SCIS 2017 2017 Symposium on
Cryptography and Information Security
Naha, Japan, Jan. 24 - 27, 2017
The Institute of Electronics,
Information and Communication Engineers

Numerical Analysis of Elias's and Peres's Deterministic Extractors

Amonrat Prasitsupparote [*] Norio Konno [†] Junji Shikata [‡]

Abstract: Generating truly random numbers is important in cryptography. Von Neumann proposed the simple procedure for extracting truly random bits from a sequence of independent, identically distributed random biased bits about half century ago. The improved algorithms of the von Neumann's extractor were later proposed by Elias and Peres. In this paper, our experimental results show that the theoretical and empirical redundancy are almost the same in each extractor. Furthermore, we show the redundancy function of Peres's extractor with iterations more than or equal to 5 is not concave. Next, we show that the redundancy of Peres's extractor is much better than Elias's extractor under the same input size and the almost same running time.

Keywords: True random number generation, von Neumann's extractor, Peres's extractor, Elias's extractor

1 Introduction

In 2012, Heninger et al. [7] and Lenstra et al. [8] explored RSA public-keys of TLS and SSH servers on the Internet. Their experiment showed the weak random numbers for generating random primes in embedded devices, thus the attackers able to reveal private keys. This tells us that, in RSA-keys generation random numbers are important, and RSA will be broken if there is not enough randomness to generate RSA-keys.

The random number generation is important to generate cryptographically secure keys as observed above. There are two basic types of generating random numbers: the true random number generator (TRNG) and the pseudo-random number generator (PRNG). TRNG is a deterministic algorithm which takes as inputs biased random sequences generated by utilizing physical phenomena and transforms them into unbiased random sequences. The PRNG is an algorithm that takes as inputs biased random sequences and seeds being short truly random sequences and outputs long pseudo-random sequences which are computationally indistinguishable from truly random sequences. In securely constructing cryptographic protocols having *computational security* (e.g., public-key cryptography), PRNG are usually used to generate secret-keys in the protocols, since PRNG can output long random sequences. However, TRNG are also important, since it is used for generating secret-keys in securely constructing cryptographic protocols having *information-theoretic security* and used even for generating seeds of PRNG.

[*] Graduate School of Environment and Information Sciences, Yokohama National University, Japan.
[†] Department of Applied Mathematics, Faculty of Engineering, Yokohama National University, Japan.
[‡] Graduate School of Environment and Information Sciences / Institute of Advanced Sciences, Yokohama National University, Japan.

Related Work. A deterministic extractor is a deterministic algorithm which takes non-uniformly random sequences as inputs and outputs uniformly random sequences. The previous works [9, 6, 11] usually consider the case that input sequences are given by the Bernoulli source $\mathsf{Bern}(p)$, where $\mathsf{Bern}(p)$ outputs $(x_1, x_2, \dots, x_n) \in \{0,1\}^n$ with i.i.d. (identically and independently distribution) according to $\Pr(x_i = 1) = p$ and $\Pr(x_i = 0) = q = 1 - p$ for some unknown $p \in (0,1)$. In this paper, we also consider that inputs of extractors are binary sequences output by $\mathsf{Bern}(p)$. The most famous extractor is the von Neumann's extractor [9]. He demonstrated a simple procedure for extracting independent *unbiased* bits from a sequence of independent, identically distributed *biased* bits.

An extension of the von Neumann's extractor was proposed by Elias [6] in 1971. The basic idea behind the Elias's method is to utilize a block coding technique to improve redundancy of the von Neumann's extractor. Elias's method requires exponential time and exponential memory size with respect to N, where N is block size, to store all 2^N input sequences with their assignment of output sequences. To improve time and space complexity of the Elias's method, Ryabko and Matchikina [12] proposed a fast method of implementing the Elias's method which we call *RM method* in this paper. The RM method utilizes enumerative encoding technique from [5] and Schonhage-Strassen algorithm [13] for fast integer multiplication in order to compute assignment of output sequences. By using RM method, time complexity and space complexity of Elias's extractor are improved.

In 1992, Peres [11] proposed the procedure for extracting independent unbiased bits from a sequence of independent, identically distributed biased bits which is an improved one from the von Neumann's extractor. The basic idea in Peres's extractor is to reuse the bit sequence which is discarded in the von Neumann's extractor by iterating similar procedures in the von Neumann's extractor.

In 2013, Pae [10] compared rates of Peres's extractor and Elias's extractor with probability $p = \frac{1}{3}$. Pae concludes Elias's extractor requires more computational costs than Peres's extractor, but there is a tradeoff between computational costs and rates. However, Pae did not consider any $p \in (0, 1)$, and one of purposes in this paper is to show results for any $p \in (0, 1)$.

In 2016, Chattopadhyay and Zuckerman [4] proposed a general two-sources extractors which each with polylogarithmic min-entropy. They combined two weak random sequences of numbers into a single sequence based on a K-Ramsey graph over N vertices and extract using a resilient function. Their extractor outputs one bit and achieve negligible error $1/n^{\Omega(1)}$.

Furthermore, most researchers interest in implement a randomness extractor in a real world. For example, in 2009, Bouda et al. [3] used mobile phones or pocket computers to generated a very close to true random data. They took 12 pictures per second then used their function for random 4 bits in each picture. After that used Carter-Wegman universal$_2$ classed of hash functions for extracted the output. Their output passed 15 out of 16 NIST statistical tests at the confidence level $\alpha = 0.01$. However, their proposed model was not simultaneous system, thus the inconvenience will be occur when apply it. Later in 2011, Voris et al. [14] investigated the use of accelerators on RFID tags as a source. They implemented the independent sources extractor and the matrix extraction which called chain extractors on RFID tags. It can produced 128 bits random output in 1.5 seconds and passed the NIST statistical tests. On the contrary, they can be change the chain extractors to the simple one because in practical terms a lot of uniform sequences is not necessary just only close to uniform, thus it can be produce an output less than a second.

