New Statistical Randomness Tests: 4-Bit Template Matching Tests

Fatih SULAK
Department of Mathematics, Atılım University, Ankara, Turkey
fatih.sulak@atilim.edu.tr

Abstract. For cryptographic algorithms, secret keys should be generated randomly as the security of the system depends on the key, therefore generation of random sequences is vital. Randomness testing is done by means of statistical randomness tests. In this work, we show that the probabilities for the overlapping template matching test in the NIST test suite is only valid for a specific template and need to be recalculated for the other templates. We calculate the exact distribution for all 4-bit templates and propose new randomness tests, namely template matching tests. The new tests can be applied to any sequence of minimum length 5504 where the overlapping template matching test in the NIST test suite can only be applied to sequences of minimum length $10^6$. Moreover, we apply the proposed tests to biased nonrandom data and observe that the new tests detect the nonrandom behaviour of the generator even for a bias 0.001, where the template matching tests in NIST cannot detect that bias.

Keywords: Cryptography, Overlapping Template Matching Test, Statistical Randomness Testing, NIST Test Suite

1 Introduction

Random sequences and random numbers are used in many fields, such as statistics, computer simulations, cryptography, and so on. In cryptography, random sequences are needed for several applications, such as the generation of primes in RSA encryption, secret keys in symmetric encryption, challenges in challenge-response protocols, initialization vectors, salts in hash functions and the like, but the most common application is the generation of secret keys.

Secret keys should be generated randomly so that the best option of the attacker should not be better than trying all possible elements in the set from which the key is chosen. If an attacker narrows down the number of possible keys, then the protocol is assumed to be broken. In 1996, Goldberg and Wagner [10] showed that the “random numbers” used to generate the keys in Netscape SSL protocol were based on the time of the processor and therefore predictable, which helped them to find a major weakness in the protocol. Thus, it is vital to use an algorithm which produces random numbers properly.

Ideally, random numbers should be produced by true random sources, like atmospheric noise, thermal noise or noise in an electrical circuit. These generators are called
true random number generators (TRNGs). However, producing random numbers by TRNGs is usually inefficient, therefore deterministic algorithms are generally used to produce random numbers. These algorithms are called pseudorandom number generators (PRNGs).

An example of a PRNG is the linear congruential generator [21] which produces a pseudorandom sequence \( x_1, x_2, x_3, \ldots \) using the linear recurrence

\[
x_n = a \cdot x_{n-1} + b \pmod{m}
\]

where \( x_0 \) is the seed and \( a, b, \) and \( m \) are parameters.

The output sequences of PRNGs should be statistically indistinguishable from truly random sequences, therefore statistical analysis of PRNGs are crucial, and their analysis are performed by statistical randomness testing. In order to test a PRNG, first an output sample is produced, then this sample is tested by various statistical randomness tests. A test suite is a collection of statistical randomness test that are designed to test the randomness properties of sequences. There are several test suites in the literature [17], [23], [18], [3], [24], [2]. Similarly there are several individual statistical randomness tests [20], [6], [12], [13], [14], [15], [26], [1], [8].

The outputs of symmetric encryption algorithms should be indistinguishable from random sequences, that is algorithms are expected to behave like PRNGs. Hence their analysis from this point of view is crucial. Generally, a sample output set is taken from a symmetric encryption algorithm, and this set is evaluated in terms of randomness by a test suite.

NIST test suite [2] is the most popular test suite for cryptographic applications. The statistical analysis of AES finalist algorithms is performed by Soto et. al. using the NIST test suite [25]. Some tests in the suite require sequences of length \( 10^6 \), while the outputs of AES finalist algorithms are 128 bits. Soto et. al. concatenate the outputs of the algorithms to obtain long sequences in order to apply all the tests. Recently, Sulak et.al. propose an alternative method where they compute and use the exact distributions instead of approximations or asymptotic distributions [27]. Having these exact probabilities, the necessity of long sequences is reduced and they apply the randomness tests directly to the outputs of the algorithms instead of concatenating them.

There are several studies for the tests of the NIST test suite [28], [7], [9], [22], [11]. Okutomi et. al. apply the tests in the NIST test suite to the random data taken from cryptographic algorithms DES and SHA-1 [22]. They observe that the Maurer’s universal statistical test and the overlapping template matching test have problems with the ratio of the random data that pass the tests. Hamano et. al. correct the probabilities for the overlapping template matching test where they take the template as \( B = 111111111 \) [11] and NIST update the probabilities accordingly. However, as noted in [28], the probability of each pattern depends on the pattern itself. In this work, we set \( m = 4 \) and classify 16 possible patterns into four groups. Then for each 16 patterns, we evaluate the exact probabilities using combinatorial approaches. Afterwards, we propose four new statistical randomness tests which can be applied to short sequences and long sequences. We observe that the probabilities are not the same for each overlapping template, which shows that the probabilities for overlapping template matching test in the NIST is valid only for \( B = 111111111 \). We apply the new tests to random data taken
from various PRNGs, and to nonrandom data to observe the power of new tests, and compare the results with NIST test suite.

The organization of the paper is as follows. In section 2, we give preliminaries. In section 3, we obtain the exact distributions. In section 4, we define new statistical randomness tests and state the corresponding bin probabilities. In section 5, we apply the new tests to random and nonrandom data and observe the power of new tests. In section 6, we conclude the paper with some future work.

2 Preliminaries

A statistical randomness test is a procedure, which takes a binary sequence as an input and tests a null hypothesis \( H_0 \) stating that the given input sequence is random. The test examines the input sequence, produces a real number between 0 and 1 which is called \( p \)-value, and accepts or rejects the hypothesis using a probabilistic approach. As it is probabilistic, the test may reject truly random sequences, and in that case, type I error has occurred. The probability of such an error is called the level of significance of the test and denoted by \( \alpha \). If \( p \)-value produced by the test is greater than \( \alpha \) then \( H_0 \) is accepted, otherwise rejected [21]. \( \alpha \) is usually set to 0.01 for cryptographic applications [2].

