P=BPP unless E has sub-exponential circuits: Derandomizing the XOR Lemma

(Preliminary Version)

Russell Impagliazzo*
Computer Science and Engineering
UC, San Diego
9500 Gilman Drive
La Jolla, CA 92093-0114
russell@cs.ucsd.edu

Avi Wigderson†
Institute of Computer Science
Hebrew University
Jerusalem, Israel
avi@cs.huji.ac.il

October 15, 1996

Abstract

Yao showed that the XOR of independent random instances of a somewhat hard Boolean problem becomes almost completely unpredictable. In this paper we show that, in non-uniform settings, total independence is not necessary for this result to hold. We give a randomness-optimal generator for n instances of a problem for which the analog of the XOR lemma holds. Combining this generator with the results of Nisan and Wigderson [NW] gives substantially improved results for hardness vs randomness trade-offs. In particular, we show that if any problem in $E = DTIME(2^{O(n)})$ has circuit complexity $2^{\Omega(n)}$, then $P = BPP$. Our generator is an intersting combination of two known ones - the random walks on expander graphs of [**?**, **?**, **?**] and the nearly disjoint subsets generator of [**?**, NW]. The quality of the new generator is proved via the recent Hard-Core theorem of [10], and a new proof of the XOR lemma.

*Research supported by NSF YI Award CCR-92-570979, Sloan Research Fellowship BR-3311, grant #93025 of the joint US-Czechoslovak Science and Technology Program, and USA-Israel BSF Grant 92-00043

†Currently visiting the Institute for Advanced Study, Princeton, N.J. 08540 and Princeton University. Research supported the Sloan Foundation, American-Israeli BSF grant 92-00106, and the Wolfson Research Awards, administered by the Israel Academy of Sciences.

1 Introduction

This paper addresses the relationship between two central questions in complexity theory. First, what properties should a pseudo-random generator have so that its outputs are "random enough" for the purpose of simulating a randomized algorithm? Secondly, if solving one instance of a problem is computationally difficult, is solving several instances of the problem proportionately harder?

Yao's seminal paper ([16]) was the first to show that these questions are related. Building on work by Blum and Micali ([2]) he showed how to build, from any cryptographically secure one-way permutation, a pseudo-random generator whose outputs are indistinguishable from truly random strings to any reasonably fast computational method. Blum, Micali, and Yao thus showed the central connection between "hardness" (the computational difficulty of a problem) and "randomness" (the utility of the problem as the basis for a pseudo-random generator to de-randomize computation).

Moreover, Yao's construction introduced the XOR Lemma, the prototypical example of a "direct product lemma" (a specific way in which several instances or a problem are more difficult than a single instance.) The XOR Lemma states that (for non-uniform computation) if any computational method of a certain complexity has a significant probability of failure when solving a single randomly chosen instance of a boolean (i.e., one bit) problem, then any method (of a slightly smaller complexity) will be unable to guess the XOR of the bits for n random instances significantly better chance of success than a random coin toss.

In this paper, we show that true independence of the instances is not necessary for a version of the XOR Lemma to hold. We show how to pick n instances of a somewhat hard problem in a way that minimizes the total number of random bits used but still guarantees that the XOR of the bits for the instances is almost as hard to predict as if the instances were independent. We then use this "de-randomized" XOR lemma to improve the construction of Nisan and Wigderson [NW]) which gives a general method of derandomizing computation using a hard to predict predicate. We also give a modified version of the result of Feigenbaum and Fortnow ([**?**]) showing that E-complete problems are random-self-reducible. Combining these ingrediants, we conclude that unless every problem in E has worst-case circuit complexity $2^{o(n)}$, $P = BPP$. In other words, randomness *never* speeds computation by more than a polynomial amount unless non-uniformity *always* helps computation more than polynomially for problems with exponential time complexity. (Using previously known techniques, one could show under the same hardness assumption that $BPP \subseteq DTIME(n^{O(\log n)})$)

2 Direct Product Lemmas: Definitions and History

Let m, ℓ, and s be positive integers. Let π an arbitrary distribution on $\{0,1\}^m$. For every function $f : \{0,1\}^m \to \{0,1\}^\ell$, distribution π and size bound s we define the *success* $SUC_s^\pi(f)$ to be the maximum, over all Boolean circuits C of size at most s of $Pr_\pi[C(x) = f(x)]$. Note that if f can be computed in size s than $SUC_s^\pi(f) = 1$. For the (important) case $\ell = 1$

we define the *advantage* $ADV_s^{\pi}(f) = 2SUC_s^{\pi}(f) - 1$. In both notations we replace the superscript π by A if π is the uniform distribution on a subset $A \subseteq \{0,1\}^m$. When π is uniform on $\{0,1\}^m$ we eliminate the superscript altogether.

Two fundamental results that we shall need are the *hard-core set* result of Impagliazzo [10] and the *hard-core predicate* result of Goldreich and Levin [GL].

