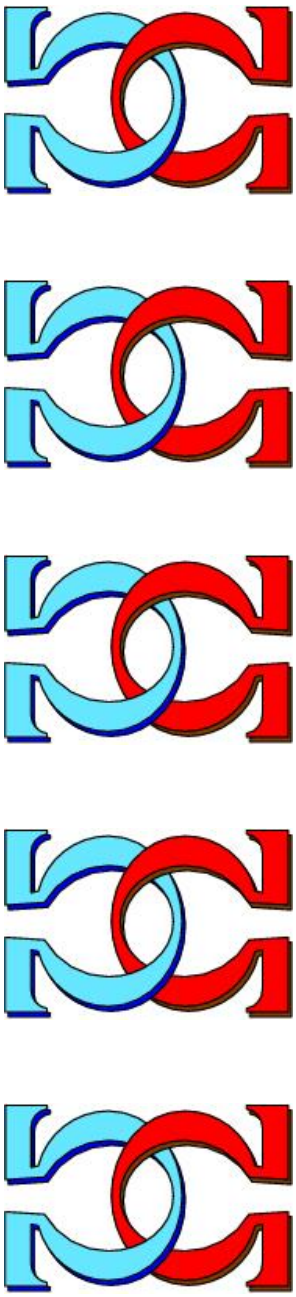

CDMTCS
Research
Report
Series

A forward-parsing randomness test based on the expected codeword length of T-codes

Ulrich Speidel
Department of Computer Science,
University of Auckland,
Auckland, New Zealand

CDMTCS-403
May 2011

Centre for Discrete Mathematics and
Theoretical Computer Science

A forward-parsing randomness test based on the expected codeword length of T-codes

Ulrich Speidel
Department of Computer Science
The University of Auckland, Private Bag 92019
Auckland, New Zealand
ulrich@cs.auckland.ac.nz

Abstract

This paper proposes an algorithm for randomness testing. It is a variant of an algorithm presented earlier by the author for the construction of random sequences. The algorithm exploits two facts: Firstly, given a complete finite prefix code C_i over an alphabet A, every semi-infinite sequence s of symbols $x_0, x_1, x_2, \ldots$ from A starts with a codeword $w_i \in C_i$, and if one presumes that s is random, one can compute the expected length h_i of w_i. Secondly, w_i can be used to extend C_i to yield a larger code C_{i+1}. In this case, the actual length $|w_i|$ of w_i determines the growth in the expected codeword length h_{i+1} for the codeword $w_{i+1} \in C_{i+1}$ that follows in s after the end of w_i. If $|w_i| > h_i$, then h_{i+1} grows less compared to h_i than if $|w_i| \leq h_i$. By comparing expected codeword lengths and actual codeword lengths cumulatively, one may assess the randomness of s: If the hypothesis that s is random holds, then the cumulative values should closely mirror each other.

1 Introduction

Randomness testing for sequences and (pseudo-)random generators has been the subject of considerable research interest over many decades, and numerous tests have been developed. A selection of these have been collated in the NIST randomness test suite [7], for example. The topic continues to be of fundamental importance in areas of digital communication such as, e.g., cryptography and sequences.

The randomness test presented here takes some of its underlying ideas from on another presented by the author in [8] and [11], which uses a parsing algorithm that decodes a sequence "left-to-right" but parses the sequence "right-to-left", effectively restricting its use to finite sequences which need to be fully loaded into computer memory. Hamano and Yamamoto [12] subsequently proposed T-complexity-based randomness test that employs a "forward parsing T-decomposition" algorithm, which both decodes and parses in a "left-to-right" direction. The randomness test presented in this paper is an applies Hamano and Yamamoto's forward parsing algorithm to the author's original randomness

test, which does not base itself on T-complexity but on Shannon entropy [1], and also lends itself to use on semi-infinite sequences.