The \( \chi^2 \) distribution is used to compare how good the observed frequencies of events fits to the corresponding expected frequencies under the hypothesized distribution.

Definition 1. [21] A random variable has a \( \chi^2 \) distribution with degrees of freedom \( v \) if the corresponding probability density function \( f(x) = 0 \) for \( x < 0 \) and

\[
f(x) = \frac{1}{\Gamma(v/2)2^{v/2}} x^{v/2 - 1} e^{-x/2}, \quad x \geq 0
\]

where \( v \) is a positive integer and \( \Gamma \) is the gamma function, that is \( \Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx \), for \( t > 0 \).

\( \chi^2 \) Goodness of Fit Test is a statistical randomness test where the distribution of the test statistic follows \( \chi^2 \) distribution, assuming \( H_0 \) is true. In other words, let \( E_i \) be the expected frequencies and \( F_i \) be the observed frequencies for \( 1 \leq i \leq k \). Then

\[
\chi^2 = \sum_{i=1}^{k} \frac{(F_i - E_i)^2}{E_i} \quad \text{and} \quad p\text{-value} = \text{igamc}\left(2, \frac{\chi^2}{2}\right)
\]

where igamc is the incomplete gamma function [2].

3 4-Bit Template Matching Tests

The subject of the 4-bit template matching tests is the frequency of a pre-specified template in a binary sequence. Similar tests are defined in the NIST test suite, namely the nonoverlapping template matching test and the overlapping template matching test.
In both tests, first an \( m \)-bit template \( B \) is chosen, and the sequence subject to the test is divided into \( N \) subsequences of length \( M \). An \( m \)-bit window is used to search the \( m \)-bit overlapping blocks of each subsequence. Then for each block, the number of the template \( B \) in that subsequence is counted. Let \( W_i \) denote the number of \( B \) in the \( i \)th block. For the overlapping template matching test \( M \) is set to 1032. Let \( \pi_j \) denote the probability that \( W_i = j \) for \( 0 \leq j \leq 4 \) and \( \pi_5 \) denote the probability that \( W_i \geq 5 \). For \( M = 1032 \) and \( B = 111111111 \), the exact probabilities \( \pi_j \) are calculated in [11] using a recursion. A \( p \)-value is produced using \( \chi^2 \) goodness of fit test using those probabilities.

For the nonoverlapping template matching test, the pre-specified template is chosen in a manner that if the template is observed somewhere in the sequence, then it should not be seen before the template is completed. As noted in NIST test suite, if the pattern is observed somewhere in the sequence, it cannot be observed again for the next \( m - 1 \) blocks, hence the \( m \)-bit window slides \( m \) bits. This shows that the distribution is same for all nonoverlapping templates. Using the similar idea, we have the following proposition.

**Proposition 1.** The distribution of the frequency of a pre-specified template depends only on the number of overlapping bits in the template.

**Proof.** Assume that the pre-specified template \( B \) of length 4 has \( k \) overlapping bits. If \( B \) is observed somewhere in the sequence, the next \( 3 - k \) blocks cannot be equal to \( B \) as \( B \) has \( k \) overlapping bits. The latter block may be equal to \( B \) with probability \( \frac{1}{2^{3-k}} \). This shows that the distribution depends only on the number of overlapping bits in the block.

Using this proposition, we classify the 4-bit templates according to their number of overlapping bits. There 4 types of blocks:

1. Non-overlapping blocks: 0001, 0011, 0111, 1000, 1100, 1110
2. One bit overlapping blocks: 0010, 0100, 0110, 1001, 1011, 1101
3. Two bit overlapping blocks: 0101, 1010
4. Three bit overlapping blocks: 0000, 1111

We choose one representative block from each type and find the exact distributions. Different from the previous approaches, we assume that the bits are circular in each subsequence.

**Example 1.** Let the subsequence be 1000011000. Then the number of 001 blocks is two, one starting from the fourth bit and one starting from the ninth bit, and the number of 000 blocks is three, starting from the second bit, the third bit and the eighth bit.

First we state some combinatorial formulae which we use in the calculation of probabilities.

**Lemma 1.** [4] The number of nonnegative integer solutions of the equation \( x_1 + x_2 + \cdots + x_b = a \) is \( \binom{a + b - 1}{b - 1} \).
Lemma 2. [4] The number of integer solutions of the equation $x_1 + x_2 + \cdots + x_b = a$ with $x_i \geq c$ for $1 \leq i \leq b$ is

$$\binom{a - b(c - 1) - 1}{b - 1}.$$

Proof. With the substitution $x_i = y_i + c$ we get,

$$(y_1 + c) + (y_2 + c) + \cdots + (y_b + c) = a$$

$$y_1 + y_2 + \cdots + y_b = a - bc.$$

From Lemma 1 it follows that the number of solutions is:

$$\binom{a - bc + b - 1}{b - 1} = \binom{a - b(c - 1) - 1}{b - 1}.$$

Lemma 3. [4] [Inclusion - Exclusion Principle] The number of nonnegative integer solutions of the equation $x_1 + x_2 + \cdots + x_b = a$ with $x_i \leq c$ for $1 \leq i \leq b$ is

$$\sum_{j=0}^{b} \binom{a + b - 1 - j(c + 1)}{b} \binom{b}{j} (-1)^j.$$

3.1 Non-overlapping Case

In order to define a randomness test, we need to find the probability that the pre-specified template $B$ occurs $k$ times in the subsequence. For the non-overlapping case, we choose $B = 0001$ and compute the probability accordingly. We assume that we know the weight $W$ and the number of runs $V$ of the sequence.