2.1 Hard-Core Sets

Definition 2.1 *Assume $\ell = 1$ and let $0 < \epsilon < 1$. A subset $H \subseteq \{0,1\}^m$ is called an ϵ-hard-core set for f and size s if $ADV_s^H(f) \leq \epsilon$.*

Note that the existance of a hard-core set for f, s trivially gives a upper bound on $SUC_s(f)$. The hard-core set theorem is a strong converse to this observation. (I am stating it just for the uniform dist - we should probably state it, and our general theorem, for arbitarary dist, or at least claim we can do that)

Theorem 2.2 *[10] For every function f, with $SUC_s(f) \leq 1 - \delta$, and every ϵ there is an ϵ-hard-core set H of size $|H|/2^m \geq \delta$ for f and size $s\epsilon^2/\log(1/\delta)$.*

2.2 Hard-Core Predicates

Definition 2.3 *Given f, define $f' : \{0,1\}^m \times \{0,1\}^\ell \to \{0,1\}$ by $f'(x,y) = < f(x), y >$, where $< \cdot, \cdot >$ denotes $GF(2)$ inner product. For a distribution π on $\{0,1\}^m$ define π' on $\{0,1\}^m \times \{0,1\}^\ell$ be π on the first coordinate and uniform on the second, independently. The function f' is called the* Goldreich-Levin predicate *for f.*

Note that trivially if $SUC_{s+O(m)}^{\pi'}(f') \geq SUC_s^{\pi}(f)$. The hard core predicate theorem is a strong converse of this inequality. The following paraphrases the main result of [GL]:

Theorem 2.4 *[GL] There are functions $s' = s(\epsilon)^{O(1)}$, $\delta = \epsilon^{O(1)}$ so that: for every function f and distribution π, if $SUC_s^{\pi}(f) \leq \delta$, then $ADV_{s'}^{\pi'}(f') \leq \epsilon$.*

Note that if $\epsilon = 1-\gamma$, where $\gamma = o(1)$, then $\delta = (1-\gamma)^O(1) = 1-O(\gamma)$. So between functions and predicates that are only weakly unpredictable the above relationship gives quite tight bounds.

2.3 The XOR Lemma

In this subsection, we state the XOR Lemma of Yao, which we will generalize in later sections.

Let $g : \{0,1\}^n \to \{0,1\}$ and assume $SUC_s(g) \leq 1-\delta$. Information theoretically (thinking of the value of g on random input from $\{0,1\}^n$ as a coin flip), the advantage of guessing the exclusive or of k independent copies is $(1-2\delta)^k$. The XOR lemma is a computational manifestation of this intuition. Define $g^{\oplus(k)} : (\{0,1\}^n)^k \to \{0,1\}$ by $g(x_1, \cdots, x_k) = (g(x_1) \oplus \cdots \oplus g(x_k))$.

Theorem 2.5 *[16] There are functions k, s' of n, s, ϵ, δ with $k = O(-\log \epsilon/\delta)$, $s' = s(\epsilon\delta)^{O(1)}$ and so that, for any $g : \{0,1\}^n \rightarrow \{0,1\}$ with $SUC_s(g) \leq 1-\delta$, we have: $ADV_{s'}(g^{\oplus(k)}) \leq \epsilon$.*

The above hard-core predicate [GL] reduces the proof of the XOR lemma to the proof of the following "direct product" theorem, which requires the circuit to output *all* values of $g(x_i)$, rather than their exclusive-or.

Define $g^{(k)} : (\{0,1\}^n)^k \rightarrow \{0,1\}^k$ by $g(x_1, \cdots, x_k) = (g(x_1), \cdots, g(x_k))$. Then

Theorem 2.6 *There are $k = O(-\log \epsilon/\delta)$, $s' = s(\epsilon\delta)^{O(1)}$ so that for any $g : \{0,1\}^n \rightarrow \{0,1\}$ with $SUC_s(g) \leq 1-\delta$. we have: $SUC_{s'}(g^k) \leq \epsilon$,*

Note that the Goldreich-Levin predicate for $g^{(k)}$ is basically equivalent to $g^{\oplus k/2}$, (The number of inputs is not fixed at $k/2$ but a binomial distribution with expectation $k/2$; however, if we restrict r in the Goldreich-Levin predicate to have exactly $k/2$ 1's, the bound on the advantage is almost the same.)

In the next section, we give a new proof of this theorem, which generalizes to distributions on $\vec{x}$ which are not uniform and hence can be sampled from using fewer than nk random bits. This allows us to give some combinatorial conditions which suffice for such a pseudo-random distribution to have direct product properties similar to the uniform distribution. We give a construction of such a pseudo-random generator which (for certain parameters of interest) uses only a linear number of bits to sample n n-bit inputs. We formalize the notion of a direct product result holding with respect to a pseudo-random generator as follows:

Definition 2.7 *Let G be a function from $\{0,1\}^m$ to $(\{0,1\}^n)^k$. G is an $(s, s', \epsilon, \delta)$ direct product generator, if for every boolean function g with $SUC_s(g) \leq 1-\delta$, we have:*

$$SUC_{s'}(g^k \circ G) \leq \epsilon$$

. (Here, the distribution is over uniformly chosen inputs to G.)