Our Contribution. Recently, most researchers are interested in two-sources or fast seeded extractor (i.e., not deterministic ones). On the other hand, since the deterministic extractors by von Neumann, Elias, and Peres are fundamental and simple, a few researchers are interested in such extractors, especially from viewpoints of practical use. In this paper, we evaluate numerical performance of Peres's extractor and the Elias's extractor with RM method in term of practical aspects. Specifically, the contribution of the paper is as follows.

In Sections 4.1 and 4.2, the theoretical and experimental redundancy are the same results of Peres's extractor and Elias's extractor with RM method. In addition in Section 4.1, we show the redundancy function of Peres's extractor with iterations $\nu \geq 5$ is not concave in term of theoretical and experimental aspects. Next, we investigate the time complexity of both extractors. As a result, Peres's extractor with $\nu = 6$ and Elias's extractor with $N = 6$ have an almost same running time, then we compare the redundancy of both extractors under the almost same running time. This result shows that Peres's extractor with is much better than Elias's extractor under the almost same running time.

2 Preliminaries

A deterministic extractor is a deterministic algorithm which takes non-uniformly random sequences (i.e., biased sequences) as inputs and outputs uniformly random sequences (i.e., unbiased ones). For simplicity, the previous works [9, 6, 11] usually consider the case that input sequences are given by the Bernoulli source $\mathsf{Bern}(p)$, where $\mathsf{Bern}(p)$ outputs $(x_1, x_2, \ldots, x_n) \in \{0, 1\}^n$ with i.i.d. (identically and independently distribution) according to $\Pr(x_i = 1) = p$ and $\Pr(x_i = 0) = q = 1 - p$ for some unknown $p \in (0, 1)$. In this paper, we also consider that inputs of extractors are binary sequences output by $\mathsf{Bern}(p)$. Suppose that a deterministic extractor takes $(x_1, x_2, \ldots, x_n) \in \{0, 1\}^n$ as input and outputs $(y_1, y_2, \ldots, y_\ell) \in \{0, 1\}^\ell$. Then, we denote the average bit-length of outputs by $\bar{\ell}(n)$ which is a function of n, and define the rate function by $r(p) := \lim_{n\to\infty} \bar{\ell}(n)/n$. In addition, we define the redundancy function by $f(p) := h(p) - r(p)$, where $h(p)$ is the binary entropy function defined by $h(p) = -p \log p - (1 - p) \log(1 - p)$. We note that the above definition of redundancy functions is meaningful, since $h(p)$ is shown to be the information bound in [6, 11].

In this paper, we define non-asymptotic functions $r(p, n) := \bar{\ell}(n)/n$, and $f(p, n) := h(p) - r(p, n)$. Furthermore, we define the maximum redundancy by $\Gamma := \sup_{p\in(0,1)} f(p)$ and $\Gamma(n) := \sup_{p\in(0,1)} f(p, n)$ for every n.

2.1 von Neumann's extractor

In 1951, von Neumann [9] proposed the simple procedure for extracting independent *unbiased* bits from a sequence of independent, identically distributed *biased* bits, which is called the von Neumann's extractor. This method divides the input sequence $(x_1, x_2, x_3, x_4, \ldots)$ into the pairs[1] $((x_1x_2), (x_3x_4), \ldots)$ and considers a mapping defined by

$$00 \mapsto \wedge, \quad 01 \mapsto 0, \quad 10 \mapsto 1, \quad 11 \mapsto \wedge, \tag{1}$$

where the symbol $\wedge$ denotes no output bit. Then, the extractor outputs the unbiased sequence obtained by concatenating the outputs of the mapping (1).

Complexity. The von Neumann's extractor is efficient in the sense that both time complexity and space complexity are small: time complexity is evaluated as $O(n)$, and its space complexity is evaluated as $O(1)$.

Redundancy. The von Neumann's extractor is not desirable, since maximum redundancy is far from zero. Actually, the rate function $r^{\mathsf{vN}}(p)$ of the von Neumann's extractor is evaluated by $r^{\mathsf{vN}}(p) = \lim_{n\to\infty} npq/n = pq$, which is 1/4 at $p = q = 1/2$ and less elsewhere. In addition, the redundancy function $f^{\mathsf{vN}}(p)$ and maximum

[1] If n is odd, we discard the last bit.

redundancy Γ^{vN} are evaluated as $f^{\mathsf{vN}}(p) = h(p) - p(1-p), \Gamma^{\mathsf{vN}} = \sup_{p\in(0,1)} f^{\mathsf{vN}}(p) = f^{\mathsf{vN}}(1/2) = 3/4$.

2.2 Elias's extractor

An extension of the von Neumann's extractor was proposed by Elias [6] in 1971. The basic idea behind the Elias's method is to utilize a block coding technique to improve redundancy of the von Neumann's extractor.

Let N be the block size used in Elias's extractor, and it is a natural number $N \in \mathbb{N}$ with $N \geq 2$. For all binary sequences with length N, we first partition them into $N+1$ sets S_k $(k = 0, 1, 2, \dots, N)$, where S_k consists of all the $\binom{N}{k}$ sequences of length N which have k ones and $N-k$ zeros. Here, we note that each sequence of S_k is equiprobable, i.e., the probability is $p^k q^{N-k}$.

Let $|S_k| = \binom{N}{k} = \alpha_m 2^m + \alpha_{m-1}2^{m-1} + \dots + \alpha_0 2^0$ $(\alpha_i \in \{0,1\})$ be the binary expression of the integer $|S_k|$, and we briefly write $|S_k| = (\alpha_m, \alpha_{m-1}, \dots, \alpha_0)$ for it. For each j $(1 \leq j \leq m)$ such that $\alpha_j = 1$, we assign 2^j distinct output sequences of length j to 2^j distinct sequences of S_k which have not already been assigned. If $\alpha_0 = 1$, one sequence of S_k is assigned to $\wedge$. In particular, since $|S_0| = |S_N| = 1$, two sequences $(0,0,\dots,0)$ and $(1,1,\dots,1)$ are assigned to $\wedge$.