Theorem 1. Let $(a_1, a_2, \ldots, a_n)$ be a binary sequence and $b_i = a_ia_{i+1}a_{i+2}a_{i+3}$ be blocks of length 4 for $1 \leq i \leq n$ with $a_{n+j} = a_j$ for $j = 1, 2, 3$, and let $K$ denote the number of 0001 blocks among $b_i$ for $1 \leq i \leq n$. Also let $w$ be the weight of the sequence and $2r$ be the number of runs in the sequence. If the sequence is not all zero or all one then

$$\Pr(K = k) = \frac{n}{r \cdot 2^r} \binom{w - 1}{r - 1} \sum_{a=0}^{r-k} \binom{r-k}{a} \binom{n - w - r - a - k - 1}{k - 1}.$$

Proof. First note that, the number of runs is even if the sequence is not all zero or all one sequence. We assume the bits are arranged on a circle and we write ‘one’s and ‘zero’s consecutively to define $2r$ runs. As a result, $w - r$ ‘one’s and $n - w - r$ ‘zero’s remain.

As all the 0001 blocks contain 01 blocks, if a run of ‘zero’s has more than 2 ‘zero’s, it produces exactly one 0001 block. Now, we find the distribution of $w - r$ many ‘one’s and $n - w - r$ many ‘zero’s so that the number of 0001 blocks is $k$. The number of such arrangements is equal to the number of nonnegative integer solutions of the system

$$x_1 + x_2 + \cdots + x_r = n - w - r$$
$$y_1 + y_2 + \cdots + y_r = w - r$$
with an additional condition that exactly $k$ of $x_i$’s satisfy $x_i \geq 2$ for $1 \leq i \leq r$. This additional condition guarantees that there are exactly $k$ many 0001 blocks. The second equation has \( \binom{w-1}{r-1} \) solutions by Lemma 1.

\[
\sum_{i=1}^{k} x_i + x_{k+1} + \cdots + x_{k+a} + x_{k+a+1} + \cdots + x_r = n - w - r
\]

In order to find the number of solutions of the first equation, we may assume that $x_i \geq 2$ for $1 \leq i \leq k$ (with a factor \( \binom{r}{k} \)), $x_j = 1$ for $k + 1 \leq j \leq k + a$ (with a factor of \( \binom{r - k}{a} \)), $x_s = 0$ for $k + a + 1 \leq s \leq r$, hence the number of integer solutions of the first equation is

\[
x_1 + x_2 + \cdots + x_k = n - w - r - a, \quad x_i \geq 2, \quad 1 \leq i \leq k
\]

which is \( \binom{n - w - r - a - k - 1}{k - 1} \) by Lemma 2. Note that if $k = 0$, we cannot apply Lemma 2, in that case we assume there is only one solution. We use this assumption throughout the paper.

Moreover, each arrangement on the circle gives $n$ sequences. However, since there are $r$ many 01 blocks, $r$ of these sequences are identical. Thus, considering the circular symmetry, other than all zero or all one sequence, we have

\[
\Pr(K = k) = \frac{n}{r \cdot 2^n} \binom{w-1}{r-1} \binom{r-k}{a} \sum_{a=0}^{r-k} \binom{n - w - r - a - k - 1}{k - 1}.
\]

**Example 2.** Assume that $n = 8$, $w = 3$, $r = 2$, $k = 1$. We need to find the number of integer solutions of the system

\[
x_1 + x_2 = 3
\]
\[
y_1 + y_2 = 1
\]

$x_1 = 3, x_2 = 0, y_1 = 0, y_2 = 1$ is a solution. The corresponding sequence is obtained as:

\[
\begin{align*}
0000 & 1 \\
1 & 0 \\
0 & x_1 = 3 \\n11 & y_2 = 1
\end{align*}
\]

Note that as $x_1 \geq 2$, it produces exactly one 0001 block. We show all the solutions and the corresponding sequences in the table.

Moreover, each arrangement gives 8 sequences, and two sequences are always identical. Consider the solution $x_1 = 3, x_2 = 0, y_1 = 0, y_2 = 1$ and its corresponding sequence 00001011. It produces 00010110, 00101100, 01011000, 10110000, 01100001, 11000010, and 10000101. But note that we also obtain 01100001 as the corresponding sequence of the solution $x_1 = 0, x_2 = 3, y_1 = 0, y_2 = 1$. As a result, there are \( \frac{8 \cdot 8}{2} = 32 \) sequences, which is consistent with Theorem 1.
Table 1. An Example for Theorem 1

<table>
<thead>
<tr>
<th>$y_1 = 0, y_2 = 1$</th>
<th>$y_1 = 1, y_2 = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1 = 3, x_2 = 0$</td>
<td>00001011 00001101</td>
</tr>
<tr>
<td>$x_1 = 2, x_2 = 1$</td>
<td>00010011 00011001</td>
</tr>
<tr>
<td>$x_1 = 1, x_2 = 2$</td>
<td>00100011 00110001</td>
</tr>
<tr>
<td>$x_1 = 0, x_2 = 3$</td>
<td>01000011 01100001</td>
</tr>
</tbody>
</table>

In the template matching test, we need to find the probabilities independent of weight and number of runs of the sequence. For this reason, we state the following corollary.

**Corollary 1.** Let $\{a_1, a_2, \ldots, a_n\}$ be a binary sequence and $b_i = a_i a_{i+1} a_{i+2} a_{i+3}$ be blocks of length 4 for $1 \leq i \leq n$ with $a_{i+j} = a_j$ for $j = 1, 2, 3$, and let $K$ denote the number of 0001 blocks among $b_i$ for $1 \leq i \leq n$. If the sequence is not all zero or all one then

$$Pr(K = k) = \sum_{w=1}^{n-1} \sum_{r=1}^{[n/2]} \frac{n}{r \cdot 2^n} \binom{w-1}{r-1} \binom{r-k}{a} \binom{n-w-r-a-1}{k-1} \sum_{a=0}^{r-k} \binom{w-2r+a-1}{r-k-a-1}.$$

**Proof.** Since we compute $Pr(K = k | W = w, V = 2r)$ in Theorem 1, by summing over all possible weights and runs, we obtain $Pr(K = k)$.

