and $Y_n$ conditioned on the event $\{Z_n > 0\}$ meaning that a random tree has depth at least $n$.

*Theorem 3:* In the subcritical case, $\mu < 1$, as $n \to \infty$, the probability distribution of $Z_n|\{Z_n > 0\}$ converges to a limit probability distribution, and if $E(Z_1 \log Z_1) < \infty$, then

$$\lim_{n\to\infty} E(Z_n|Z_n > 0) = \frac{1}{c} \tag{22}$$

$$\lim_{n\to\infty} \mathrm{Var}(Z_n|Z_n > 0) = \frac{\sigma^2}{c\,\mu(1-\mu)} - \frac{1}{c^2} \tag{23}$$

where $c$ is the same positive constant as in (14).

In the critical case $\mu = 1$, if $0 < \sigma^2 < \infty$, then

$$\lim_{n\to\infty} \Pr\left\{\frac{Z_n}{n} > z|Z_n > 0\right\} = e^{-2z/\sigma^2}, \qquad z \ge 0 \tag{24}$$

$$E(Z_n|Z_n > 0) \sim \frac{\sigma^2}{2}n \tag{25}$$

$$\mathrm{Var}(Z_n|Z_n > 0) \sim \frac{\sigma^4}{4}n^2. \tag{26}$$

The probability distribution of the conditioned random variable $Y_n|\{Z_n > 0\}$ is not treated in the standard books on branching processes like [8] and [2]. Nevertheless, the previous theorems and the results regarding the conditioned random variable $Z_n|\{Z_{n+k} > 0\}$ presented in [2] lead us to conclude that in the subcritical case

$$E(Y_n|Z_n > 0) = O(n)$$

and

$$\mathrm{Var}(Y_n|Z_n > 0) = O(n^2)$$

whereas in the critical case

$$E(Y_n|Z_n > 0) = O(\sigma^2 n^2)$$

and

$$\mathrm{Var}(Y_n|Z_n > 0) = O(\sigma^4 n^4).$$


## Acknowledgment

The author wishes to thank the two anonymous referees for their helpful comments.


## References


[1] R. J. Anderson, Internet communication, 1994.

[2] K. B. Athreya and P. E. Ney, *Branching Processes*. Berlin, Germany: Springer-Verlag, 1972.

[3] W. G. Chambers, "On random mappings and random permutations," in *Fast Software Encryption—Leuven'94 (Lecture Notes in Computer Science)*, B. Preneel, Ed. Berlin, Germany: Springer-Verlag, 1995, vol. 1008, pp. 22–28.

[4] J. Dj. Golić and M. J. Mihaljević, "A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance," *J. Cryptol.*, vol. 3, no. 3, pp. 201–212, 1991.

[5] J. Dj. Golić, "Toward fast correlation attacks on irregularly clocked shift registers," in *Advances in Cryptology—Eurocrypt '95 (Lecture Notes in Computer Science)*, L. C. Guillou and J.-J. Quisquater, Eds. Berlin, Germany: Springer-Verlag, 1995, vol. 921, pp. 248–262.

[6] J. Dj. Golić and R. Menicocci, "Edit distance correlation attack on the alternating step generator," in *Advances in Cryptology—Crypto '97 (Lecture Notes in Computer Science)*, B. Kaliski, Ed. Berlin, Germany: Springer-Verlag, 1997, vol. 1294, pp. 499–512.

[7] J. Dj. Golić, "Recent advances in stream cipher cryptanalysis," *Publications de l'Institut Mathematique*, vol. 64/78, pp. 183–204, 1998.

[8] T. H. Harris, *The Theory of Branching Processes*. Berlin, Germany: Springer-Verlag, 1963.

[9] M. E. Hellman, "A cryptanalytic time-memory trade-off," *IEEE Trans. Inform. Theory*, vol. IT-26, pp. 401–406, July 1980.

[10] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1997.

[11] R. A. Rueppel, "Stream ciphers," in *Contemporary Cryptology: The Science of Information Integrity*, G. Simmons, Ed. New York: IEEE Press, 1991, pp. 65–134.

[12] B. Schneier, *Applied Cryptography*. New York: Wiley, 1996.