Example 1 *Suppose that the block size is $N = 4$. Then, we partition the set $\{0,1\}^4$ of possible input sequences into the following subsets:*

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

Then, we have $|S_0| = |S_4| = 1 = (1), |S_1| = |S_3| = 4 = (1,0,0), |S_2| = 6 = (1,1,0)$. For instance, we consider the following assignment of output sequences:

$$\begin{aligned}
&(0,0,0,0) \mapsto \wedge, && (1,1,1,1) \mapsto \wedge,\\
&(1,0,0,0) \mapsto (0,0), && (1,1,1,0) \mapsto (0,0),\\
&(0,1,0,0) \mapsto (0,1), && (1,0,1,1) \mapsto (1,0),\\
&(0,0,1,0) \mapsto (1,0), && (1,1,0,1) \mapsto (1,1),\\
&(0,0,0,1) \mapsto (1,1), && (0,1,1,1) \mapsto (0,1),\\
&(0,0,1,1) \mapsto (0,1), && (1,0,1,0) \mapsto (1,0),\\
&(0,1,1,0) \mapsto (0,0), && (1,0,0,1) \mapsto (1,1),\\
&(0,1,0,1) \mapsto (0), && (1,1,0,0) \mapsto (1).
\end{aligned}$$

Suppose that an input sequence $x = (1,0,0,1,0,0,1,1)$ is given. Since $N = 4$, the sequence is divided as $x = ((1,0,0,1),(0,0,1,1))$. Then, by the assignment of output sequences above, the output sequence is $y = ((1,1)(0,1)) = (1,1,0,1)$.

It should be noted that Elias's extractor with $N = 2$ is equivalent to the von Neumann's extractor, or equivalently the mapping (1). In this sense, Elias's extractor is an extension of the von Neumann's extractor.

Redundancy. In general, the rate function and redundancy function of the Elias's extractor depend on block size N. For given n-bit input sequence, if we take the maximum block size $N := n$, the rate function and maximum redundancy would be best. Hence, for simplicity, we assume that $N = n$ in the following explanation. The rate function $r^{\mathsf{E}}(p,N)$ is evaluated by

$$r^{\mathsf{E}}(p,N) \approx \frac{1}{N}\sum_{k=0}^{N}\binom{N}{k}p^k(1-p)^{N-k}\log\binom{N}{k} \qquad (2)$$

Elias [6] showed that the rate function[2] $r^{\mathsf{E}}(p,N)$ of the Elias's extractor converges to $h(p)$ as $N \to \infty$, or equivalently, the redundancy function $f^{\mathsf{E}}(p,N) := h(p) - r^{\mathsf{E}}(p,N)$ converges to zero as $N \to \infty$. More precisely, it was shown that $f^{\mathsf{E}}(p,N) = O(1/N)$ for any fixed p.

Complexity. A naive construction of the Elias's extractor requires much space complexity and time complexity to make a table of the assignment of output sequences as illustrated by Example 1. Actually, it requires exponential time and exponential memory size with respect to N to store all 2^N input sequences with their assignment of output sequences. To improve time and space complexity of the Elias's extractor, Ryabko and Matchikina [12] proposed a fast method of implementing the Elias's extractor which we call *RM method* in this paper. The RM method utilizes enumerative encoding technique from [5] and Schonhage-Strassen algorithm [13] for fast integer multiplication in order to compute assignment of output sequences instead of making a large table as illustrated by Example 1. The RM method is executed as follows. Suppose that a binary input sequence $x^N = (x_1, x_2, \dots, x_N)$ contains k ones and $N-k$ zeros. Then, the number $\mathrm{Num}(x^N)$ is defined by

$$\mathrm{Num}(x^N) = \sum_{t=1}^{N}\binom{x_t N - t}{k - \sum_{i=1}^{t-1} x_i}. \qquad (3)$$

Then, a binary codeword $\mathit{code}(x^N)$ of x^N, which is assignment of an output sequence of x^N, is computed as follows:

(i) Compute $\mathrm{Num}(x^N)$ in the set S_k, if x^N contains k ones.

(ii) Let $|S_k| = \binom{N}{k} = 2^{j_0} + 2^{j_1} + \dots + 2^{j_m}$ for $0 \leq j_0 < j_1 < \dots < j_m$.

(iii) If $j_0 = 0$ and $\mathrm{Num}(x^N) = 0$, then $\mathit{code}(x^N) = \wedge$.

(iv) If $0 \leq \mathrm{Num}(x^N) < 2^{j_0}$, then $\mathit{code}(x^N)$ is defined to be the j_0 low-order binary string of $\mathrm{Num}(x^N)$.

(v) If $\sum_{s=0}^{t} 2^{j_s} \leq \mathrm{Num}(x^N) < \sum_{s=0}^{t} 2^{j_s} + 2^{j_{t+1}}$ for some $0 \leq t \leq m$, then $\mathit{code}(x^N)$ is defined to be the suffix consisting of the j_{t+1} binary string of $\mathrm{Num}(x^N)$.

[2] In Elias's paper [6], it is called *efficiency*.

By using RM method, time complexity and space complexity of Elias's extractor are improved as follows: Time complexity is $O(N \log^3 N \log\log N)$, and space complexity is $O(N \log^2 N)$ (see [12] for details).

2.3 Peres's extractor

The basic idea in Peres's extractor is to reuse the bit sequence which is discarded in the mapping (1) to improve redundancy of the von Neumann's extractor. In the following, we denote the von Neumann's extractor by Ψ_1. For an n-bit sequence $(x_1, x_2, \ldots, x_n)$, we describe the von Neumann's extractor by

$$\Psi_1(x_1, x_2, \ldots, x_n) = (y_1, y_2, \ldots, y_\ell),$$

where $y_i = x_{2m_i - 1}$ and $m_1 < m_2 < \cdots < m_\ell$ are all the indices satisfying $x_{2m_i-1} \neq x_{2m_i}$ with $m_i \leq n/2$. In Peres's extractor, Ψ_ν ($\nu \geq 2$) is defined inductively as follows: For an even n,

$$\begin{aligned}\Psi_\nu(x_1, x_2, \ldots, x_n) \quad = \quad & \Psi_1(x_1, x_2, \ldots, x_n) * \\ & \Psi_{\nu-1}(u_1, u_2, \ldots, u_{\frac{n}{2}}) * \\ & \Psi_{\nu-1}(v_1, v_2, \ldots, v_{\frac{n}{2}-\ell}), \quad (4)\end{aligned}$$

where $*$ is concatenation; $u_j = x_{2j-1} \oplus x_{2j}$ for $1 \leq j \leq n/2$; $v_s = x_{2i_s - 1}$ and $i_1 < i_2 < \cdots < i_{\frac{n}{2}-\ell}$ are all the indices satisfying $x_{2i_s-1} = x_{2i_s}$ with $i_s \leq n/2$. For an odd input size n, $\Psi_\nu(x_1, x_2, \ldots, x_n) := \Psi_\nu(x_1, x_2, \ldots, x_{n-1})$, i.e., the last bit is discarded and utilize the case of an even n above.