### 3.2 One-bit Overlapping Case

For one-bit overlapping case, we choose $B = 0110$ and obtain the probability accordingly.

**Theorem 2.** Let $\{a_1, a_2, \ldots, a_n\}$ be a binary sequence and $b_i = a_i a_{i+1} a_{i+2} a_{i+3}$ be blocks of length 4 for $1 \leq i \leq n$ with $a_{i+j} = a_j$ for $j = 1, 2, 3$, and let $K$ denote the number of 0110 blocks among $b_i$ for $1 \leq i \leq n$. Also let $w$ be the weight of the sequence and $2r$ be the number of runs in the sequence. If the sequence is not all zero or all one then

$$Pr(K = k) = \frac{n}{r \cdot 2^n} \binom{n-w-1}{r-1} \sum_{a=0}^{r-k} \binom{r-k}{a} \binom{w-2r+a-1}{r-k-a-1}.$$

**Proof.** Using the similar idea of the proof of the Theorem 1, we assume the bits are arranged on a circle and we write ‘one’s and ‘zero’s consecutively to define 2$r$ runs. As a result, $w - r$ ‘one’s and $n - w - r$ ‘zero’s remain.

As all the 0110 blocks contain 01 blocks, if a run of ‘one’s contains exactly two ‘one’s, it produces exactly one 0110 block. Therefore, we need to find the distribution of $w - r$ many ‘one’s and $n - w - r$ many ‘zero’s so that the number of 0110 blocks is $k$. The number of such arrangements is equal to the number of nonnegative integer solutions of the system

$$x_1 + x_2 + \cdots + x_r = n - w - r$$

$$y_1 + y_2 + \cdots + y_r = w - r$$
Corollary 2. Let \( y_i \) be blocks among \( b_i \) for \( 1 \leq i \leq n \) with \( a_{i+j} = a_j \) for \( j = 1, 2, 3 \), and let \( K \) denote the number of 0110 blocks among \( b_i \) for \( 1 \leq i \leq n \). If the sequence is not all zero or all one then

\[
Pr(K = k) = \sum_{w=1}^{n-1} \sum_{r=1}^{\lfloor n/2 \rfloor} \frac{n}{r \cdot 2^n} \binom{n-w-1}{r} \binom{r-k}{a} \sum_{a=0}^{r-k} \binom{w-2r+a-1}{r-k-a-1}.
\]

Proof. Since we compute \( Pr(K = k|W = w, V = 2r) \) in Theorem 2, by summing over all possible weights and runs, we obtain \( Pr(K = k) \).

3.3 Two-bit Overlapping Case

In this case we choose the pre-specified block as 1010. Different from Theorem 1 and Theorem 2, to obtain the probabilities, we use another model where \( x_i \)'s are modeled as red boxes and \( y_i \)'s are modeled as white boxes.

Theorem 3. Let \( \{a_1, a_2, \ldots, a_n\} \) be a binary sequence and \( b_i = a_i a_{i+1} a_{i+2} a_{i+3} \) be blocks of length 4 for \( 1 \leq i \leq n \) with \( a_{i+j} = a_j \) for \( j = 1, 2, 3 \), and let \( K \) denote the number of 1010 blocks among \( b_i \) for \( 1 \leq i \leq n \). Also let \( w \) be the weight of the sequence and \( 2r \) be the number of runs in the sequence. If the sequence is not all zero or all one then

\[
Pr(K = k) = \frac{n}{r \cdot 2^n} \sum_{a=0}^{r-1} \binom{r}{a} \binom{n-w-1}{r-a} \binom{w-a-1}{r-k-1}.
\]
Proof. Using the similar idea of the proof of the Theorem 1, we assume the bits are arranged on a circle and we write ‘one’ and ‘zero’ consecutively to define 2r runs. As a result, \( w - r \) ‘one’ and \( n - w - r \) ‘zero’ remain.

Now, we find the distribution of \( n - w - r \) many ‘zero’ and \( w - r \) many ‘one’ so that the number of 1010 blocks is \( k \). We use another model to solve this problem. Assume that there are \( r \) red-white box pairs and we distribute \( w - r \) balls into white boxes and \( n - w - r \) balls into red boxes. \( \text{Pr}(K = k) \) is the probability where exactly \( k \) pairs are empty.

In other words, we find the number of nonnegative integer solutions of the system

\[
x_1 + x_2 + \cdots + x_r = n - w - r \\
y_1 + y_2 + \cdots + y_r = w - r
\]

with the condition that \( x_i + y_i = 0 \) is satisfied for exactly \( k \) different values of \( i \) where \( 1 \leq i \leq r \).

Let \( a \) of the red boxes \( x_i \) be empty. Assume that the empty boxes are the first \( a \) boxes (with a factor of \( \binom{r}{a} \)). If we consider the first \( a \) white boxes, \( k \) of them should be empty. Assume that the empty boxes are the first \( k \) boxes (with a factor of \( \binom{a}{k} \)). Now, we need to find the number of integer solutions of the system

\[
\begin{align*}
\frac{x_1 + \cdots + x_a + x_{a+1} + \cdots + x_r}{=0} &= n - w - r \\
\frac{y_1 + \cdots + y_k + y_{k+1} + \cdots + y_a + y_{a+1} + \cdots + y_r}{=0 & \geq 1} &= w - r.
\end{align*}
\]

The first equation has \( \binom{n - w - r - 1}{r - a - 1} \) solutions and the second equation has \( \binom{w - a - 1}{r - k - 1} \) solutions by Lemma 2.

