COMBINATORICA 7 (4) (1987) 375—392

STRONG COMMUNICATION COMPLEXITY OR GENERATING QUASI-RANDOM SEQUENCES FROM TWO COMMUNICATING SEMI-RANDOM SOURCES

U. V. VAZIRANI

Received 1 September 1985
Revised 29 October 1986

The semi-random source, defined by Sántha and Vazirani, is a general mathematical mode for imperfect and correlated sources of randomness (physical sources such as noise diodes). In this paper an algorithm is presented which efficiently generates "high quality" random sequences (quasi-random bit-sequences) from two independent semi-random sources. The general problem of extracting "high quality" bits is shown to be related to communication complexity theory, leading to a definition of strong communication complexity of a boolean function. A hierarchy theorem for strong communication complexity classes is proved; this allows the previous algorithm to be generalized to one that can generate quasi-random sequences from two communicating semi-random sources

1. Introduction

Studying the fundamental connections between randomness and computation has been an important theme in contemporary theoretical computer science research. A large part of this research has focussed on using a source of independent unbiased bits to speed up such diverse computational tasks as combinatorial algorithms, synchronization and deadlock resolution protocols, encrypting data and cryptographic protocols. Yet, whether or not such a source can be constructed is an open question — even assuming Quantum Physics, which postulates random outcomes in elementary events. The major difficulty is that making any measurement disturbs the system — thus the bit-sequence resulting from a series of measurements might be correlated. This is not just a philosphical concern: the available physical sources of randomness, such as Zener diodes and geiger counters, suffer seriously from this imperfection of correlations. Sántha and Vazirani [11] proposed a mathematical framework in which this problem could be studied: their *semi-random source* provides a general model for such correlations. In this paper we work within this mathematical framework and prove that under very general conditions such semi-random sources can be algorithmically transformed into "high quality" random sources.

Let S be a source, and let $b_1 b_2 \ldots b_i b_{i+1} \ldots$ denote its output. Say that S satisfies the δ-condition, for some constant $0 < \delta < 1$, if $\delta \leqq \Pr[b_k = 1 | b_1 b_2 \ldots b_{k-1}] \leqq 1 - \delta$

This research was supported in part by an IBM Doctoral Felloswhip and by NSF Grant MCS 82-04506.

AMS subject classification: 94 A 05.

for every k. A δ semi-random source is obtained by allowing a malicious adversary to pick a source among all sources satisfying the δ-condition. An equivalent but more graphic description of the semi-random source is to regard it as a coin of variable bias controlled by an adversary; the bias of the next coin flip is set by the adversary based on the previously generated sequence. The only condition imposed on the adversary is that the bias of the coin must lie between δ and $1-\delta$. The worst case nature of the correlations make this a general model while abstracting away complicated details about the actual correlations. Now the problem of obtaining "high quality" randomness can be precisely stated:

Say that two sources S_1 and S_2 producing output distributions D_1 and D_2 are independent if D_1 and D_2 are independent distributions (i.e. the two physical sources are shielded from each other so that the output of one cannot affect the output of the other). We give an algorithm which produces "high quality" randomness from the outputs of independent semi-random sources S_1 and S_2. The algorithm is efficient, both in the number of semi-random bits used per output bit and computational time. It is also easy to implement in hardware. In contrast to previous empirical and engineering methods, our results suggest the following radically new scheme for generating "high quality" random sequences: sample the imperfect physical noise source very fast to implement semi-random sources; then use the algorithm described here, thereby maintaining a fast rate of "high quality" output bits.

The core of the above algorithm lies in extracting a single "almost unbiased" bit from the outputs of the two independent semi-random sources. In the second part of the paper, a basic connection is established between this unbiasing ability of a boolean function (when its two arguments are independent semi-random sequences) and its distributional communication complexity. This connection leads naturally to the definition of strong communication complexity. We prove a hierarchy theorem for strong communication complexity classes. One consequence of this theorem is that the independence condition imposed on the two semirandom sources for the "high quality" randomness extracting algorithm can be relaxed by allowing the two sources to communicate. In practical terms this means that the two noise diodes need not be perfectly shielded from each other. Another consequence of the hierarchy theorem is that it yields an $\Omega(n/\log n)$ lower bound on the distributional communication complexity of the inner product function, improving the previous $\Omega(\log^2 n)$ lower bound obtained by Yao [25].

2. Two Independent Sources

First we review some definitions and theorems from the paper of Santha and Vazirani [16]:

Definition [Santha and Vazirani]. A probability distribution on $\{0, 1\}^\omega$ is *semi-random* if there exists a constant δ, $0<\delta<1/2$, such that for every i and for every string s of length $i-1$:

$$\delta \leqq \Pr[x_i = 1 | x_1 \ldots x_{i-1} = s] \leqq 1-\delta.$$

A *semi-random source* is one whose output probability distribution is semi-random.

It is convenient to regard a semi-random source as a process, controlled by an adversary, which generates a sequence of 0's and 1's by flipping a coin of variable bias. The bias of the next coin-flip depends upon the outcome of the previous coin-flips. We shall call the function describing this dependence the strategy adopted by the adversary. More formally,

Definition. A *strategy* for the adversary is a map T: $\{0, 1\}^* \to [\delta, 1-\delta]$, where $T(x) =$ $=$ Pr [next bit is 1| the string x has been output so far].

The adversary nature of the semi-random source makes it impossible to obtain a sequence of absolutely unbiased bits from the outputs of such sources (a precise lower-bound on the bias can be found in Theorem 6 [16]). However, it is possible to obtain sequences whose distribution is so close to the uniform distribution that there is provably no loss in performance of algorithms that use such sequences in place of coin flips. This is made precise in the technical definition given below:

Definition [Santha and Vazirani]. Let S be a source which on input n, generates n-length strings $x \in \{0, 1\}^n$ with probability $p_n(x)$. S is *quasi-random* if for every $t > 0$, and for sufficiently large n: $\sum_{|x|=n}' |p_n(x) - 1/2^n| < 1/n^t$.

To prove quasi-randomness of a source, it is more convenient to show that it satisfies the condition in the following theorem:

Theorem 2.1. [Santha and Vazirani]. *Let S be the following "high quality" source: on input n, it generates an n-length string $x \in \{0, 1\}^n$ according to probability distribution D_n satisfying: the probability that the k^{th} output bit is 1, conditioned on the first $k-1$ output bits lies between $1/2 - \varepsilon(n)$ and $1/2 + \varepsilon(n)$. If for every $t > 0$ and for sufficiently large n, $\varepsilon(n) < 1/n^t$, then S is quasi-random.*

In this paper, we show how to extract quasi-random sequences from two independent semi-random sources. This is best possible because no algorithm can extract a better than $1-\delta$ biased bit from a single semi-random source:

Theorem 2.2. [Sántha and Vazirani]. *Let f: $\{0, 1\}^m \to \{0, 1\}$ denote any arbitrary boollean function. Then there is an adversary strategy for the δ semi-random source such that* $\Pr[f(x) = 1]$ *does not lie in the open interval* $(\delta, 1-\delta)$.