Note that, in Peres's extractor, the number of iterations ν is at most $\lfloor \log n \rfloor$, since Ψ_ν for every $\nu \geq 2$ is defined by $\Psi_{\nu-1}$ having an input sequence whose bit-length is at most $n/2$, i.e., the bit-length of both $(u_1, u_2, \ldots, u_{\frac{n}{2}})$ and $(v_1, v_2, \ldots, v_{\frac{n}{2}-\ell})$ in the equation (4) is at most $n/2$. Obviously, Peres's extractor with $\nu = 1$ is the same as the von Neumann's extractor, and Peres's extractor with a large ν (say, $\nu = \lfloor \log n \rfloor$) is considered to be an elegantly improved version from von Neumann's one by utilizing a recursion mechanism.

Complexity. We denote time complexity of Ψ_ν by $T_\nu(n)$. By the equation (4), we have

$$T_\nu(n) = T_1(n) + n/2 + T_{\nu-1}(n/2) + T_{\nu-1}(n/2 - \ell), \quad (5)$$

and $T_1(n) = O(n)$ (see Section 2.1 for time complexity of the von Neumann extractor). From the condition (5), we obtain $T_\nu(n) = O(\nu n)$ for Ψ_ν with $1 \leq \nu \leq \lfloor \log n \rfloor$. In particular, time complexity of Peres's extractor with the maximum iterations $\nu = \lfloor \log n \rfloor$ is evaluated as $T_\nu(n) = O(n \log n)$.

Redundancy. The rate function $r^{\mathsf{P}}_\nu(p)$ of the procedure Ψ_ν can be computed inductively by the equation

$$\begin{aligned} r^{\mathsf{P}}_\nu(p) \quad &= pq + \tfrac{1}{2} r^{\mathsf{P}}_{\nu-1}(p^2+q^2) + \\ & \tfrac{1}{2}(p^2+q^2) r^{\mathsf{P}}_{\nu-1}\left(\tfrac{p^2}{p^2+q^2}\right) \qquad (6)\end{aligned}$$

for $\nu \geq 2$, and $r^{\mathsf{P}}_1(p) = pq$. Note that $r^{\mathsf{P}}_1(p)$ is the rate of the von Neumann's extractor. It is shown in [11] that $r^{\mathsf{P}}_\nu(p) \leq r^{\mathsf{P}}_{\nu+1}(p)$ for all $\nu \in \mathbb{N}$ and for all $p \in (0,1)$, and $\lim_{\nu\to\infty} r^{\mathsf{P}}_\nu(p) = h(p)$ uniformly in $p \in (0,1)$.

In other words, the above result is described in terms of redundancy as follows: The redundancy function $f^{\mathsf{P}}_\nu(p) = h(p) - r^{\mathsf{P}}_\nu(p)$ satisfies

$$f^{\mathsf{P}}_\nu(p) = \frac{1}{2} f^{\mathsf{P}}_{\nu-1}(p^2+q^2) + \frac{1}{2}(p^2+q^2) f^{\mathsf{P}}_{\nu-1}\left(\frac{p^2}{p^2+q^2}\right), \quad (7)$$

for $\nu \geq 2$ and $f^{\mathsf{P}}_1(p) = h(p) - p(1-p)$, where the above equation (7) follows from the equation (6). Furthermore, it holds that $f^{\mathsf{P}}_\nu(p) \geq f^{\mathsf{P}}_{\nu+1}(p)$ for all $\nu \in \mathbb{N}$ and for all $p \in (0,1)$, and $\lim_{\nu\to\infty} f^{\mathsf{P}}_\nu(p) = 0$ uniformly in $p \in (0,1)$.

3 Theoretical Analysis of Peres's extractor

Let $f^{\mathsf{P}}_\nu(p) = h(p) - r^{\mathsf{P}}_\nu(p)$ be the redundancy function for Peres's extractor with ν iterations. Then, we show that $f^{\mathsf{P}}_\nu(p)$ is not concave in $p \in (0,1)$ for $\nu \geq 5$ as follows. The proof is given in Appendix A.

Proposition 1 *The redundancy function $f^{\mathsf{P}}_\nu(p)$ in the Peres's extractor with ν iterations is not concave in $p \in (0,1)$ if $\nu \geq 5$. Generally, for the Peres's extractor with ν iterations, the redundancy function $f^{\mathsf{P}}_\nu(p)$ satisfies*

$$\frac{d^2 f^{\mathsf{P}}_\nu(\frac{1}{2})}{dp^2} = 8 - \frac{4}{\ln 2} - 6\left(\frac{3}{4}\right)^{\nu-1}. \quad (8)$$

In particular, $\frac{d^2 f^{\mathsf{P}}_\nu}{dp^2}\left(\frac{1}{2}\right) < 0$ for $1 \leq \nu \leq 4$ and $\frac{d^2 f^{\mathsf{P}}_\nu}{dp^2}\left(\frac{1}{2}\right) > 0$ for $\nu \geq 5$.

4 Numerical Analysis

To evaluate the performance of Peres's extractor and Elias's extractor with RM method, we use Java language version 1.8 for our implementation in the environment of Intel 3.70 GHz with RAM 4 GB on Window 8 64 bits. We consider four kinds of questions:

(i) Do theoretical and experimental redundancy of the Peres's extractor and Elias's extractor with RM method show the same results?