Considering the circular symmetry, other than all zero or all one sequence, we have

\[
\text{Pr}(K = k) = \frac{n}{r \cdot 2^n} \sum_{a=k}^{r-1} \binom{r}{a} \binom{n - w - r - 1}{r - a - 1} \binom{w - a - 1}{r - k - 1}.
\]

Corollary 3. Let \( \{a_1, a_2, \ldots, a_n\} \) be a binary sequence and \( b_i = a_ia_{i+1}a_{i+2}a_{i+3} \) be blocks of length 4 for \( 1 \leq i \leq n \) with \( a_{i+j} = a_i \) for \( j = 1, 2, 3 \), and let \( K \) denote the number of 1010 blocks among \( b_i \) for \( 1 \leq i \leq n \). If the sequence is not all zero or all one then

\[
\text{Pr}(K = k) = \sum_{w=2}^{n-2} \sum_{r=2}^{n} \frac{n}{r \cdot 2^n} \sum_{a=k}^{r-1} \binom{r}{a} \binom{n - w - r - 1}{r - a - 1} \binom{w - a - 1}{r - k - 1}.
\]

Proof. Since we compute \( \text{Pr}(K = k | W = w, V = 2r) \) in Theorem 3, by summing over all possible weights and runs, we obtain \( \text{Pr}(K = k) \).
3.4 Three-bit Overlapping Case

We choose the pre-specified block as 1111 for the three-bit overlapping case. We apply inclusion-exclusion principle to obtain the probability.

**Theorem 4.** Let \( \{a_1, a_2, \ldots, a_n\} \) be a binary sequence and \( b_i = a_ia_{i+1}a_{i+2}a_{i+3} \) be blocks of length 4 for \( 1 \leq i \leq n \) with \( a_{i+j} = a_j \) for \( j = 1, 2, 3 \), and let \( K \) denote the number of 1111 blocks among \( b_i \) for \( 1 \leq i \leq n \). Also let \( w \) be the weight of the sequence and \( 2r \) be the number of runs in the sequence. If the sequence is not all zero or all one then

\[
Pr(K = k) = \frac{n}{r \cdot 2^n} \binom{n - w - 1}{r - 1} \sum_{t=0}^{r} \binom{r}{t} \binom{k - 1}{t - 1}
\cdot \sum_{i=0}^{r} \binom{w - k - 3t - 3i - 1}{r - t - 1} \binom{r - t}{i} (-1)^i.
\]

**Proof.** Using the similar idea of the proof of the Theorem 1, we assume the bits are arranged on a circle and we write ‘one’s and ‘zero’s consecutively to define 2r runs. As a result, \( w - r \) ‘one’s and \( n - w - r \) ‘zero’s remain. Now, we find the distribution of \( n - w - r \) many ‘zero’s and \( w - r \) many ‘one’s so that the number of 1111 blocks is \( k \) and \( V = 2r \). Similarly we should find the the number of integer solutions of the system

\[
x_1 + x_2 + \cdots + x_r = n - w - r, \quad x_i \geq 0, \quad 1 \leq i \leq r
\]

\[
y_1 + y_2 + \cdots + y_r = w - r, \quad y_j \geq 0, \quad 1 \leq j \leq r
\]

which should also satisfy that the number of 1111 blocks is \( k \).

The first equation has \( \binom{n - w - 1}{r - 1} \) solutions by Lemma 1. Without loss of generality (or multiplying by \( \binom{r}{t} \)), assume \( y_j \geq 3 \) for \( 1 \leq j \leq t \), and \( 0 \leq y_s \leq 2 \) for \( t + 1 \leq s \leq r \).

We find the number of solutions of the second equation in two parts:

Each run of ‘one’s with length \( l \geq 4 \) defines \( l - 3 \) many 1111 blocks. Hence, in order to have \( k \) many 1111 blocks, we should have

\[
y_1 - 2 + y_2 - 2 + \cdots + y_t - 2 = k, \quad y_j \geq 3
\]

\[
y_1 + y_2 + \cdots + y_t = k + 2t, \quad y_j \geq 3
\]

therefore, the number of solutions are \( \binom{k - 1}{t - 1} \) (and \( t = 0 \Leftrightarrow k = 0 \)) by Lemma 2.

As the weight of the sequence is \( w \), we have:

\[
y_{t+1} + y_{t+2} + \cdots + y_r = w - r - k - 2t, \quad 0 \leq y_j \leq 2, \quad t + 1 \leq j \leq r.
\]

We apply the Inclusion Exclusion principle to find the number of the solutions and obtain:

\[
\sum_{i=0}^{r} \binom{w - k - 3t - 3i - 1}{r - t - 1} \binom{r - t}{i} (-1)^i.
\]
Considering the circular symmetry, other than all zero or all one sequence, we have

\[
Pr(K = k) = \frac{n}{r \cdot 2^n} \left( n - w - 1 \right) \sum_{t=0}^{r} \binom{r}{t} \binom{k-1}{t-1} \cdot \sum_{i=0}^{r} \binom{w - k - 3i - 1}{r-t} \binom{r-t}{i} (-1)^i.
\]

Note that we may use other models to prove the theorem, but inclusion-exclusion principle can be generalized to other \( m \) values. We can still apply inclusion-exclusion principle if we choose \( m = 9 \) and \( B = 1111111111 \).

**Corollary 4.** Let \( \{a_1, a_2, \ldots, a_n\} \) be a binary sequence and \( b_i = a_i a_{i+1} a_{i+2} a_{i+3} \) be blocks of length 4 for \( 1 \leq i \leq n \) with \( a_{n+j} = a_j \) for \( j = 1, 2, 3 \), and let \( K \) denote the number of 1111 blocks among \( b_i \) for \( 1 \leq i \leq n \). If the sequence is not all zero or all one then

\[
Pr(K = k) = \sum_{w=1}^{n-1} \sum_{r=1}^{[n/2]} \frac{n}{r \cdot 2^n} \left( n - w - 1 \right) \sum_{t=0}^{r} \binom{r}{t} \binom{k-1}{t-1} \cdot \sum_{i=0}^{r} \binom{w - k - 3i - 1}{r-t} \binom{r-t}{i} (-1)^i.
\]

**Proof.** Since we compute \( Pr(K = k \mid W = w, V = 2r) \) in Theorem 4, by summing over all possible weights and runs, we obtain \( Pr(K = k) \).