With a complete finite prefix code C (also frequently known as an "exhaustive prefix-free code" in engineering literature) over some alphabet A, we can parse *any* semi-infinite sequence s over A. E.g., the binary code

$$C = \{00, 010, 011, 10, 110, 111\}$$

lets us parse the semi-infinite string $s = 10011101111010\ldots$ as $10.011.10.111.10.10\ldots$. If we know the probabilities with which the codewords in C occur in s, we can compute the average codeword length h (or Shannon entropy) for s and C. Let $P(w)$ denote the probability of occurrence of w in s and let $\#A$ be the cardinality (number of symbols) of the alphabet. Then:

$$h = -\sum_{w \in C} P(w) log_{\#A} P(w), \tag{1}$$

where h carries the unit of symbols per codeword.

If s is "perfectly random", we expect $P(w)$ to depend on $\#A$ and the codeword length $|w|$ only, i.e., $P(w) = \#A^{-|w|}$ and thus, with $b = 1/\#A$:

$$\begin{aligned} h &= -\sum_{w \in C} P(w) log_{\#A} P(w) \\ &= \sum_{w \in C} b^{|w|} |w|. \end{aligned} \tag{2}$$

Moreover, if s is perfectly random and we view decoding as a process that starts at the beginning of s (i.e., to the left) and progresses in time to the right, we can expect the average length of the actually decoded codewords in s to converge to h. If we can demonstrate that this is not the case for a particular s, then clearly that s is not random. This is nothing new, of course – we are simply observing Shannon entropy here.

One problem with using C is that the reverse conclusion – that s is random if the average length of codewords decoded converges to h – is not true. This is easily seen for the trivial code $C = \{0, 1\}$ and $s = 010101\ldots$. Here, both h and the average codeword length are 1. The problem in this case is that C is small and not large enough to "detect" that the pattern "01" repeats. If we choose $C = \{1, 00, 01\}$, the result is quite different: $h = 1.5$, but the average codeword length is 2. This small example demonstrates that the code we use has an influence on the answer we get.

This paper proposes a solution to this dilemma, by modifying the above proposal in two ways:

1. use a code C_i that grows in size each time we decode a codeword in s and therefore progressively "acquires" more patterns that it then "knows", and

2. compute an h_i for the C_i used at each such decoding step, and compare it against the actual codeword length w_i decoded over C_i in this step. As h_i is an estimate for $|w_i|$, we can still claim that $\sum_i h_i$ converges to $\sum_i |w_i|$ for random s as $i \to \infty$.

The question that remains is how to obtain C_i. Note that C_i needs to be complete and prefix-free. It seems also reasonable to demand that the extension of C_i to C_{i+1} should in some way depend on w_i to enable the code to recognise patterns that have occurred in s.

This paper proposes the use of simple T-augmentation [3, 4, 6] for this purpose. In [8] and [11], the author presented an algorithm for fixed-length s using generalised T-augmentation and a right-to-left parsing algorithm. For semi-infinite s, however, the right-to-left parsing does not work (there is no right end to s that one might start from), but there is also no need to take recourse to generalised T-augmentation. While generalised T-augmentation *may* be used as well in this case, there is no obvious advantage to it.

2 The proposed algorithm

Like its right-to-left parsing cousin, the proposed algorithm successively constructs a series of T-codes $A^{(k_1,k_2,\ldots,k_i)}_{(p_1,p_2,\ldots,p_i)}$, such that $C_i = A^{(k_1,k_2,\ldots,k_i)}_{(p_1,p_2,\ldots,p_i)}$. The algorithm also sources its construction parameters p_i from s as $p_i = w_i$. The simpler version of the algorithm, which will mainly be discussed here, assumes $k_i = 1$, such that we may use the simplified notation $C_i = A_{(p_1,p_2,\ldots,p_i)}$. With $C_0 = A$, we thus have the recurrence relationship of simple T-augmentation,

$$\begin{aligned} C_{i+1} &= A_{(p_1,p_2,\ldots,p_{i+1})} \\ &= \left[p_{i+1}A_{(p_1,p_2,\ldots,p_i)} \cup A_{(p_1,p_2,\ldots,p_i)}\right] \setminus \{p_{i+1}\} \\ &= [p_{i+1}C_i \cup C_i] \setminus \{p_{i+1}\} \end{aligned} \tag{3}$$

where $p_{i+1}X$ denotes the set of words obtained by prefixing each codeword in a code X with p_{i+1}.