# Fast and Efficient Construction of an Unbiased Random Sequence


Boris Ya. Ryabko and Elena Matchikina



***Abstract*—The problem of converting a sequence of symbols generated by a Bernoulli source into an unbiased random sequence is well-known in information theory. The proposed method is based on Elias' algorithm [5] in which the sequence of symbols is divided into blocks of length $N$, $N \ge 1$. We suggest a new method of constructing an unbiased random sequence which uses $O(N \log^2 N)$ bits of memory and takes $O(\log^3 N \log\log(N))$ bit operations per letter.**



***Index Terms*—Efficiency, fast coding, redundancy, unbiased random sequence.**


## I. Introduction

We consider the problem of fast and efficient construction of an output sequence $z = (z_1, z_2, \cdots, z_m, \cdots)$ of statistically independent and equiprobable binary digits from an input binary sequence $x = (x_1, x_2, \cdots, x_n, \cdots)$ generated by a Bernoulli source $S$ which chooses $x_n$ from $\{0, 1\}$ independently with bias $p : \Pr\{x_n = 1\} = p$, $\Pr\{x_n = 0\} = 1–p$ for all $n$, $p$ unknown but fixed, $0 < p < 1$. In recent years many investigators have been interested in this problem (see, for example, the survey in [10] and [11]). In general, the efficiency of such a construction is characterized by redundancy $r$ which is defined as the difference between the Shannon entropy $H$ and the efficiency $\eta_N$ introduced by Elias in [5]. Elias in [5] gives the following definition of the efficiency $\eta_N$ of method of converting the block $x^N = (x_1, \cdots, x_N)$ of the input sequence $x$ into the output block $z^K = (z_1, \cdots, z_K)$: $\eta_N$ is the average of the ratios $K/N$, averaged with respect to probabilities of appearing $x^N$ in an input sequence. The redundancy, of course, depends on the Bernoulli source, so we consider the maximum (supreme) redundancy over the set of all Bernoulli sources generating letters from $\{0, 1\}$, and, for brevity, we call it redundancy, as before. We consider the problem of constructing an unbiased random sequence with arbitrarily small redundancy.

Besides the redundancy, the complexity of the construction can be assessed by the memory size $(M)$ required by the encoder, as well as by the average time $(T)$ of encoding one symbol measured by the number


Manuscript received March 23, 1998; revised July 5, 1999. This work was supported by the Russian Foundation of Basic Research under Grant 99-01-00586. The material in this correspondence was presented in part at the IEEE International Symposium on Information Theory, Cambridge, MA, August 16–21, 1988.



The authors are with the Department pf Applied mathematics and Cybernetics, Siberian State University of Telecommunications and Information Sciences, Novosibirsk, Russia.



Communicated by T. E. Fuja, Associate Edtor At Large.

Publisher Item Identifier S 0018-9448(00)03105-9.

of binary operations on single-bit words when it is implemented on a computer with random-access memory. For a discussion of this natural model, see [1].

In 1951, von Neumann described in [7] a procedure of generating an unbiased random sequence using on each of the pairs $x_1x_2,\ x_3x_4, \cdots$, the mapping

$$00 \to \Lambda,\ 01 \to 0,\ 10 \to 1,\ 11 \to \Lambda \tag{1}$$

where $\Lambda$ represents no output digit. The redundancy of this procedure

$$r = p\log\frac{1}{p} + (1-p)\log\frac{1}{1-p} - \frac{1}{2}(2p(1-p)+2p(1-p))$$

and attains its maximal value when $p = \frac{1}{2}$. (Here and below $\log x = \log_2 x$.)

Elias proposed in [5] a block encoding for converting an input sequence of symbols generated by a stationary random process into a sequence of independent, equiprobable output symbols with $r \to 0$ when $N \to \infty$, where $N$ is the length of a block. We give a short description of the main idea of encoding proposed by Elias. Divide the set of all $2^N$ possible input binary blocks of length $N$ into $N+1$ classes $S_k,\ k = 0, \cdots, N$. Every class $S_k$ is composed of all the binary blocks of length $N$ with $k$ ones.

