A RANDOMNESS TEST BASED ON POSTULATE R-2 ON THE NUMBER OF RUNS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

OKAN ŞEKER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CRYPTOGRAPHY

AUGUST 2014

Approval of the thesis:

A RANDOMNESS TEST BASED ON POSTULATE R-2 ON THE NUMBER OF RUNS

submitted by **OKAN ŞEKER** in partial fulfillment of the requirements for the degree of **Master of Science in Department of Cryptography, Middle East Technical University** by,

Prof. Dr. Bülent Karasözen
Director, Graduate School of **Applied Mathematics** ______________

Prof. Dr. Ferruh Özbudak
Head of Department, **Cryptography** ______________

Assoc. Prof. Dr. Ali Doğanaksoy
Supervisor, **Department of Mathematics, METU** ______________

Examining Committee Members:

Dr. Muhiddin Uğuz
Department of Mathematics, METU ______________

Assoc. Prof. Dr. Ali Doğanaksoy
Department of Mathematics, METU ______________

Assist. Prof. Dr. Çetin Ürtiş
Department of Mathematics, TOBB ETU ______________

Assist. Prof. Dr. Fatih Sulak
Department of Mathematics, Atılım University ______________

Dr. Cihangir Tezcan
Department of Mathematics, METU ______________

Date: ______________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: OKAN ŞEKER

Signature :

ABSTRACT

A RANDOMNESS TEST BASED ON POSTULATE R-2 ON THE NUMBER OF RUNS

Şeker, Okan
M.S., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Ali Doğanaksoy

August 2014, 39 pages

Random values are considered as an indispensable part of cryptography, since they are necessary for almost all cryptographic protocols. Most importantly, key generation is done by random values and key itself should behave like a random value. Randomness is tested by statistical tests and hence, security evaluation of a cryptographic algorithm deeply depends on statistical randomness tests.

In this thesis we focus on randomness postulates of Solomon W. Golomb in particular, second postulate which is about runs of a sequence and their distributions. The distributions of runs of length one, two and three are underlined. And by these distributions we state three new statistical randomness tests. New tests use χ^2 distribution therefore, exact probabilities are needed. We calculate the probabilities in a combinatorial approach. In order to using in the tests, probabilities are divided into five intervals, which are called as subintervals. Subinverval are selected in such a manner that each interval have nearly equal probabilities. Finally, three new statistical tests are defined and pseudocodes for new statistical tests are given.

New statistical tests are designed to detect deviations of number of different length from a random sequence. Since other tests are not interested in runs of different length, they cannot be detected this deviation. The tests are implemented with some other statistical tests, on some well-known algorithms and binary expansion of irrational numbers. Experiment results show the performance and sensitivity of our tests.

Keywords : Statistical Randomness Tests, Golomb's Randomness Postulates, Run Tests.

ÖZ

ÖBEK SAYILARI HAKKINDAKİ R-2 POSTÜLASINA DAYALI BİR RASTGELELİK TESTİ

Şeker, Okan
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Ali Doğanaksoy

Ağustos 2013, 39 sayfa

Rastgele değerler neredeyse bütün kriptografik protokollerde ihtiyaç¸ duyulduğu için kriptografinin ayrılmaz bir parçası olarak görülmektedir. En önemlisi, anahtar üretimi rastgele değerler ile sağlanmaktadır ve anahtarın kendisi de bir rastgele değer gibi davranmalıdır. Rastgelelik testleri istatistiksel testleri ile yapılmaktadır, bu yüzden kriptografik algoritmaların güvenliği derin bir şekilde istatistiksel rastgelelik testlerine bağlıdır. Bu tezde Solomon W. Golomb tarafından tanımlanan rasgelelik postülalarına odaklanılmıştır, özellikle öbek sayıları ve dağılımları üzerine olan ikinci postülası üzerine çalışılmıştır. Postülada geçen öbek terimi, bir seride geçen kesintisiz ve aynı bitlerden oluşan en uzun alt-seriler olarak tanımlanmıştır. Birlik, ikilik ve üçlük öbek sayılarının dağılımı vurgulanmış ve bu dağılımlar ile yeni istatistiksel rastgelelik testlerini tanımlanmıştır. Yeni testler chi-kare dağılımını kullandıkları için, gerçek olasılıklara ihtiyaç duyulmuştur. Bu olasılıkları kombinatorik bir yaklaşım ile hesaplanmıştır. Testlerde kullanmak üzere bu olasılıkları alt-aralık denilen beş aralığa bölünmüştür. Alt-aralıklar neredeyse eşit olasılıklara sahip olacak şekilde seçilmiştir. Son olarak da testler tanımlanmış¸ ve kodları verilmiştir. Yeni tanımlanan testler farklı uzunluktaki öbek sayılarının dağılımındaki sapmaları ortaya çıkarmak üzere tasarlanmıştır. Diğer testler farklı uzunluktaki öbek sayıları ile ilgilenmedikleri için bu sapma belirlenemememektedir. Yeni testler bazı tanınmış algoritmalar ve irrasyonel sayıların ikili açılımları üzerinde uygulanmıştır. Bu deneyler testleri performansını ve hassasiyetini göstermiştir

Anahtar Kelimeler : İstatistiksel Rassalık Testleri, Golomb'un Rassallık Postülaları, Run Testleri.

To My Family

ACKNOWLEDGMENTS

First of all, I would like to express my great thanks to my supervisor Assoc. Prof. Ali Doğanaksoy. His excellent mentoring, guidance and motivation make this work finish.