The string s is thus parsed as[1]: $s = p_1p_2p_3p_4\ldots$.

The initial Shannon entropy h_0 is computed simply under the hypothesis that s is random, i.e. that all letters of the alphabet occur with equal probability. Hence $h_0 = 1$.

Computing h_{i+1} from h_i needs to consider that the code now consist of two parts: the original codewords in $A_{(p_1,p_2,\ldots,p_i)}$ less p_i, and those contributed by the augmentation part $p_{i+1}A_{(p_1,p_2,\ldots,p_i)}$. We may thus split h_{i+1} into two contributions h'_{i+1} and h''_{i+1} from these respective parts, such that $h_{i+1} = h'_{i+1} + h''_{i+1}$. We have, for $b = 1/\#A$:

$$\begin{aligned} h'_{i+1} &= h_i - b^{|p_{i+1}|}|p_{i+1}|, \\ h''_{i+1} &= \sum_{w\in C_i} b^{|p_{i+1}|+|w|}(|p_{i+1}| + |w|) \\ &= b^{|p_{i+1}|}\sum_{w\in C_i} b^{|w|}(|p_{i+1}| + |w|) \end{aligned} \tag{4}$$

[1]Readers familiar with the parsing algorithm in LZ78 [2] may interpret Hamano and Yamamoto's algorithm as a modified LZ78 where the initial vocabulary consist of the full alphabet and the "innovation symbol" in each step is replaced by a copy of the entire hitherto derived voculary tree.

$$
\begin{aligned}
&= b^{|p_{i+1}|}\left[\sum_{w\in C_i} b^{|w|}|w| + |p_{i+1}|\sum_{w\in C_i} b^{|w|}\right] \\
&= b^{|p_{i+1}|}h_i + b^{|p_{i+1}|}|p_{i+1}|, \qquad (5)
\end{aligned}
$$

where we have used the identity $\sum_{w\in C_i} b^{|w|} = 1$. Hence,

$$h_{i+1} = (1 + b^{|p_{i+1}|})h_i. \qquad (6)$$

The derivation of this recurrence relation for h_i mirrors that of the H_1 entropy in [8, 11]: the underlying problem is the same, i.e., computation of the code entropy for a T-code under the assumption that each codeword w in the code has a probability of occurrence of $b^{|w|}$. The difference to the previous algorithm, however, is how we find p_{i+1}: in the previous algorithm, p_{i+1} is sourced from the T-decomposition [5, 10] of a finite s; in the algorithm presented here, it is simply the respective next codeword that we decode in s using $A_{(p_1,p_2,\ldots,p_i)}$ as the code, after we have decoded $p_1p_2\ldots p_i$.

Note that h_i is the average codeword length in $A_{(p_1,p_2,\ldots,p_i)}$ and thus the expected value for $|p_{i+1}|$ if s is random. The randomness test proposed here thus works on the hypothesis that s is random, in which case we expect to find:

$$\sum_{i=0}^{n} |p_{i+1}| \approx \sum_{i=0}^{n} h_i. \qquad (7)$$

Note that $|p_{i+1}| < h_i$ results in a larger h_{i+1} than $|p_{i+1}| \geq h_i$ and vice versa. If s is indeed random, this relationship acts like a "regulation mechanism".

In the following,

$$\Delta_n(s) = |\sum_{i=0}^{n} |p_{i+1}| - \sum_{i=0}^{n} h_i|$$

will be called the "deviation" of s after the n'th parsing step. The remaining question to solve is now how large $\Delta_n(s)$ may become before we need to abandon our hypothesis that s is random.