Define $m_k = \lfloor \log_2 |S_k| \rfloor$, where $\lfloor y \rfloor$ is the largest integer not greater than $y$. Let $|S_k| = (\alpha_{m_k}, \alpha_{m_k-1}, \cdots, \alpha_0)$ be a binary notation of the integer $|S_k| = \binom{N}{k},\ \alpha_{m_k} = 1,\ \alpha_j \in \{0, 1\},\ m_k > j \ge 0$. If $\alpha_j = 1,\ 0 \le j \le m_k$ then arbitrarily assign the $2^j$ possible output binary sequences of length $j$ to $2^j$ distinct members from $S_k$ which have not been yet assigned. If $|S_k|$ is odd then one member of $S_k$ will be assigned to the empty sequence. $S_0$ and $S_N$ have only one element each, which is therefore assigned to the empty sequence. To help illustrate the Elias encoding we provide the following example. Let $N = 4$. Then

$$\begin{aligned}
S_0 &= \{(0,0,0,0)\}\\
S_1 &= \{(1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1)\}\\
S_2 &= \{(0,0,1,1), (0,1,0,1), (0,1,1,0), (1,1,0,0),\\
&\qquad (1,0,1,0), (1,0,0,1)\}\\
S_3 &= \{(1,1,1,0), (1,0,1,1), (1,1,0,1), (0,1,1,1)\}\\
S_4 &= \{(1,1,1,1)\}.
\end{aligned}$$

The binary notations of $|S_i|,\ i = 0, 1, 2, 3, 4$, are $|S_0| = (1)$, $|S_1| = (1, 0, 0)$, $|S_2| = (1, 1, 0)$, $|S_3| = (1, 0, 0)$, $|S_4| = (1)$. According to the Elias encoding we have the mapping

$$\begin{array}{ll}
(0,0,0,0) \to \Lambda, & (1,1,1,1) \to \Lambda,\\
(1,0,0,0) \to (0,0), & (0,1,1,1) \to (0,1),\\
(0,1,0,0) \to (0,1), & (1,0,1,1) \to (1,0),\\
(0,0,1,0) \to (1,0), & (1,1,0,1) \to (1,1),\\
(0,0,0,1) \to (1,1), & (1,1,1,0) \to (0,0),\\
(1,1,0,0) \to (1), & (0,0,1,1) \to (0,1),\\
(0,1,0,1) \to (0), & (1,0,1,0) \to (1,0),\\
(0,1,1,0) \to (0,0), & (1,0,0,1) \to (1,1).
\end{array}$$

In case $N = 2$ we have the von Neumann mapping (1). Elias proved that the redundancy of this coding is $r = O(1/N)$ and, therefore, $r \to 0$ when $N \to \infty$. A naive implementation of this method required one to store all $2^N$ codewords. That is why the memory size of the encoder increases exponentially when $N$ grows. In Section II, we suggest a fast method for the Elias encoding which does not require exponential memory size. This method is based on the method of enumerative encoding from [8] and uses the Schönhage–Strassen method for fast integer multiplication.

## II. The Fast Method

Let $x = (x_1, x_2, \cdots, x_n, \cdots)$ be a sequence of binary digits generated by a Bernoulli source with probabilities $\Pr\{x_n = 1\} = p, \Pr\{x_n = 0\} = 1-p$. We shall not encode individual digits, but blocks of length $N$. Denote by $x^N$ such a block.

Let $x^N$ contain $k$ ones. Let $\mathrm{Num}\,(x^N)$ be a number which corresponds to $x^N$ when we lexicographically order set $S_k$. To enumerate the set of binary words of length $N$ with fixed numbers of ones in the lexicographical order we exploit an enumerative code from [2]–[4], [6]. If $x^N$ has $k$ ones the number $\mathrm{Num}\,(x^N)$ is given by

$$\mathrm{Num}(x^N) = \sum_{t=1}^{N} \binom{x_t N - t}{k - \sum_{i=1}^{t-1} x_i}. \tag{2}$$

Let us enumerate the elements of $S_2$ from the previous example. According to (2) we have

$$\begin{aligned}
\mathrm{Num}\,((1,1,0,0)) &= \binom{4-1}{2} + \binom{4-2}{2-1} = 5,\\
\mathrm{Num}\,((0,1,1,0)) &= \binom{4-2}{2} + \binom{4-3}{2-1} = 2,\\
\mathrm{Num}\,((0,0,1,1)) &= \binom{4-3}{2} + \binom{4-4}{2-1} = 0,\\
\mathrm{Num}\,((0,1,0,1)) &= \binom{4-2}{2} + \binom{4-4}{2-1} = 1,\\
\mathrm{Num}\,((1,0,0,1)) &= \binom{4-1}{2} + \binom{4-4}{2-1} = 3,\\
\mathrm{Num}\,((1,0,1,0)) &= \binom{4-1}{2} + \binom{4-3}{2-1} = 4.
\end{aligned}$$