Extracting a single „high quality" bit:

Let S_1 and S_2 be two independent δ semi-random sources. Let A_1 and A_2 be the adversaries associated with the two sources. Let x and y denote k-bit outputs of S_1 and S_2. The "high quality" bit extraction problem is the following:

> Exhibit a boolean function b: $\{0, 1\}^* \times \{0, 1\}^* \to \{0, 1\}$ such that for every $\varepsilon > 0$ there exists k such that $b(x, y)$ is at most ε biased. i.e. for every δ and for every $\varepsilon > 0$, there exists k such that for every pair of independent δ semirandom sources S_1 and S_2 $|\Pr[b(x, y) = 1] - 1/2| \leqq \varepsilon$, where x and y are k-bit outputs of S_1 and S_2.

Example. Let $b(x, y)$ be the xor of the bits of x and y. Then A_1 and A_2 have strategies such that the probability that the output $b(x, y)$ is 0 is at least $1-2\delta$ independent of k

(recall that $|x|=|y|=k$). Both adversaries follow the following simple strategy: the first $k-1$ bits are independent and unbiased. The adversary computes the xor, u, of the $k-1$ bits that he has generated so far, and for the k^{th} bit he flips a coin which has $1-\delta$ bias towards u. Each adversary succeeds in setting the xor of his bits to 0 with probability $1-\delta$. If both adversaries succeed then $b(x, y)$ is 0 and this happens with probability exceeding $1-2\delta$.

Why did the xor function fail? The problem with the xor function is that the effect of x on the xor of x and y can be expressed in just one bit: the xor of the bits of x. i.e. in other words the communication complexity of the xor function is just one bit. This relationship between extracting good bits from two sources and communication complexity is formalized in Chapter 3.

We prove below that the inner product function produces a high-quality bit from the outputs of two independent sources.

Definition. The *inner product function*, $b(x, y)=\vec{x}\cdot\vec{y}$, where $\vec{x}$ and $\vec{y}$ are just x and y represented as bit vectors, and $\cdot$ denotes GF(2) inner product.

Let $|x|=|y|=k$. $b(x, y)$ is 0 if the number of bit positions in which both x and y are 1 is even, and $b(x, y)$ is 1 if this number is odd. The following intuitive argument (and its shortcommings) might be helpful in understanding why the inner product function works: although each adversary knows the positions in which he has generated 1's so far, he does not know how many of them will pair up with 1's from the other source and how many with 0's. Therefore, the direction of bias of the next bit as dictated by his strategy is almost independent of the inner product of the sequences generated so far. Thus the effect of the $k+1$st bit is to xor the answer obtained so far with an independent bit which has bias between δ^2 and $1-\delta^2$. If such an inductive argument could be formalized, it would show that $b(x, y)$ is at most $(1-2/\delta^2)^k$ biased.

However, the inner product function works for a more subtle reason than the above intuition would suggest. The above argument is flawed in that it does not take into account the possibility that the adversaries can implicitly decrease their uncertainty about the effect of the previously generated bits. As an extreme example, consider the modified scenario in which the first $k/2$ bits of each source are generated by the flips of a fair coin; the next $k/2$ bits are then deterministically output by the adversary, who can of course see the first $k/2$ bits. In this setting, do the adversaries have a pair of matching strategies to bias the inner-product function? Certainly, after the first $k/2$ bits of x and y are generated, neither adversary has any idea about the inner-product of the bits generated so far. The intuitive argument from the previous paragraph would now suggest that the adversaries have no strategies that could bias the output even slightly. Yet the two adversaries have strategies which *always* set $b(x, y)$ to 0! Let u be the $k/2$ bits that were generated by the fair coin fllips. The adversary simply outputs the string u again (deterministically)! Clearly, any proof that the inner product function works must show that in the case that the control of the adversaries over the output bits is never absolute, there are no such clever strategies for the adversaries which allow them to implicitly control the bias of the output.

Indeed our proof is along very different lines than the above intuition might suggest. Fix any strategy T for adversary A_1. We show that there is no corresponding strategy for adversary A_2 to achieve more than ε-bias in $b(x, y)$.

Definition. The *label* of y with respect to strategy T, $label_T(y) = \Pr[b(x, y) = 1 | x$ was produced using strategy $T]$.

Definition. The *potency* of y with respect to strategy T, $\Delta_T(y) = label_T(y) - 1/2$. y is *ε-potent* with respect to strategy T if its potency is greater than ε.

The proof is divided into two parts. First we prove an upper bound on the number of $(\varepsilon/2)$-potent strings with respect to T. The rest of the strings can contribute at most an $\varepsilon/2$ bias to $b(x, y)$. In the second part of the proof, we show that even for the placement of the potent strings which is most favourable to the adversary, his probability of hitting an $\varepsilon/2$-potent string is at most $\varepsilon/2$.

Main Lemma. *The number of $(\varepsilon/2)$-potent strings with respect to any k-bit strategy is at most $(1+(1-2\delta)^m)^k/(\varepsilon/2)^m$ for every even, positive integer m.*

Proof. We introduce the following measure of the aggregate potency of the k-bit strings with respect to T:

Definition. Let m be any positive, even integer. The *m-potential* of a k-bit strategy T is defined to be $\sum_{|y|=k} \Delta(y)^m$ where $\Delta(y)$ is the potency of the string y with respect to T.

This potential function has two useful properties:

1) There is an upper bound on the potential function that increases very slowly as a function of k. Thus the average contribution of each k-bit string to the potential function decreases exponentially with k.

2) If the potential function of strategy T is sufficiently small, there are very few $(\varepsilon/2)$-potent strings with respect to strategy T.

We prove the upper bound on the m-potential function by induction on k. Let γ denote the probability that the first bit output by A_1 is 1, $\delta \leqq \gamma \leqq 1-\delta$. Let T_1 denote the $k-1$ bit strategy used if the first bit is 1, and T_0 denote the $k-1$ bit strategy used if the first bit is 0. Consider the strings $y = 0z$ & $y' = 1z$, where z is a $k-1$ bit string. Let the label of z with respect to the strategy T_0 be $1/2+\Delta_0$, and let the label of z with respect to the strategy T_1 be $1/2+\Delta_1$. Then the label of $0z$ w.r.t. T is $(1-\gamma)(1/2+\Delta_0)+\gamma(1/2+\Delta_1) = 1/2+(1-\gamma)\Delta_0+\gamma\Delta_1$. And the label of $1z$ w.r.t. T is $(1-\gamma)(1/2+\Delta_0)+\gamma(1/2-\Delta_1) = 1/2+(1-\gamma)\Delta_0-\gamma\Delta_1$.