(ii) Proposition 1 shows that the redundancy function $f^{\mathsf{P}}_\nu(p)$ ($\nu \geq 5$) of the Peres's extractor is not concave, but what is an actual graph of the function in the interval $(0,1)$?

(iii) What is the actual difference of running time required in Peres's extractor and Elias's extractor with RM method?

(iv) Under the almost same running time, which extractor is better in terms of redundancy?

We perform analysis to answer the questions as follows.

Firstly, we analyze redundancy of Peres's extractor in Section 4.1. This experiment uses probability $p =$

$0.001, 0.002, \ldots, 0.999$. In a real world, we cannot exactly know probability p, thus we use a pseudo-random number generation program **rand()** in MATLAB [2] to generate biased input sequences by controlling the probability p, since it can produce a random numbers by changing p. The reasonability of using a pseudo-random number by **rand()** is; first, we can control the probability p for each input sequence; second, if theoretical and experimental redundancy results are almost the same, we can rely on **rand()** and can use it to generate biased input sequences. Specifically, we generate 180-bit input sequences 100 times for each p. Then, we calculate the average on $f_\nu^{\mathsf{P}}(p)$ for each p. The number of iterations satisfies $\nu \leq \lfloor \log 180 \rfloor = 6$, and we investigate $f_\nu^{\mathsf{P}}(p)$ for $\nu = 1, 2, \ldots, 6$.

Secondly, we analyze redundancy of the Elias's extractor with RM method in Section 4.2. This experiment use Schonhage-Strassen algorithm of fast multiplication for computing $\binom{N}{k}$. For computing $\binom{N}{k}$, we consider the following:

1. Schonhage-Strassen multiplication algorithm requires $O(N^{1+\epsilon})$ which is asymptotically better than $O(N^2)$ (i.e., the normal multiplication). However, the advantage of Schonhage-Strassen method over the normal multiplication seems to appear when N is large enough. In addition, there may be the case that Schonhage-Strassen multiplication algorithm is not supported in some software, and in this case users need to implement it by themselves.

2. We want to avoid multiplication operations and use only addition operations, since it is simple and makes the basic operations lighter that it can be used in various applications and environments.

By taking into account the above remarks, we use recursive formula $\binom{N}{k} = \binom{N-1}{k-1} + \binom{N-1}{k}$ for $10 \leq N \leq 180$ in order to compute $\binom{N}{k}$ only by additions. We use **rand()** to generate 180-bit input sequences 100 times for each $p = 0.1, 0.2, \ldots, 0.9$. Then, we calculate the average on $f_N^{\mathsf{E}}(p)$ for each p with $N = 10, 20, 30, 60, 90, 180$.

Thirdly, we analyze running time of both extractors in Section 4.3. This experiment does not use p as a parameter, thus we change the random number generator to RANDOM.ORG [1]. It produces the sequences close to true random with unknown p by using randomness of atmospheric noises. It provides 131,072 random bits in each time, thus we set the bit-length of inputs $n = 100, 200, \ldots, 1000$ 100 times for each n, then we calculate the average on the running time.

Finally, we compare the redundancy of both extractors under the almost same running time in Section 4.4. We can clarify which is better in practice in terms of redundancy under the almost same time complexity.

4.1 Analysis redundancy of Peres's extractor

We show the redundancy of Peres's extractor from theoretical aspects in Fig. 1. We calculated $f_\nu^{\mathsf{P}}(p)$ by using (7) with $\nu = 1, 2, \ldots, 6$ and $p = 0.1, 0.2, \ldots, 0.9$, then depicted the graphs of $f_\nu^{\mathsf{P}}(p)$, where x-axis means probability p and y-axis means redundancy. It can be seen that, the redundancy becomes smaller as the number of iterations becomes bigger, for all $p \in (0, 1)$.

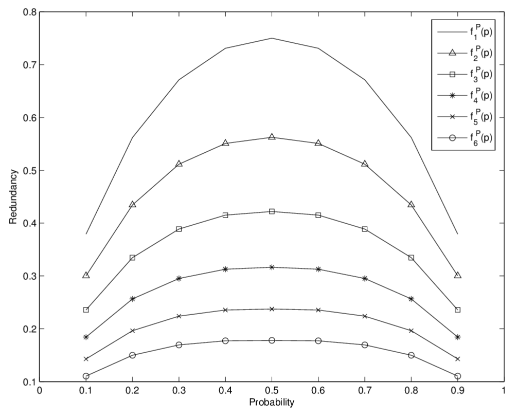


Figure 1: Theoretical estimate.

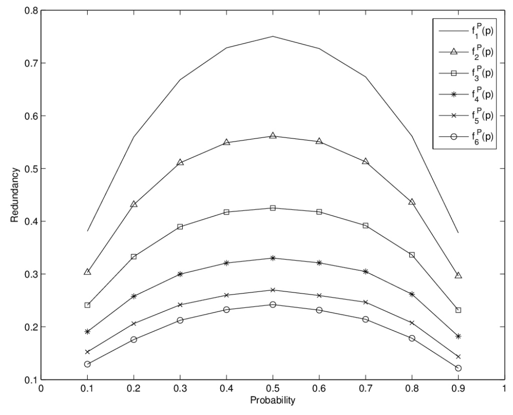
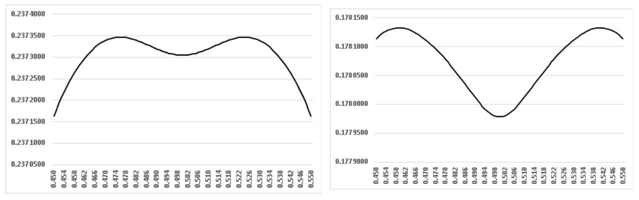


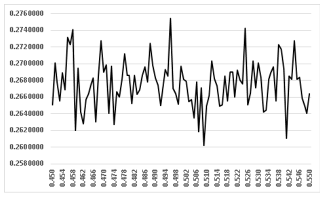
Figure 2: Experimental estimate with 180-bit.



