Online Entropy Estimation for Non-Binary Sources and Applications on iPhone

Cédric Lauradoux, Julien Ponge, Andrea Roeck

Online Entropy Estimation for Non-Binary Sources and Applications on iPhone

Cédric Lauradoux — Julien Ponge — Andrea Roeck

N° 7663

June 2011

Networks and Telecommunications

ISSN 0249-6399 ISRN INRIA/RR--7663--FR+ENG

Online Entropy Estimation for Non-Binary Sources and Applications on iPhone

Cédric Lauradoux , Julien Ponge , Andrea Roeck

Theme : Networks and Telecommunications
Networks, Systems and Services, Distributed Computing
Équipe-Projet SWING

Rapport de recherche n° 7663 — June 2011 — 19 pages

Abstract: The design of a random number generator is a challenging task on systems in changing environment such as smartphones. Finding reliable and high-throughput sources of entropy is difficult. This paper proposes an online entropy estimation algorithm to test the quality of an entropy source when nothing is known *a priori* on the source statistics. Our estimator can be executed at a low cost and is adapted for any type of sources. It extends the results of Bucci and Luzzi to non-binary sources and introduces a parameter that allows to trade time and memory for a better estimate. Our estimator is then applied to several sources available on an iPhone and compare to the state of the art.

Key-words: Random number generator, online entropy estimation, iPhone, random sources.

Online Entropy Estimation for Non-Binary Sources and Applications on iPhone

Résumé : La conception de générateur aléatoire est un challenge ayant un environnent changeant tel que les smartphones. Dans ces environnements, trouver des sources d'entropie fiables et ayant un haut débit est très difficile. Nous proposons dans ce rapport un estimateur d'entropie qui permet de tester la qualité d'une source a la volée et sans faire d'hypothèses *a priori* sur les caractéristiques statistiques de la source. Notre estimateur peut être exécuté à faible coût et il s'adapte a n'importe quelle type de source. Notre algorithme est une généralisation des résultats de Bucci et Luzzi a des sources non-binaires qui permet aussi de faire un compromis entre la ressource consommée par l'algorithme (temps CPU et mémoire) et la précision de l'estimation. Notre estimateur est testé pour différentes sources sur un iPhone.

Mots-clés : Générateur de nombre aléatoire cryptographique, estimateur d'entropie, iPhone, source d'aléa.

1 Introduction

Random numbers are the backbone of many security solutions and random number generators (RNGs) must be designed carefully. Since the NETSCAPE case [1], many RNGs weaknesses have been discovered on operating systems such as LINUX [2] and WINDOWS [3, 4], on numerous cryptographic libraries [5] and on other environment such as JAVA [6] or PHP [7]. The conclusion of all these "failures" is that inadequate choices for the RNG can annihilate all the efforts to enforce security. The cryptographic primitive considered in this paper is known under several names in the literature like *cryptographic* RNG, *true* RNG, physical RNG or pseudo random number generator (PRNG) with entropy input. In the remaining part, we simply use the term RNG without distinction as all these primitives are concerned by entropy estimation.

The research on RNGs has been particularly active in the latest decade. Several original designs have been proposed such as YARROW-160 [8], FORTUNA [9], HAVEGE [10] or TINYRNG [11] to name a few. Roughly speaking, the design of an RNG is composed of three elements: (1) some entropy sources, (2) a sampling component that gathers the entropy and (3) a cryptographic post-processing to thwart some attacks. Practical high-rate entropy sources have been studied by the hardware/software community: physically unclonable functions [12], air turbulence in disk drive [13] and operating system sources [14, 15]. A theoretical study of the sampling component has been done in [16] and the cryptanalytic work of Kelsey *et al.* [5] has benefited to the cryptographic post-processing of RNGs.

In the design of modern RNGs, the contribution of entropy estimation is controversial. Indeed, two visions of RNG designs are confronted. On the one hand, we have designs in which entropy estimation is critical. The entropy estimation is in charge of determining if enough entropy is available from the sources to be injected in the RNG. Such a design represents the majority of the literature [17, 16, 9, 14, 15] and standards (ISO/ISEC 18031 [18] and NIST [19]). On the other hand, the model proposed by Barak and Haveli in [20] and the design of FORTUNA [9] exclude entropy estimation as it provides a false sense of security.

The main argument of Barak and Haveli [20] is that "*measuring of entropy from the point of view of an attacker is well beyond what can be expected from a computer program*". On this statement, the authors strongly agree, but it does not mean that entropy estimation has to be excluded in the design of RNGs. In their own work, the authors of [20] use the assumption that there exist at least from time to time high entropy inputs. However, without any entropy estimation this property cannot be guaranteed. Entropy estimators are of no use against an adversary who is able to observe a source. They are meant to ensure that a legitimate user operates the generator properly when reseeding and does not have to deal with the adversary.

The contribution of this paper is an entropy estimator for sources with unknown probability distribution. Our result extends the work of Bucci and Luzzi [21] to the case of non-binary sources. Moreover, we demonstrate that our estimator converges towards the Shannon entropy of the source by increasing the time and memory requirements. As an application, we apply our entropy estimator on different sources of the iPhone.

The rest of the paper is organized as follows. Section 2 provides the fundamental definitions and notations related to the study of entropy. Section 3 describes the properties expected from an entropy estimator. Subsequently, we compare these properties with previous works (Section 4). The main contribution of the paper is given in Section 5 where we introduce the estimator and give a mathematical proof for sources producing uncorrelated output. Then, we evaluate the quality of several sources available on an iPhone (Section 6), show the behavior of the estimator on these sources and compare the efficiency of our estimator with previous works.

2 Notations

Throughout this paper, we denote random variables $X, Y, \ldots$ by capital letters and their corresponding sample spaces by $\mathcal{X}, \mathcal{Y}, \ldots$ Realizations $x \in \mathcal{X}, y \in \mathcal{Y}, \ldots$ of random variables are marked by small letters.