### 4 Tests Descriptions

The subject of the 4-bit template matching tests are the number of a pre-specified template in a sequence. We apply \( \chi^2 \) Goodness of Fit Test to measure how well the observed values fit the expected values. For this purpose, we divide the sequence into 128-bit blocks and find the number of occurrences of the template in each block. Afterwards, we apply \( \chi^2 \) test with 5 bins and produce \( p \)-value using the Table 2. The probabilities in Table 2 are evaluated using Corollaries 1, 2, 3, 4.

**Table 2.** Bin Probabilities for 4-Bit Template Matching Tests

<table>
<thead>
<tr>
<th>0-bit</th>
<th>1-bit</th>
<th>2-bit</th>
<th>3-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-6</td>
<td>0.24205627</td>
<td>0.16353105</td>
<td>0.19990529</td>
</tr>
<tr>
<td>7</td>
<td>0.17082354</td>
<td>0.27433485</td>
<td>0.18737326</td>
</tr>
<tr>
<td>8</td>
<td>0.18629990</td>
<td>0.15466458</td>
<td>0.08721848</td>
</tr>
<tr>
<td>9</td>
<td>0.16401892</td>
<td>0.24482145</td>
<td>0.16931656</td>
</tr>
<tr>
<td>10-128</td>
<td>0.23680138</td>
<td>0.16264780</td>
<td>0.11861450</td>
</tr>
</tbody>
</table>

Assume that we want to test a binary sequence of length \( n \) using the template matching test. We can summarize the procedure as follows:
– Choose a 4-bit template $B$.
– Divide the sequence into $M = \left\lfloor \frac{n}{128} \right\rfloor$ many 128-bit blocks.
– For each block, write the first 3 bits to the end of the sequence.
– Find the occurrence of $B$ among the first block in an overlapping manner and increment the corresponding bin value, call them $F_i$ for $1 \leq i \leq 5$. Repeat the same procedure for all blocks.
– Apply $\chi^2$ Goodness of Fit Test, that is evaluate

$$\chi^2 = \sum_{i=1}^{5} \left( \frac{(F_i - M \cdot p_i)^2}{M \cdot p_i} \right)$$

and $p$-value $= \text{igamc}(2, \chi^2 / 2)$

where $p_i$’s are obtained from Table 2 according to the number of overlapping bits in the template $B$.
– If $p$-value $< 0.01$ conclude as nonrandom, else conclude as random.

Let us demonstrate the template matching test using a simple example.

**Example 3.** Assume the sequence subject to template matching test is

$$\{0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0\}$$

and assume that we choose a template $B = 0010$. Note that $B$ is a one-bit overlapping block. We divide the sequence into three 8-bit blocks, and extend the blocks.

– Block 1: $\{0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0\}$, there are two occurrences of $B$, the first one starts from the first bit and the second one starts from the fourth bit.
– Block 2: $\{0, 1, 0, 0, 1, 0, 1, 0, 1, 0\}$, $B$ does not occur in this block.
– Block 3: $\{1, 0, 1, 0, 1, 0, 0, 1, 0, 1\}$, there is one occurrence of $B$ starting from the seventh bit.

As a result we find that $F_1 = 1$, $F_2 = 1$ and $F_3 = 1$.

In this example we divide the sequence into 8-bit blocks, instead of 128-bit blocks. Note that we use different values for $F_i$’s and thus we cannot produce a $p$-value using Table 2, as the block size is not 128. The pseudocode of the test is stated in Algorithm 4.1. As the expected number of items in each bin should be at least 5, and the minimum probability in Table 2 is 0.1186145, the sequence subject to the test should be at least $128 \cdot \left\lfloor \frac{5}{0.1186145} \right\rfloor = 5504$ bits.
Algorithm 4.1: Template Matching Test \( \{a_1, a_2, \ldots, a_n\}, B \)

\[
F_1 = 0, \quad F_2 = 0, \quad F_3 = 0, \quad F_4 = 0, \quad F_5 = 0;
\]

\[
M = \left\lfloor \frac{n}{128} \right\rfloor;
\]

for \( i \leftarrow 0 \) to \( M - 1 \)
do
for \( j \leftarrow 1 \) to \( 128 \)
do

\[
b_j = a_{128i+j};
\]

\[
b_{129} = a_{128i+1}, b_{130} = a_{128i+2}, b_{131} = a_{128i+3};
\]

\[
count = 0;
\]

for \( j \leftarrow 1 \) to \( 128 \)
do

\[
\begin{cases}
\text{if } b_j b_{j+1} b_{j+2} b_{j+3} = B \\
\text{then } count + +;
\end{cases}
\]

Increment \( F_i \) according to Table 2;

Apply \( \chi^2 \) of Goodness of Fit test to \( F_1, F_2, F_3, F_4, F_5 \);

return \( (p \text{-value}) \)

Example 4. Assume the sequence subject to template matching test is the first 5504 bits of the binary expansion of \( \pi \) and assume that we choose the template as \( B = 1111 \). Note that \( B \) is a three-bit overlapping block. We divide the sequence into forty three 128-bit blocks, and we find that \( F_1 = 11, F_2 = 7, F_3 = 10, F_4 = 6, \) and \( F_5 = 9 \). Using Table 2, we find the \( p \)-value as \( 0.861602 \).