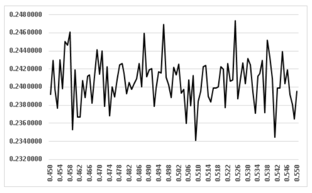
(a) Graph of $f_5^{\mathsf{P}}(p)$ (b) Graph of $f_6^{\mathsf{P}}(p)$

Figure 3: Theoretical estimate on redundancy of Peres's extractor with $\nu = 5, 6$ and $p = 0.450, 0.454, \ldots, 0.550$.

In Fig. 1, the graphs of $f_\nu^{\mathsf{P}}(p)$ $(1 \leq \nu \leq 4)$ are concave, while the graphs of $f_\nu^{\mathsf{P}}(p)$ $(\nu = 5, 6)$ are not concave though they look to be concave. Actually, in Fig. 3, we can see that $f_\nu^{\mathsf{P}}(p)$ $(\nu = 5, 6)$ is not concave as theoretically shown by Proposition 1, where x-axis means probability $0.450 \leq p \leq 0.550$. In addition, it is observed that each of $f_5^{\mathsf{P}}(p)$ and $f_6^{\mathsf{P}}(p)$ takes the maximum at two different points. Specifically, our experiment shows that: $\max f_5^{\mathsf{P}}(p) \approx 0.2373467$ at $p \approx 0.476$ and



(a) Graph of $f_5^{\mathsf{P}}(p)$



(b) Graph of $f_6^{\mathsf{P}}(p)$

Figure 4: Experimental estimate on redundancy of Peres's extractor for 180-bit input sequences with $\nu = 5, 6$ and $p = 0.450, 0.454, \ldots, 0.550$.

$p \approx 0.524$; and $\max f_6^{\mathsf{P}}(p) \approx 0.1781326$ at $p \approx 0.459$ and $p \approx 0.541$. In Fig. 2, we show experimental redundancy of Peres's extractor with 180-bit inputs. The results are almost the same as theoretical in Fig. 1. Even in our experiment in Fig. 4, it is observed that the graph of $f_\nu^{\mathsf{P}}(p)$ ($\nu = 5, 6$) are not concave but there is much fluctuation, although $f_\nu^{\mathsf{P}}(p)$ ($\nu = 5, 6$) in Fig. 2 look concave.

4.2 Analysis of redundancy of Elias's extractor with RM method

In Fig. 5, we calculated $f^E(p, N) = h(p) - r^E(p, N)$ by using (2) with $p = 0.1, 0.2, \ldots, 0.9$ and $N = 10, 20, 30, 60, 90, 180$. It can be easily seen that redundancy becomes smaller as block size becomes larger. There is slightly difference between theoretical estimate in Fig. 5 and experimental estimate in Fig. 6, thus, the experimental redundancy in our implementation is similar to theoretical redundancy.

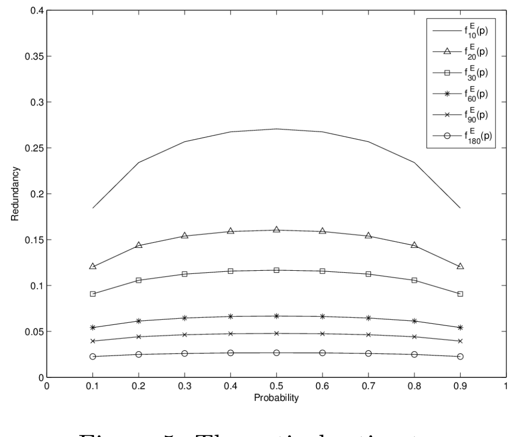


Figure 5: Theoretical estimate.

4.3 Analysis running time of both extractors

Fig. 7 shows the running time of Peres's extractor with $\nu = 1, 2, \ldots, 6$. As the number of iterations becomes larger, the running time is required more. In addition, the running time increases almost linearly but slope depends on ν, as supported by theoretical estimate of time complexity $O(\nu n)$. Furthermore, the running time of Peres's extractor with all parameters in our experiment is at most 0.32 milliseconds, which implies that the Peres's extractor is quite practical.

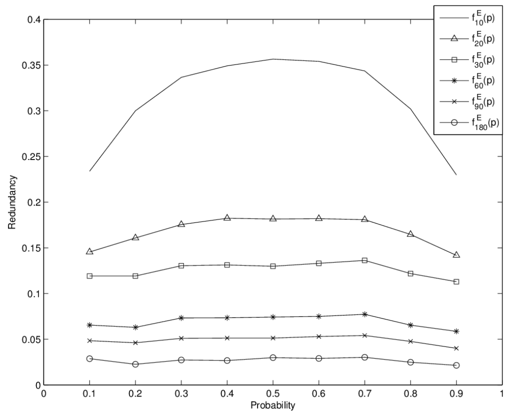


Figure 6: Experimental estimate with 180-bit.

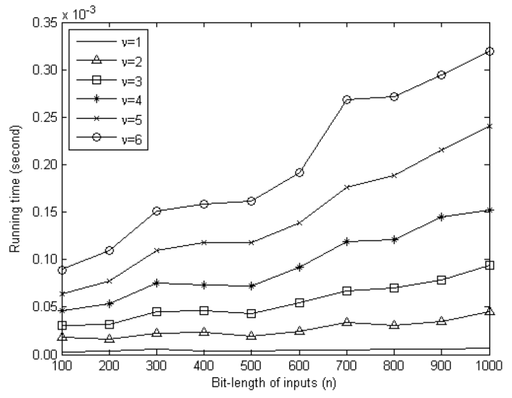


Figure 7: Peres's extractor.

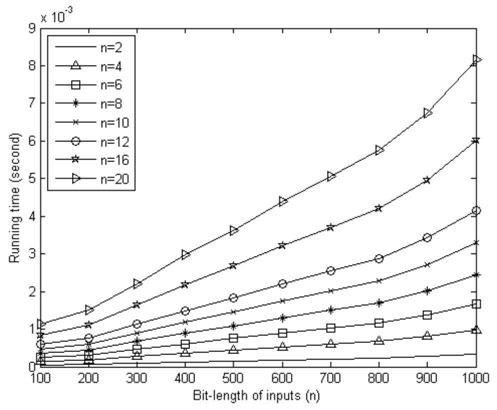


Figure 8: Elias's extractor with RM method.