Let us assume that a sequence of length n, $\mathbf{X} = X_1, \ldots, X_n$, is composed of independent and identical distributed (i.i.d.) random variables. The whole sequence is mentioned with bold letters. By $\mathbf{X}_{[i,j]} = X_i, \ldots, X_j$, we define the subsequence from index i to index j. We always consider a finite sample space, thus we can denote by $M = |\mathcal{X}|$ the size of the sample space and without loss of generality we assume that $\mathcal{X} = \{0, 1, \ldots, M-1\}$. The probability distribution of the source is given by $p = \{p_0, p_1, \ldots, p_{M-1}\}$, where $p_\eta = Pr[X_i = \eta]$ for all $\eta \in \mathcal{X}$ and $1 \leq i \leq n$. The expectation of a random variable is denoted $\mathrm{E}(X)$.

When speaking of entropy, we refer to Shannon's entropy [22] (Equation 1) which is a measure of the average number of binary guesses an attacker has to perform to predict the output of a random source. It can also be seen as the expected information of an element of the source, where the element η has information $-\log_2 p_\eta$. Some other entropy notions used in connection with RNGs are the Rényi entropy [23] (Equation 2) which is a measure of the probability of collisions or the Min entropy (Equation 3) which is a lower bound to the other entropy notions. Their definitions for a random source with probability distribution p are given by:

$$H(p) = -\sum_{\eta \in \mathcal{X}} p_\eta \log_2 p_\eta, \tag{1}$$

$$H_\alpha(p) = \frac{1}{1-\alpha} \log_2 \left(\sum_{\eta \in \mathcal{X}} p_\eta^\alpha \right), \tag{2}$$

$$H_\infty(p) = -\log_2 (\max_{\eta \in \mathcal{X}} p_\eta). \tag{3}$$

For $\alpha > 1$ we have the following relation between the different entropy notions:

$$H_\infty(p) \leq H_\alpha(p) \leq H(p) \leq |\mathcal{X}|$$

with equality if and only if p is uniformly distributed. We use the usual convention that $0 \log_2 0 = 0$.

If not mentioned otherwise, we always assume that the source outputs are an i.i.d. sequence of random data and that the entropy is defined by the probability $p = \{p_\eta\}_{\eta \in \mathcal{X}}$. In this case, we write $H(p)$ to denote the entropy. Some estimators are able to estimate the entropy of a source generating correlated sequences. If we consider the entropy of a source with dependencies between the symbols we write only H.

Let $\mathbf{x} = x_1, x_2, \ldots, x_n$ denote the outcome of an entropy source. Then we define the empirical distribution of this sequence as $\hat{p} = \{\hat{p}_\eta\}_{\eta\in\mathcal{X}}$, where $\hat{p}_\eta = \frac{1}{n}\#\{1 \leq j \leq n | x_j = \eta\}$ is the empirical frequency of η. We use the notation $\hat{H}(\mathbf{X}_{[i-\ell+1,i]})$ to denote an entropy estimator that takes as input ℓ consecutive values of the sequence $\mathbf{X}$.

3 Requirements

Entropy estimators for RNGs must fulfill several requirements which are of two types. Theoretical requirements (1 and 2) are related to the hypothesis made on the entropy. They are fundamental for the correctness of the estimation. Implementation requirements (3, 4, and 5) concern the constraints for the execution of the estimator.

Requirement 1 *The entropy of the source is unknown which implies that the probability distribution p is unknown.*

This first requirement is fundamental in the context of an RNG working in a changing environment. The entropy of the different sources sampled by the RNG may fluctuate depending on the context. If the probability distribution of the entropy source is known, the estimation is easy. However, this is not the case in general.

Requirement 2 *An entropy estimator must satisfy the following inequality:*

$$\mathrm{E}\left(\hat{H}(\mathbf{X}_{[i-\ell+1,i]})\right) \leq H(p).$$

The estimator must be pessimistic: an overestimation of the entropy could give the user a false sensation of security. Thus it is always better to underestimate the entropy than overestimating it.

Requirement 3 *The estimator $\hat{H}$ must be efficient in memory and computations.*

The algorithm is applied at run-time and should not represent an heavy load for the system. Since there is no information available on the source, we want to have an estimator with a "good behavior" in any cases. Thus, an estimator whose time and memory complexity depends on the sample size violates this requirement.

Requirement 4 *For any $i \geq \ell$, we must have:*

$$\hat{H}\left(\mathbf{X}_{[i-\ell+1,i]}\right) \text{ must be defined.}$$

We want to know when we have collected enough entropy. Therefore, we need an estimate for each input sample. The initialization time ℓ of the estimator must be as short as possible.

Requirement 5 *The estimator must work with any types of sources.*

The more entropy sources we can use, the better. In a modern RNG, we cannot afford to implement a dedicated entropy estimator for each different sources. It is more convenient to have an entropy estimator which can be adapted to any kind of sources.

Option 6 *A parameter allows better estimation if we use more time and memory.*

This property allows to adapt the estimator to the actual needs and preferences of the user.

4 Related Works

Entropy estimation is a very active field in information theory, mathematics and physics. We analyze several estimators with respect to the requirements defined in Section 3. In Table 1, we give a summary of the results.

Known distribution. In the case of a physical source with known behavior, we can use an online test like the one described by Schindler [24]. This test checks if the physical source behaves like expected. However, it is of no help in our problem as it cannot be applied for sources with unknown distribution. Therefore, this kind of tests do not satisfy Requirement 1.

Plug-in estimator. The most straightforward estimator of the entropy applies the entropy function on the empirical distribution, $H(\hat{p})$. This estimator is often called maximum-likelihood estimator or plug-in estimator, see for example [25, 26]. The main problem of this estimator is that it only gives an estimate of the total data set. This is useful to evaluate off-line a physical source with fixed behavior, however, in the case of a source with changing behavior, this is not practical. The estimator clearly violates Requirement 4. Moreover, it needs to store the number of occurrences for each element which can contradict Requirement 3 for a large sample space.

One might try to cut the whole sequence in smaller chunks of size $N' \ll N$ and apply an estimator for insufficient sampling, i.e. where the length of the sequence N' is only slightly larger than the size of the sample space. Such estimators are discussed in [26, 27, 28]. However, they still remain impractical for large sample spaces. Requirement 3 and 4 are not fully satisfied.

Compression. Source-coding can be a way to measure the entropy of a sequence. Lossless compression[1] can be used to determine the most concise way to represent the sequence. In our problem, we can distinguish two classes of estimators based on compression: frequency counting (*e.g.* Huffman coding or arithmetic coding) and match length (*e.g.* Lempel-Ziv).