We suggest a new method for encoding which uses (2). According to our method the codeword $code\,(x^N)$ for every block $x^N$ is constructed as follows:

i) We begin by computing the number $\mathrm{Num}\,(x^N)$ in the set $S_k$, if $x^N$ contains $k$ ones.

ii) Let the integer $|S_k| = \binom{N}{k}$ be presented by $2^{j_0} + 2^{j_1} + \cdots + 2^{j_m},\ 0 \le j_0 < j_1 < \cdots < j_m$. If $0 \le \mathrm{Num}(x^N) < 2^{j_0}$ then the codeword $code\,(x^N)$ is $j_0$ low-order binary digits of $\mathrm{Num}\,(x^N)$. In particular, if $j_0 = 0$ then the codeword for $x^N$ with $\mathrm{Num}\,(x^N) = 0$ will be the empty string. If

$$\sum_{s=0}^{t} 2^{j_s} \le \mathrm{Num}(x^N) < \sum_{s=0}^{t} 2^{j_s} + 2^{j_{t+1}}$$

for some $t,\ t = 0, \cdots, m$ then the codeword $code\,(x^N)$ is a suffix consisting of $j_{t+1}$ binary digits of $\mathrm{Num}\,(x^N)$.

It may be shown that the running time of calculation according to (2) is not greater than $cN^2$ bit operations, $c > 0,\ c = const$. In order to develop a faster method, rewrite (2) in the following way:

$$\mathrm{Num}(x^N) = \binom{N}{k} \cdot \left( x_1 \frac{\binom{N-1}{k}}{\binom{N}{k}} + x_2 \frac{\binom{N-2}{k - \sum_{i=1}^{1} x_i}}{\binom{N-1}{k - \sum_{i=1}^{1} x_i}} \cdot \frac{\binom{N-1}{k - \sum_{i=1}^{1} x_i}}{\binom{N}{k}} + \cdots \right).$$

We shall define the subsidiary values for $t = 1, \cdots, N$ needed in the encoding $x^N$ as in [9]. Define

$$p(x_t/x_1, \cdots, x_{t-1}) = \frac{\binom{N-t}{k-\sum_{i=1}^{t} x_i}}{\binom{N-t+1}{k-\sum_{i=1}^{t-1} x_i}}$$

$$q(x_t/x_1, \cdots, x_{t-1}) = x_t \frac{\binom{N-t}{k-\sum_{i=1}^{t-1} x_i}}{\binom{N-t+1}{k-\sum_{i=1}^{t-1} x_i}}.$$

Obviously,

$$\mathrm{Num}\,(x_1, x_2, \cdots, x_N) = |S_k|(q(x_1) + q(x_2/x_1)p(x_1) + q(x_3/x_2, x_1)p(x_2/x_1)p(x_1) + \cdots). \tag{3}$$

To compute $\mathrm{Num}\,(x^N)$ we use the fast method of enumerative coding which is proposed in [9]. For simplicity we suppose $\log N$ to be an integer. In general, we can add the letters 0 to every word of $S_k$ in order to make $\log N$ an integer. It does not change $|S_k|$ or the complexity of the code. To carry out (3) define the values

$$\begin{aligned}&\rho_1^0 = p(x_1),\ \rho_2^0 = p(x_2/x_1), \cdots, \rho_N^0 = p(x_N/x_1, \cdots, x_{N-1})\\ &\lambda_1^0 = q(x_1),\ \lambda_2^0 = q(x_2/x_1), \cdots, \lambda_N^0 = q(x_N/x_1, \cdots, x_{N-1}).\end{aligned} \tag{4}$$

All calculations are performed in the following way:

$$\rho_t^s = \rho_{2t-1}^{s-1}\rho_{2t}^{s-1},\ \lambda_t^s = \lambda_{2t-1}^{s-1} + \lambda_{2t}^{s-1}\rho_{2t}^{s-1}$$
$$s = 1, 2, \cdots, \log N,\ t = 1, \cdots, N/2^s. \tag{5}$$