Let s, ϵ, m, δ and k be fixed functions of n, and let G_n be a polynomial-time computable function from $\{0,1\}^m$ to $(\{0,1\}^n)^k$. The family of functions G is called a (s, ϵ, δ) direct product generator *if each G_n is a $(s, s\epsilon^{O(1)}, \epsilon, \delta$ direct product generator, In the case $\delta = O(1)$, we say G is a (s, ϵ) direct product generator.*

In this notation, [10] proved:

Theorem 2.8 *There is a function G from $\{0,1\}^{2n}$ to $(\{0,1\}^n)^n$ which, for any $s = s(n)$, $c \geq 1$, is a $(s, O(n^{-(c-1)}), n^{-c})$ direct product generator,*

The G they constructed produces samples that are pairwise independent. They use the hardcore set lemma to show that this suffices. Combining this result with the Goldreich-Levin predicate allows us to convert any boolean function f on inputs of length n with $SUC_s(f) \leq 1 - n^{O(1)}$ to a boolean function g with $SUC_{sn^{-O(1)}}(g) \leq 1 - \Omega(1)$ so that inputs to g are of size $O(n)$ and g is computable in polynomial-time with an oracle for f. This allows us, in the rest of the paper, to only consider the case when δ is a fixed constant.

2.4 The Card Guessing Game: A new proof of the direct product lemma

The XOR Casino offers the following game. A sequence of k cards each labelled either 0 or 1 is put on the table facing down. The cards are chosen by the casino, with the only restriction being that (by gambling laws of the state of Nevada) they must all be 1 with probability p. One card is chosen at random, and the casino turns over all cards except that one. You must guess the value of the chosen card, and recieve pay-off of \$1 if your guess is correct. What is the value of this game? (A game with a similar flavor was considered in [?].)

The correspondence with the direct product lemma is, intuitively, as follows. Let g be a boolean function. We want to use an algorithm A advertised as having a probability p of computing $g^k(x_1, ..x_k)$ to get an algorithm A' that predicts g on a single x. We know that A gets all k answers correct with probability p. But when A does not get the answers all correct, then the answers it gives might be anything, so we assume they are picked adversarily (by the "casino"). The value of the i'th "card" is 1 when A predicts $g(x_i)$ correctly, 0 otherwise, so we know all are 1 with probability p. We "pick" a card i by setting $x_i = x$, and "turn over" the other cards by having the values $g(x_j)$ as "advice" (if we are in a non-uniform model of computation) or by choosing x_j in a way so that we can compute $g(x_j)$ (in cryptographic settings.) A' will then incorporate our strategy for the card game to get a strategy for either believing or disbelieving the prediction of $g(x) = g(x_i)$ given by A.

We will give an almost optimal strategy for the card guessing game, and then formalize the above intuitive correspondence. Observe that if the casino, when not obliged by law to make all cards 1's, fixes the values as independent coin tosses, your probability is at most $p + 1/2(1-p) = 1/2 + p/2$. We will show that you can get a pay-off quite close to this trivial bound, and get the answer correct with probability $1/2 + p/2 - 2^{-k/3}$.

Let $\bar{a} = (a_1, a_2, \cdots, a_k) \in \{0,1\}^k$ denote the random sequence chosen by the casino, Consider the following strategy S: Let t_I be the number of cards you see marked 0. S guesses 1 with probability 2^{-t_I}), and otherwise guesses 0 or 1 uniformly at random. (i.e., the overall probability of guessing 0 is $1/2(1-2^{-t_I})$ and that of guessing 1 is $1/2(1+2^{-t_I})$.)

Theorem 2.9 *For any distribution on $\bar{a}$ which is equal to 1^k with probability at least p,,*

$$Pr[S = a_I] \geq 1/2 + p/2 - 2^{-k/3}$$

.

Proof.

Let $\bar{a}$ be a fixed sequence of cards. Clearly, if $\bar{a} = i^k$, then S always guesses correctly. We'll show that, for any other sequence, the conditional probability that S guesses correctly is at least $1/2 - 2^{-k/3}$.

Let T denote the number of coordinates i in which $a_i = 0$. Then with probability T/k, $a_I = 0$, and the obseved number of 0's is $t_I = T-1$. In this case, S outputs 0 with probability $1/2(1-2^{-t_i}) = 1/2 - 2^{-T}$. In the other case, which occurs with probability $1 - T/k$, $a_I = 1$, we see all T 0's and output 1 with probability $1/2(1+2^{-T}) = 1/2 + (1/2)2^{-T}$. Thus, the overall probability that S outputs the correct answer is $1/2 + 2^{-T}((1-T/k)(1/2) - T/k) = 1/2 + 2^{-T}\frac{k-3T}{2k}$. If $T \geq k/3$, this is at $1/2$, and if $T < k/3$ it is at most $1/2 - 2^{-T} \geq 1/2 - 2^{-k/3}$.

Thus the overall probability that S gets the right answer is at least $p + (1-p)(1/2 - 2^{-k/3}) \geq 1/2 + p/2 - 2^{-k/3}$.

We shall actually need a small variant of the above game. In the variant game, the casino picks, in addition to $\bar{a}$, a subset K of k' cards (which can depend on $\bar{a}$). The casino does not reveal K to you, but does give you a uniformly chosen element $I \in K$. You get to see the values of all cards except a_I and try to predict a_I. (Intuitively, K represents those instances of the problem that are in the hard-core set.) The player can still use the strategy S described above, and a similar calculation shows:

Theorem 2.10 *For any distribution on $\bar{a}$ and K so that $\bar{a} = 1^k$ with probability at least p ,*

$$Pr[S = a_I] \geq 1/2 + p/2 - 2^{-k'/3}$$

.