Fig. 8 shows the running time of Elias's extractor with RM method with $N = 2, 4, 6, 8, 10, 12, 16, 20$. As the block size becomes larger, the running time is required more. The running time increases almost linearly but slope depends on N, as time complexity can be theoretically evaluated as $(n/N) \cdot O(N^2 \log N) = O(nN \log N)$. Furthermore, the running time of Elias's extractor with RM method with all parameters in our experiment is at most 8.1 milliseconds. By comparing the running time of both extractors, we can conclude that the Peres's extractor is faster than the Elias's extractor with RM method for the same bit-length of inputs.

4.4 Comparison of redundancy under the almost same running time

From Fig. 7 and 8, the running time of Peres's extractor with $\nu = 6$ is almost the same as Elias's extractor with RM method $N = 6$. Hence, we compare $f_6^{\mathsf{P}}(p)$ and $f_6^{\mathsf{E}}(p)$ in terms of practical aspects that shown in Fig. 9. It shown that Peres's extractor is much better than Elias's extractor with RM method in terms of redundancy, though Peres's extractor is worse than Elias's extractor in terms of maximum redundancy in theoretical analysis in Section 3[3].

Furthermore, we depicted the graph of $f_N^{\mathsf{E}}(p)$ with $N = 10, 20$ in addition to $f_6^{\mathsf{E}}(p)$ in Fig. 9. This result shows that: even if $f_{10}^{\mathsf{E}}(p)$ is allowed to use, $f_6^{\mathsf{P}}(p)$ is better than $f_{10}^{\mathsf{E}}(p)$; if $f_{20}^{\mathsf{E}}(p)$ is allowed to use, $f_6^{\mathsf{P}}(p)$ is almost the same as $f_{20}^{\mathsf{E}}(p)$, though running time of Elias's extractor with RM method $N = 10, 20$ is quite larger than Peres's extractor with $\nu = 6$.

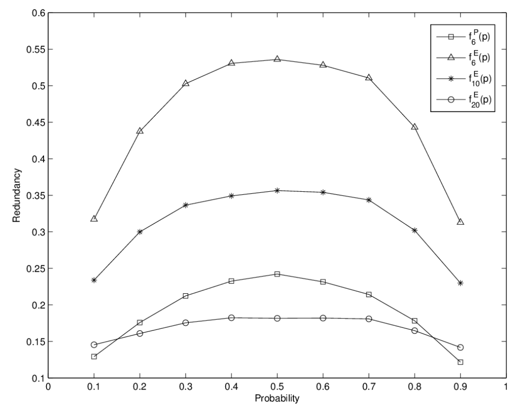


Figure 9: Comparison of redundancy of Peres's extractor and Elias's extractor with RM method with 180-bit inputs.

5 Conclusion

We evaluated numerical performance of Peres's extractor and Elias's extractor with RM method. For analysis of the redundancy, we used a pseudo-random number from **rand()** in MATLAB to generate 180-bit biased input sequences by controlling the probability p, then calculated the redundancy of both extractors in term of theoretical and experimental aspects (see Fig 1-6). Our result shows that the theoretical and experimental redundancy are almost the same results of both extractors. For analysis of the running time, our results shows that Peres's extractor was faster than Elias's extractor with RM method, and the redundancy of Peres's extractor was much better than Elias's extractor with RM method under the almost same running time. Consequently, Peres's extractor will be better to use in applications such as cryptography.

[3] Note that we do not take into account time complexity in Section 3, and the results in Section 3 and this section are not contradicting.

Acknowledgments. This work was in part supported by JSPS KAKENHI Grant Number 15H02710, and in part conducted under the auspices of the MEXT Program for Promoting the Reform of National Universities.

References

[1] *RANDOM.ORG - Byte Generator.*

[2] *Uniformly distributed random numbers - MATLAB rand.*

[3] Jan Bouda, Jan Krhovjak, Vashek Matyas, and Petr Svenda, *Towards True Random Number Generation in Mobile Environments*, Identity and Privacy in the Internet Age (Audun Jsang, Torleiv Maseng, and Svein Johan Knapskog, eds.), Lecture Notes in Computer Science, no. 5838, Springer Berlin Heidelberg, October 2009, DOI: 10.1007/978-3-642-04766-4_13, pp. 179–189 (en).

[4] Eshan Chattopadhyay and David Zuckerman, *Explicit Two-source Extractors and Resilient Functions*, Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (New York, NY, USA), STOC 2016, ACM, 2016, pp. 670–683.

[5] T. Cover, *Enumerative source encoding*, IEEE Transactions on Information Theory **19** (1973), no. 1, 73–77.

[6] Peter Elias, *The Efficient Construction of an Unbiased Random Sequence*, Ann. Math. Statist. **43** (1972), no. 3, 865–870 (EN).

[7] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman, *Mining your Ps and Qs: Detection of widespread weak keys in network devices*, Proceedings of the 21st USENIX Security Symposium, August 2012.

[8] Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter, *Public Keys*, Advances in Cryptology CRYPTO 2012 (Reihaneh Safavi-Naini and Ran Canetti, eds.), Lecture Notes in Computer Science, no. 7417, Springer Berlin Heidelberg, 2012, DOI: 10.1007/978-3-642-32009-5_37, pp. 626–642 (en).

[9] John 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.

[10] Sung-il Pae, *Exact output rate of Peres's algorithm for random number generation*, Information Processing Letters **113** (2013), no. 56, 160–164.

[11] Yuval Peres, *Iterating Von Neumann's Procedure for Extracting Random Bits*, Ann. Statist. **20** (1992), no. 1, 590–597 (EN). MR MR1150365

[12] B.Ya. Ryabko and E. Matchikina, *Fast and efficient construction of an unbiased random sequence*, IEEE Transactions on Information Theory **46** (2000), no. 3, 1090–1093.