It is not difficult to see that

$$\mathrm{Num}\,(x^N) = |S_k|\lambda_1^{\log N}. \tag{6}$$

Let us give an example. Given $N = 4$, $k = 2$, and block $x^N = (1, 0, 0, 1)$. From (4) we obtain

$$\begin{aligned}
\rho_1^0 &= p(x_1) = \frac{2}{4-1+1} = \frac{1}{2}\\
\rho_2^0 &= p(x_2/x_1) = 1 - \frac{2-1}{4-2+1} = \frac{2}{3}\\
\rho_3^0 &= p(x_3/x_1, x_2) = 1 - \frac{2-1}{4-3+1} = \frac{1}{2}\\
\rho_4^0 &= p(x_4/x_1, x_2, x_3) = \frac{2-1}{4-4+1} = 1\\
\lambda_1^0 &= q(x_1) = \frac{1}{2}\\
\lambda_2^0 &= q(x_2/x_1) = 0\\
\lambda_3^0 &= q(x_3/x_1, x_2) = 0\\
\lambda_4^0 &= q(x_4/x_1, x_2, x_3) = 0.
\end{aligned}$$

According to (5) we compute the values

$$\begin{aligned}
\rho_1^1 &= \rho_1^0\rho_2^0 = \frac{1}{2}\cdot\frac{2}{3} = \tfrac{1}{3}\\
\rho_2^1 &= \rho_3^0\rho_3^0 = \frac{1}{2}\cdot 1 = \frac{1}{2}\\
\lambda_1^1 &= \lambda_1^0 + \lambda_2^0\rho_2^0 = \frac{1}{2}\\
\lambda_2^1 &= \lambda_3^0 + \lambda_4^0\rho_4^0 = 0\\
\lambda_2^2 &= \lambda_1^1 + \lambda_2^1\rho_2^1 = \frac{1}{2}.
\end{aligned}$$

So from (6) we obtain $\mathrm{Num}\,((1, 0, 0, 1)) = 3$. Of course, the calculation according to the formula (2) gives the same result.

*Theorem:* Let $x = (x_1, x_2, \cdots, x_n, \cdots)$ be a sequence of binary digits generated by a Bernoulli source with probabilities $\Pr\{x_n = 1\} = p$, $\Pr\{x_n = 0\} = 1 - p$, $p$ unknown but fixed, $0 < p < 1$. The proposed method for converting the input sequence $x$ into the sequence $z$ of independent equiprobable output symbols has the following properties:

i) the redundancy $r = O(1/N)$;
ii) the time of encoding per letter $T = O(\log^3 N \log\log N)$ bit operations;
iii) the memory size of the encoder $M = O(N \log^2 N)$ bits;

where $N$ is the length of block.

*Proof:* Obviously, the redundancy of Elias' method and the redundancy of our method are the same. This implies immediately the claim i).

For the sake of simplicity of the proof of ii) we assume that $\log N$ is an integer. All calculations are carried out with rational numbers and all $\rho_t^s$ and $\lambda_t^s$ are fractions and presented as pairs of integers. The Shönhage–Strassen method is used for multiplication (see [1]). For this method the time $\mathcal{T}(L)$ of multiplication of two binary numbers with $L$ digits (and the time of division into a number with $L$ digits) is given by

$$\mathcal{T}(L) = O(L \log L \log\log L). \tag{7}$$

It is not difficult to see that the notation of every $p(x_t/x_1, \cdots, x_{t-1})$ and $q(x_t/x_1, \cdots, x_{t-1})$ uses $2\log N$ bits ($\log N$ for the numerator and $\log N$ for the denominator). That is why the calculation of $\rho_t^1$, $t = 1, \cdots, N/2$ according to (5) takes $2(N/2)$ multiplications of numbers with $\log N$ digits and the calculation of $\lambda_t^1$, $t = 1, \cdots, N/2$ according to (5) and the common formula $a/b + c/d = (ad+bc)/(bd)$ takes $3(N/2)$ multiplications of numbers of length $\log N$ bits. The calculations of $\rho_t^2, \lambda_t^2$, $t = 1, \cdots, N/4$, take $5(N/4)$ multiplications of numbers of length $2\log N$ bits. Similarly, the calculation of $\rho_t^i, \lambda_t^i$, $t = 1, \cdots, N/2^i$ takes $5(N/2^i)$ multiplications of numbers of length $2^i \log N$ bits. From (7) we obtain that the general time of calculations is