2.5 (Yet Another) Proof of the XOR Lemma

Three completely different proofs of the XOR lemma are known[?, 10, ?], with the last reference containing a survey of all three. We are now ready for our new proof, after which we discuss its benefits for the purpose of derandomization.

Proof. (of Theorem 2.6) Let $q(k,\delta)$ be the probability that if k independent coins are tossed each with probability δ of heads, fewer than $\delta k/2$ heads are tossed. Then there is a constant $0 < c < 1/6$ so that $q(k,\delta) \leq 2^{-c\delta k}$. For $k = 1/c(-\log\epsilon + 3)/delta = O(-\log\epsilon/\delta)$, $q(k,\delta) \leq \epsilon/8$. Similarly, since we chose $c < 1/6$, $2^{-k\delta/6} \leq \epsilon/8$. This is the value of k we use. We'll show the contrapositive of the statement in Theorem 2.6: If $SUC_{s'}(g^{(k)} \geq \epsilon$, then $ADV_s(g) \geq 1-\delta$ for some $s = s'n^{O(1)}(\epsilon^{-O(1)})$

Assume $SUC_{s'}(g^{(k)}) \geq \epsilon$, and let C be a circuit achieving this succees probability. We shall use it to construct a distribution on circuits F of approximately the same size, for which $Exp_F[ADV_F^H(g)] > \epsilon/2 - 2^{-c\delta k}$ for some absolute constant $c > 0$ and *any* subset H of inputs with $|H| \geq \delta 2^n$.

Then it follows that taking $t = O(n/p^2)$ samples from this distribution $F_1, ...F_t$, $Maj(F_1, ..F_t)$ (the Boolean majority function of the samples) is likely to predict f correctly on all but an at most δ fraction of inputs, contrary to the hardness assumption. the desired upper bound on p.

The distribution on F is merely implementing the strategy S of the previous section using $\bar{a} = C(\bar{x}) \oplus g^{(k)}(\bar{x})$.

Consider the following distribution on probabilistic circuits with one $n - bit$ input x. $I \in [k]$ is chosen uniformly at random, and for each $J \neq I$, x_J is chosen uniformly. F sets $x_I = x$ and simulates $C(\bar{x})$. F then computes t, the number of $J \neq I$ where the J'th output of $C(\bar{x})$ differs from $g(x_J)$. (In non-uniform settings, $g(x_J)$ is given as advice; in uniform cryptographic settings, x_J can be sampled so that $g(x_J)$ is known.) F outputs the I'th output of $C(\bar{x})$ with probability 2^{-t} and otherwise outputs 0 or 1 with equal probability.

Lemma 2.11 *Let $H \subseteq \{0,1\}^n, |H| \geq \delta 2^n$. Let $q = q(\delta, k)$. Then $Exp_F[ADV_F^H(g)] > \epsilon/2 - (1.5)q - 22^{-\delta k/6} \geq \epsilon/16$.*

Proof. Without loss of generality, assume $|H| = \delta 2^n$. (If not, apply the result to a random $H' \subseteq H$ of the above size.) Consider the distribution on $x_1, ...x_k$ chosen by F given that $x \in H$. Any sequence $x_1, ..x_k$ with s elements from H has s values of I for which it can be chosen by F. Thus, the probabilities of such vectors are proportional to s, and since the average value of s is δk, the exact probability is $s/(\delta k)$ times the probability in the uniform distribution.

Now, there are at least a $\epsilon - q$ fraction of such tuples so that both $s \geq \delta k/2$ and $C(\bar{x}) = g^k(x)$. For each such sequence, the probability that it is chosen by F given $x \in H$ is at least 1/2 the probability that it is chosen under the uniform distribution. Therefore, the probability that F simulates C on such a tuple is at least $(\epsilon - q)/2$. Note that given that F simulates C on $\bar{x}$, I is uniformly distributed among the indices i with $x_i \in H$.

Consider the following casino strategy for the variant card-guessing game, with $k' = \delta k/2$. $I, x_1, ..x_k$ are chosen as F does, given that at least k' indices i have $x_i \in H$. The casino sets $a_i = 1$ if and only if the i'th bit of $C(\vec{x})$ is $g(x_i)$. The casino picks a subset K of k' such indices with $I \in K$ at random. ¿From the above discussion, the casino sets each $a_i = 1$ with probability at least $(\epsilon - q)/2$, and I is uniformly distributed in K. Thus, the payoff of the casino when the strategy S from Theorem 2.10 is used is at least $1/2 + (\epsilon - q)/4 - 2^{k'/3}$. Also, given that $x_1..x_k$ has at least k' elements of H, this payoff is equal to the probability that $F(x) = g(x)$, the expected success probability for F. Since the probability that $x_1, ..x_k$ does not have its quota of elements from H is at most $q/2$, the expected success probability for F is at least $1/2 + \epsilon/4 - 3q/4 - 2^{k'/3}$, and so the expected advantage is at least $\epsilon/2 - 3q/2 - 22^{k'/3}$, as claimed. □