[13] A. Schnhage and V. Strassen, *Schnelle multiplikation groer zahlen*, Computing **7**, no. 3-4, 281–292 (de).

[14] Jonathan Voris, Nitesh Saxena, and Tzipora Halevi, *Accelerometers and Randomness: Perfect Together*, Proceedings of the Fourth ACM Conference on Wireless Network Security (New York, NY, USA), WiSec '11, ACM, 2011, pp. 115–126.

Appendix A: Proof of Proposition 1

Let n be a large natural number. For a natural number $\nu \in \mathbb{N}$, we define $a_\nu := r^{\mathsf{P}}_\nu(1/2)$. Then, by the equation (6) we have

$$a_1 = \frac{1}{4}, \qquad a_\nu = \frac{1}{4} + \frac{3}{4}a_{\nu-1} \text{ for } \nu \ge 2.$$

By solving the equation above, we have

$$a_\nu = 1 - \left(\frac{3}{4}\right)^\nu \text{ for } \nu \ge 1. \tag{9}$$

We note that, for $\nu \ge 1$,

$$f^{\mathsf{P}}_\nu(1/2) = h(1/2) - r^{\mathsf{P}}_\nu(1/2) = \left(\frac{3}{4}\right)^\nu, \tag{10}$$

where the last equality follows from (9).

For $p \in (0,1)$, we define $\tilde{p} := p^2 + (1-p)^2$ and $\hat{p} := p^2/\tilde{p}$. Then, it holds that

$$\frac{d\tilde{p}}{dp} = 2(2p-1), \quad \frac{d\hat{p}}{dp} = \frac{2p(1-p)}{\tilde{p}^2}. \tag{11}$$

Next, for the first order derivative of $f^{\mathsf{P}}_\nu(p)$, we have

$$\frac{df^{\mathsf{P}}_1(p)}{dp} = \frac{1}{\ln 2}\ln\frac{1-p}{p} + 2p - 1, \tag{12}$$

$$\frac{df^{\mathsf{P}}_\nu(p)}{dp} = (2p-1)\left(f^{\mathsf{P}}_{\nu-1}(\hat{p}) + \frac{df^{\mathsf{P}}_{\nu-1}(\tilde{p})}{dp}\right) + \frac{p(1-p)}{\tilde{p}}\frac{df^{\mathsf{P}}_{\nu-1}(\hat{p})}{dp} \quad \text{for } \nu \ge 2. \tag{13}$$

Then, by setting $p = 1/2$ in (13), for $\nu \ge 2$, we have

$$\begin{aligned} \frac{df^{\mathsf{P}}_\nu(1/2)}{dp} &= \frac{1}{2}\frac{df^{\mathsf{P}}_{\nu-1}(1/2)}{dp} \qquad (14)\\ &= \left(\frac{1}{2}\right)^{\nu-1}\frac{df^{\mathsf{P}}_1(1/2)}{dp} \\ &= 0, \qquad (15) \end{aligned}$$

where (14) follows from (13), and (15) follows from (12).

Moreover, for the second order derivative of $f^{\mathsf{P}}_\nu(p)$, we obtain

$$\frac{d^2 f^{\mathsf{P}}_1(p)}{dp^2} = -\frac{1}{\ln 2}\frac{1}{p(1-p)} + 2, \tag{16}$$

$$\begin{aligned} \frac{d^2 f^{\mathsf{P}}_\nu(p)}{dp^2} &= 2f^{\mathsf{P}}_{\nu-1}(\hat{p}) + 2\frac{df^{\mathsf{P}}_{\nu-1}(\tilde{p})}{dp} + \frac{1-2p}{\tilde{p}}\frac{df^{\mathsf{P}}_{\nu-1}(\hat{p})}{dp} \\ &\quad + 2(2p-1)^2\frac{d^2 f^{\mathsf{P}}_{\nu-1}(\tilde{p})}{dp^2} \\ &\quad + \frac{2p^2(1-p)^2}{\tilde{p}^3}\frac{d^2 f^{\mathsf{P}}_{\nu-1}(\hat{p})}{dp^2} \quad \text{for } \nu \ge 2. \qquad (17) \end{aligned}$$

And, by setting $p = 1/2$ in (17), for $\nu \ge 2$, we have

$$\begin{aligned} \frac{d^2 f^{\mathsf{P}}_\nu(1/2)}{dp^2} &= 2f^{\mathsf{P}}_{\nu-1}(1/2) + 2\frac{df^{\mathsf{P}}_{\nu-1}(1/2)}{dp} + \frac{d^2 f^{\mathsf{P}}_{\nu-1}(1/2)}{dp^2} \\ &= 2\left(\frac{3}{4}\right)^{\nu-1} + \frac{d^2 f^{\mathsf{P}}_{\nu-1}(1/2)}{dp^2}, \qquad (18) \end{aligned}$$

where the first equality follows from (17), and the second equality (18) follows from (10) and (15). Then, by solving the equation (18) ($\nu \ge 2$) and $\frac{d^2 f^{\mathsf{P}}_1(1/2)}{dp^2} = 2 - 4/\ln 2$, we get

$$\begin{aligned} \frac{d^2 f^{\mathsf{P}}_\nu(1/2)}{dp^2} &= \frac{d^2 f^{\mathsf{P}}_1(1/2)}{dp^2} + 2\sum_{k=1}^{\nu-1}\left(\frac{3}{4}\right)^k \\ &= 2 - \frac{4}{\ln 2} + 6\left\{1 - \left(\frac{3}{4}\right)^{\nu-1}\right\} \\ &= 8 - \frac{4}{\ln 2} - 6\left(\frac{3}{4}\right)^{\nu-1}. \qquad (19) \end{aligned}$$

From the equation (19), it follows that $\frac{d^2 f^{\mathsf{P}}_\nu(1/2)}{dp^2} < 0$ for $1 \le \nu \le 4$, and $\frac{d^2 f^{\mathsf{P}}_\nu(1/2)}{dp^2} > 0$ for $\nu \ge 5$.