$$\begin{aligned}
&(5N/2)O(\log N \log\log N \log\log\log N)\\
&+ (5N/4)O(2\log N \log(2\log N)\log\log(2\log N))\\
&+ \cdots (5N/2^i)O(2^i \log N \log(2^i \log N)\log\log(2^i \log N))\\
&+ \cdots + 5O(N\log N \log(N\log N)\log\log(N\log N))
\end{aligned}$$

bit operations. It is easy to see that the last value is not greater than

$$O(N\log^2 N \log(N\log N)\log\log(N\log N))$$

bit operations. It yields $O(\log^3 N \log\log N)$ bit operations per letter for the calculation of $\lambda_1^{\log N}$. In order to obtain $\mathrm{Num}\,(x_N)$ we should calculate the product $|S_k|\lambda_1^{\log N}$ from (6). Obviously, $|S_k| < 2^N$ and the binary notation of the numbers $|S_k|$ and $\lambda_1^{\log N}$ takes not more than $N \log N$ bits. From (7) we obtain that the time of calculation of $|S_k|\lambda_1^{\log N}$ is $O(\log^2 N \log\log N)$ bit operations per letter. Computing the codeword $code\,(x^N)$ using $\mathrm{Num}\,(x^N)$ takes not more than $O(1)$ bit operations per letter. Finally, we have the claim ii).

In order to estimate the memory size we note that when we calculate $\rho_k^i, \lambda_k^i$ we can store only $\rho_k^{i-1}, \lambda_k^{i-1}$, $i = 2, \cdots, \log N$, the same memory is used to store $\{\rho_k^{i-1}, \lambda_k^{i-1}, k = 1, \cdots, N/2^{i-1}\}$ and $\{\rho_k^i, \lambda_k^i, k = 1, \cdots, N/2^i\}$. The memory size required for computing the codeword $code\,(x^N)$ using $\mathrm{Num}\,(x^N)$ is $O(N)$ bits (we have to store the binary notation of the integer $|S_k| = \binom{N}{k}$). From this we easily obtain iii) and the theorem is proved.

Let us consider an example of constructing an unbiased random sequence from the given input sequence. Let

$$x = (1, 1, 1, 0, 1, 1, 0, 0, \cdots).$$

We shall encode the block of length $N = 4$. First, we compute $\mathrm{Num}((1, 1, 1, 0))$. According to (4)–(6) we obtain $\mathrm{Num}((1, 1, 1, 0)) = 3$. The binary notation of the integer $|S_3|$ is $(1, 0, 0)$. So $0 \le \mathrm{Num}((1, 1, 1, 0)) < 2^2$, and we have to take the first two binary digits of $\mathrm{Num}((1, 1, 1, 0))$ as a codeword $(code((1, 1, 1, 0)) = (1, 1))$. In the same way we calculate $code((1, 1, 0, 0)) = (0, 1)$. To obtain the output sequence $y$ we have to concatenate all codewords. Finally, we have the output sequence $y = (1, 1, 0, 1, \cdots)$.

## REFERENCES


[1] A. V. Aho, L. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA: Addison-Wesley, 1976.

[2] V. F. Babkin, "A method of universal coding with nonexponential labour consumption," *Probl. Inform. Transm.*, vol. 7, pp. 13–21, 1971.

[3] T. M. Cover, "Enumerative source encoding," *IEEE Trans. Inform. Theory*, vol. IT-10, pp. 73–77, Jan. 1973.

[4] L. D. Davisson, "Comments on 'Sequence time coding for data compression'," *Proc. IEEE*, vol. 54, p. 2010, 1966.

[5] P. Elias, "The efficient construction of an unbiased random sequence," *Ann. Math. Statist.*, vol. 43, no. 3, pp. 864–870, 1972.

[6] T. Y. Lynch, "Sequence time coding for data compression," *Proc. IEEE*, vol. 54, pp. 1490–1491, 1966.

[7] J. von Neumann, "Various techniques used in connection with random digits," *Nat. Bur. Stand. Appl. Math. Ser.*, vol. 12, pp. 36–38, 1951 (Reprinted in the Collected Works of von Neumann, vol. 5).

[8] B. Ya. Ryabko, "Fast and efficient coding of information sources," *IEEE Trans. Inform. Theory*, vol. 40, pp. 96–99, Jan. 1994.