3 Acceptable tolerances in randomness

The expected difference δ_i between h_i and the actual length of p_{i+1} in a random string is given by the absolute difference in length between each codeword w in C_i and h_i, times the probability $b^{|w|}$ that w will actually be encountered:

$$\delta_i = \sum_{w\in C_i} b^{|w|}||w| - h_i|. \qquad (8)$$

Knowing δ_i lets us derive an expected value for the difference between $\sum_{i=0}^{n} |p_{i+1}|$ and $\sum_{i=0}^{n} h_i$ as

$$E[\Delta_n(s)] = \sqrt{\sum_{i=0}^{n} \delta_i^2}. \qquad (9)$$

In order to derive a recurrence relation for δ_i, we must manage the absolute value in Eq. (8). For this purpose, define the following subsets of C_i, keeping in mind that we always have $h_{i+1} > h_i$:

$$\begin{aligned}
C_{i,1} &= \{w | w \in C_i \setminus \{p_{i+1}\}, |w| < h_i\}, \\
C_{i,2} &= \{w | w \in C_i \setminus \{p_{i+1}\}, |w| \geq h_i, |w| < h_{i+1}\}, \\
C_{i,3} &= \{w | w \in C_i \setminus \{p_{i+1}\}, |w| \geq h_{i+1}\}, \\
C_{i,4} &= \{w | w \in C_i, |p_{i+1}w| \geq h_{i+1}, |w| < h_i\}, \\
C_{i,5} &= \{w | w \in C_i, |p_{i+1}w| \geq h_{i+1}, |w| < h_{i+1}, |w| \geq h_i\}, \\
C_{i,6} &= \{w | w \in C_i, |p_{i+1}w| \geq h_{i+1}, |w| \geq h_{i+1}\}, \\
C_{i,7} &= \{w | w \in C_i, |p_{i+1}w| < h_{i+1}, |w| < h_i\}, \\
C_{i,8} &= \{w | w \in C_i, |p_{i+1}w| < h_{i+1}, |w| \geq h_i\}
\end{aligned}$$

Note that some of these subsets may be empty in some circumstances and that the following relationships hold:

$$C_{i,1} = (C_{i,4} \cup C_{i,7}) \setminus \{p_{i+1}\} \tag{10}$$

$$C_{i,2} = (C_{i,5} \cup C_{i,8}) \setminus \{p_{i+1}\} \tag{11}$$

$$C_{i,3} = C_{i,6} \setminus \{p_{i+1}\} \tag{12}$$

$$C_i \setminus \{p_{i+1}\} = C_{i,1} \cup C_{i,2} \cup C_{i,3} \tag{13}$$

$$C_i = C_{i,4} \cup C_{i,5} \cup C_{i,6} \cup C_{i,7} \cup C_{i,8}. \tag{14}$$

Furthermore, the intersection of any two of $C_{i,1}$, $C_{i,2}$ and $C_{i,2}$ is empty, as is that of any two of $C_{i,4}$, $C_{i,5}$, $C_{i,6}$, $C_{i,7}$ and $C_{i,8}$. Rewriting Eq. (8) for $i+1$, we get

$$\begin{aligned}
\delta_{i+1} &= \sum_{w \in C_{i+1}} b^{|w|} ||w| - h_{i+1}| \\
&= \sum_{w \in C_i \setminus \{p_{i+1}\}} b^{|w|} ||w| - h_{i+1}| + b^{|p_{i+1}|} \sum_{w \in C_i} b^{|w|} ||p_{i+1}w| - h_{i+1}|,
\end{aligned} \tag{15}$$

where we use Eq. (3) to split C_{i+1} into words from $C_i \setminus \{p_{i+1}\}$ and words of the form $p_{i+1}w$ with $w \in C$. We may thus re-write Eq. (15) as:

$$\begin{aligned}
\delta_{i+1} &= \sum_{w \in C_{i,1}} b^{|w|}(h_{i+1} - |w|) \\
&+ \sum_{w \in C_{i,2}} b^{|w|}(h_{i+1} - |w|) \\
&+ \sum_{w \in C_{i,3}} b^{|w|}(|w| - h_{i+1}) \\
&+ b^{|p_{i+1}|} \sum_{w \in C_{i,4}} b^{|w|}(|p_{i+1}| + |w| - h_{i+1}) \\
&+ b^{|p_{i+1}|} \sum_{w \in C_{i,5}} b^{|w|}(|p_{i+1}| + |w| - h_{i+1}) \\
&+ b^{|p_{i+1}|} \sum_{w \in C_{i,6}} b^{|w|}(|p_{i+1}| + |w| - h_{i+1})
\end{aligned}$$

$$
\begin{aligned}
&+ \quad b^{|p_{i+1}|} \sum_{w \in C_{i,7}} b^{|w|}(h_{i+1} - |p_{i+1}| - |w|) \\
&+ \quad b^{|p_{i+1}|} \sum_{w \in C_{i,8}} b^{|w|}(h_{i+1} - |p_{i+1}| - |w|). \qquad (16)
\end{aligned}
$$

Note that this now resolves the absolute values, as the relationship between h_{i+1} and $|w|$ and $|p_{i+1}|$ respectively is known for all contributing subsets. Substituting $(1 + b^{|p_{i+1}|})h_i$ for h_{i+1} as per Eq. (6), we get:

$$
\begin{aligned}
\delta_{i+1} \quad = \quad & \sum_{w \in C_{i,1}} b^{|w|}||w| - h_i| + b^{|p_{i+1}|}h_i \sum_{w \in C_{i,1}} b^{|w|} \\
- \quad & \sum_{w \in C_{i,2}} b^{|w|}||w| - h_i| + b^{|p_{i+1}|}h_i \sum_{w \in C_{i,2}} b^{|w|} \\
+ \quad & \sum_{w \in C_{i,3}} b^{|w|}||w| - h_i| - b^{|p_{i+1}|}h_i \sum_{w \in C_{i,3}} b^{|w|} \\
+ \quad & b^{|p_{i+1}|} \Bigg[- \sum_{w \in C_{i,4}} b^{|w|}||w| - h_i| + (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w \in C_{i,4}} b^{|w|} \\
+ \quad & \sum_{w \in C_{i,5}} b^{|w|}||w| - h_i| + (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w \in C_{i,5}} b^{|w|} \\
+ \quad & \sum_{w \in C_{i,6}} b^{|w|}||w| - h_i| + (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w \in C_{i,6}} b^{|w|} \\
+ \quad & \sum_{w \in C_{i,7}} b^{|w|}||w| - h_i| - (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w \in C_{i,7}} b^{|w|} \\
- \quad & \sum_{w \in C_{i,8}} b^{|w|}||w| - h_i| - (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w \in C_{i,8}} b^{|w|} \Bigg] \qquad (17)
\end{aligned}
$$

Using the identity:

$$
\sum_{w \in C_i} b^{|w|} = 1 \qquad (18)
$$

and its corrollary

$$
\sum_{w \in C_i \setminus \{p_{i+1}\}} b^{|w|} = 1 - b^{|p_{i+1}|}, \qquad (19)
$$

we can rewrite the 2nd, 4th, and 6th term after the second equal sign in Eq. (17) as:

$$
\begin{aligned}
& b^{|p_{i+1}|}h_i \sum_{w \in C_{i,1} \cup C_{i,2}} b^{|w|} - b^{|p_{i+1}|}h_i \sum_{w \in C_{i,3}} b^{|w|} \\
= \quad & b^{|p_{i+1}|}h_i \left[1 - b^{|p_{i+1}|} - 2 \sum_{w \in C_{i,3}} b^{|w|} \right] \\
= \quad & b^{|p_{i+1}|}h_i \left[1 + b^{|p_{i+1}|} - 2 \sum_{w \in C_{i,6}} b^{|w|} \right]
\end{aligned}
$$

and the 2nd, 4th, 6th, 8th, and 10th term in the square brackets as:

$$\begin{aligned}
& (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w\in C_{i,4}\cup C_{i,5}\cup C_{i,6}} b^{|w|} - (|p_{i+1}| - b^{|p_{i+1}|}h_i) \sum_{w\in C_{i,7}\cup C_{i,8}} b^{|w|} \\
= \; & (|p_{i+1}| - b^{|p_{i+1}|}h_i) \left[1 - 2\sum_{w\in C_{i,7}} b^{|w|} - 2\sum_{w\in C_{i,8}} b^{|w|}\right].
\end{aligned}$$