Let $H = \{x | Prob_F(F(x) = g(x)) < 1/2 + \epsilon/32\}$. Since the expected success probability of F on H is the average of the probabilities used to define H, $Exp(Adv_F^H(g)) < \epsilon/16$. Thus, from Lemma 2.11, $|H| \leq \delta 2^n$. Standard arguments show that for t as mentioned earlier, if $F_1, ..F_t$ are chosen independently, $Maj(F_1, ...F_t)(x) = g(x)$ for all $x \notin H$ with high probability. Thus, there are $F_1, ..F_t$ so that $Maj(F_1, ..F_t)$ achieves success probability $1 - \delta$. This function will have circuit size $O(ts') = s' n^{O(1)} \epsilon^{-O(1)}$ as claimed. □

3 Direct product lemmas for pseudo-random distributions

In this section, we present some general conditions on a pseudo-random distribution on $x_1, ..x_k$ so that the proof of the direct product theorem from the last section goes through basically unchanged. Note that the above argument could be simplified somewhat if we applied Theorem /refhcset to get a fixed hard-core set H rather than having the same strategy work for all such sets. However, the above argument also works in the uniform cryptographic setting, as well as the non-uniform circuit complexity setting. In the generalizations, it will be helpful to have H be the fixed hard-core set, since these will only hold for non-uniform complexity anyway.

Note that the argument from the last section basically used the independence of the x_i in two ways;

- With very high probability, the number of $x_i \in H$ is at least some minimal value k'.
- It is possible (and computationally feasible) to fix the values of all the $x_J, J \neq I$ while leaving x_I free to be any value.

Of course the last condition really does force true independence. However, if instead of x_J being *completely fixed*, x_J could be restricted to a *relatively small set of possibilities*, we could still store $g(x_J)$ for each possible x_J. This motivates the following definitions:

Definition 3.1 *Let $G_n : \{0,1\}^m \rightarrow (\{0,1\}^n)^k$ be a polynomial time computable pseudo-random generator. We say is* M-marginalizable *if there is a polynomial-time computable function $h(i, x, \alpha), h : [1, ..n] \times \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}^m$, satisfying:*

- *For any i and uniformly chosen x,α, $h(i, x, \alpha) \in \{0,1\}^m$ is uniformly distributed.*
- *For any inputs i, x, α, let $G(h(i, x, \alpha) = x_1, ..x_k$. Then $x_i = x$.*
- *For any inputs i, x, α, let $G(h(i, x, \alpha) = x_1, ..x_k$. Then for any $i, j \neq i, \alpha$, there is a set $S \subseteq \{0,1\}^n$, $|S| \leq M$, so that for any value of x, $x_j \in S$.*

We also need to quantify the first, statistical use of independence:

Definition 3.2 *We say that a distribution π pn $(\{0,1\}^n)^k$ is k', q, δ-hitting if for any sets $H_1, ..H_k \subseteq \{0,1\}^n, Prob_\pi[|\{i|x_i \in H_i\}| < k'] \leq q$.*

One nice feature of the above definitions is that, if we can find two pseudo-random generators that satisfy them separately, we can merely take the bitwise XOR of the two to get a single marginalizable, hitting distribution. This is formalized as follows:

Definition 3.3 *Let $x \oplus y$ denote the bit-wise xor of the two strings. For pseudo-random generators $G(r), G'(s)$ with outputs in $(\{0,1\}^n)^k$, let $G \oplus G'(r, s)$ denote the function $x_1 \oplus y_1, ..., x_k \oplus y_k$ where $x_1, ..x_k = G(r)$ and $y_1, ..y_k = G'(s)$.*

Lemma 3.4 *Let $G(r), G'(s)$ be polynomial-time pseudo-random generators with outputs in $(\{0,1\}^n)^k$.*

- *If G is (k', q, δ)-hitting then $\pi \oplus \rho$ is (k', q, δ)-hitting.*

- *If G is M-marginalizable then $\pi \oplus \rho$ is M-marginalizable.*

Proof.

- Let $H_1, ..H_k \subseteq \{0,1\}^n, |H_i| \geq \delta 2^n$ be given, and fix $y_1, ..y_k$. Let $H_i' = H_i \oplus y_i$; clearly, $|H_i'| = |H_i|$. Then $x_i \oplus y_i \in H_i \iff x_i \in G_i$, so applying the hitting condition for G to $H_1', , ..H_k'$ gives us a bound on the hitting probability of $G \oplus G'$.

- Let h be the marginalizing function for G. Let $G(s) = y_1, ..y_k$. Then let $h'(i, x, \alpha \circ s) = (h(i, x \oplus y_i, \alpha), s)$. It is easy to check that h' is a marginalizing function for $\pi \oplus \rho$.

We now show that these conditions are sufficient for a direct product theorem to hold for a generator:

Theorem 3.5 *If $G(r) : \{0,1\}^m \rightarrow (\{0,1\}^n)^k$ is $(.5\delta k, q, \delta)$-hitting, and is M-marginalizable, then for any s, G is a $(s, O(sq^2/(Mn^{O(1)}), 3q + 2 * 2^{-\delta k/6}, \delta)$-direct product generator.*

Proof. Let $\epsilon = 5q + 4 * 2^{-\delta k/6}$, We give a probabilistic construction which takes a circuit C that has success probability ϵ for $g^{(k)} \circ G$ and outputs a circuit F whose expected advantage on any H is at least q.