[9] ——, "Fast enumeration of combinatorial objects," *Discr. Math. and its Appl.*, vol. 10, no. 2, pp. 101–110, 1998.

[10] Q. F. Stout and B. Warren, "Tree algorithms for unbiased coin tossing with a biased coin," *Ann. Probab.*, vol. 12, no. 1, pp. 212–222, 1984.

[11] S. Vembu and S. Verdú, "Generating random bits from arbitrary source: Fundamental limits," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1322–1332, Sept. 1995.


# Extraction of Optimally Unbiased Bits from a Biased Source

Mats Näslund and Alexander Russell


***Abstract*—We explore the problem of transforming $n$ independent and identically biased $\{-1, 1\}$-valued random variables $X_1, \cdots, X_n$ into a single $\{-1, 1\}$ random variable $f(X_1, \cdots, X_n)$, so that this result is as unbiased as possible. In general, no function $f$ produces a completely unbiased result. We perform the first study of the relationship between the bias $b$ of these $X_i$ and the *rate* at which $f(X_1, \cdots, X_n)$ can converge to an unbiased $\{-1, 1\}$ random variable (as $n \to \infty$).**

**A $\{-1, 1\}$ random variable has bias $b$ if $\mathbb{E}(X_i) = b$. Fixing a bias $b$, we explore the rate at which the output bias $|\mathbb{E}(f(X_1, \cdots, X_n))|$ can tend to zero for a function $f: \{-1, 1\}^* \to \{-1, 1\}$. This is accomplished by classifying the behavior of the natural normalized quantity**

$$\Xi(b) \triangleq \inf_f \left[\lim_{n\to\infty} \sqrt[n]{|\mathbb{E}(f(X_1, \cdots, X_n))|}\right]$$

**this infimum taken over all such $f$.**

**We show that for rational $b$, $\Xi(b) = (1/s)$, where $(1 + b/2) = (r/s)$ ($r$ and $s$ relatively prime). Developing the theory of uniform distribution of sequences to suit our problem, we then explore the case where $b$ is irrational. We prove a new metrical theorem concerning multidimensional Diophantine approximation type from which we show that for (Lebesgue) almost all biases $b$, $\Xi(b) = 0$. Finally, we show that algebraic biases exhibit curious "boundary" behavior, falling into two classes.**

**Class 1. Those algebraics $b$ for which $\Xi(b) > 0$ and, furthermore, $c_1 \le \Xi(b) \le c_2$ where $c_1$ and $c_2$ are positive constants depending only on $b$'s algebraic characteristics.**

**Class 2. Those algebraics $b$ for which there exist $n > 0$ and $f : \{-1, 1\}^n \to \{-1, 1\}$ so that $\mathbb{E}(f(X_1, \cdots, X_n)) = 0$.**

**Notice that this classification excludes the possibility that**

$$\sqrt[n]{|\mathbb{E}(f(X_1, \cdots, X_n))|}$$

***limits* to zero (for algebraics).**

**For rational and algebraic biases, we also study the computational problem by restricting $f$ to be a polynomial time computable function. Finally, we discuss natural extensions where output distributions other than the uniform distribution on $\{-1, 1\}$ are sought.**


***Index Terms*—Bias, computation, Diophantine approximation, randomness, random variables, type.**

## I. INTRODUCTION

The general problem of producing unbiased random bits from an imperfect random source has received enormous attention. This study essentially began with von Neumann [1] in 1951 and, following his work, a variety of models of such "imperfect sources" have been defined and studied. We study the problem of transforming $n$ independent random bits $X_1, \cdots, X_n$, each of fixed bias $b$, into a single bit

Manuscript received March 25, 1998; revised November 8, 1999. The material in this correspondence was presented in part at the IEEE Information Theory Workshop, Killarney, Ireland, June 1998. This work was performed at the Royal Institute of Technology, Stockholm, Sweden.

M. Näslund is with Ericsson Research, SE-164 80 Stockholm, Sweden (e-mail: mats.naslund@era-t.ericsson.se).

A. Russell is with Department of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269 USA (e-mail: acr@cse.uconn.edu).

Communicated by S. Shamai, Associate Editor for Shannon Theory.

Publisher Item Identifier S 0018-9448(00)02900-X.