We express below the the m-potential of strategy T in terms of the sub-strategies T_0 and T_1:

$$m\text{-potential}(T) = \sum_{|z|=k-1} \left((1-\gamma)\Delta_0+\gamma\Delta_1\right)^m + \left((1-\gamma)\Delta_0-\gamma\Delta_1\right)^m =$$

$$= 2\sum_{|z|=k-1} \left((1-\gamma)^m\Delta_0^m + \binom{m}{2}(1-\gamma)^{m-2}\gamma^2\Delta_0^{m-2}\Delta_1^2 + \ldots + \gamma^m\Delta_1^m\right) =$$

$$= 2(1-\gamma)^m \sum_{|z|=k-1} \Delta_0^m + 2\binom{m}{2}(1-\gamma)^{m-2}\gamma^2 \sum_{|z|=k-1} \Delta_0^{m-2}\Delta_1^2 + \ldots + 2\gamma^m \sum_{|z|=k-1} \Delta_1^m.$$

The previous step is justified since γ is independent of z. In the two extreme terms, the factors $\sum_{|z|=k-1} \Delta_0^m$, and $\sum_{|z|=k-1} \Delta_1^m$ are the m-potentials of the $k-1$ bit

strategies T_0 and T_1 respectively. However, the factors in the middle terms $\sum_{|z|=k-1} \Delta_0^{m-2i}\Delta_1^{2i}$ have no nice interpretation. We use the following generalization of Schwartz's Inequality (see Holder's Inequality [15]): if m is even, then $\sum_i a_i^{m-1}b_i \leqq$ $\leqq \max\left[\sum_i a_i^m, \sum_i b_i^m\right]$:

Thus m-potential $(T) \leqq$

$$\leqq 2\left((1-\gamma)^m + \binom{m}{2}(1-\gamma)^{m-2}\gamma^2 + \ldots + \gamma^m\right)\max\left(\sum_{|z|=k-1}\Delta_0^m, \sum_{|z|=k-1}\Delta_1^m\right) =$$

$$= \left(((1-\gamma)+\gamma)^m + ((1-\gamma)-\gamma)^m\right)\max\left(\sum_{|z|=k-1}\Delta_0^m, \sum_{|z|=k-1}\Delta_1^m\right) =$$

$$= \left(1+(1-2\gamma)^m\right)\max\left(m + \text{potential}\,(T_0),\ m\text{-potential}\,(T_1)\right).$$

Let m-potential (k) denote the maximum over all k-bit strategies of m-potential (T).

Then m-potential $(T) \leqq \left(1+(1-2\gamma)^m\right)$ m-potential $(k-1) \leqq \left(1+(1-2\delta)^m\right)$ m-potential $(k-1)$.

The previous step follows from the definition of m, and the fact that γ lies between δ and $1-\delta$. Since the above inequality holds for every k-bit strategy T, it follows that

$$m\text{-potential}\,(k) \leqq \left(1+(1-2\delta)^m\right)\ m\text{-potential}\,(k-1).$$

Hence m-potential $(k) \leqq \left(1+(1-2\delta)^m\right)^k$.

Each $(\varepsilon/2)$-potent string y contributes at least $(\varepsilon/2)^m$ to the potential function.

$$\text{Therefore the number of } (\varepsilon/2)\text{ potent sequences} \leqq \frac{m\text{-potential}\,(k)}{(\varepsilon/2)^m} \leqq$$

$$\leqq \frac{\left(1+(1-2\delta)^m\right)^k}{(\varepsilon/2)^m}. \quad \blacksquare$$

The next two lemmas are useful for the second part of the proof. We omit their proofs, which are elementary.

Call a strategy *extreme* if each bit is output by the flip of a δ or $1-\delta$ biased coin.

Lemma 1. *For every set B, there is an extreme strategy for the adversary, which maximizes his chances of generating a string from the set B.* ∎

Lemma 2. *Let T and T' be any extreme strategies. Let B be any set of strings. Then there is a set B' such that $|B| = |B'|$, and the probability that T' hits B' is equal to the probability that T hits B.* ∎

Theorem 2.3. *Let $k = (16/\delta^2)\log 1/\varepsilon$. Then for every A_1 strategy and for every A_2 strategy, $b(x, y)$ has bias in the range $[1/2-\varepsilon, 1/2+\varepsilon]$.*

Proof. The main lemma provides an upper bound on the number, U, of $(\varepsilon/2)$-potent strings. It suffices to prove that the adversary A_2 has no more than an $(\varepsilon/2)$ chance

of generating an $(\varepsilon/2)$ potent string, since the rest of the strings contribute no more than $(\varepsilon/2)$ to the bias of $b(x, y)$. By lemma 1 and lemma 2, we may fix A_2's strategy to be "always bias $1-\delta$ towards 0". Then the source is simply a biased coin. It remains to show that the U most likely k bit sequences produced by flips of this biased coin have total probability at most $(\varepsilon/2)$. Let $s=s_1 s_2 \ldots s_k$ denote the sequence of k tosses of the biased coin. Then the probability that the k coin tosses produce s is:

$$(1-\delta)^{k-\Sigma s_i}\delta^{\Sigma s_i}.$$

The U most likely k bit strings are the U strings with the fewest 1's in their binary expansion. To complete the proof of the theorem, we prove the following two claims:

Claim 1. $U \leqq \left|\{s:|s|=k \text{ and } \sum s_i \leqq \delta k/2\}\right|$.

Proof. Choose $m=(2/\delta)\log(1/\delta)$, and apply the main lemma to obtain:

$$|U| \leqq \frac{(1+(1-2\delta)^m)^k}{(\varepsilon/2)^m}.$$

First we show that the right hand side: $(1+(1-2\delta)^m)^k/(\varepsilon/2)^m \leqq (2/\varepsilon)^{2m}$.

This is so since: $k\log(1+(1-2\delta)^m) \leqq k\log(1+\delta^4) \leqq k\delta^4 = 16\delta^2\log\frac{1}{\varepsilon} \leqq$

$$\leqq \frac{2}{\delta}\log\frac{2}{\varepsilon} = m\log\frac{2}{\varepsilon}.$$

Finally we show that

$$\left(\frac{2}{\varepsilon}\right)^{2m} \leqq \binom{k}{\frac{\delta}{2}k}.$$

Use Stirling's formula to obtain

$$\binom{k}{\frac{\delta}{2}k} \geqq \frac{2^{k(\delta/2)\log(2/\delta)}}{2\sqrt{2\pi k}}.$$

Finally, it is easy to check the inequality by making the above substitution, taking $\log s$, and observing that $\delta \leqq 1/2$. Thus

$$|U| \leqq \frac{(1+(1-2\delta)^m)^k}{(\varepsilon/2)^m} \leqq (2/\varepsilon)^{2m} \leqq \binom{k}{\frac{\delta}{2}k}. \quad \blacksquare$$

Claim 2.

$$\Pr\left[\sum s_i \leqq \frac{\delta}{2}k\right] \leqq \frac{\varepsilon}{2}.$$

Proof. We use the following version of the Chernoff bound [14]: Let $X_1, X_2, \ldots, X_n$ be independent random variables with mean value p

$$\Pr\left[\sum_{i=1}^{n} X_i \leqq np-\alpha\right] \leqq e^{-(\alpha^2/2np)}.$$

Substituting $p=\delta$, and $n=k$, we have

$$\Pr\left[\sum s_i \leqq \frac{\delta}{2}k\right] \leqq e^{-\delta(k/8)} = e^{(-2/\delta)\log(1/\varepsilon)} \leqq 2^{-2\log(1/\varepsilon)} = \varepsilon^2 \leqq \varepsilon/2. \quad \blacksquare$$

Implementing a quasi-random source:

A straightforward implementation of a quasi-random source follows from the previous theorem:
On input n;

Let $n=\omega\left(\frac{1}{\delta^2}\log n\right)$.