Estimators based on frequency counting are all using dynamic versions of classical compression algorithms. They suffer all from an excessive memory consumption (which violates Requirement 3). For instance, dynamic Huffman coding needs to use a tree of size $|\mathcal{X}|$ to store the code and a counter for each symbol to count the number of its occurrences. For every new event, we search for the element in the tree, increase the count for this symbol and if necessary

[1]All the compression algorithms given throughout the paper can be found in [29].

adapt the tree. Especially for large sample spaces this estimator contradicts Requirement 3. A similar flaw can be found for arithmetic coding.

Lempel-Ziv compression is not based on the frequencies of symbols, but on match lengths. For a sequence $X_0, X_1, \ldots$, let us define the match length L_i^r for a given window size r as:

$$L_i^r(\mathbf{x}) = 1 + \max\left\{\ell : \mathbf{x}_{[i,i+\ell-1]} = \mathbf{x}_{[j,j+\ell-1]}\ , i-r \leq j \leq i-1\right\} \tag{4}$$

It has been shown that for a stationary and ergodic process:

$$\frac{L_i^r(\mathbf{X})}{\log_2(r)} \to \frac{1}{H} \quad \text{with prob. } 1 \quad (r \to \infty)\ .$$

Other entropy estimators based on match length are proposed in [30, 31]. From Requirement 4 follows that we would like to have for each output i a value m_i such that $\frac{1}{n}\sum_{i=1}^{n} m_i \xrightarrow{n\to\infty} H(p)$. Neither the estimator in the original paper of Wyner and Ziv [32] nor the estimators proposed by Kontoyiannis et al. [30] fulfill this requirement. However, the work of Gao et al. [31] contains a suitable candidate. This estimator comes closest to fulfilling all the requirements of the previous section. We consider it more closely in our comparison. It is also interesting because its proof is not limited to independent sources. The estimator is defined by:

$$\hat{H}_{\mathrm{LZ}}^r(\mathbf{x}_{[i-r,i]}) = \frac{\log_2(r)}{L_i^r(\mathbf{x})}\ . \tag{5}$$

Then Gao et al. [31] showed that for a stationary and ergodic process which satisfies the Doeblin condition, one gets:

$$\frac{1}{n-r}\sum_{i=r+1}^{n} \hat{H}_{\mathrm{LZ}}^r(\mathbf{x}_{[i-r,n]}) \to H \quad \text{with prob. } 1 \quad (n, r \to \infty)\ .$$

This estimator has the disadvantage that it uses future data, which means that we can only give an estimate as soon as we know that we cannot find any longer match.

Transition frequency. Bucci and Luzzi [21] propose a different approach. They consider a binary source and count the number of transitions from 0 to 1 or from 1 to 0. The authors did not mention the mathematical connection between the entropy of a binary source $-p_0 \log_2 p_0 + (1-p_0)\log_2(1-p_0)$ and the average number of transition in $n+1$ bits: $n 2 p_0(1-p_0)$. However, the connection follows directly from the fact that $-log_2 x \geq \frac{1}{ln(2)}(1-x)$ for $0 < x < 1$. This estimator works only for binary sources and, thus, contradicts Requirement 5. The estimator that we introduce in Section 5 extends this approach in a non trivial way to non-binary sources and adds an additional parameter that allows to refine the estimation by spending more time and memory.

Correlated sources. Most of the estimators mentioned above are defined for sequences of i.i.d. random variables. Only the estimators based on LZ-compression consider sources with dependencies that can be modeled as a Markov chain. If we want to estimate directly the entropy of a Markov chain of order k we need the frequencies of all $(k+1)$-tuples [33]. For large sample spaces this gets quickly impractical. Thus, such an estimator does not satisfy

Requirements 3 and Requirement 4. Another possibility would be the use of higher order adaptive arithmetic coding. However, as for any method using empirical frequencies, we need even more memory to consider the dependencies which contradicts Requirement 3.

Table 1: Alternative Entropy Estimators. Notes: *(a)* The estimate is for a block of elements, not for the whole message, however, still not for each element; *(b)* Can get time consuming if we have long matches; *(c)* We have to wait until we are sure that no longer matches exist.

Name	Violated Requirements
Tests for physical sources [24]	1
Plug-in estimator [25, 26]	3, 4
Insufficient sampling [26, 27, 28]	3, 4 *(a)*
Dynamic Huffman coding	3
Arithmetic coding	3
Based on LZ-compression on whole data [32, 30]	3 *(b)*, 4
Based on LZ-compression value for each element [31]	3 *(b)*, 4 *(c)*
Transition Frequencies [21]	5
Plug-in for Markov chains	3,4
Higher order adaptive arithmetic coding	3

5 Entropy Estimator

In this section we introduce our new algorithm in details. A corresponding patent is pending [34]. Besides the description of the estimator, we give the mathematical justifications and show that in the case of i.i.d sequences, when the message length n and the parameter r go to infinity, the expected value of the estimator converges towards the correct value of the Shannon entropy.

Remark 1 *In the proof of Theorem 1, we consider a source which outputs a sequence of i.i.d. random variables. This is a simplified model. In practice, most of the sources will have some dependencies in the data. When ignoring these dependencies, we risk to overestimate the entropy. For example in English text the entropy for each letter is around 4 bits, the entropy considering also the dependencies between the letters is only around 1.34 bits [35]. In Section 6 we compare our estimator with the* $\hat{H}^r_{\mathrm{LZ}}$ *estimator which is based on LZ-compression and which takes the dependencies into account. In all the sources that we tested, our estimator is much more pessimistic than the one taking the dependencies into account. This shows that in these practical situations our estimator does not suffer from the fact that it was designed for uncorrelated sources.*

5.1 Algorithm

Our estimator is inspired by the one used in [21]. In their work, Bucci and Luzzi use the transition frequency to estimate the entropy of a binary source. By a transition we mean a change from 0 to 1 or vice versa.

Let p_0 be the probability that the binary source outputs 0. Then, the average number of transitions in $n+1$ bits is $n2p_0(1-p_0)$. The average bits of information in an n-bit string is $n\left(-p_0\log_2(p_0)-(1-p_0)\log_2(1-p_0)\right)$. This two values can be connected by the fact that $-log_2 x \geq \frac{1}{ln(2)}(1-x)$ for $0<x<1$.