I deeply thank to Assist. Prof. Dr. Fatih Sulak who patiently guide me and encourage me throughout the research with his expertise and to Dr. Muhiddin Uguz for his for his ideas, valuable comments from the beginning. Special thanks should be given to Ziya Akcengiz for his support, collaboration and friendship.

I am grateful to my colleges in TÜBITAK-UEKAE and in Institute of Applied Mathematics. They have been exquisite friends and supporters.

Finally, I would like to thank to my family for their endless support and love. And I thank to my sister for being an excellent mentor and friend through my entire life. Last but not the least, I wish to thanks all my friends with whom I have worked on stage and my dearest friends for their wonderful friendship.

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

LIST OF ABBREVIATIONS

TRNG	True Random Number Generator
PRNG	Pseudorandom Number Generator
H_0	Null Hypothesis
NIST	National Institute of Standards and Technology
AES	Advanced Encryption Standard
r_t	Number of Runs
r_1	Number of Runs of Length One
r_2	Number of Runs of Length Two
r_3	Number of Runs of Length Three

CHAPTER 1

INTRODUCTION

1.1 Random Sequences and Random Numbers

Random numbers and random sequences are extensively used in many areas such as, *game theory, numerical analysis, quantum mechanics, cryptography* etc. They constitute a necessary part of cryptography. Need for random sequences emerges in *challenge and response authentication systems, generation of digital signatures, zero-knowledge protocols* etc. But, most important feature is; key generation is highly depend on random values. As stated in [29], if each term of a sequence takes any value from a finite set with equal probability, then the sequence is said to be random. The sequence is called binary random sequence if the set is chosen as $\{0,1\}$. Another definition stated by Kolmogorov [17] is about the length of the shortest description of a sequence. Accordingly, if a sequence is random then the shortest description should be the sequence itself. To prevent any weaknesses in the sequences, three basic properties of a random bit sequence are stated in [27] as follows:

1. **Unpredictability:** Let $S = s_1, s_2, \ldots, s_n, s_{n+1}, \ldots$ be a random bit sequence then, knowing first t element of a sequence should not give any information about s_{t+1}, the next element of the sequence S. All elements of the sequence should be generated independently.
2. **Uniformity:** Given any subsequence of S, there should be nearly equal number of 1's and 0's.
3. **Independence:** Each element of S is independent from other elements, That is, $Pr(s_i{=}s_j) = 1/2$ for any $i \neq j$.

Random sequences should have some additional properties also. First of all, they should be reproducible in order to use in an algorithm. Moreover, they should not need too much time and power consumption and they should have large period. Such sequences can be generated by two type of sources.

First one is called true random generator (TRNG). TRNG is a combination of a non-deterministic source and some processing function which is required to overcome any weaknesses in the source. Physical sources like *thermal noise, sound samples from an* *environment, radioactive decay etc.* are generally used as a non-deterministic source. As it is needed, outputs of these sources have good statistical properties. They are unpredictable and not periodic. However, using these sources in algorithms causes various problems. For example, production of random numbers can become time and power consuming. Since, reproducing the outputs of these sources is nearly impossible, large number of bits should be transmitted. Therefore, this transmission makes the systems unpractical.

As a consequence of these problems another type of random number generators are used in cryptosystems, which are called pseudorandom number generators (PNRG). A PRNG is used one or multiple inputs or seeds to generate pseudorandom sequences, that is *random looking* sequences of a specific length which are produced by deterministic processes [21].

The seeds should be also unpredictable and random, therefore PNRG's are designed to work with a TRNG. Three important features of PNRG's are; they don't need too much time and power compared to TRNG's, they are reproducible and have large period. Moreover, if an PRNG is designed properly, each value, which is produced by previous values via transformations, introduce additional randomness. That is, a series of such transformations can eliminate undesired statistical properties. Hence PRNG's can have better statistical properties compared to TRNG's.

1.2 Randomness

As stated in Section 1.1 need for random sequences arises in many different areas in cryptography. Therefore, randomness is a highly important issue for cryptographic systems. According to the Kerckhoff's principle [14], security of the cryptographic systems should depend only on the key, so that security of the system depends on random values which are used in the algorithm. Also, using weak random values in key generations can cause a leakage in the system and hence an adversary can gain ability to break the whole cryptosystem.

In all applications, used values should be sufficient size and be *random*, in such a manner that probability of any chosen quantity should be small enough to eliminate an adversary to gain any specific information. Thus, sequences and numbers, which are used as a key in cryptographic algorithms such as, block ciphers and synchronous stream ciphers should be pseudorandom. These sequences should have good statistical properties and they are usually generated by a PRNG, defined in Section 1.1

Another place where *random looking* plays an important role is that, outputs of algorithms should be pseudorandom and hence, should be indistinguishable from random mappings. That is, an algorithm's output should not give any information about input of the system or the key itself. Therefore, round number of a block cipher is decided according to concept of random looking. Hence, security of the system depends on testing of pseudorandom sequences. For these reasons, testing pseudorandom sequences is an important topic and it is done by *statistical randomness tests*, which is considered as an important part of evaluating the security of cryptographic algorithms.

1.3 Testing

Statistical tests are defined to detect weaknesses that a sequence could have. For a finite sequence there is no definition of randomness. Therefore, statistical tests are defined to detect weaknesses that a sequence could have.

For each statistical test a random variable, whose distribution function is known, is chosen, such as number of ones or number of uninterrupted sequence of identical bits. A *critical value* which corresponds to *far out* in the tails of distribution, is decided according to the distribution. Depending on the distribution and random variable a real number between zero and one, called *p-value*, is calculated. If the *p-value* for a test is calculated as one, then this result indicates that the sequence is completely random. On the other hand, the sequence is completely non-random, if *p-value* is determined as zero.