Let $\alpha_1, \ldots, \alpha_n$ and $\beta_1, \ldots, \beta_n$ be n blocks, each of k bits, output by the two sources.

Output $\vec{\alpha}_1 \cdot \vec{\beta}_1, \ldots, \vec{\alpha}_n \cdot \vec{\beta}_n$.

end;

Proposition 2.4. *The source defined above is quasi-random.*

Proof. By Theorem 2.3, the above source is "high-quality", and thus by Theorem 2.1 it is quasi-random. ∎

The method described above extracts 1 bit from every pair of blocks of $k=\omega(1/\delta)\log n$ semi-random bits. Next we show how to extract $O(k)$ bits from each pair of blocks by using a construction of Wyner [22]. Wyner studied the problem of communicating in perfect secrecy in the presence of an eavesdropper whose wiretap channel is noisy; each communicated bit is correctly seen by the eavesdropper with probability $1-\delta$ and flipped with probability δ, $0<\delta<1/2$. He showed how to achieve secrecy even though the coding scheme is known to the eavesdropper.

The following simple method uses the parity function to achieve secrecy, but does not give a good rate of transmission: Encode each message bit m as a k-bit word $c_1c_2\ldots c_k$ picked uniformly among k-bit strings such that parity $(c_1c_2\ldots c_k)=m$. Clearly m is easy to recover from $c_1c_2\ldots c_k$. However, the eavesdropper sees the sequence $e_1e_2\ldots e_k$ where $e_i=c_i$ with probability $1-\delta$ and $e_i=\bar{c}_i$ with probability δ. It is not hard to see that $H(m|e_1e_2\ldots e_k)=(h/2)(1-(1-2\delta)^k)$ which tends to 1 as k tends to infinity. A rough analogy between these two situations may be made: the eavesdropper corresponds to the adversary; the noisy channel to the δ uncertainty in the outputs of the semi-random sources; the use of the parity function in the Wyner scheme to the parity of the ands in the inner-product function.

Wyner showed how to achieve optimal rate of communication, using parity-check codes. We use a similar method to extract quasi-random sequences efficiently. Let $x_1, x_2, \ldots, x_k$ and let $y_1, y_2, \ldots, y_k$ be the two k-bit blocks produced by the two independent sources. Let $b_i=x_i$ and y_i. Let us extract l bits $r_1, r_2, \ldots, r_l$ as follows:

each r_i is the parity of some subset of $b_1, b_2, \ldots, b_k$. To make sure that the r_i's are almost unbiased, the subsets must be large, and to ensure small correlations, they most be "well spread-out". This is precisely achieved by the rows of an $l \times k$ parity-check code generator matrix (the rows are 0/1 vectors of dimension k, and represent subsets in a natural manner) which detects αk errors for some constant $\alpha > 0$ [5]. An explicit contruction of such asymptotically efficient codes appears in [9]; for practical values of k, the BCH code is more useful, although it it not asymptotically efficient. The following algorithm puts together these ideas:

proc qgen(n):

Let G be a $l \times k$ parity check code generator matrix, which detects αk errors, where $k = \omega((1/\delta^2) \log n)$, and $l = \delta^2 \alpha k/32$.

Let $x_{11}x_{12}\ldots x_{1k}, x_{21}x_{22}\ldots x_{2k}, \ldots, x_{k1}x_{k2}\ldots x_{mk}$ and

$y_{11}y_{12}\cdots y_{1k}, y_{21}y_{22}\cdots y_{2k}, \cdots, y_{k1}y_{k2}\cdots y_{mk}$

be m blocks of k bits each output by the two independent sources; $m = (n/l)$.

Let $b_{ij} = x_{ij}$ and y_{ij}.

Let $\vec{\mathbf{u}}_1 = b_{11}\ldots b_{1k}$,

$\vec{\mathbf{u}}_2 = b_{21}\ldots b_{2k}, \ldots$

$\vec{\mathbf{u}}_m = b_{m1}\ldots b_{mk}$.

Output $G \cdot \vec{\mathbf{u}}_1, G \cdot \vec{\mathbf{u}}_2, \ldots, G \cdot \mathbf{u}_m$.

end;

Theorem 2.5. *The source defined above is quasi-random. It converts $O(n/\delta^2)$ semi-random bits into n-bit quasi-random sequence.*

Proof. Since G is an $l \times k$ parity check code generator matrix with minimum distance αk, by Theorem 2.3, the exclusive-or of every non-empty subset of bits in a block has bias $\leqq 2^{-\delta^2 \alpha k/16}$. By the Parity Lemma below, the sum of the deviations of the extracted l-bit block from the uniform distribution is at most $(2^l - 1)2^{-\delta^2 \alpha k/16}$. By the choice of l and k, this expression is smaller than inverse polynomial in n. Thus the source is "high quality". By Theorem 2.1 it is quasi-random. ∎

Let D be any probability distribution on $\{0, 1\}^n$.

Let $\Pr_D[E(x)]$ denote the probability that event $E(x)$ occurs if x is picked according to the distribution D.

Definition.

$$\text{bias}_i(D) = \sum_{x \in \{0,1\}^n} \Pr_D[\vec{\mathbf{i}} \cdot \vec{\mathbf{x}} = 1] - \sum_{x \in \{0,1\}^n} \Pr_D[\vec{\mathbf{i}} \cdot \vec{\mathbf{x}} = 0].$$

Parity-Lemma. For any distribution D on $\{0, 1\}^n$,

$$\sum_{x \in \{0,1\}^n} \left| \Pr_D[x] - \frac{1}{2^n} \right| \leqq \sum_{i \in \{0,1\}^n - 0^n} \text{bias}_i(D).$$

Proof. By induction on n. Let

$$p = \sum_{z \in \{0,1\}^{n-1}} \Pr_D[1z].$$

D induces two distributions D_0 and D_1 on $\{0, 1\}^{n-1}$; D_0 being the distribution on n bit strings whose first bit is 0. Now

$$\sum_{x \in \{0,1\}^n} \left| \Pr_D[x] - \frac{1}{2^n} \right| = \sum_{z \in \{0,1\}^{n-1}} \left| \Pr_D[0z] - \frac{1}{2^n} \right| + \sum_{z \in \{0,1\}^{n-1}} \left| \Pr_D[1z] - \frac{1}{2^n} \right| =$$

$$= \sum_{z \in \{0,1\}^{n-1}} \left| (1-p) \Pr_{D_0}[z] - \frac{1}{2^n} \right| + \sum_{z \in \{0,1\}^{n-1}} \left| p \Pr_{D_1}[z] - \frac{1}{2^n} \right| \leqq$$

$$\leqq \sum_{z \in \{0,1\}^{n-1}} \left| (1-p) \Pr_{D_0}[z] - \frac{1-p}{2^{n-1}} \right| + \sum_{z \in \{0,1\}^{n-1}} \left| \frac{1-p}{2^{n-1}} - \frac{1}{2^n} \right| +$$

$$+ \sum_{z \in \{0,1\}^{n-1}} \left| p \Pr_{D_1}[z] - \frac{p}{2^{n-1}} \right| + \sum_{z \in \{0,1\}^{n-1}} \left| \frac{p}{2^{n-1}} - \frac{1}{2^n} \right|.$$