5 Simulation Results

In this section, we apply the new statistical randomness tests to various sequences. We compare the new tests with the randomness tests in the NIST test suite [2]. There are 16 possible templates, and we choose a sample template from each class. For non-overlapping case we choose \( B = 0001 \), for one-bit overlapping case we choose \( B = 0010 \), for two-bit overlapping case we choose \( B = 0101 \), and for three-bit overlapping case we choose \( B = 1111 \). For the tests in the NIST test suite, we choose \( M = 128 \) for Frequency Test within a Block, \( M = 10^4 \) for Test for the Longest Run of Ones in a Block, \( M = 32 \) for Binary Matrix Rank Test, \( m = 9 \) and \( B = 000000001 \) for Non-overlapping Template Matching Test, \( m = 9 \) and \( B = 111111111 \) for Overlapping Template Matching Test, \( L = 7 \) and \( L = 10 \) for Maurer’s Universal Statistical Test, \( M = 500 \) for Linear Complexity Test, \( m = 16 \) for Serial Test, \( m = 14 \) and \( m = 14 \) for Approximate Entropy Test, \( state = 1 \) for Random Excursions Test, \( state = -1 \) for Random Excursions Variant Test. We produce two \( p \)-values for the Serial Test and the Cumulative Sums Test.

First, we apply the randomness tests to the binary expansions of \( e, \pi, \sqrt{2} \) and \( \sqrt{3} \). For this purpose, we produce approximately \( 10^6 \) bits (7812 × 128 = 999936 bits) from
Table 3. Test results for the binary expansions of e, π, √2 and √3

<table>
<thead>
<tr>
<th></th>
<th>e</th>
<th>π</th>
<th>√2</th>
<th>√3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>0.953749</td>
<td>0.578211</td>
<td>0.811881</td>
<td>0.610051</td>
</tr>
<tr>
<td>Block Freq</td>
<td>0.211072</td>
<td>0.380615</td>
<td>0.833222</td>
<td>0.473961</td>
</tr>
<tr>
<td>Runs</td>
<td>0.561917</td>
<td>0.419268</td>
<td>0.313427</td>
<td>0.261123</td>
</tr>
<tr>
<td>Long Run of Ones</td>
<td>0.718945</td>
<td>0.243990</td>
<td>0.012117</td>
<td>0.446726</td>
</tr>
<tr>
<td>Bin Matrix Rank</td>
<td>0.306156</td>
<td>0.083553</td>
<td>0.823810</td>
<td>0.314498</td>
</tr>
<tr>
<td>Non-over Temp</td>
<td>0.078790</td>
<td>0.165737</td>
<td>0.569461</td>
<td>0.532235</td>
</tr>
<tr>
<td>Over Temp</td>
<td>0.110434</td>
<td>0.296897</td>
<td>0.791982</td>
<td>0.082716</td>
</tr>
<tr>
<td>Maurer Univ</td>
<td>0.282568</td>
<td>0.669012</td>
<td>0.130805</td>
<td>0.165981</td>
</tr>
<tr>
<td>Linear Comp</td>
<td>0.826335</td>
<td>0.255475</td>
<td>0.317127</td>
<td>0.346469</td>
</tr>
<tr>
<td>Serial Test 1</td>
<td>0.766182</td>
<td>0.143005</td>
<td>0.861925</td>
<td>0.157500</td>
</tr>
<tr>
<td>Serial Test 2</td>
<td>0.462921</td>
<td>0.034354</td>
<td>0.629225</td>
<td>0.171100</td>
</tr>
<tr>
<td>App Entropy</td>
<td>0.700073</td>
<td>0.361595</td>
<td>0.884740</td>
<td>0.180481</td>
</tr>
<tr>
<td>CuSum Forw</td>
<td>0.669886</td>
<td>0.628308</td>
<td>0.879009</td>
<td>0.917121</td>
</tr>
<tr>
<td>CuSum Back</td>
<td>0.724265</td>
<td>0.663369</td>
<td>0.957296</td>
<td>0.689519</td>
</tr>
<tr>
<td>Rand Excur</td>
<td>0.786868</td>
<td>0.844143</td>
<td>0.216235</td>
<td>0.783283</td>
</tr>
<tr>
<td>Rand Excur Var</td>
<td>0.826009</td>
<td>0.769066</td>
<td>0.566118</td>
<td>0.798247</td>
</tr>
<tr>
<td>0-bit Temp (0001)</td>
<td>0.766497</td>
<td>0.975645</td>
<td>0.383993</td>
<td>0.884000</td>
</tr>
<tr>
<td>1-bit Temp (0010)</td>
<td>0.903124</td>
<td>0.717759</td>
<td>0.898930</td>
<td>0.849536</td>
</tr>
<tr>
<td>2-bit Temp (0101)</td>
<td>0.631473</td>
<td>0.981607</td>
<td>0.508969</td>
<td>0.336139</td>
</tr>
<tr>
<td>3-bit Temp (1111)</td>
<td>0.907699</td>
<td>0.294869</td>
<td>0.839803</td>
<td>0.600553</td>
</tr>
</tbody>
</table>

the binary expansions of each number and apply the randomness tests. These four sequences are random according to all tests. The test results are presented at Table 3.

Second, we apply the randomness tests to four PRNGs. The random data are taken from the Mersenne Twister (Matsumoto et al. 1998), the Random and RNGCryptoServiceProvider classes of C#, and the outputs of AES (Daeman et al. 2002). The data from AES is produced using a fixed random key and low weight inputs. For both PRNGs $2^{17} \times 128 = 2^{24}$ bits are tested. Similar to the previous experiment, both generators pass all the statistical randomness tests. The test results are presented at Table 5.

Finally, in order to measure the power of the tests we produce biased nonrandom data and observe that which statistical randomness tests detect the bias. Using random source, we produce sequences of length $2^{24}$ that satisfy $Pr(a_i = 1) = \frac{1}{2} + q$, for each $i$, where $q$ is the bias. We then find that for which values of $q$ the tests detect the nonrandom behaviour of the generator. The results are presented in Table 5. We observe that three instances of our new randomness test can detect the nonrandom behaviour of the generator even for $q = 0.001$, where the template matching tests in NIST cannot detect that bias.