1.4 Test Suites

Numerous statistical tests can be implemented to a sequence to compare with a truly random sequence. A large number of statistical tests can be described in such a manner that each of them evaluates a different characteristic of the sequence. Therefore, there is no special set of tests assumed to be *complete*. For different purposes, different finite set of test can be arranged. These sets are called as test suites. In the literature, there exists various statistical test suites. The most important test suites are the suite given in the Knuth's book [16], test suite presented by Rukhin [24], DIEHARD [19], CRYPT-X [6], TestU01 [18] and the test suite published by NIST [3] so far.

- Knuth's Test suite is stated in Donald Knuth's book *The Art of Computer Programming, Volume 2* [16] in 1969. It is one of the most quoted reference for the statistical randomness testing. The reason is that, it contains most of the standard and required tests such as, frequency test, runs test, serial test etc.
- DIEHARD Test Suite was developed by Marsaglia in 1995 on a CDROM. 15 statistical tests were included in the suite. In most of the tests, sequence sizes are fixed, therefore it is not suitable to test sequences with different lengths. In 2010 Alani [1] uses the suite to evaluate the AES finalists.
- NIST(National Institute of Standards and Technology) test suite consist of firstly 16 then 15 various statistical tests. The suite is used as an evaluation tool for the Advanced Encryption Standard (AES) by Soto and Bassham [28]. Since its first publication some revisions are made. In 2004 test setting of discrete fourier transform test and lempel-ziv test are found wrong [15] and new test, which can be used instead of lempel-ziv test, is defined in [10] and correction of overlapping template matching is stated in 2007 [9].
- Crypt-X suite was developed in the Information Security Research Centre at Queensland University of Technology in 1992. It consists of 7 different tests

which are; frequency, binary derivative, change point, runs, sequence complexity and linear complexity tests.

- *TestU01* is developed in 2007 and it consists of two parts and each parts has 2 sub parts.

    1. Tests for a sequence of real numbers in (0,1)
        - Tests on a single stream of n numbers.
        - Tests based on n subsequences of length t:
    2. Tests for a sequence of random bits
        - One long binary stream of length n.
        - Tests based on n bit strings of length m.

Besides the suites there are works, focusing on statistical tests individually. Such as, a universal statistical test, stated by Maurer [20], a test based on diffusion characteristic of a block cipher [13], topological binary test defined by Alcover et.al. [2], a new randomness test based on SAC [11], work done by Ryabko [25] in 2004

1.5 Golomb's Randomness Postulates

As it is stated, there are no complete set of tests to be assumed complete. Apart from the statistical tests, there is works on deciding pseudorandomness, which is a difficult task. Golomb's postulates constitute a base for randomness tests. This work is considered as one of the most important attempts to create some necessary properties for a finite (or periodic) pseudorandom sequence to be random looking. Sequences satisfying following three properties are called *pseudo-noise sequence* [8].

Let $S = s_0, s_1, \ldots, s_{n-1}, \ldots$ be an infinite binary sequence periodic with n (or a finite sequence of length n). A *run* is defined as an uninterrupted maximal sequence of identical bits. Runs of 0's are called *gap*, runs of 1's are called *block*. R1, R2 and R3 are Golomb's randomness postulates which are given as follow:

R1) In a period of S, the number of 1's should differ from the number of 0's by at most 1. In other words, sequence should be *balanced*.

R2) In a period of S, at least half of the all runs of 0's or 1's should have length one, at least one-fourth should have length 2, at least one-eight should have length 3, and the like. Moreover, for each of these lengths, there should be (almost) equally many *gaps* and *blocks*.

R3) The auto correlation function $C(t)$ should be two valued. That is for some integer K and for all $t = 0, 1, 2, \ldots, n-1$

$$C(t) = \sum_{i=0}^{n-1}(-1)^{s_i+s_{i+t}} = \begin{cases} n \text{ if } \ t = 0 \\ K \text{ if } \ 1 \leq t \leq n-1 \end{cases} .$$

The first postulate states that in an $n - bit$ sequence, the difference of number of ones and zeros should be 1 or 0. In other words, the number of ones in a sequence, that is weight of the sequence, should be approximately $n/2$. Frequency test, which measures the difference of number of ones and zeros in an $n - bit$ sequence, is defined to check this the first postulate of Golomb. Since being balanced for an algorithm's output is very important in cryptography, frequency test is used as initial step for almost all test suites. If an algorithm fails the frequency test, then other tests are not even applied.

The second postulate of Golomb is about number of runs in sequences. Tests, which deal with number of runs, are called as *run tests* and these are also included in many test suites as frequency test. However, in most of test suites, they consider only the total number of runs in the sequence and do not concern about the number of runs of different length. The main reason, for this is calculating the expected number of runs of specified length in a random sequence is a difficult task especially when specified length becomes large.

Third postulate gives information about amount of similarities between the sequence and shifted version of it. If S is a random sequence, the autocorrellation should be constant, that is; correlation between i^{th} and $(i + t)^{th}$ bits should give no information about the sequence for $t = 1, 2, \ldots, (n - 1)$.

1.6 Motivation

Apart from the statistical tests, which are used in test suites given in Section 1.4, there are various statistical tests. In this thesis, we focus on the second postulate of Golomb and define a new test based on this postulate. Our main motivation is to decrease the number of tests in the suites nby using fundamental postulates and hence, design more efficient test suites with less statistical tests.