Applying the inductive hypothesis and simplifying, this is:

$$\leqq |\mathrm{bias}_{10^{n-1}}(D)| + \sum_{i \in \{0,1\}^{n-1} - 0^{n-1}} (1-p)|\mathrm{bias}_i(D_0)| + p|\mathrm{bias}_i(D_1)| \leqq$$

$$\leqq |\mathrm{bias}_{10^{n-1}}(D)| + \sum_{i \in \{0,1\}^{n-1} - 0^{n-1}} \left| \frac{1}{2} \mathrm{bias}_i(D_0) + \frac{1}{2} \mathrm{bias}_i(D_1) \right| +$$

$$+ |(1-p)\,\mathrm{bias}(D_0) - p\,\mathrm{bias}_i(D_1)| =$$

$$= |\mathrm{bias}_{10^{n-1}}(D)| + \sum_{i \in \{0,1\}^{n-1} - 0^{n-1}} |\mathrm{bias}_{0i}(D)| + |\mathrm{bias}_{1i}(D)|. \quad \blacksquare$$

3. Strong Communication Complexity

In this chapter we establish a basic connection between communication complexity theory and random bit-extraction. The communication complexity of a boolean function was defined by Yao [23]. Suppose that two distant computers C_1 and C_2 must cooperatively compute a boolean function $f(x, y)$. Initially x is known to C_1 and y is known to C_2. The computation now goes as follows: C_1 sends a bit $\alpha_1(x)$ to C_2. C_2 then sends a bit $\beta_1(y, \alpha_1(x))$ to C_1. They alternate in this manner; each α_k and β_k are functions of all the previously communicated bits. The computation ends when both C_1 and C_2 know the answer. The cost $C(A)$ of the computation A is defined to be the maximum number of bits exchanged for x, y of length n. The communication complexity of f is defined to be:

$$C(f, 1 \leftrightarrow 2) = \min \{C(A) \mid A \text{ computes } f\}.$$

The above model was inspired by the work of Abelson, who studied a similar problem for smooth real valued functions. A related question was studied by Thompson [17] to establish AT^2 lowerbounds for VLSI circuits. In a series of interesting papers, the

relationships between deterministic, non-deterministic, probabilistic and k-round communication complexity have been studied; these include the work of Lipton and Sedgewick [10], Melhorn and Schmidt [11], Papadimitriou and Sipser [13], Aho, Ullman and Yannakakis [2].

In 1984, Yao [25] defined the distributional communication complexity of a boolean function. In this model, x and y are uniformly distributed among all inputs of length n. The value of $f(x, y)$ is to be computed by a deterministic algorithm A, which gives the correct answer on $1/2+\varepsilon$ fraction of the inputs. The cost of A is the average number of bits exchanged. Then the distributional communication complexity of f: $D_\varepsilon(f)$ is the minimum such cost. Yao's motivation for studying $D_\varepsilon(f)$ was that it provides a lower-bound on the probabilistic communication complexity, $C_\varepsilon(f)$ of the boolean function; this is a consequence of Yao's general game-theoretic lemma which states that the probabilistic complexity of a problem is equal to its distributional complexity on the worst possible input distribution. The next theorem provides a different motivation for studying the distributional communication complexity of a boolean function; it lower-bounds the distributional communication complexity of any boolean function which is a good bit-extractor.

Theorem 3.1. *If $D_\varepsilon(f) \geqq k$, then there exist two independent δ semi-random sources S_1 and S_2 such that if x and y denote k-bit outputs of S_1 and S_2 then $|\Pr[f(x, y)]-(1/2)| > \varepsilon 2^{-c(\delta)k \log k}$,* where $c(\delta)$ is a positive constant, and $k \geqq 2$. ∎

Theorem 3.1 follows easily from the more general Theorem 3.2. Therefore we shall omit the proof here.

Corollary. *For every $\varepsilon > 0$, $D_\varepsilon(f) = \Omega\left(\frac{n}{\log n}\right)$, when f is the inner-product function.* ∎

The corollary follows immediately from Theorems 2.3 and 3.1. This bound was recently improced to $\Omega(n)$ by El-Gamal and Orlitzky [8] and by Chor and Goldreich [4].

Does the lower bound on the communication complexity give a sufficient condition for a boolean function to be a good bit-extractor? Below we give a counter-example. The reason that this condition is not sufficient is that the distributional communication complexity is measured for a fixed distribution on the inputs, whereas the adversaries can produce any semi-random distributions.

Example. Define $g(x, y)$ as follows: Let $n=2m$. Let $x=x'x''$ and $y=y'y''$ where $|x'|=|x''|=|y'|=|y''|=m$. x' and y' are used to determine a $\log m$ bit pointer, which picks out a bit of x. $g(x, y)$ is defined to be this bit. More precisely, divide x' and y' into $\log m$ blocks, each of length $m/\log m$. Let $x_1, \ldots, x_{\log m}$ and $y_1, \ldots, y_{\log m}$ denote these blocks. Let $l(u, v)$ be a function having linear distributional communication complexity. Let

$$b_1 = l(x_1, y_1).$$
$$\vdots$$
$$b_{\log m} = l(x_{\log m}, y_{\log m}).$$

Let i denote the $\log m$ bit number $i=b_1 \ldots b_{\log m}$.

Definition. $g(x, y)=(m+i)^{th}$ bit of x.

The function $g(x, y)$ has high communication complexity in the distributional sense. This is because at least $m/\log m$ bits must be exchanged to know even one bit of the address i with $1/2+\varepsilon$ certainty. On the other hand since x is picked from the uniform distribution, roughly half the bits of x'' are 0 and roughly half are 1. So the chance of correctly guessing $g(x, y)$ is less than $1/2+\varepsilon$ for n sufficiently large.

However, the high communication complexity of $g(x, y)$ relies critically on the particular distribution of x and y: that they are generated by a fair coin. Let us pick x and y from a different (but still semi-random) distribution: generate x and y by the flips of a $1-\delta$ biased coin. $g(x, y)$ is now $1-\delta$ biased and therefore not a good bit-extractor. In particular the communication complexity to achieve ε bias for this distribution is 0. This drastic dependence on the input distribution contrasts sharply with the inner product function, whose communication complexity appears insensitive to the bias of the input bits.

Strong Communication Complxity:

We wish to say that a function has high communication complexity in a strong sense, if the number of bit exchanges required to compute it is large under any reasonable input distribution. We propose the following strong definition of communication complexity: x and y are n-bit sequences generated by two semirandom sources S_1 and S_2. They would like to *set* the function $f(x, y)$ to 1 (or 0) with probability at least $1/2+\varepsilon$. The Strong Communication complexity of the function $f(x, y)$ is the minimum number of bits, k, that the two adversaries must exchange to achieve this ε bias in $f(x, y)$. We shall denote this strong communication complexity of $f(x, y)$ by $S_{\varepsilon,\delta}(f)=k$, where δ is the parameter associated with the semi-random sources.