Noting here that

$$\delta_i \;=\; \sum_{j=1}^{3}\sum_{w\in C_{i,j}} b^{|w|}||w| - h_i| + b^{|p_{i+1}|}||p_{i+1}| - h_i|,$$

we get for the 1st, 3rd and 5th term:

$$\begin{aligned}
& \sum_{w\in C_{i,1}\cup C_{i,3}} b^{|w|}||w| - h_i| - \sum_{w\in C_{i,2}} b^{|w|}||w| - h_i| \\
= \; & \delta_i - b^{|p_{i+1}|}||p_{i+1}| - h_i| - 2\sum_{w\in C_{i,2}} b^{|w|}||w| - h_i| \\
= \; & \delta_i - b^{|p_{i+1}|}||p_{i+1}| - h_i| - 2\sum_{w\in C_{i,5}\cup C_{i,8}\setminus\{p_{i+1}\}} b^{|w|}||w| - h_i|.
\end{aligned}$$

Similarly, with:

$$\delta_i = \sum_{j=4}^{8}\sum_{w\in C_{i,j}} b^{|w|}||w| - h_i| \tag{20}$$

the odd-numbered terms in the square bracket of Eq. (17) sum up to

$$\begin{aligned}
& -\sum_{w\in C_{i,4}\cup C_{i,8}} b^{|w|}||w| - h_i| + \sum_{w\in C_{i,5}\cup C_{i,6}\cup C_{i,7}} b^{|w|}||w| - h_i| \\
= \; & \delta_i - 2\sum_{w\in C_{i,4}} b^{|w|}||w| - h_i| - 2\sum_{w\in C_{i,8}} b^{|w|}||w| - h_i|.
\end{aligned}$$

Combining these results, we get:

$$\begin{aligned}
& \delta_{i+1} \\
= \; & (1 + b^{|p_{i+1}|})\delta_i - b^{|p_{i+1}|}||p_{i+1}| - h_i| - 2\sum_{w\in C_{i,5}\cup C_{i,8}\setminus\{p_{i+1}\}} b^{|w|}||w| - h_i| \\
+ \; & b^{|p_{i+1}|}h_i\left[1 + b^{|p_{i+1}|} - 2\sum_{w\in C_{i,6}} b^{|w|}\right] \\
- \; & b^{|p_{i+1}|}\left[2\sum_{w\in C_{i,4}\cup C_{i,8}} b^{|w|}||w| - h_i| + (b^{|p_{i+1}|}h_i - |p_{i+1}|)\left[1 - 2\sum_{w\in C_{i,7}\cup C_{i,8}} b^{|w|}\right]\right].
\end{aligned} \tag{21}$$

The nonlinearity introduced by the absolute value in Eq. (8) means that we cannot eliminate the need to classify each codeword – Eqns. (11) and (12) imply that the remaining terms still amount to an iteration over each codeword length in C_i if we wish to be precise.

As the number of codewords in a T-code grows exponentially in i and even the number of different codeword lengths grows by $|p_i|$ or $|p_i|-1$ at each step, this precise approach is not practical if we wish to analyse long s.

We can however consider what we may expect to find for large i. The number of T-code codewords of any particular length ℓ is limited by the number of cyclic equivalence classes for ℓ [9], meaning that $(1+b^{|p_{i+1}|})$ is almost always very close to 1 for large i. This has a number of consequences in strings that exhibit generally accepted characteristics of randomness – i.e., the type of strings that are of particular interest in a randomness test.