Our contribution is to extend their idea to non-binary sources, give the corresponding mathematical proof and introduce a parameter r which allows to achieve better estimates by spending more time and memory.

To extend the estimator of Bucci and Luzzi to the non-binary case, we use the following idea. If we compare two consecutive values, the probability that $x_i \neq x_{i-1}$ is $\sum_{\eta\in\mathcal{X}} p_\eta(1-p_\eta)$. By using the properties of the natural logarithm we can write $-\sum_{\eta\in\mathcal{X}} p_\eta \log_2 p_\eta \geq \frac{1}{\ln(2)}\sum_{\eta\in\mathcal{X}} p_\eta(1-p_\eta)$. Our estimator expands this idea to the comparison to r previous values.

This estimator is applied on $r+1>0$ consecutive values of the sequence $\mathbf{X}=X_1,X_2,\ldots$ It compares the actual value to the r previous ones, thus we can give an estimate for $i \geq r+1$.

Definition 1 *For any $i \geq r+1$, let us define*

$$\ell_i^r = \begin{cases} r & \text{if } x_{i-j} \neq x_i, 1 \leq j \leq r \\ \min\{0 \leq j \leq r-1 : x_{i-1-j} = x_i\} & \text{otherwise.} \end{cases}$$

Then the entropy estimator comparing to previous values *is defined by*

$$\hat{H}_{\mathrm{pv}}^r(\mathbf{x}_{[i-r,i]}) = \frac{1}{\ln(2)}\sum_{j=1}^{\ell_i^r}\frac{1}{j}\ .$$

The differences between the match length L_i^r and the value ℓ_i^r is that ℓ_i^r compares for equality between the single element x_i and the previous values, whereas L_i^r searches for equal sequences. Moreover, ℓ_i^r compares at most to r elements while L_i^r can go infinitely into the future.

In Algorithm 1 we show the computation of $\hat{H}_{\mathrm{pv}}^r(\mathbf{x}_{[i-r,i]})$, for $i=r+1,\ldots,n$ and any specific outcome $x_1,x_2,\ldots,x_n$.

Algorithm 1 $\hat{H}_{\mathrm{pv}}^r(\mathbf{x}_{[i-r,i]})$, for $i=r+1,\ldots,n$

```
Require: $\mathbf{x}_{[i-r,i]} = x_{i-r}, x_{i-r+1}, \ldots x_i \in \mathcal{X}$
Ensure: $\hat{H}_{\mathrm{pv}}^r(\mathbf{x}_{[i-r,i]})$
  $w \leftarrow 0$
  $j \leftarrow 1$
  while $j \leq r$ and $x_i \neq x_{i-j}$ do
    $w \leftarrow w + \frac{1}{j}$
    $j \leftarrow j + 1$
  end while
  $\hat{H}_{\mathrm{pv}}^r(\mathbf{x}_{[i-r,i]}) \leftarrow \frac{1}{\ln(2)} w$
```

Mathematical Justification We use the Taylor series of the natural logarithm, $\ln \frac{1}{x} = \sum_{i=1}^{\infty} \frac{(1-x)^i}{i}$, for $0 < x \leq 2$. From this follows directly for any integer $r \geq 1$ and any probability distribution p

$$\sum_{\eta \in \mathcal{X}} p_\eta \log_2 \frac{1}{p_\eta} \geq \frac{1}{\ln(2)} \sum_{\eta \in \mathcal{X}} p_\eta \sum_{i=1}^{r} \frac{(1-p_\eta)^i}{i} \ , \tag{6}$$

where we omit all values for which $p_\eta = 0$. In the following, we will denote the sum on the right hand side by

$$H^r(p) = \frac{1}{\ln(2)} \sum_{\eta \in \mathcal{X}} p_\eta \sum_{i=1}^{r} \frac{(1-p_\eta)^i}{i} \ . \tag{7}$$

Theorem 1 *For any fixed r and $n \to \infty$, the average of the estimation converges* almost surely *to $H^r(p)$, i.e.*

$$\frac{1}{n-r} \sum_{i=r+1}^{n} \hat{H}^r_{\text{pv}}\left(\mathbf{X}_{[i-r,i]}\right) \to H^r(p) \quad a.s. \quad (n \to \infty) \ , \tag{8}$$

and for $r, n \to \infty$ we have

$$\frac{1}{n-r} \sum_{i=r+1}^{n} \hat{H}^r_{\text{pv}}\left(\mathbf{X}_{[i-r,i]}\right) \to H(p) \quad a.s. \quad (n, r \to \infty) \ . \tag{9}$$

Remark 2 *Note that for a fixed value of r, we estimate only $H^r(p)$ and not $H(p)$. For example, in the uniform case over a set of M elements, $H^r(p) = \frac{1}{\ln(2)} \sum_{i=1}^{r} \frac{(1-1/M)^i}{i}$. The exact value of the entropy can only be achieved for $n \to \infty$ and $r \to \infty$.*

Proof 1 *Let us define a new sequence of random variables*

$$Y_i = \hat{H}^r_{\text{pv}}\left(\mathbf{X}_{[i-r,i]}\right)$$

for all $r+1 \leq i \leq n$. These variables have the following expectation:

$$\begin{aligned}
E(Y_i) &= E\left(\hat{H}^r_{\text{pv}}(\mathbf{X}_{[i-r,i]})\right) \\
&= \frac{1}{\ln(2)} \sum_{\eta \in \mathcal{X}} p_\eta \left[\sum_{\eta_{i-1} \neq \eta} p_{\eta_{i-1}} + \sum_{\eta_{i-1}, \eta_{i-2} \neq \eta} p_{\eta_{i-1}} p_{\eta_{i-2}} \frac{1}{2} + \right. \\
&\left. \cdots + \sum_{\eta_{i-1}, \eta_{i-2}, \ldots \eta_{i-r} \neq \eta} p_{\eta_{i-1}} p_{\eta_{i-2}} \cdots p_{\eta_{i-r}} \frac{1}{r} \right] \\
&= \frac{1}{\ln(2)} \sum_{\eta \in \mathcal{X}} p_\eta \sum_{i=1}^{r} \frac{(1-p_\eta)^i}{i} = H^r(p) \ .
\end{aligned}$$

Clearly, the variables Y_i are not independent, thus we cannot apply directly the law of large numbers. However, they are $(r+1)$-dependent, which means that Y_i and Y_j are independent if $|i-j| > r+1$. Since we have a finite state space, *the variance of Y_i is finite and constant for all i. Thus by using results on $(r+1)$-dependent random variables like in [36] it follows that*

$$\frac{1}{n-r}\sum_{i=r+1}^{n} Y_i \to E(Y_i) \quad a.s. \quad (n \to \infty) \, .$$

This gives us directly Equation 8. From the Taylor series of the logarithm follows that $\lim_{r\to\infty} H^r(p) = H(p)$ *and thus Equation 9.*

Special Case of $r = 1$ In the case of $r = 1$ we can use [37, Theorem II.6 for $k = 1$] which states that

$$H(p) \geq 2(1 - \sum p_i^2) = 2\sum p_i(1-p_i).$$

Thus, in the case of $r = 1$, we can add 2 instead of $1/\ln(2)$ each time $x_i \neq x_{i-1}$. This has the additional advantage that we only have to add an integer value.