Let us define the model more precisely: The two adversaries A_1 and A_2 alternately send each other bits $b_1 b_2 \ldots b_k$ while generating x and y. A_1 sends the odd numbered bits and A_2 sends the even numbered bits. The protocol may be divided into k phases. Phase i, i odd, starts when A_1 receives b_{i-1} from A_2. Let $x_1, \ldots, x_j$ be the subequence of x generated by A_1 before phase i. During phase i, A_1 generates $x_{j+1}, \ldots, \ldots, x_{j'}$, and the phase ends when A_1 sends bit b_i to A_2. The sequence $x_{j+1}, \ldots, x_{j'}$ is semi-random. Each bit depends not only on the subsequence generated so far, but also on the communicated bits $b_1, \ldots, b_{i-1}$. The δ-constraint is:

$$\delta \leqq \Pr[x_k | x_1, \ldots, x_{k-1}, b_1, \ldots, b_{i-1}] \leqq 1-\delta \quad \text{for} \quad k = j+1, \ldots, j'.$$

Moreover, the number of bits generated may be decided dynamically; i.e. j' is allowed to depend on the generated bits $x_1, \ldots, x_{j'}$, as well as the communicated bits $b_1, \ldots, b_{i-1}$. Finally b_i is some boolean function of $b_1, \ldots, b_{i-1}$, and $x_1, \ldots, x_j$.

There is another motivation for the definition given above. A function f: $S_{\varepsilon,\delta}(f) \geqq k$ can be used to obtain bits that are at most ε-biased from the outputs of two sources that are allowed to communicate up to k bits. This would relax the independence condition imposed on the two sources in the previous section.

Theorem 1 can be restated in terms of strong communication complexity:

Restatement of Theorem 2.3.

$$S_{2-\delta^2 n/16,\delta}(f) > 0,$$

when f is the inner product function.

Theorem 3.2. *For every* $0<\alpha<1$ *and* $k\geqq 1$, $S_{\varepsilon,\delta}(f)\leqq k$, *then*

$$S_{((1-\alpha)\varepsilon/2(1+\alpha)),\alpha\delta}(f)\leqq k-1.$$

Proof. Consider a pair of strategies for the two adversaries that achieves the ε bias in the value of f, using just k bits of communication. We shall show that by giving the adversaries more power, the first bit of communication, b_1, can be dispensed with, while still achieving a not much smaller bias in the value of f. Let V_0 be the set of (n-bit) strings for which the communicated bit b_1 is 0, and let V_1 be the set of strings on which b_1 is 1. Let T_0 be A_2's strategy if he receives a 0 from A_1, and T_1 if he receives a 1. Let γ be the probability that A_1 generates a V_0 string (i.e. $b_1=0$), and $1-\gamma$, the probability that he generates a V_1 string. Let $p_{i,j}$ be the probability that the value of f is 1 given that A_1 generated a V_i string and A_2 followed strategy T_j. Then the bias in the value of f is

$$|\gamma p_{0,0}+(1-\gamma)p_{1,1}-1/2|\geqq\varepsilon.$$

Clearly either

$$\left|\gamma p_{0,0}-\frac{\gamma}{2}\right|\geqq\frac{\varepsilon}{2}\quad\text{or}\quad\left|(1-\gamma)p_{1,1}-\frac{(1-\gamma)}{2}\right|\geqq\frac{\varepsilon}{2}.$$

Without loss of generality assume that $|\gamma-p_{0,0}-\gamma/2|\geqq\varepsilon/2$. Let us modify the protocol so that A_1 does not communicate bit b_1, and A_2 follows startegy T_0. The bias in the value of f is now $|\gamma p_{0,0}+(1-\gamma)p_{1,0}-1/2|$. This bias could be as small as 0 because the contribution due to $|(1-\gamma)p_{1,1}-(1-\gamma)/2|$ may cancel the contribution due to $|\gamma p_{0,0}-\gamma/2|$. However, A_1 is allowed the extra power to set the bits of x with probability $1-\alpha\delta$ instead of $1-\delta$. By the redistribution lemma below, this added power allows A_1 to change the distribution on his strings such that $\Pr[V_0]/\Pr[V_1]$ can be varied between $\alpha\gamma/(1-\gamma)$ and $\gamma/(\alpha(1-\gamma))$, *without* changing the relative probabilities of strings within V_0, and the relative probabilities of strings within V_1. At one of the two extremes, the bias in the value of f is at least:

$$\frac{\varepsilon}{2(1+\alpha)}-\frac{\alpha\varepsilon}{2(1+\alpha)}=\frac{(1-\alpha)\varepsilon}{2(1+\alpha)}.\quad\blacksquare$$

Redistribution Lemma. *Let* D *be a* δ *semi-random distribution on* $\{0,1\}^n$. *Let* V_0, V_1 *be a partition of* $\{0,1\}^n$. *Let* $\Pr_D[V_0]=\gamma$ *and* $\Pr_D[V_1]=1-\gamma$. *Then for every* $0<\alpha<1$ *and for every* $\alpha\gamma/(1-\gamma)\leqq\beta\leqq\gamma/(\alpha(1-\gamma))$, *there is an* $\alpha\delta$ *semi-random distribution* D' *on* $\{0,1\}^n$ *such that*

$$\frac{\Pr_{D'}[V_0]}{\Pr_{D'}[V_1]}=\beta\quad\text{and}\quad\frac{\Pr_{D'}[u]}{\Pr_{D'}[v]}=\frac{\Pr_D[u]}{\Pr_D[v]}$$

whenever $u, v\in V_0$ *or* $u, v\in V_1$.

Proof. Clearly D' is uniquely specified by D and β. It remains to check that D' is an $\alpha\delta$ semi-random distribution. Fix a binary string s of length less than n. Let T be the set of n-bit strings having prefix $s0$, and let U be the set of n-bit string having prefix $s1$.

Let $T_0=T\cap V_0$, $T_1=T\cap V_1$, $U_0=U\cap V_0$, $U_1=U\cap V_1$. Then

$$\frac{\Pr_{D'}[T]}{\Pr_{D'}[U]}=\frac{\Pr_{D'}[T_0]+\Pr_{D'}[T_1]}{\Pr_{D'}[U_0]+\Pr_{D'}[U_1]}=$$

$$=\frac{\frac{\beta}{(1+\beta)\gamma}\Pr_D[T_0]+\frac{1}{(1+\beta)(1-\gamma)}\Pr_D[T_1]}{\frac{\beta}{(1+\beta)\gamma}\Pr_D[U_0]+\frac{1}{(1+\beta)(1-\gamma)}\Pr_D[U_1]}=$$

$$=\frac{\frac{\beta(1-\gamma)}{\gamma}\Pr_D[T_0]+\Pr_D[T_1]}{\frac{\beta(1-\gamma)}{\gamma}\Pr_D[U_0]+\Pr_D[U_1]}.$$

By hypothesis $\alpha\leqq\beta(1-\gamma)/\gamma\leqq 1/\alpha$. Assume w.l.o.g. $1\leqq\beta(1-\gamma)/\gamma$. Then

$$\frac{\beta(1-\gamma)}{\gamma}\frac{\Pr_D[T_0]+\Pr_D[T_1]}{\Pr_D[U_0]+\Pr_D[U_1]}\geqq\frac{\Pr_{D'}[T]}{\Pr_{D'}[U]}\geqq\frac{\gamma}{\beta(1-\gamma)}\frac{\Pr_D[T_0]+\Pr_D[T_1]}{\Pr_D[U_0]+\Pr_D[U_1]}.$$