6 Conclusion

Random sequences are used widely in cryptographic applications and it is vital to use a proper random number generator to produce keys. Randomness testing is done by statistical randomness tests, and NIST test is suite the most popular suite for crypto-
Table 4. Test results for the four PRNGs

<table>
<thead>
<tr>
<th>Test Type</th>
<th>MerTwis</th>
<th>c# random</th>
<th>RNGCrypto</th>
<th>AES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>0.597616</td>
<td>0.906325</td>
<td>0.251393</td>
<td>0.156628</td>
</tr>
<tr>
<td>Block Freq</td>
<td>0.121173</td>
<td>0.958137</td>
<td>0.845587</td>
<td>0.482953</td>
</tr>
<tr>
<td>Runs</td>
<td>0.110322</td>
<td>0.686351</td>
<td>0.374532</td>
<td>0.348251</td>
</tr>
<tr>
<td>Long Run of Ones</td>
<td>0.191338</td>
<td>0.097943</td>
<td>0.213666</td>
<td>0.698123</td>
</tr>
<tr>
<td>Bin Matrix Rank</td>
<td>0.356745</td>
<td>0.269794</td>
<td>0.329875</td>
<td>0.698390</td>
</tr>
<tr>
<td>Non-over Temp</td>
<td>0.055189</td>
<td>0.804045</td>
<td>0.089171</td>
<td>0.276687</td>
</tr>
<tr>
<td>Over Temp</td>
<td>0.275223</td>
<td>0.024611</td>
<td>0.328769</td>
<td>0.249247</td>
</tr>
<tr>
<td>Maurer Univ</td>
<td>0.044998</td>
<td>0.854632</td>
<td>0.364266</td>
<td>0.423608</td>
</tr>
<tr>
<td>Linear Comp</td>
<td>0.693782</td>
<td>0.364427</td>
<td>0.105382</td>
<td>0.956454</td>
</tr>
<tr>
<td>Serial Test 1</td>
<td>0.147844</td>
<td>0.557261</td>
<td>0.126045</td>
<td>0.157290</td>
</tr>
<tr>
<td>Serial Test 2</td>
<td>0.382965</td>
<td>0.279386</td>
<td>0.129355</td>
<td>0.058659</td>
</tr>
<tr>
<td>App Entropy</td>
<td>0.252337</td>
<td>0.823391</td>
<td>0.037039</td>
<td>0.640333</td>
</tr>
<tr>
<td>CuSum Forw</td>
<td>0.715825</td>
<td>0.961562</td>
<td>0.764521</td>
<td>0.416899</td>
</tr>
<tr>
<td>CuSum Back</td>
<td>0.302646</td>
<td>0.418822</td>
<td>0.809574</td>
<td>0.241198</td>
</tr>
<tr>
<td>Rand Excur</td>
<td>0.589143</td>
<td>0.279374</td>
<td>0.421991</td>
<td>0.152072</td>
</tr>
<tr>
<td>Rand Excur Var</td>
<td>0.497518</td>
<td>0.483294</td>
<td>0.980570</td>
<td>0.450652</td>
</tr>
<tr>
<td>0-bit Temp (0001)</td>
<td>0.250408</td>
<td>0.762958</td>
<td>0.369484</td>
<td>0.804434</td>
</tr>
<tr>
<td>1-bit Temp (0010)</td>
<td>0.724940</td>
<td>0.960639</td>
<td>0.129355</td>
<td>0.058659</td>
</tr>
<tr>
<td>2-bit Temp (0101)</td>
<td>0.930115</td>
<td>0.194722</td>
<td>0.380861</td>
<td>0.865790</td>
</tr>
<tr>
<td>3-bit Temp (1111)</td>
<td>0.930115</td>
<td>0.194722</td>
<td>0.380861</td>
<td>0.865790</td>
</tr>
</tbody>
</table>

Table 5. Test results for the biased non-random data

<table>
<thead>
<tr>
<th>Test Type</th>
<th>0</th>
<th>0.001</th>
<th>0.002</th>
<th>0.003</th>
<th>0.004</th>
<th>0.005</th>
<th>0.01</th>
<th>0.02</th>
<th>0.03</th>
<th>0.04</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Block Freq</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Runs</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Long Run of Ones</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>Bin Matrix Rank</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>Non-over Temp</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Over Temp</td>
<td>Ran</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Maurer Univ</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>Linear Comp</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>Serial Test 1</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>Serial Test 2</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>App Entropy</td>
<td>Ran</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>CuSum Forw</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>CuSum Back</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Rand Excur</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>Rand Excur Var</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>0-bit Temp (0001)</td>
<td>Ran</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
<td>Non</td>
</tr>
<tr>
<td>1-bit Temp (0010)</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>2-bit Temp (0101)</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
<tr>
<td>3-bit Temp (1111)</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
<td>Ran</td>
</tr>
</tbody>
</table>

graphic applications. One of the randomness tests in this suite is the overlapping template matching test.
In this work, we classify all templates according to their number of overlapping bits and show that the probabilities used in NIST’s overlapping template matching test is valid only for $B = 111111111$ and should be recalculated for different overlapping blocks. Moreover, we find the exact distributions for all 4-bit templates and propose new randomness tests, namely 4-bit template matching tests.

We apply the proposed tests to biased random data and observe that the new tests detect the nonrandom behaviour of the generator even for $q = 0.001$, where the template matching tests in NIST cannot detect that bias. Moreover, NIST’s overlapping template matching test can only be applied to long sequences, that is the sequences of minimum length $10^6$, where the new proposed tests can be applied to any sequence whose length is bigger than 5504. Furthermore, for the new tests, it is also possible to change the subsequence length by calculating the bin probabilities for the new subsequence length.

As a future work, the exact distributions can be obtained for all 5-bit templates. Also, the probabilities for the other overlapping templates in NIST’s overlapping template matching test can be calculated.

References