6 Analysis and Application

First, a complexity analysis of our estimator is given. Then, the experiments made on a second-generation iPhone 3GS are described. Finally, we compare the different merits of several estimators on the data collected.

6.1 Complexity Analysis of the Estimator

We have to store r intermediate values and perform up to r comparisons. This leads to a memory complexity of r words and a time complexity of up to r comparisons. If we store the results $\sum_{i=1}^{R} \frac{1}{i}$ in a table for $1 \leq R \leq r$ we only need one table look-up after the comparisons.

Comparison Between $\hat{H}^r_{\mathrm{LZ}}$ and $\hat{H}^r_{\mathrm{pv}}$ The estimator $\hat{H}^r_{\mathrm{LZ}}$ stores r previous values. However, it searches the maximal match length, when considering future outputs. Only when it is sure that there exists no longer match, it finishes the estimate for the value at time i. The number of future values that we have to consider is not fixed and depends on the source. This has several inconvenient effects on the complexity of the estimator. We do not know how long we have to wait before we can output an estimate. We do not know how much future values we have to store and how much comparisons we have to do. Thus the time and memory complexity depends on the data and is not known beforehand. This is especially a problem for a source with low entropy and thus long match lengths or for the extreme case where we have a broken source that always outputs the same value. This last case can lead to an infinite complexity if no maximum length are considered. It might also happen if a source has temporarily no entropy. We can avoid such situations by defining a maximal number of comparisons into the future. However, when using such a maxima we change the algorithm and the convergence proofs are not valid any more. Taking too short match lengths can lead to an overestimation of the entropy.

The estimator presented in Section 5 has the advantage that the time and memory complexity is fixed by the parameter r and does not depend on the data. In addition, it can stop as soon as it finds the new value in the stored data. The estimator $\hat{H}^{r}_{\mathrm{LZ}}$ has two nested loops and always has to test all the stored values to make sure that it finds the longest match.

6.2 iPhone's Sources

The tests conducted where made on Apple iOS devices such as the iPhone, the iPad and the iPod Touch. Not every iOS device supports the same range of hardware extensions, and our application caters with this heterogeneity. Indeed, iPod Touch or iPad devices do not ship with a GPS chip, while compasses appeared only with the iPhone 3GS. The device that we used in the experiments hereafter is a second-generation iPhone 3GS, which ships with all the hardware extensions an iOS device can offer presently. As entropy sources, we used the IEEE 754 floating point values provided by:

- the three accelerometers (one per axis), giving floating values within a $[-1.0, 1.0]$ range,
- the geo-positioning using a combination of GPS and GSM triangulation, yielding longitude, latitude and altitude,
- the heading data using a compass, giving three dimensional data towards the magnetic north.

Sampling rate The application used to collect the data from the different sources runs in user mode. It affects significantly the sampling rate: running our data collection tools in the kernel space would have offered more sampling rate options. However, the authors believe that the deployment of an RNG on such a device can only be done in the user space. The access to the kernel space of iOS is highly restricted. It is preferable that the source code of an application manipulating highly sensitive data (geo-positioning) can be inspected by the user at any time. Otherwise it will rise the suspicion from the users concerning their privacy despite the security benefits of running in the kernel space. This is why we use a limited sampling rate. The sampling rate used was not fixed and was driven by iOS. Indeed, we collected data each time iOS successfully obtains values from the different sensors. On average, one capture is done per second. It justifies the fluctuations on the sampling rate found in Table 2.

Accuracy Details on the hardware used in those devices are not publicly available, hence it is hard to say anything related to the precision of the data that we capture. Still, we made the following observations. The accelerometers were primarily introduced to know the device orientation, and rotate the screen display accordingly. They are also used to detect shaking gestures. As such, this piece of hardware does not require a strong precision. One can easily check this by writing a small application that reads the accelerometer values: even with the device firmly standing on a stable table, the returned values are unstable! Exploiting such data needs smoothing through Fourier transforms, as it is mentioned in the iOS SDK documentation.

The geo-positioning data depends on where the user is. GPS positioning only works outdoors. GSM triangulation is leveraged to speed up with a "close-enough" position until more precise data can be obtained from the GPS. It may be the only data that can be fetched when indoors. An application specifies how precise positioning shall be, with accuracy parameters ranging from three kilometers to ten meters at best. In our case, we opted for the best possible accuracy. Point-to-point road navigation applications exist on the iPhone, and to the very best of our knowledge, it is as capable as a reasonable GPS navigation device.

Finally, heading directions are based on a compass chip that measures values towards the magnetic north. Again, not much can be said regarding the accuracy of the data, but it seems to give good indications in use-cases such as pedestrian navigation. However, magnetic field measures can easily suffer perturbations. Anyone running the built-in Apple Compass application will more than often experiment requests to move the device around so as to "describe an 8" and hopefully get rid of electrical interferences.

6.3 Experiments

We have collected two different types of data: outdoor measurements and indoor measurements. The subject holding the device is a "standard" academic researcher. The outdoor measurements were made while the device's holder was cycling or running. The indoor measurement were made in the laboratory during office hours.

For each of data files, we have computed (1) the plug-in estimate of the Shannon entropy, (2) the Rényi entropy, (3) the Min entropy, (4) the LZ estimator $\hat{H}^r_{\mathrm{LZ}}$ for $r = 2, 10, 100$ and (5) our new estimator $\hat{H}^r_{\mathrm{pv}}$ for $r = 1, 2, 10, 100$. Note that the LZ estimator is only defined for $r > 1$. We also computed the average number of comparisons for $\hat{H}^r_{\mathrm{LZ}}$ and $\hat{H}^r_{\mathrm{pv}}$.