Therefore

$$\frac{\beta(1-\gamma)}{\gamma}\frac{\Pr_D[T]}{\Pr_D[U]}\geqq\frac{\Pr_{D'}[T]}{\Pr_{D'}[U]}\geqq\frac{\gamma}{\beta(1-\gamma)}\frac{\Pr_D[T]}{\Pr_D[U]}.$$

Thus

$$\frac{(1-\delta)}{\alpha\delta}\geqq\frac{\Pr_{D'}[T]}{\Pr_{D'}[U]}\geqq\frac{\alpha\delta}{(1-\delta)}.$$

Therefore D' is $\alpha\delta$ semi-random. ∎

The above theorem trades off the number of bits communicated by making the adversaries (of the semi-random sources) more powerful (smaller δ), but setting the output with smaller probability of success. This tradeoff, combined with Theorem 1 imply the following lower bound for the strong communication complexity of the inner product function.

Corollary. *For every fixed* $0<\varepsilon<1/2$, $S_{\varepsilon,\delta}(\vec{\mathbf{x}},\vec{\mathbf{y}})=\Omega(n/\log n)$.

Proof. By using Theorem 3.2 iteratively, if $S_{\varepsilon,\delta}(f)\leqq k$ then

$$S_{((1-\alpha)^k\varepsilon/2^k(1+\alpha)^k),\,\alpha^k\delta}(f)\leqq 0.$$

Substituting $\alpha=1-1/k$, and combining with Theorem 2.3 yields the desired bound on k. ∎

In other words the inner product function extracts good bits even from two communicating semi-random sources. This is important from a practical point of view, because it says that even if the shielding between the two physical noise sources is not perfectly designed, the inner-product function will still extract good sequences from their output.

Memory Bounded Source:

Definition. Source S is *memory bounded* if the adversary strategy is a map T: $\{0,1\}^m \times \mathbf{N} \to [\delta, 1-\delta]$, where the first argument denotes the contents of the m-bit memory of the adversary, and the second argument is the time (assume that one bit is output per unit time). The memory is updated by the function g: $\{0,1\}^m \times \mathbf{N} \times \times \{0,1\} \to \{0,1\}^m$; the first argument to g is the contents of the memory, the second is time and the third is the latest bit output by the source.

Notice that whereas in the case of the semi-random source, there was no need to explicitly make the strategy T a function of time, this does increase the class of distributions that can be obtained when the memory is bounded. Interpreted physically, this time dependence is very important: the bias of the output of a Zener diode depends upon operating conditions such as temperature and humidity. Such effects cannot be expressed in the bounded memory, and must be represented separately as the time dependence of the bias.

The advantage of putting this additional memory constraint on the adversary is that this makes it possible to obtain quasi-random sequences from a single source. Let x and y be k-bit outputs of the source, and suppose that x is output before y.

Theorem 3.3. $|\Pr[\vec{x} \cdot \vec{y} = 1] - 1/2| \leqq (4m)^m 2^{-\delta^2 k/64}$.

Proof. x and y may be regarded as the outputs of two communicating sources. The communication is at most m bits. Now apply Theorem 1 and Theorem 3.2. ∎

Thus if $m \ll k/\log k$, then the bit $\vec{x} \cdot \vec{y}$ is "high quality". Using the Wyner construction in exactly the same way as in Chapter 2, we obtain an alogirhtm which produces quasi-random sequences efficiently; both in time efficiency and in the number of source bits used per output bit.

4. Discussion and Open Questions

If the adversary strategy in the memory bounded source does not depend on time, but only on the m-bit finite memory, then the source is a finite state Markov process on 2^m states. The problem of obtaining random bit-sequences from a deterministic finite state Markov process has been studied by Elias [7] and Blum [3]. In particular, Blum showed that whereas the natural generalization of von Neumann's unbiasing algorithm [12] yields correlated bits when applied to a deterministic Markov process, a counter-intuitive generalization yields independent, unbiased bit-sequences.

Deterministic Markov processes and memory bounded sources produce different classes of distributions. A deterministic Markov process may have states whose output is completely determined; the δ condition is not satisfied by such a source. On the other hand, the memory bounded source might be a nondeterministic Markov process, or more generally a time dependent process. Such processes cannot be described by a finite number of states. There is also a difference in the output distributions produced by the two algorithms — Blum's algorithm produces independent, unbiased bit-sequences, whereas the algorithm for the memory bounded source produces quasi-random output. Due to this, Blum's algorithm requires an explicit knowledge

of the underlying state transition diagram of the Markov process, whereas the algorithm for the memory bounded source works even though the source is controlled by an unknown adversary.

All the theorems proved in this paper use some form of independence or lack of communication between two semi-random sources to extract quasi-random sequences. On the other hand, Theorem 2.2. shows the futility of attempting to extract even a single "high quality" bit from a single semi-random source. The recent result of Vazirani and Vazirani [20] that Random Polynomial Time is equal to Semi-random Polynomial time stands in sharp contrast to this impossibility theorem. Random Polynomial time is the class of languages that can be recogonized in polynomial time by an algorithm that can flip a fair coin at each step. Semi-random Polynomial time is analogously the class of languages that can be recognized in polynomial time by an algorithm that can obtain a bit from a semi-random source at each step. Since a semi-random source is part way between a truly-random source and a deterministic source, this provides some insight into the relationship between randomness and efficient computation. In a companion paper, they showed that BPP (two sided error probabilistic polynomial time) can also be simulated in polynomial time using a single semi-random source. This stronger result also yielded an improvement in simulation time: they showed that $\mathrm{RTIME}(T) \subseteq \mathrm{SRTIME}(T \log T)$. It is interesting that the Main Lemma of chapter 2 played an important role in their proof.

In another recent development, Chor and Goldreich [4] studied a generalization of the semi-source, in which the adversary outputs blocks of k bits at a time. The constraint on the adversary is that no sequence of length k can be output with probability greater than $(1-\delta)^k$. This provides a strict generalization of the semi-random source because some k-bit sequences may never be output by the source. Chor and Goldreich consider whether in this stronger model it is still possible to generate quasi-random sequences from two independent block-wise sources. The show by a counting argument that most boolean functions extract "high quality" bits from two independent sources. However, the problem of explicitly giving a boolean function which yields "high quality" bits is more difficult and remains an interesting open question. Clearly, the inner product function does not work for this block-wise source (this easily follows from the comments in Chapter 2 on the "modified scenario"). However unlike the two source theorem, the BPP simulation result of Vazirani and Vazirani [20], does not require the full strength of the Main Lemma of Chapter 2; (it suffices to prove the case $m=2$). Chor and Goldreich showed that this form of the lemma holds for the block-wise source as well; the algorithm of Vazirani and Vazirani then yields that BPP can be simulated by a block-wise random source as well. Alon [1] obtained a similar lemma (with somewhat sharper bounds) by a simple proof.