Our secondary goal is to state statistical tests in order to test short sequences like outputs of block ciphers and hash functions. This problem is actually stated in the NIST test suite. In the test suite, it is assumed that sequence length n is of order 10^3 to 10^7. For this reason, asymptotic reference distributions were derived and used for tests. But, asymptotic reference distributions are misleading for small values of n, As stated in [3] "*the asymptotic reference distributions would be inappropriate and would need to be replaced by exact distributions that would commonly be difficult to compute*". In other words, asymptotic reference distributions can lead some errors while testing the outputs of block ciphers or hash functions. In 1999, to overcome this problem, Soto [28] proposed to concatenate short sequences. This method is used for testing the randomness of Advanced Encryption Standard candidates. Another method has been proposed by Sulak et.al. [30], in which distribution functions used in NIST test suite replaced by exact distribution and a similar method is used for producing the *p-values*. We use the methods stated in [30] and [29].

Outline of the thesis is as follows:

In Chapter 2, we give the proofs of our fundamental theorems. Also probabilities associated to runs of length one, two and three are stated and the probabilities are divided into five intervals, which are called as subintervals, in such a way that each interval has nearly equal probabilities. In Chapter 3, three new statistical tests are defined according to the subinterval probabilities and pseudocodes for new statistical tests are given. In Section 4, we apply our tests to five finalists of advanced encryption standard competition. After that we implement the tests on binary expansion of e, π and $\sqrt{2}$, which are obtained from NIST package [3]. In the last part of implementation we generate some non-random data sets to emphasize the sensitivity of our tests. Finally In Section 5, we summarize our results and state the topics for further research.

CHAPTER 2

RUN TESTS

2.1 Brief Overview of Run Tests

In Chapter 2 we propose three new statistical randomness tests and a variation of run test which is stated in [30] . First, we give a brief information about run tests in the literature. Then the proofs of our fundamental theorems and calculation of probabilities, which are needed to introduce our approach to the run tests, are given. Using these calculations, we state our new run tests and give the pseudocodes.

Run tests depend on the Golomb's second postulate and they investigate number of runs in a sequence and their distribution. Run tests take place in most of the test suites. Almost in all of these suites, run tests concern only about the total number of runs in the sequences. The most important ones of these are the suite given in [16], [19], [18] and [3].

Knuth and DIEHARD test suite defines the run test on random numbers. They define runs as *runs up* and *runs down* in a sequence. To illustrate their definition; consider a sequence of length 10, $S_n = 138742975349$; putting a vertical line between s_j's when $s_j > s_{j+1}$, this runs of the sequence 138742975349 can be seen as |138|7|4|29|7|5|349|.In other words, the run test examines the length of monotone subsequences.

TestU01 defines *run and gap tests* for testing the randomness of long binary stream of length n. This test collects runs of 1's and 0's until the total number of runs is $2r$. Then for each length $j = 1, 2, \ldots, k$ it counts the number of runs of 1's and 0's of length in this collection and and records this $2k$ counts. Then it applies χ^2 test on these counts. *Longest run of 1's test* is also defined for the collection of strings of length m which are obtained from the original long binary string of length n.

In the NIST test suite, 2 of 15 tests are variations of run tests. They are called as *run test* and *longest run of ones in a block test*. First one deals with the total number of runs in a sequence. It calculates the total number of runs in a sequence and determine whether it is consistent with the expected number of runs, which is supposed to be close to $n/2$ in a sequence or not. Second one determines whether the longest run of ones in the sequence is consistent with the length of longest runs of ones which is in a random sequence. In NIST test suite the reference distribution for the run tests is a χ^2 distribution.

In this thesis we use the approach stated in [30], thus we need the exact probabilities and exact distribution of tests statics. Finding the number of sequences having a selected number of runs of length i is a hard problem. We find the number with the help of combinatorial formulas. After that we calculate the desired probabilities by just dividing the number by total number of sequences. Calculating the exact probabilities of the number of runs of length i in a random sequence enables us to investigate the number of the runs of same length in an random sequence. We calculate for number of runs of length 1, 2 and 3 and we give the detailed information in the following section. However as the length grows, calculations are getting complex and time required for these calculations grows exponentially. Therefore, tests involving number of runs of length j ($j > 3$) unpractical for statistical test suites.

2.2 Computation of Probabilities

In this section, we give detailed information about the calculations for finding the number of sequences, having a number of runs of lengths 1, 2 or 3 and hence, we state exact probabilities. Probabilities depend on the number of existing shorter runs. That is; probabilities for the number of runs of length two depend on both *total number of runs* and *number of runs of length one* , similarly number of runs of length three depends on *total number of runs* and *number of runs of length one and two* and so on. These probabilities are not directly used in the test, since they include some dependence with number of runs etc. Therefore, after stating the theorems we give the algorithm to find the exact probabilities which we need for describing the tests.

In the calculations of probabilities we frequently use the following combinatorial formula.

Remark 2.1 ([23] Number of Non-Negative Integer Solutions of a Linear Equation). The number of non-negative integer solutions of $x_1 + x_2 + \ldots + x_r = n$, $n \in \mathbb{Z}^+$ is $\binom{n+r-1}{r-1}$.

Remark 2.2. The number of positive integer solutions of $x_1 + x_2 + \ldots + x_r = n$, $n \in \mathbb{Z}^+$ is $\binom{n-1}{r-1}$.