The construction is based on the strategy for the variant card guessing game, and is similar to the previous section. We pick I from $1, ..k$ uniformly at random, and select α_0 at random. Let $x_1, ..x_k = G(h(I, x, \alpha_0))$, For each $j \neq I$, we construct (non-uniformly) a table of $g(x_j)$ for any x_j that is a possible output of $h(I, x, \alpha_0)$ (as we vary over x). There are at most M such values. Our circuit F, on input x simulates C on $h(I, x, \alpha_0)$. It then (via the tables) computes t, the number of indices $j \neq i$ where the j'th output of C differs from $g(x_j)$. F outputs the i'th output of C with probability 2^{-t}, and otherwise outputs a randomly chosen bit.

Lemma 3.6 *Let $H \subseteq \{0,1\}^n, |H| \geq \delta 2^n$. $Exp(Adv_F^H(g)) \geq q$*

Proof. Let $x \in H$ be uniformly chosen. Let s be the number of $x_i \in H$. As in the previous section, the probability of a sequence $x_1, ..x_k$ given that $x \in H$ is $s/(\delta k)$ times its probability of being output by $G(r)$. The probability that the value of s for $G(r)$ is at least $.5\delta k$ and that $C(r) = g^k(G(r))$ is at least $\epsilon - q$ from the hitting property and the success probability of C being $5q$. Thus, the corresponding probability for $x_1, ..x_k$ constructed by F on $x \in H$ is at least $(\epsilon - q)/2$, since each such sequence has at least 1/2 its probability under $G(r)$. Also, $r = h(I, x, \alpha_0)$ is independent of I so given r (and hence $x_1, ..x_k$), I is a uniformly distributed index so that $x_I \in H$.

So again, the expected success of F on $x \in H$ given that $s \geq .5\delta k$ is the same as the expected advantage of S on the variant game with $k' = .5\delta k$ when the casino sets $a_j = 1$ iff the j'th output of C is $g(x_i)$, and picks K a random set of k' positions where $x_i \in H$. This last is at least $1/2 + (\epsilon - q)/4 - 2^{k'/3} = q$ So the expected success probability of F is at least $(1 - q)(1/2 + q) \geq 1/2 + q/2$, since $s \geq k'$ with probability at least $1 - q$, which gives the claimed advantage. Thus, there is at most a δ fraction of x where F has expected probability $< 1/2 + q/2$, and taking the majority of approximately n/q^2 circuits F gives a circuit that predicts $g(x)$ on all x not in this fraction. Each F has size approximately $O(|C| + kM)$, so the combined circuit has size $O((|C| + kM)n/q^2)$. Thus, if $|C| \leq s'$, there is a circuit of size $\leq s$ that has success probability at least $1 - \delta$, a contradiction.

Definition 3.7 *A distribution π on $(\{0,1\}^n)^k$ is called* t-good *if it satisfies the following conditions.*

1. *The marginals x_i are uniformly distributed (can be generalized).*
2. *For every set $H \in \{0,1\}^n$ with $|H| = \delta 2^n$, $Pr[|\{i : x_i \in H\}| < t] < 2^{-\Omega(t)}$.*
3. *For every $i \in [k]$, every fixed α and every $j \neq i$ $h_i(\alpha, x)$ depends on at most t bits of x.*

Theorem 3.8 *Assume that $SUC_s^{\pi}(g^{(k)}) \geq p$ for some t-good distribution π. Then for every set $H \subseteq \{0,1\}^n$ with $|H|/2^n = 1/3$, $ADV^H_{s2^{O(t)}}(g) > p/3 - 2^{O(t)}$ (and thus, by the hard-core set theorem, $SUC_{s2^{O(t)}}(g) > 2/3$).*

Proof. (Sketch) By the above discussion, we can repeat all the steps of the above proof of the XOR lemma with our good distribution π. The only point to add is that the third property allows the nonuniform construction of the function values $g(x_j)$ as functions of x_I as truth tables of size 2^t.

4 First Generator

(THis is stuff we never discussed, that is what solves our problem of amplification for every circuit size - please read carefully).

In this section we present a generator which uses $O(n + k^2)$ bits to generate k strings which "fool" the XOR lemma. Note that $O(n)$ bits suffice for optimal amplification against circuits of size at most $2^{\sqrt{n}}$. First however, we decribe the proerties of the well known "pairwise independent" generator of Chor and Goldreich [**?**].

Let F be a finite field, and (wlog) let $1, 2, \cdots, r$ be distinct elements of F. For every two elements $a, b \in F$ define the sequence $\bar{y}(a, b) = y_1, \cdots y_r$ of r elements from F by $y_i = a + ib$ for all $i \in [r]$.

Theorem 4.1 *[**?**] Let T be an arbitrary subset of F with $|T|/|F| = \rho$. Say that a pair (a, b)* estimates *T if $||\{i \in [r] : y_i \in T\}|/|F| - \rho| \leq r^{1/4}$ If a, b are chosen uniformly at random from F, then the probability that (a, b) do not estimate T is at most $r^{1/4}$.*

Let $F = GF(2^n)$ and given ϵ set $k = 4 \log \epsilon$ and $r = 2^k$. Our first XOR generator is $XG1 : (\{0,1\}^n)^2 \times (\{0,1\}^k)^k \rightarrow (\{0,1\}^n)^k$, defined by $XG1((a, b); (j_1, \cdots, j_k)) = (x_1, \cdots x_k)$ with $x_i = a + bj_i$.