Several open questions and issues remain to be explored. Is the $\Omega(n/\log n)$ lower bound for the strong distributional complexity of the inner product function tight? We conjecture that $\Theta(n)$ is the correct complexity. One approach towards improving the lower bound might be to prove a sharper hierarchy theorem.

The Bit Extraction Problem:

Consider the following question:

Input: Positive integer k, n-bit sequence generated as follows: m of the bits are picked deterministically the rest of the $n-m$ bits are generated by the flips of a fair coin.

Output: *Without knowing* which bits were deterministically fixed, generate k independent, unbiased bits from the sequence.

A solution to the problem in general would entail outputting $b_1(x_1, \ldots, x_n)$, $b_2(x_1, \ldots, x_n)$, $\ldots$, $b_k(x_1, \ldots, x_n)$, where the b_i's are boolean predicates. The core of the problem lies in showing that no subset of m bits can be fixed so that the k output bits become correlated.

The case $m=1$ is easy. Let b_i be x_i xor x_n. Then the b_i's are independent, unbiased bits. However, for general m, such xor schemes perform poorly. If $m=2n/3$, then we can prove that there is no xor scheme for extracting *even* 2 *bits* this is independent of the value of n. What about general boolean functions b_i? We conjecture that there is always an xor scheme which is optimal for the problem; for the case of extracting two bits, our conjecture was recently proved [6]. For the general case, they give bounds that show that no function can do substantially better than an xor scheme. However, the conjecture that xor schemes are optimal in general, is still open.

Probabilistic Analysis of Algorithms:

The worst-case complexity of an algorithm is a useful and fundamental measure of how "good" the algorithm is. However, for some algorithms such as the Simplex method, this measure appears to be too pessimistic. To get around this difficulty, Karp introduced an alternative measure: the average case complexity of the algorithm on a given input distribution. Several algorithms, including some versions of the simplex algorithm have been successfully analyzed under this new measure.

The major problem with average case analysis is that the input distribution is often unknown in practice. The philosophy behind the semi-random source suggests a way around this difficulty. Allow the input distribution to be picked from among a class of distributions (for example the semi-random distributions with parameter δ). Let the complexity of an algorithm be the maximum over distributions D in this class, of the average running time of the algorithm when the input is picked according to distribution D. In these terms, the worst-case complexity of an algorithm is the extreme situation where the adversary is allowed to pick a distribution whose entire weight is concentrated on a single point.

Acknowledgements. It is a pleasure to thank: Dick Karp for useful suggestions and encouragement throughout the evolution of this work, Manuel Blum, Gilles Brassard and Sampath Kannan for listelning patiently to my ideas and providing valuable feedback, and Ron Rivest, Miklós Sántha, Mike Sipser, Vijay Vazirani and Andy Yao for some very interesting discussions. I also wish to thank the referee for useful comments on the write-up.

References

[1] N. ALON, On VV-s result SRP=RP, unpublished manuscript, 1985.
[2] A. AHO, J. ULLMAN and M. YANNAKAKIS, On Notions of Information Transfer in VLSI Circuits, *15th Annual Symp. on Theory of Computing*, (1983), 133—139.
[3] M. BLUM, Independent Unbiased Coin Flips From a Correlated Biased Source: a Finite State Markov Chain, *25th, IEEE Symposium on the Foundations of Computer Science* (1984).
[4] B. CHOR and O. GOLDREICH, Unbiased Bits from Weak Sources of Randomness, *26th IEEE Symposium on the Foundations of Computer Science* (1985).
[5] R. GALLAGER, *Information Theory and Reliable Communication*, New York: John Wiley, (1986).
[6] B. CHOR, O. GOLDREICH, J. HASTAD, J. FRIEDMAN, S. RUDICH and R. SMOLENSKY, The Bit Extraction Problem or t-Resilient Functions, *26th IEEE Symposium on the Foundations of Computer Science*, (1985).
[7] P. ELIAS, The Efficient Construction of an Unbiased Random Sequence, *Ann. Math. Statist.* **43** (1972), 865—870.
[8] EL-GAMAL and ORLITZKY, Randomized Communication Complexity, preprint.
[9] J. JUSTESEN, A Class of Constructive Asymptotically Good Algebraic Codes, *IEEE Trans. Inform. Theory*, **18** (1972), 652—656.
[10] R. LIPTON and R. SEDGEWICK, Lower bounds for VLSI, *Proc. 13th Ann. ACM Symp. on Theory of Computing*, (1981), 300—307.
[11] K. MELHORN and E. SCHMIDT, Las Vegas is better than Determinism in VLSI and Distributed Computing, *14th Annual Symp. on Theory of Computing*, (1982), 330—337.
[12] J. VON NEUMANN, Various Techniques Used in Connection with Random Digits, Notes by G. E. Forsythe, National Bureau of Standards, *Applied Math. Series*, **12**, (1951), 36—38. Reprinted in von Neumann's Collected Works, Vol 5, Pergamon Press (1963), 768—770.
[13] C. H. PAPADIMITRIOU and M. SIPSER, Communication complexity, *Proc. 14th Ann. ACM Symp. on Theory of Computing*, (1982), 196—200.
[14] P. RAGHAVAN, Probabilistic Construction of Deterministic Algorithms: Approximating Packing Integer Programs, *27th. IEEE Symposium on the Foundations of Computing*, (1986).
[15] W. RUDIN, *Principles of Mathematical Analysis*, Third Edition, McGraw-Hill Book Company.
[16] M. SÁNTHA and U. V. VAZIRANI, Generating Quasi-random Sequences from Semi-random Sources, *Journal of Computer Systems and Sciences*, **33**, (1986), 75—87.
[17] C. THOMPSON, Area-Time Complexity for VLSI, *11th Annual Symp. on Theory of Computing*, (1979), 81—88. to appear in JCSS (currently, Proc. STOC 1985).
[18] U. V. VAZIRANI, Randomness, Adversaries and Computation, *Ph. D. Dissertation, U. C. Berkeley*, (1986).
[19] U. V. VAZIRANI and V. V. VAZIRANI, Trapdoor Pseudo-random Number Generators with Applications to Protocol Design, *15th Annual ACM Symp. on Theory of Computing*, (1983).
[20] U. V. VAZIRANI and V. V. VAZIRANI, Random Polynomial Time is Equal to Semi-Random Polynomial Time, *26th. IEEE Symposium on the Foundations of Computer Science*, (1985).
[21] U. V. VAZIRANI and V. V. VAZIRANI, Sampling a Population with a Semirandom source, *Proccedings Sixth Ann. FST-TCS Conference*, New Delhi, (1986).
[22] A. WYNER, Wire-tap Channel, *Bell System Technical Journal*, (1975).
[23] A. C. YAO, Some Complexity Questions related to Distributive computing, *Proc. 11th Ann. ACM Symp on the Theory of Computing*, (1979), 209—213.
[24] A. YAO, Theory and Applications of Trapdoor Functions, *23th IEEE Symposium on the Foundations of Computer Science*, (1982).
[25] A. C. YAO, Lower Bounds by Probabilistic Arguments, *Proc. 24 Ann. Symp on the Theory of Computing*, (1983), 420—428.

Umesh V. Vazirani

Department of Computer Science
U. C. Berkeley
Berkeley, CA 94720 U.S.A.