Proof. With the substitution $x_i = x'_i + 1$ we get,

$$\begin{aligned} (x'_1 + 1) + (x'_2 + 1) + \ldots + (x'_r + 1) &= n \\ x'_1 + x'_2 + \ldots + x'_r &= n - r. \end{aligned}$$

From Remark 2.1 it follows that the number of solutions is:

$$\binom{(n-r) + (r) - 1}{r-1} = \binom{n-1}{r-1}.$$

□

2.2.1 Number of Runs

In the rest of the thesis we denote the total number of runs, number of runs of length one, two and three as r_t, r_1, r_2 and r_3 respectively and we use samples of these variables r, l_1, l_2, l_3 respectively. We denote by $Pr(r_t = r)$ the probability of randomly chosen binary sequence with r runs. In the same way, $Pr(r_i = l_i)$ is the probability of randomly chosen binary sequence with l_i runs of length i. Also we use subscripts $S_1, S_2, \ldots, S_m$ to differentiate the blocks of a long sequence or outputs of block ciphers and hash functions.

Also to illustrate runs of a sequence we use the equation $x_1 + x_2 + \ldots + x_r = n$ for a sequence with length n and having r runs. An important property of this illustration is that; it gives no information about content of x_i's, that is; x_i $(i = 1, 2 \ldots, r)$ can be a run of 0's or 1's. Thus, each positive integer solution of the equation $x_1 + x_2 + \ldots + x_r = n$ corresponds to two sequences, one starting with 1, the other start with 0. Hence, the number of sequences with length n and having exactly r runs is $2\binom{n-1}{r-1}$.

Example 2.1. Let S=01100010011111001100011101010000 be a binary sequence length of 32, having 15 runs. Then;

$$x_1 + x_2 + \ldots + x_{15} = 32$$
$$\underbrace{0}_{x_1}\underbrace{11}_{x_2}\underbrace{000}_{x_3}\underbrace{1}_{x_4}\underbrace{00}_{x_5}\underbrace{11111}_{x_6}\underbrace{00}_{x_7}\underbrace{11}_{x_8}\underbrace{000}_{x_9}\underbrace{111}_{x_{10}}\underbrace{0}_{x_{11}}\underbrace{1}_{x_{12}}\underbrace{0}_{x_{13}}\underbrace{1}_{x_{14}}\underbrace{0000}_{x_{15}}.$$

$$x_1 = 1,\, x_2 = 2,\, x_3 = 3,\, x_4 = 1,\, x_5 = 2,$$
$$x_6 = 5,\, x_7 = 2,\, x_8 = 2,\, x_9 = 3,\, x_{10} = 3,$$
$$x_{11} = 1,\, x_{12} = 1,\, x_{13} = 1,\, x_{14} = 1,\, x_{15} = 4.$$

Probabilities are calculated in a similar way as in [30]. The main difference is that, in the previous approach, sequences are viewed in a circular form. Probabilities depend on weight of the sequence and parity of number of runs. We calculate the probabilities with the above notation, which is not based on circular form and they depend on the number of runs and number of shorter runs.

Theorem 2.1. *Let $S = s_1, s_2, \ldots, s_n$ be a binary sequence of length n having total a of r runs then;*

$$Pr(r_t = r) = \frac{\binom{n-1}{r-1}}{2^{n-1}}.$$

Proof. We can illustrate the sequence of length n, having r runs as follows;

$$x_1 + x_2 + \ldots + x_r = n.$$

The number of all binary sequences $S = s_1, s_2, \ldots, s_n$ of length n, having total number of r runs is $2\binom{n-1}{r-1}$ which is the result of Remark 2.2. Hence probability of a randomly chosen such sequence to have exactly r runs is;

$$Pr(R = r) = \frac{2 \cdot \binom{n-1}{r-1}}{2^n}.$$

□

After finding the exact probabilities we calculate the subinterval probabilities. Following example shows the calculations of subinterval probabilities for 128 bit sequences.

Example 2.2. *Calculating the Subinterval Probabilties.*

- **Step 1:** Calculate $Pr(r_t = r)$ for $l_1 = 1, 2, \ldots, 128$ by using Theorem 2.1.
- **Step 2:** Determine subintervals such that; $(\alpha_0, \alpha_1), (\alpha_1, \alpha_2), \ldots, (\alpha_4, \alpha_5)$ such that, $Pr_i(\alpha_i < R < \alpha_{i+1}) \approx 0,2$. In our example subinterval probability can be calculated as follows;

$$\text{Box1} = \sum_{r=0}^{58} Pr_1(r_t = r) \quad \text{Box2} = \sum_{r=59}^{61} Pr_2(r_t = r)$$

$$\text{Box3} = \sum_{r=62}^{64} Pr_3(r_t = r) \quad \text{Box4} = \sum_{r=65}^{67} Pr_4(r_t = r)$$

$$\text{Box5} = \sum_{r=68}^{128} Pr_5(r_t = r)$$

- **Step 3:** Finally, we get the Table 2.1 for subinterval probabilities;

In the same way we calculate the subinterval probabilities for different block lengths. They can be seen in Table 2.2.1

Table 2.1: Subinterval probabilities for 128 bit sequences.

	Intervals	Probability
Box 1	0-58	0.187478
Box 2	59-61	0.173915
Box 3	62-64	0.208992
Box 4	65-67	0.190652
Box 5	68-128	0.238960

Table 2.2: Interval and probability values for number of runs for 64, 128, 256, 512 bits blocks.

	n=64		n=128		n=256		n=512	
	interval	prob	interval	prob	interval	prob	interval	prob
Box 1	1-28	0.224981	1-58	0.187478	1-120	0.190339	1-246	0.212950
Box 2	29-30	0.175671	59-61	0.173915	121-125	0.210794	247-253	0.216834
Box 3	31-32	0.198693	62-64	0.208992	126-129	0.197731	254-259	0.208488
Box 4	33-34	0.175671	65-67	0.190652	130-134	0.210794	260-265	0.173571
Box 5	35-64	0.224981	68-128	0.238960	135-256	0.190339	266-512	0.188154

2.2.2 Number of Runs of Length One

In this section, probabilities for an $n - bit$ sequence having l_1 runs of length one are given in a combinatorial approach. We use the same notation and similar ideas of Section 2.2.1 to compute the number of sequences having total of r runs, l_1 of which are of length one and hence we calculate the probabilities. As a result of this calculations, we state the first new run test, which depends on the idea in Golomb's second postulate.