Theorem 4.2 *$XG1$ is an efficient (s, ϵ) XOR generator of seed length $O(n + k^2)$.*

Proof. (Sketch) The only thing to prove, by Theorem 3.8, is that the output distribution of $XG1$ is $\Omega(k)$-good. The first condition is trivial, and the third satified with $t = 0$. For the second condition, by Theorem 4.1, all but 2^{-k}-fraction of the pairs (a, b) estimate any set to within additive error 2^{-k}, so neglecting such error we can take $t = k$ here. Finally we assume that g has an ϵ^4 hard-core set for size s, to conclude the proof.

The intuition behind this proof can be stated as follows. Somehow the savings in the number of bits occured by flipping (aside from the $O(n)$ bits for (a, b)) only k independent k-bit vectors instead of n-bit vectors. This effective reduction of input size is achieved by defining, (conceptually,) for any function $g : \{0,1\}^n \to \{0,1\}$ and $a, b \in \{0,1\}^n$ a function $g_{(a,b)}\{0,1\}^k \to \{0,1\}$. Theorem 4.1 asserts that in H is a hard-core set for g, then for most (a, b) the set $H \cap y(a, b)$ is a hard core set for $g_{(a,b)}$. For each of these, we use only the XOR lemma on independent inputs.

5 The Second Generator

In this section we present a generator which uses $O(n^2/k)$ bits to generate k strings which "fool" the XOR lemma for $s > 2^k$. Note that $O(n)$ bits suffice for optimal amplification against circuits of size at least $2^{\Omega(n)}$. Before descibing it we again recall properties of two well known generators, which produce k vectors of n bits each from a short seed, and their (pseudo) random properties.

5.1 Expander Walks

Definition 5.1 *Fix an explicit expander graph G on vertex set $\{0,1\}^n$, of constant degree (say 16) and small second eigenvalue (say < 8) [?, ?]. Let the generator $EW : \{0,1\}^n \times [16]^{k-1} \to (\{0,1\}^n)^k$ be defined by $EW(v; \bar{d}) = (v_1, v_2, \cdots, v_k)$ where $v_1 = v$ and v_{i+1} is the d_ith neighbour of v_i in G (here $d = (d_1, \cdots, d_{k-1})$.*

The amplification property of this generator, formalized below, asserts that with exponentially high probability, many of these k pseudo random points will satisfy arbitrary large events.

Theorem 5.2 *[?, ?, ?] Let $T_1, \cdots, T_k \subseteq \{0,1\}^n$ be arbitrary subsets with $|T_i|/2^n \geq 1/3$. Then, $Pr[|\{i : v_i \in T_i\}| < k/10] < 2^{-k/10}$.*

5.2 Nearly Disjoint Subsets

Definition 5.3 *Set $m = 100n^2/k$. Let $p = \binom{m}{n}$ and encode (efficiently and 1-1) the n-subsets of m by the integers modulo p. Let the generator $ND : \{0,1\}^m \times (Z_p)^2 \to (\{0,1\}^n)^k$ be defined by $ND(x; (a, b)) = y_1, y_2, \cdots, y_k$, with $y_i \in \{0,1\}^n$ are the bits of x in the n-subset S_i (in the natural order), where S_i is the encoding of $a + ib(mod p)$.*

Lemma 5.4 *$Pr[\exists i < j : |S_i \cap S_j| > k/10] < k^2 2^{-k}$ (and thus at most $k/10$ positions of every pair of output vectors are shared, except with probability 2^{-k}).*

Remark 5.5 *Note that for $k < \sqrt{n}$ it is more efficient to take disjoint subsets (resulting in $m = kn < 100n^2/k$). However the case of most inerest for us is $k = \Theta(n)$, for which $m = O(n)$.*

Proof. (Sketch) The subsets S_i are pairwise independent, and for each pair the probability of intersection $> k/10$ is exponentially small.

5.3 The Generator XG2

The generator we use here to "fool" the XOR lemma is the simplest combination of the two generators above - we take the componentwise exclusive-or of their outputs.

Definition 5.6 *With the parameters above, we define the XOR generator $XG2(x; (a, b); v; \bar{d}) = (x_1, \cdots x_k)$ by $x_i = y_i \oplus v_i$, with $\bar{y} = DS(x; (a, b))$ and $\bar{v} = EW(v, \bar{d})$.*

Definition 5.7 *We will use the shorthand $z =< (a, b); v; \bar{d} >$ to denote all parts of the seed to $XG2$ except x, and let $G_z(x) = XG2(x, z)$. Fix a set $S \subseteq \{0, 1\}^n$ with $|S|/2^n > 1/3$. Call z* good *if the following two conditions hold.*

1. *The subsets T_i defined by (a, b) have pairwise intersection $< k/10$.*
2. *$Pr[|\{i : x_i \in S\}| < k/10] < 2^{-k/10}$, where x_i are the output of $G_z(x)$ for a uniformly chosen $x \in \{0, 1\}^m$.*

Call z bad *otherwise.*

The main property of the generator $XG2$ we shall use in the proof is

Lemma 5.8 *For every fixed subset $S \subseteq \{0, 1\}^n$ with $|S|/2^n > 1/3$, $Pr[zbad] < 2^{-k/20}$.*

Proof. The failure bound of $k^2 2^{-k}$ for the first property (of small intersections) follows directly from lemma 5.4. The second is argued as follows. Fix $x; (a, b)$ and define $S_i = S \oplus y_i$ (namely add y_i to every element of S). Applying lemma 5.2 we find that only $2^{-k/10}$ fraction of choices of $v; \bar{d}$ cause the output of XG to violate $|\{i : x_i \in S\}| < k/10$. Changing order of summation (fixing $v; \bar{d}$ and freeing x we get the desired bound.

