepsilon})$. Then the following is immediate.

Corollary 16. *If $\mathcal{B}$ is a general family of hard-core predicates, then it must contain a function which, for large enough n, cannot be expressed as the sign of a symmetric polynomial of degree $d = O(n^{1/2-\varepsilon})$, for any $\varepsilon > 0$.*

6. Bounds on Circuit Size and Depth

Instead of characterizing functions by smoothness, we can use circuit complexity as a measure of simplicity. In this section we derive strong lower bounds on the size of small-depth circuits for general hard-core predicates.

Definition 6. A *circuit* is a directed acyclic graph having *gates* as vertices. A gate can be of type OR or AND 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, $i = 1, 2, \dots, n$, or a negated input. 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;

as mentioned above, we actually treat this output as a ± 1 value. The output gate can therefore be considered as a sink, and the vertices corresponding to inputs are sources in the graph. The *size* of the circuit is the number of gates and the *depth* is the maximum path length from an input to the output gate.

A circuit c computing a Boolean function b is said to *depend* on m bits if there is a fixed m-subset, $J \subseteq \{1, 2, \ldots, n\}$, such that for all x, $|x| = n$, $c(x)$ is uniquely determined by x_J.

We assume that the circuit is "leveled." That is, the gates at level i take their inputs from gates at level $i - 1$ and all gates at a given level are of the same type (AND/OR), types alternating from level to level. Hence all inputs x_i (and negations $\bar{x}_i$) are at level 0, and the depth of the circuit is thus the number of levels. The restrictions that all negations are at the input level and that the circuit is leveled are not severe; converting an arbitrary circuit with unbounded fan-in gates to this form increases its depth by at most 1, and its size increases by a constant factor. As usual, AC^0 denotes the set of Boolean functions computable by circuits of constant depth, polynomial size, and unbounded fan-in.

Consider the function f_ε in Theorem 4, and let b be a hard-core predicate for f_ε. The lower bound on the sensitivity of b implies a lower bound on the size of a depth-d AND/OR circuit that computes b.

Theorem 17. *If b is a hard-core predicate for f_ε, then a depth-d AND/OR circuit for b must have size at least $2^{\Omega(n^{(1-\varepsilon)/d})}$.*

Proof. We use Theorem 4 and the following result due to Linial et al.

Lemma 18 [22]. *Let b be a Boolean function computed by a circuit of depth d and size M, and let t be any integer. Then*

$$\sum_{|\alpha|>t} \widehat{b}_\alpha^{\,2} \le 2M2^{-t^{1/d}/20} .$$

Let $t = n^{1-\varepsilon}/40$. If $2M2^{-n^{(1-\varepsilon)/d}/800} \le \frac{1}{40}$, then b is $(n^{1-\varepsilon}/40, \frac{1}{40})$-smooth, and cannot be a hard-core predicate for f_ε. The lower bound on M follows. □

7. Hard-Core Predicates for Padded Functions

We return to the Goldreich–Levin construction. It asserts that there is a fixed function b_{GL} that is a hard-core predicate for g_f, a padded version of an arbitrary one-way function, f.

One can consider a generalized construction, any one-way function $f(x)$ yielding a padded version $g_f(x, w) = f(x) \circ w$, where $|w| = p(|x|)$ for some polynomial p, so that a fixed predicate b (independent of f) is a hard-core predicate for g_f. What kind of lower bounds can one show for, for instance, the sensitivity of b?

Our results can be extended to such padded functions: a predicate such as b above must have average sensitivity greater than $|x|^{1-\varepsilon}$ for all $\varepsilon > 0$. Note that this is not that far from optimal since the Goldreich–Levin predicate has average sensitivity $\Theta(|x|)$.

The lower bound follows from the following simple argument. For an arbitrary ε, take f to be the function f_ε constructed in Section 4.2. Now, the task of the adversary is to guess $b(x, w)$ given $f(x)$ and w (using x to mean both inputs of f_ε). However, once w is known, b depends only on x, and hence is easy to guess from $f(x)$ if it has average sensitivity less than $|x|^{1-\varepsilon}$.

Finally, one can consider more general ways than padding to modify a one-way function f so as to offer a guarantee that a fixed predicate b will be a hard-core predicate for the result. In this general setting, however, the spectral properties of b cannot be bounded as above. Consider the example of a one-way function $f(x)$ modified to produce $h_f(x, w, z)$, where $|x| = |w|$, z is a single bit, and we define

$$h_f(x, w, z) = f(x) \circ w \circ z',$$

where

$$z' = (1 + (2z - 1) \cdot b_{\mathrm{GL}}(x, w))/2$$

and let $b'(x, w, z) = (2z - 1)$. Clearly, h_f is one-way if f is one-way, and it is not hard to see that predicting $b'(x, w, z)$ from $h_f(x, w, z)$ is as hard as predicting $b_{\mathrm{GL}}(x, w)$ from $f(x)$ and w. On the other hand, the sensitivity of b' is 1 for all inputs.