For one, h_{i+1} will generally not be much larger than h_i. For the difference to be larger than one symbol, $|p_{i+1}|$ must be significantly smaller than its expectation value h_{i+1}, which is the exception rather than the rule. Consequently, the probability that the interval $[h_i, h_{i+1})$ will contain an integer decreases progressively with growing i. $C_{i,5}$ and $C_{i,8}$ will thus generally be empty, and $C_{i,4}$ generally contains all codewords shorter than h_i.

Similarly, we may drop terms that contain $b^{2|p_{i+1}|}$ as they contribute only very insignificantly in the limit. Observing further that the expected value for $||p_{i+1}|-h_i|$ is δ_i, and that for $|p_{i+1}|$ is h_i, we remain with the estimate:

$$\delta_{i+1} \approx \delta_i - 2b^{h_i}\left[h_i \sum_{w\in C_{i,6}} b^{|w|} - \sum_{w\in C_{i,4}} b^{|w|}(|w|-h_i) + h_i \sum_{w\in C_{i,7}} b^{|w|}\right]. \tag{22}$$

If, as we assume, $C_{i,5}$ and $C_{i,8}$ are empty, the union in Eqn. (14) simplifies and we can leverage the identity in Eq. (18) to obtain:

$$\delta_{i+1} \approx \delta_i + 2b^{h_i} \sum_{w\in C_{i,4}} b^{|w|}|w| \leq \delta_i + 2b^{h_i}h_i. \tag{23}$$

The estimated upper bound on the right can be justified by Eq. (2) and the fact that $C_{i,4} \subset C$. The bound may not necessarily be overly tight, however

4 Experimental results

Experimental verification of a randomness test is inherently problematic since there are no sequences whose randomness is known for certain – verifiably non-random sequences, on the other hand, are easier to supply. In a recent paper [13], Calude, Dinneen, Dumitrescu and Svozil have investigated very long bit sequences from four different sources: the digits of π, output from pseudo-random generators in the software packages Maple and Mathematica, and from two quantum random generators (Quantis and a proprietary generator). They concluded that the quantum randum generators showed a statistically significant behaviour difference compared to the pseudo-random generators, but only in a subset of the tests they conducted.

The experiments conducted for this paper used a subset of sequences from these sources to supply the "random" base case. For each source[2], 128 subsequences of 2^{20}

[2]the data from the proprietary generator was not available

Source	$E[\Delta_n(s)]$	$\Delta_n(s)$	avg. $\frac{\Delta_n(s)}{E[\Delta_n(s)]}$	R
Maple	1326 – 2425	16 – 4891	0.89	83
π	1352 – 2363	3 – 5352	0.90	79
Mathematica	1415 – 2394	2 – 6637	0.86	85
Quantis	1371 – 2373	6 – 5043	0.83	88

Table 1: $E[\Delta_n(s)]$ and $\Delta_n(s)$ for 1 Mbit sequences from various sources

bits (1 megabit) each were investigated to see whether $\Delta_n(s) \leq E[\Delta_n(s)]$ for the final n. As shown in Table 4, the difference for around two thirds of the subsequences did indeed fall within the respective tolerance band, i.e., under this test they would have to be regarded as random ("R"). The largest final $\Delta_n(s)/E[\Delta_n(s)]$ observed for any of the 512 sequences investigated was 4.25.

For comparison, 100 sequences of the same length were analysed that had been generated by the pseudo-random generator from the standard C library. This generator is known to produce weak random numbers. This analysis did not classify a single one of these sequences as "random".

Having established this base case sensitivity, the first 1 Mbit Qantis sequence that had been classified as random was subjected to "k-bit stuffing", i.e., the insertion of a 1 after each contiguous run of k 0's in the sequence, with $k = 1, \ldots, 18$. For $k = 18$, this added only a single bit to the entire sequence; for $k = 3$, 74,764 bits were added. All sequences with $k > 8$ classified as random, all those for $k \leq 8$ as non-random. For $k = 8$, 2049 1-bits were inserted - i.e., approximately one for every 512 bits. Moreover, the $\Delta_n(s)$ for each $k \leq 8$ clearly increased with decreasing k, while $\Delta_n(s)$ values for $k > 8$ did not show a clear trend.