Remark 3 *For the LZ estimator, the estimated entropy is inverse proportional to the match length. For a sequence* $\mathbf{x}$ *of fixed length* n*, the match length* $L^r_i(\mathbf{x})$ *can never be longer than* $n-i+2$*. For* i *close to* n *this might lead to shorter match lengths than expected from an infinite data stream, and thus to an overestimation of the entropy. To avoid this we compute* $\hat{H}^r_{\mathrm{LZ}}$ *only for* $r + 1 \leq i \leq n - 100$*. Except for the cases where the source outputs a constant value, in all our tests the maximal length that* $\hat{H}^r_{\mathrm{LZ}}$ *looked into the future was less than 100. Thus, by considering only* $r + 1 \leq i \leq n - 100$ *we get the same results than if we would have an infinite data stream.*

Remark 4 *We did not take directly the values from the sources but considered the differences to the previous value. This was to eliminate redundancy. On the implementation side, this has the disadvantage that we have to store the previous value for each source.*

We collected several traces and Table 2 shows the average number of collected events per minute. The Figures 1-4 describe respectively the results obtained for the indoor and outdoor trace. In the first two figures, we give the estimates of the entropy. In the last two figures, we give the average number of comparisons needed in $\hat{H}^r_{\mathrm{LZ}}$ and $\hat{H}^r_{\mathrm{pv}}$.

6.3.1 Conclusion from the tests

We can see from Figures 1 and 2 that both estimators $\hat{H}^r_{\mathrm{pv}}$ and $\hat{H}^r_{\mathrm{LZ}}$ are always smaller than Shannon's entropy. Except for the cases with constant sources

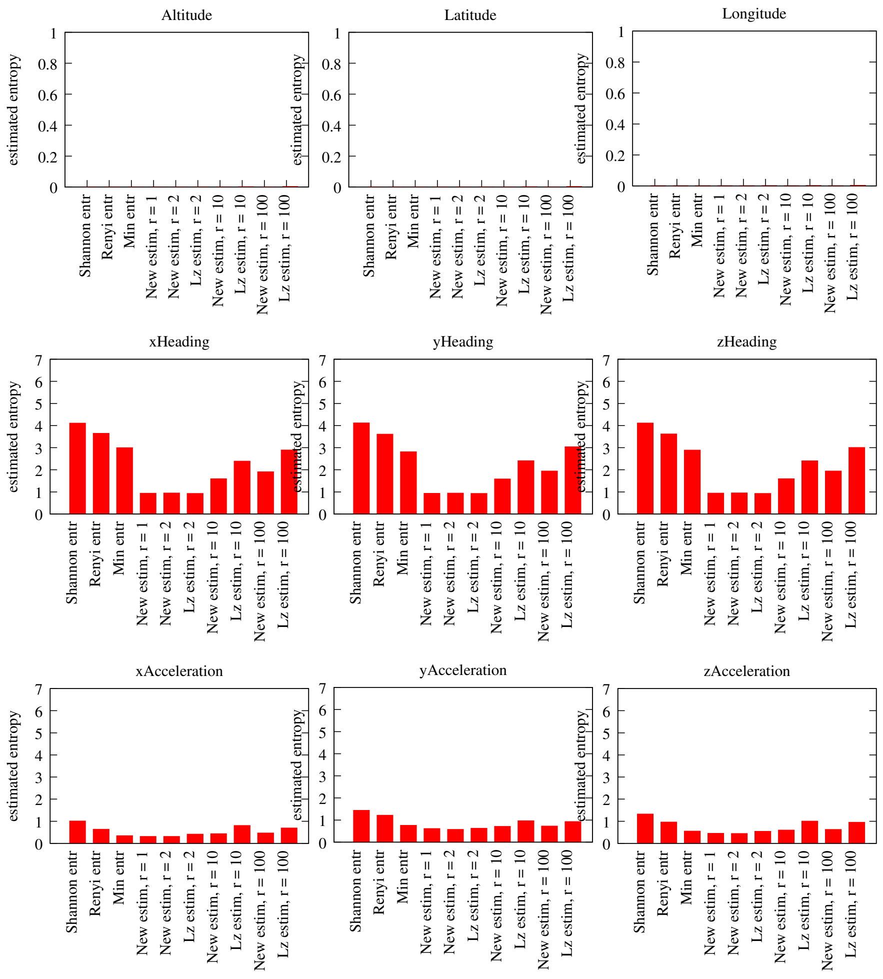

Figure 1: Indoor measurements: Entropy estimates.