Theorem 2.2. *The probability of randomly chosen binary sequence $S = s_1, s_2 \ldots, s_n$ with length n, to have a total of r runs, l_1 of which are runs of length one is,*

$$Pr(r_t = r, r_1 = l_1) = \frac{\binom{n-r-1}{r-l_1-1} \cdot \binom{n}{l_1}}{2^{n-1}}.$$

Proof. As in the proof of the Theorem 2.1, we illustrate the sequence as follows;

$$x_1 + x_2 + \ldots + x_r = n \tag{2.1}$$

Let's first assume that the last l_1 runs are the runs of length one and the rest are of length at least two. That is,

$$x_{r-l_1+1} = \ldots = x_{r-1} = x_r = 1.$$

$$\begin{aligned} x_1 + x_2 + \ldots + x_{r-l_1} + \overbrace{1 + 1 \ldots + 1}^{l_1} &= n \\ x_1 + x_2 + \ldots + x_{r-l_1} &= n - l_1. \end{aligned}$$

Notice that here, $x_i \geq 2$ so, we use the change of variable $y_i = x_i - 2$ for $i = 1, 2, \ldots r - l_1$.

$$\begin{aligned} (x_1 - 2) + (x_2 - 2) + \ldots + (x_{r-l_1} - 2) &= n - l_1 - 2(r - l_1) \\ y_1 + y_2 + \ldots + y_{r-l_1} &= n - 2r + l_1. \end{aligned} \tag{2.2}$$

The number of sequences having conditions, which are stated above, is equal to the number of non-negative solutions of the equation 2.2. Consequently, by the Remark 2.1, number of desired solutions is,

$$\binom{n-r-1}{r-l_1-1}.$$

Selection of l_1 runs of length 1 give us a factor of $\binom{r}{l_1}$. Since, each positive integer solution of the equation 2.1 corresponds to two sequences. First one starts with 1 and the other starts with 0. Therefore we multiply the number of solutions by 2. Therefore, the number of all binary sequences of length n, having total number of r runs, l_1 of which of length one is equal to $2\binom{n-r-1}{r-l_1-1}\binom{r}{l_1}$. Hence probability of a randomly chosen such sequence to have exactly r runs, l_1 of which, is length one is:

$$Pr(r_t = r, r_1 = l_1) = \frac{2 \cdot \binom{n-r-1}{r-l_1-1} \cdot \binom{r}{l_1}}{2^n}. \tag{2.3}$$

□

Number of sequences having r runs, l_1 of which are length one can be found using formula above. Our aim is to compute total number of sequences of length n having l_1 runs of length one without depends on the total number of runs. In order to compute the aimed number we use Corollary 2.3.

Corollary 2.3. *Let $N_1(l_1)$ denote the number of sequences with exactly l_1 runs of length one. Then,*

$$N_1(l_1) = \sum_{r=1}^{n} 2 \cdot \binom{n-r-1}{r-l_1-1} \cdot \binom{r}{l_1}. \tag{2.4}$$

Since the number of all sequences of length n is 2^n, probabilities follows immediately;

$$Pr(r_1 = l_1) = \frac{N_1(l_1)}{2^n}.$$

Moreover, Algorithm 1 enable the calculation for $N_1(l_1)$. So that we can investigate number of length one independently.

After finding the $N_1(l_1)$ for $l_1 = 0, 1, 2, \ldots, n$ by using Corollary 2.3 and Algorithm 1 we calculate the subinterval probabilities in the same way that is showed in example 2.2. They can be seen in Table 2.2.2

Example 2.3. Let S_n be a random sequence of length 8, having 4 runs and 2 runs of length one.
Since, we have exactly 4 runs, x_i's must be at least 1;

Algorithm 1 Calculating $Pr(r_1 = l_1)$ for $l_1 = 0, 1 \ldots, n$

```
$l_1 \leftarrow 1$ , $r \leftarrow 1$, $N_1(l_1) \leftarrow 0$,
while $l_1 \leq n$ do
    while $r \leq n$ do
        $Pr(r_1 = l_1) \leftarrow Pr(r_1 = l_1) + \binom{n-r-1}{r-l_1-1}\binom{r}{l_1}/2^{n-1}$
        $r \leftarrow r + 1$
    end while
    $l_1 \leftarrow l_1 + 1$
end while
return $N_1$
```

Table 2.3: Interval and probability values for runs of length one for 64, 128, 256, 512 bits blocks.

	n=64		n=128		n=256		n=512	
	interval	prob	interval	prob	interval	prob	interval	prob
Box 1	0-13	0.190082	0-27	0.173171	0-56	0.187255	0-117	0.193566
Box 2	14-16	0.238877	28-31	0.21426	57-61	0.189280	118-125	0.218630
Box 3	17-18	0.174560	32-34	0.186977	62-66	0.219859	126-132	0.217076
Box 4	19-21	0.211470	35-38	0.21339	67-72	0.218775	133-140	0.199515
Box 5	22-64	0.185009	39-128	0.21219	73-256	0.184827	141-512	0.171211

$$x_1 + x_2 + x_3 + x_4 = 8, \;\; x_i \geq 1 \text{ for } i = 1, 2, 3, 4.$$

Fix $x_3 = x_4 = 1$ then;

$$x_1 + x_2 = 6 \; x_i \geq 2 \text{ for } i = 1, 2$$

We want $x_i \geq 2$. Define $x_i' = x_i + 2$ for i=1,2.