5 Conclusions

The randomness test proposed here makes no further assumptions about random strings other than those already accepted widely in probability theory – it simply predicts what we would expect from random input to the algorithm and then compares this to the actual output. The discriminator tolerance given by $E[\Delta_n(s)]$ is generous due to the upper bound on the contributing δ_i. The recurrence algorithm that computes h_i and δ_i also allows for a random string to exhibit a significant degree of local non-randomness. One would thus rather expect false positives (non-random s classified as random) than false negatives. The behaviour of the test under deliberate degradation of randomness by bit stuffing also indicates that it sets reasonable criteria. It is thus a little surprising to see sequences with a good "random reputation" such as the above "fail" the test so frequently, especially when minor degradation of pattern variety seems to have little effect. The number of sequences studied for this paper was too small to detect statistically significant differences between the different "random" sources. Future work will investigate more of the data by Calude et al. to see whether the test is able to do this.

6 Acknowledgments

The author would like to thank Cristian Calude and Michael Dinneen for providing the data for the experiments, Kenji Hamano and Hirosuke Yamamoto for contributing such an elegant parsing algorithm, and Hirosuke Yamamoto and the School of Frontier Sciences at the University of Tokyo for hosting me in 2010 and thus providing the fertile ground on which the idea for this paper grew.

References

[1] C. E. Shannon: *A Mathematical Theory of Communications*. Bell Systems Technical Journal, 27:379, July 1948

[2] J. Ziv and A. Lempel: *Compression of Individual Sequences via Variable-Rate Coding*, IEEE Trans. Inform. Theory, vol 24, no. 5, pp. 530-536, September 1978.

[3] M. R. Titchener, "Digital encoding by means of new T-codes to provide improved data synchronisation and message integrity," *IEE Proc. – Computers and Digital Tech.*, vol. 131, no. 4, pp. 151–153, July 1984.

[4] M. R. Titchener, "Generalized T-codes: extended construction algorithm for self-synchronizing variable-length codes," *IEE Proc. – Commun.*, vol. 143, no. 3, pp. 122–128, June 1996.

[5] R. Nicolescu and M. R. Titchener, "Uniqueness theorems for T-codes," *Romanian J. Inform. Sci. Tech.*, vol. 1, 1998.

[6] U. Guenther, *Robust Source Coding with Generalized T-Codes*, PhD thesis, The University of Auckland, 1998, `http://www.tcs.auckland.ac.nz/~ulrich/phd.pdf`.

[7] National Institute of Standards and Technology, "A statistical test suite for random and pseudorandom number generators for cryptographic applications", NIST special publication 800-22, 2001.

[8] U. Speidel: *Constructing Finite Pseudo-Random Strings using a Codeset-Based Entropy*, *Fourth International Symposium on Communication Systems, Networks and Digital Signal Processing*, p.91, Newcastle, UK, 2004.

[9] T. A. Gulliver, I. Makwakwa, and U. Speidel, "On the generation of aperiodic and periodic necklaces via T-augmentation," *Fundamenta Informaticae*, vol. 83, pp. 91–107, 2008.

[10] J. Yang and U. Speidel, "A T-decomposition algorithm with $O(n \log n)$ time and space complexity," *Proc. IEEE Int. Symp. on Inform. Theory*, Sept. 2005, pp. 23–27.

[11] U. Speidel, "Constructing finite pseudo-random strings using codeset-based entropy measures", IEE Proceedings - Circuits, Devices and Systems, 153(4), pp. 315–319, 2006.

[12] K. Hamano and H. Yamamoto, "A Randomness Test Based on T-codes", pp. 1095–1100, Proceedings of the International Symposium on Information Theory and its Applications, Auckland, New Zealand, 2008

[13] C. Calude and M. Dinneen and M. Dumitrescu and K. Svozil, "Experimental evidence of quantum randomness incomputability", Physical Review A 82, 022102, 2010.