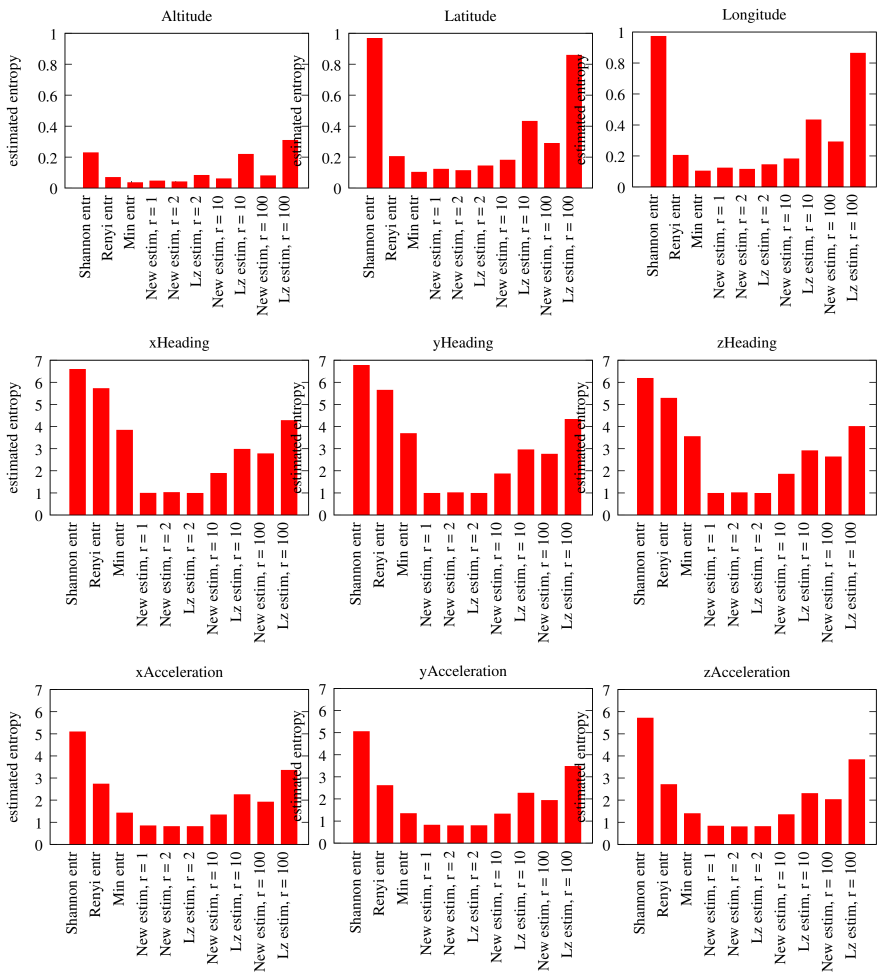

Figure 2: Outdoor measurements: entropy estimates.

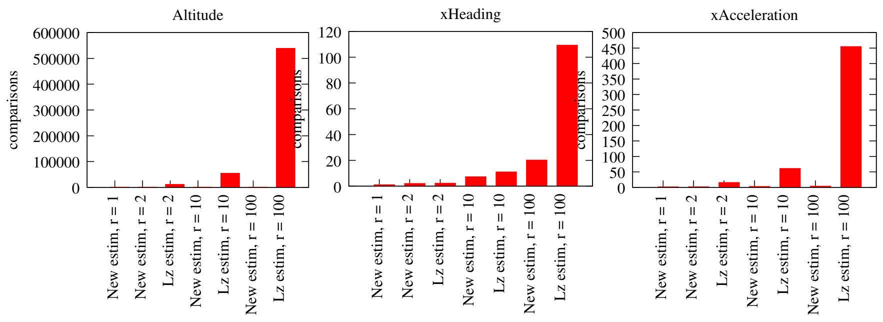

Figure 3: Indoor measurements: complexity.

Table 2: Examples of trace collected.

	indoor	outdoor
size	10765	8032
outputs/min	59.24	76.53

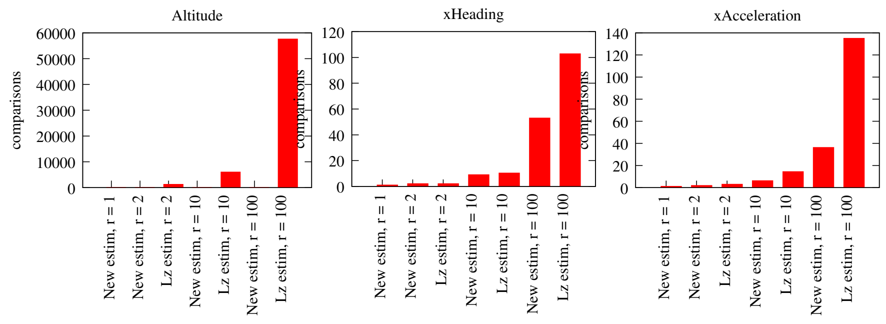

Figure 4: Outdoor measurements: complexity.

(see Remark 5), $\hat{H}^r_{\mathrm{LZ}}$ gives the same output than for an infinite data sequence, thus it takes dependencies between outputs into account. In our tests, the $\hat{H}^r_{\mathrm{pv}}$ is always smaller than the $\hat{H}^r_{\mathrm{LZ}}$. This means that for this test data, the fact that $\hat{H}^r_{\mathrm{pv}}$ was designed for independent sources clearly does not lead to any overestimation. We can also see that in some cases the estimates of $\hat{H}^r_{\mathrm{pv}}$ and $\hat{H}^r_{\mathrm{LZ}}$ are higher than the Rényi and Min entropy. This comes from the fact that these estimators are made for the Shannon entropy. We can also see that for both estimators higher values of r lead to a better estimate.

Remark 5 *If a source always outputs the same value, the LZ estimator will fail. We have such a situation in the indoor examples for the altitude, longitude and latitude. The estimator will always search for a match length until the end of the data stream. In an online situation this means that the estimator blocks and will ask for an infinite amount of memory. For our test data, this means that the match length is bounded by the size of the data file. Thus* $\hat{H}^r_{\mathrm{LZ}}$ *has an average value larger than zero. Our proposed estimator* $\hat{H}^r_{\mathrm{pv}}$ *estimates correctly a zero entropy and has an average time complexity of one.*

Remark 6 *In all our experiments, the longitude and the latitude always have exactly the same entropy estimates, for all estimators. Thus, it seems that their information is redundant. We strongly suggest to use only one of them as an entropy source.*

From Figures 3 and 4, we see that the number of comparisons for our estimator is always smaller than for the LZ estimator. In the indoor example this effect is even stronger, not only in the extreme case of a source which always outputs the same value.

7 Conclusion

We proposed a new entropy estimator for non-binary sources and we have given a mathematical justification for its design. We show how to underestimate the Shannon entropy while fitting the implementation constraints (computation and memory). Our empirical study shows that our estimator is the most pessimistic compared to the existing works. We proposed three new possible sources of entropy on the iPhone, which were used for the empirical tests.

Several questions are still left opened in the field of entropy estimation in cryptography. Finding efficient algorithms to approximate the Rényi entropy or the Min entropy is an important alternative to the current design of entropy estimators.

References

[1] Goldberg, I., Wagner, D.: Randomness and the Netscape browser. Dr. Dobbs Journal (1996) http://www.cs.berkeley.edu/ daw/papers/ddj-netscape.html.