$$x_1 + x_2 = 6, \;\; x_i \geq 2 \text{ for } i = 1, 2$$

$$(x_1 + 1) + (x_2 + 1) = 2$$

$$x_1' + x_2' = 2, \;\; x_i' \geq 0 \text{ for } i = 1, 2$$

$$x_1' = 2, x_2' = 0 \Leftrightarrow x_1 = 4, x_2 = 2, x_3 = 1, x_4 = 1 \begin{cases} 11110010 \\ 00001101 \end{cases}$$
$$x_1' = 1, x_2' = 1 \Leftrightarrow x_1 = 3, x_2 = 3, x_3 = 1, x_4 = 1 \begin{cases} 11100010 \\ 00011101 \end{cases}$$
$$x_1' = 0, x_2' = 2 \Leftrightarrow x_1 = 2, x_2 = 4, x_3 = 1, x_4 = 1 \begin{cases} 11000010 \\ 00111101 \end{cases}$$

The above construction gives us 6 different sequences of length 8 with 2 runs of length one. Also selecting x_3 and x_4 gives us a factor of $\binom{4}{2}$. Hence, the total number of sequences of length 8 with 4 runs, 2 of which are of length one is $2 \cdot \binom{8-4-1}{4-2-1} \cdot \binom{4}{2} =$ 36.

2.2.3 Number of Runs of Length Two

In this section, we calculate the number of sequences having l_1 runs of length one in a combinatorial approach. As in the previous section we use the same notation and the similar ideas in Section 3.1 to compute the number of sequences having total of r runs, l_2 of which are of length two and hence we calculate the probabilities. Then using these calculations, we state the second new run test.

Theorem 2.4. *The probability of randomly chosen binary sequence $S = s_1, \ldots, s_n$ with length n, having r runs, l_1 of which are length one and l_2 of which are length and two is ;*

$$Pr(r_t = r, r_1 = l_1, r_2 = l_2) = \frac{\binom{n-2r+l_1-1}{r-l_1-l_2-1} \cdot \binom{n}{l_1} \cdot \binom{n-l_1}{l_2}}{2^{n-1}}.$$

Proof. As in the previous Theorems 2.1 and 2.2 we illustrate the sequence as follows;

$$x_1 + x_2 + \ldots + x_r = n \tag{2.5}$$

Let us first assume that the last l_1 runs are of length one and l_2 runs are the runs of length two. The rest are of length at least three. That is,

$$\begin{aligned} x_{r-l_1+1} = \ldots = x_{r-1} = x_r = 1, \\ x_{r-l_1-l_2+1} = \ldots = x_{r-l_1-1} = x_{r-l_1} = 2. \end{aligned}$$

$$\begin{aligned} x_1 + x_2 + \ldots + x_{r-(l_1+l_2)} + \overbrace{2 + 2 + \ldots + 2}^{l_2} + \overbrace{1 + 1 \ldots + 1}^{l_1} &= n, \\ x_1 + x_2 + \ldots + x_{r-(l_1+l_2)} &= n - l_1 - 2l_2. \end{aligned}$$

Notice that here, $x_i \geq 3$. We use the change of variables $y_i = x_i - 3$ for $i = 1, 2, \ldots, r - (l_1 + l_2)$.

$$(x_1 - 3) + (x_2 - 3) + \ldots + (x_{r-(l_1+l_2)} - 3) = n - (l_1 + 2l_2) - 3(r - l_1 - l_2)$$

$$y_1 + y_2 + \ldots + y_{r-(l_1+l_2)} = n - 3r + 2l_1 + l_2. \tag{2.6}$$

The number of sequences having conditions, which are stated above, is equal to the number of non-negative solutions of the equation 2.6. Consequently, by the Remark 2.1, number of desired solutions is,

$$\binom{n-2r+l_1-1}{r-l_1-l_2-1}.$$

Selection of l_1 and l_2 runs of length 1 and length 2 give us a factor of $\binom{r}{l_1}\binom{r-l_1}{l_2}$. Since, each positive integer solution of the equation 2.5 corresponds to two sequences, one starting 1, the other start with 0. We multiply the number of solutions by 2. Therefore; the number of all binary sequences of length n, having total number of runs, l_1 and l_2 of which length one and two respectively, is equal to,

$$2\binom{n-2r+l_1-1}{r-l_1-l_2-1}\binom{r}{l_1}\binom{r-l_1}{l_2}.$$

Hence the probability of a randomly chosen sequence to have these conditions is;

$$Pr(r_t = r, r_1 = l_1, r_2 = l_2) = \frac{2\cdot\binom{n-2r+l_1-1}{r-l_1-l_2-1}\cdot\binom{r}{l_1}\cdot\binom{r-l_1}{l_2}}{2^n}.$$

□

We find the number of sequences having r runs, l_1 and l_2 of which are length one and two respectively, using formula above. In order to define the second new run test, we need number of sequences of length n having l_2 runs of length two, without dependingon the other variables such as, number of runs and number of runs of length one. Corollary 2.5 enables us to compute the probabilities that are needed for defining the new statistical test.