Theorem 5.9 *The generator $XG2$ is an efficient generator that fools (s, ϵ) XOR tests with $s, 1/\epsilon > 2^1 0k$. It uses a seed of length $O(n + k + min\{kn, n^2/k\}$ bits.*

6 Final Genrator

The idea is to define first a variant or $XG1$ where instead of k independent samples in $\{0,1\}^k$ we use the ND generator. THis means that in the ND construction we need only $O(k)$ bits. The output of this variant is exclusive-or with the output of EW, just like in the construction of $XG2$. Call this generator XG

Theorem 6.1 *The generator XE with $k = -10 \log \epsilon$ is efficient and fools (s, ϵ)-XOR tests for every s, ϵ. It uses seed length $O(n + k)$ bits, which is optimal up to a constant factor.*

References

[1] [BFNW] L. Babai, L. Fortnow, N. Nisan and A. Wigderson, "BPP has Subexponential Time Simulations unless EXPTIME has Publishable Proofs", *Complexity Theory*, to appear.

[2] M. Blum and S. Micali. "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits", *SIAM J. Comput.*, Vol. 13, pages 850–864, 1984. Preliminary version in *FOCS*, 1982.

[3] O. Goldreich. *Foundation of Cryptography – Class Notes*, Spring 1989, Computer Science Department, Technion, Haifa, Israel.

[4] O. Goldreich. *Foundation of Cryptography – Fragments of a Book*, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel, February 1995. Appears as an ECCC book, 1995.

[5] O. Goldreich and L.A. Levin. "A Hard-Core Predicate for all One-Way Functions", in *ACM Symp. on Theory of Computing*, pp. 25–32, 1989.

[6] O. Goldreich, N. Nisan and A. Wigderson. "On Yao's XOR-Lemma", unpublished manuscript, March 1995. ECCC TR95-050, 1995.

[7] S. Goldwasser. Probabilistic Encryption: Theory and Applications, Ph.D. Thesis, UC-Berkeley, 1984.

[8] S. Goldwasser and S. Micali. "Probabilistic Encryption", *JCSS*, Vol. 28, pages 270–299, 1984. Preliminary version *STOC*, 1982.

[9] J. Hastad, R. Impagliazzo, L.A. Levin and M. Luby, "Construction of Pseudorandom Generator from any One-Way Function", to appear in *SICOMP*. See preliminary versions by Impagliazzo et. al. in *21st STOC* and Hastad in *22nd STOC*.

[10] [Im] R. Impagliazzo, "Hard-core Distributions for Somewhat Hard Problems", in *36th FOCS*, pages 538–545, 1995.

[11] [L87] L.A. Levin, "One-Way Functions and Pseudorandom Generators", *Combinatorica*, Vol. 7, No. 4, 1987, pp. 357–363.

[12] [L86] L.A. Levin, "Average Case Complete Problems", *SICOMP*, Vol. 15, 1986, pp. 285–286.

[13] [NRS] N. Nisan, S. Rudich, and M. Saks. "Products and Help Bits in Decision Trees", in *35th FOCS*, pages 318–329, 1994.

[14] N. Nisan and A. Wigderson. "Hardness vs. Randomness", *Proc. 29th FOCS*, pages 2–11, 1988.

[15] C. Rackoff, private communication, 1985.

[16] A.C. Yao, "Theory and Application of Trapdoor Functions", in *23st FOCS*, pages 80–91, 1982.

[17] [Z] Zuckerman, D., "Security Preserving Amplification of Harness", 31^{st} *FOCS*, 1990, pp. 318-326.

[GL] Goldreich, O., and L.A. Levin, "A Hard-Core Predicate for any One-way Function", 21^{st} *STOC*, 1989, pp 25-32.

[L] Levin, L.A., "One-way Function and Pseudorandom Generators", *Combinatorica*, Vol. 7, No. 4, 1987, pp. 357-363.

[Li] R. Lipton, "New directions in testing", In J. Fegenbaum and M. Merritt, editors, *Distributed Computing and Cryptography*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science Volume 2, pp. 191-202. American Mathematical Society, 1991.

[LY] R. Lipton and N. Young, "Simple Strategies for Large Zero-Sum Games with Applications to Complexity Theory" , 26^{th} *STOC*, 1994, pp.734-740.

[Ni] N. Nisan, personal communication, 1994.

[NRS] N. Nisan, S. Rudich M. Saks, "Products and Help Bits in Decision Trees", 35^{th} *FOCS* 1994, pp. 318-329.

[NW] N. Nisan and A. Wigderson, Hardness vs. Randomness, Journal of Computer and System Sciences, , vol. 49, No. 2 (Oct., 1994), pp. 149-67.

[vN] J. von Neumann, "Zur Theorie der Gesellschaftspiel", Mathematishe Annalen, 100(295-320),1928.