8. Conclusion and Open Questions

The results presented here indicate a certain degree of optimality on behalf of the Goldreich–Levin construction (see also [6]). (Observe that with probability 1 a function selected from $\mathcal{B}_{\mathrm{GL}}$ will have linear average sensitivity.) Also, it suggests a connection between families of universal hash functions and general hard-core predicates. On the one hand, several well-known examples of universal hash functions have been shown to be general hard-core predicates [9], [25], [26], and, on the other hand, smooth functions make poor hash functions as well as poor hard-core predicates. An interesting problem is to determine if there is a more precise connection between universal hash functions and hard-core predicates.

Acknowledgments

We thank Johan Håstad for several helpful discussions. We thank Oded Goldreich and an anonymous referee for their comments and suggestions.

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. Comput.*, 17(2):194–209, Apr. 1988.

[2] M. Blum. Coin flipping by telephone: A protocol for solving impossible problems. In A. Gersho, editor, *Advances in Cryptology*: *A Report on* CRYPTO 81, pages 11–15, Department of Electrical and Computer Engineering, U.C. Santa Barbara, 24–26 Aug. 1981. ECE Report 82-04, 1982.

[3] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, Nov. 1984.

[4] R. Dudley. *Real Analysis and Probability*. Chapman and Hall Mathematics Series. Chapman and Hall, New York, 1989.

[5] H. Dym and H. P. McKean. *Fourier Series and Integrals*, volume 14 of Probability and Mathematical Statistics. Academic Press, New York, 1972.

[6] M. Goldmann and M. Näslund. The complexity of computing hard core predicates. In B. S. Kaliski Jr., editor, *Advances in Cryptology—CRYPTO* '97, volume 1294 of Lecture Notes in Computer Science, pages 1–15. Springer-Verlag, Berlin, 1997.

[7] M. Goldmann and A. Russell. Spectral bounds on general hard-core predicates. In H. Reichel and S. Tisson, editors, *Proceedings of the* 17*th Annual Symposium on Theoretical Aspects of Computer Science—STACS* 2000, volume 1770 of Lecture Notes in Computer Science, pages 614–625. Springer-Verlag, Berlin, Feb. 2000.

[8] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Springer-Verlag, New York, 1999.

[9] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, pages 25–32, Seattle, Washington, 15–17 May 1989.

[10] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. System Sci.*, 28(2):270–299, Apr. 1984.

[11] C. Gotsman and N. Linial. Spectral properties of threshold functions. *Combinatorica*, 14(1):35–50, 1994.

[12] J. Håstad. *Computational Limitations for Small Depth Circuits*. Ph.D. thesis, ACM Doctoral Dissertations Award. MIT Press, Cambridge, Massachusettes, 1986.

[13] J. Håstad. Pseudo-random generators under uniform assumptions. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 395–404, Baltimore, Maryland, 14–16 May 1990.

[14] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396 (electronic), 1999.

[15] J. Håstad and M. Näslund. The security of all RSA and discrete log bits. In *Proceedings of the* 39*th Annual Symposium on Foundations of Computer Science*, pages 510–519, Palo Alto, California, 8–11 Nov. 1998.

[16] J. Håstad and M. Näslund. The security of individual RSA bits. Manuscript, 1999. An Earlier version appeared in [15].

[17] J. Håstad, A. W. Schrift, and A. Shamir. The discrete logarithm modulo a composite hides $O(n)$ bits. *J. Comput. System Sci.*, 47(3):376–404, Dec. 1993.

[18] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions (extended abstracts). In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, Washington, 15–17 May 1989.

[19] J. Kahn, G. Kalai, and N. Linial. The influence of variables on Boolean functions (extended abstract). In *Proceedings of the* 29*th Annual Symposium on Foundations of Computer Science*, pages 68–80, White Plains, New York, 24–26 Oct. 1988.

[20] M. Kharitonov, A. V. Goldberg, and M. Yung. Lower bounds for pseudorandom number generators. In *Proceedings of the* 30*th Annual Symposium on Foundations of Computer Science*, pages 242–247, Research Triangle Park, North Carolina, 30 Oct.–1 Nov. 1989.

[21] N. Linial. Private communication. December, 1998.

[22] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *Assoc. Comput. Mech.*, 40(3):607–620, July 1993.

[23] M. Luby. *Pseudorandomness and Cryptographic Applications*. Princeton University Press, Princeton, New Jersey, 1996.

[24] Y. Mansour, N. Nisan, and P. Tiwari. The computational complexity of universal hashing. *Theoret. Comput. Sci.*, 107(1):121–133, 1993.

[25] M. Näslund. Universal hash functions and hard core bits. In *EUROCRYPT*: *Advances in Cryptology*: *Proceedings of EUROCRYPT*, volume 921 of Lecture Notes in Computer Science, pages 356–366. Springer-Verlag, Berlin, 1995.

[26] M. Näslund. All bits in $ax + b \bmod p$ are hard (extended abstract). In N. Koblitz, editor, *Advances in Cryptology—CRYPTO* '96, volume 1109 of Lecture Notes in Computer Science, pages 114–128. Springer-Verlag, Berlin, Aug. 1996.

[27] A. C. Yao. Theory and applications of trapdoor functions (extended abstract). In *Proceedings of the* 23*rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, 3–5 Nov. 1982.