[2] Gutterman, Z., Pinkas, B., Reinman, T.: Analysis of the Linux random number generator. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P 2006), Oakland, California, USA, IEEE Computer Society (2006) 371–385

[3] Dorrendorf, L., Gutterman, Z., Pinkas, B.: Cryptanalysis of the random number generator of the Windows operating system. ACM Transactions on Information and System Security - TISSEC **13**(1) (2009) 1–32

[4] Dorrendorf, L.: Cryptanalysis of the Windows random number generator. Master's thesis, The Hebrew University of Jerusalem - School of Engineering and Computer Science (2009)

[5] Kelsey, J., Schneier, B., Wagner, D., Hall, C.: Cryptanalytic attacks on pseudorandom number generators. In: Fast Software Encryption - FSE'98. Lecture Notes in Computer Science 1372, Paris, France, Springer Verlag (1998) 168–188

[6] Gutterman, Z., Malkhi, D.: Hold your sessions: An attack on Java session-id generation. In: Topics in Cryptology - CT-RSA 2005. Lecture Notes in Computer Science 3376, San Francisco, CA, USA, Springer (2005) 44–57

[7] Kamkar, S.: phpwn: Attacking sessions and pseudo-random numbers in PHP. In: Blackhat USA 2010, Las Vegas, NV, USA (2010)

[8] Kelsey, J., Schneier, B., Ferguson, N.: Yarrow-160: Notes on the design and analysis of the yarrow cryptographic pseudorandom number generator. In: Selected Areas in Cryptography - SAC'99. Lecture Notes in Computer Science 1758, Kingston, ON, Canada, Springer (1999) 13–33

[9] Ferguson, N., Schneier, B.: Practical Cryptography. John Wiley & Sons, Inc. (2003)

[10] Seznec, A., Sendrier, N.: HAVEGE: A user-level software heuristic for generating empirically strong random numbers. ACM Transactions on Modeling and Computer Simulation - TOMACS **13**(4) (2003) 334–346

[11] Francillon, A., Castelluccia, C.: TinyRNG, a cryptographic random number generator for wireless sensor network nodes. In: Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, IEEE WiOpt 2007, Limassol, Cyprus, IEEE (2007)

[12] Donnell, C.W.O., Suh, G.E., Devadas, S.: PUF-based random number generation. MIT CSAIL CSG Technical Memo 481, Massachusetts Institute of Technology, Cambridge, MA, USA (2004)

[13] Davis, D., Ihaka, R., Fenstermacher, P.: Cryptographic randomness from air turbulence in disk drives. In: Advances in Cryptology - CRYPTO '94. Lecture Notes in Computer Science 839, Santa Barbara, CA, USA, Springer (1994) 114–120

[14] Kelsey, J.: Entropy sources. In: NIST RNG Workshop. (2004) `http://csrc.nist.gov/groups/ST/toolkit/random_number.html`.

[15] Gutmann, P.: Testing issues with OS-based entropy sources. In: NIST RNG Workshop. (2004) `http://csrc.nist.gov/groups/ST/toolkit/random_number.html`.

[16] Barak, B., Shaltiel, R., Tromer, E.: True random number generators secure in a changing environment. In: Cryptographic Hardware and Embedded Systems - CHES 2003. Lecture Notes in Computer Science 2779, Cologne, Germany, Springer (2003) 166–180

[17] Gutmann, P.: Software generation of practically strong random numbers. In: USENIX Security Symposium - SSYM'98, San Antonio, Texas, USENIX Association (1998)

[18] International Organization for Standardization: International standard ISO/IEC 18031 - Information technology - Security techniques - Random bit generation (2005)

[19] Barker, E., Kelsey, J.: Recommendation for random number generation using deterministic random bit generators (revised). Technical Report SP800-90, National Institute of Standards and Technology - NIST (2007)

[20] Barak, B., Halevi, S.: A model and architecture for pseudo-random generation with applications to /dev/random. In: ACM conference on Computer and communications security - CCS '05, Alexandria, VA, USA, ACM (2005) 203–212

[21] Bucci, M., Luzzi, R.: Design of testable random bit generators. In: Cryptographic Hardware and Embedded Systems - CHES 2005. Volume 3659 of Lecture Notes in Computer Science., Edinburgh, UK, Springer (2005) 147–156

[22] Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal **27**(5) (1948) 379–423

[23] Rényi, A.: On measures of entropy and information. In: Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, CA, USA, Univ. of Calif. Press (1961) 547–561

[24] Schindler, W.: Efficient online tests for true random number generators. In: Cryptographic Hardware and Embedded Systems - CHES 2001. Lecture Notes in Computer Science 2162, Paris, France, Springer (2001) 103–117

[25] Antos, A., Kontoyiannis, I.: Convergence properties of functional estimates for discrete distributions. Random Struct. Algorithms **19** (2001) 163–193

[26] Paninski, L.: Estimation of entropy and mutual information. Neural Comput. **15** (2003) 1191–1253

[27] Paninski, L.: Estimating entropy on m bins given fewer than m samples. Information Theory, IEEE Transactions on **50**(9) (2004) 2200 – 2203

[28] Grassberger, P.: Entropy estimates from insufficient samplings. arXiv:physics/0307138v2 (2008)

[29] Salomon, D., Motta, G.: Handbook of Data Compression. Springer (2009)

[30] Kontoyiannis, I., Algoet, P.H., Suhov, Y.M., Wyner, A.J.: Nonparametric entropy estimation for stationary processes and random fields, with applications to English text. IEEE Transactions on Information Theory **44**(3) (1998) 1319–1327

[31] Gao, Y., Kontoyiannis, I., Bienenstock, E.: Estimating the entropy of binary time series: Methodology, some theory and a simulation study. Entropy **10**(2) (2008) 71–99

[32] Wyner, A.D., Ziv, J.: Some asymptotic properties of the entropy of a stationary ergodic data source with applications to data compression. Information Theory, IEEE Transactions on **35**(6) (1989) 1250–1258

[33] Wegenkittl, S.: Entropy estimators and serial tests for ergodic chains. Information Theory, IEEE Transactions on **47**(6) (2001) 2480 –2489

[34] Roeck, A.: Blank for anonymity (2010)

[35] Cover, T.M., Thomas, J.: Elements of Information Theory. 2 edn. Wiley-Interscience (2006)

[36] Móricz, F.: Strong limit theorems for blockwise m-dependent and blockwise quasiorthogonal sequences of random variables. Proceedings of the American Mathematical Society **101**(4) (1987) 709–715

[37] Harremoës, P., Topsøe, F.: Inequalities between entropy and index of coincidence derived from information diagrams. IEEE Transactions on Information Theory **47**(7) (2001) 2944–2960

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399