Corollary 2.5. *Let $N_2(l_2)$ denote the number of runs of sequences with exactly i runs of length two. Clearly, we have maximum $\left\lfloor \frac{n}{2} \right\rfloor$ runs of length two. Otherwise sequence length exceeds n. Then for $l_2 = 0, 1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor$,*

$$N_2(l_2) = \sum_{l_1=0}^{n}\sum_{r=1}^{n} 2\cdot\binom{n-2r+l_1-1}{r-l_1-l_2-1}\cdot\binom{r}{l_1}\cdot\binom{r-l_1}{l_2}. \tag{2.7}$$

Since the number of all sequences of length n is 2^n, probabilities follows immediately;

$$Pr(r_2 = l_2) = \frac{N_2(l_2)}{2^n}$$

Also Algorithm 2 enable the calculation for the number of sequences with desired conditions, and hence subinterval probabilities can be stated in the same way in example 2.2. The subinterval probabilities can be seen in the Table 2.2.3

Algorithm 2 Calculating $Pr(r_2 = l_2)$ for $l_2 = 1, 2, \dots, \left\lfloor \frac{n}{2} \right\rfloor$

```
$i \leftarrow 1, l_1 \leftarrow 0, r \leftarrow 1, N_2(l_2) \leftarrow 0.$
while $l_2 \leq \left\lfloor \frac{n}{2} \right\rfloor$ do
  while $l_1 \leq n$ do
    while $r \leq n$ do
      $Pr(r_2 = l_2) \leftarrow Pr(r_2 = l_2) + \binom{n-2r+l_1-1}{r-l_1-l_2-1}\binom{r}{l_1}\binom{r-l_1}{l_2}/2^{n-1}$
      $r \leftarrow r + 1$
    end while
    $l_1 \leftarrow l_1 + 1$
  end while
  $l_2 \leftarrow l_2 + 1$
end while
return $N_2$
```

Table 2.4: Interval and probability values for runs of length two test for 64,128,256,512 bits blocks.

	n=64		n=128		n=256		n=512	
	Interval	Prob.	Interval	Prob.	Interval	Prob.	Interval	Prob.
Box 1	0-5	0.161344	0-12	0.167075	0-27	0.192579	0-57	0.188938
Box 2	6-7	0.260964	13-14	0.174075	28-30	0.194051	58-61	0.178794
Box 3	8	0.149093	15-16	0.209794	31-33	0.222923	62-65	0.210496
Box 4	9-10	0.245287	17-19	0.266590	34-36	0.187853	66-70	0.225615
Box 5	11-32	0.183309	20-64	0.182464	37-128	0.202591	71-256	0.196154

2.2.4 Number of Runs of Length Three

In the last section of this chapter, we focus on the number of sequences having exactly l_3 runs of length three. We use same constructions with the previous sections to compute the number of sequences having total of r runs, l_3 of which are of length three and hence we calculate the probabilities. Then using these calculations, we state the last new statistical test in the next chapter.

Theorem 2.6. *The probability of chosen binary sequence $S = s_1, s_2, \dots, s_n$ with length n, to have r runs, l_1 runs of length one, l_2 runs of length two and l_3 runs of length three is*

$$Pr(r_t = r, r_1 = l_1, r_2 = l_2, r_3 = l_3) = \frac{\binom{n-3r+2l_1+l_2-1}{r-l_1-l_2-l_3-1}.\binom{r}{l_1}.\binom{r-l_1}{l_2}.\binom{r-l_1-l_2}{l_3}}{2^{n-1}}.$$

Proof. As in Theorems 2.1, 2.2 and 2.4 we illustrate the sequence as follows;

$$x_1 + x_2 + \dots + x_r = n \tag{2.8}$$

Let us first assume that the last l_1 are of length 1 l_2 are of length 2 and l_3 of length 3. Rest are at least length four.

$$\begin{gathered} x_{r-l_1+1} = \ldots = x_{r-1} = x_r = 1, \\ x_{r-l_1-l_2+1} = \ldots = x_{r-l_1-1} = x_{r-l_1} = 2, \\ x_{r-l_1-l_2-l_3+1} = \ldots = x_{r-l_1-l_2-1} = x_{r-l_1-l_2} = 3, \end{gathered}$$

$$\begin{gathered} x_1 + x_2 + \ldots + x_{r-l_1-l_2-l_3} + \overbrace{3 + 3 + \ldots + 3}^{l_3} + \overbrace{2 + 2 + \ldots + 2}^{l_2} + \overbrace{1 + 1 \ldots + 1}^{l_1} = n \\ x_1 + x_2 + \ldots + x_{r-l_1-l_2-l_3} = n - r - l_1 - 2l_2 - 3l_3. \end{gathered}$$

Notice that $x_i \geq 4$, we use the change of variables $y_i = x_i - 4$ for $i = 1, 2, \ldots r - (l_1 + l_2 + l_3)$.
The number of cases is equal to the number of non-negative solutions of following equation.

$$(x_1 - 4) + (x_2 - 4) + \ldots + (x_{r-(l_1+l_2+l_3)} - 4) = n - (l_1 + 2l_2 + 3l_3) - 4(r - l_1 - l_2 - l_3)$$

$$y_1 + y_2 + \ldots + y_{r-(l_1+l_2+l_3)} = n - 4r + 3l_1 + 2l_3 + l_3 \tag{2.9}$$

The number of sequences having conditions, which are stated above, is equal to the number of non-negative solutions of the equation 2.9. Consequently, by the Remark 2.1, number of desired solutions is

$$\binom{n - 3r + 2l_1 + l_2 - 1}{r - l_1 - l_2 - l_3 - 1}.$$

Selection of l_1, l_2 and l_3 runs gives us a factor of $\binom{r}{l_1}\binom{r-l_1}{l_2}\binom{r-l_1-l_2}{l_3}$. As stated in Theorems 2.2 and 2.4 we multiply the number by 2. Therefore, the number of all binary sequences of length n with conditions stated above is,

$$2 \cdot \binom{n - 3r + 2l_1 + l_2 - 1}{r - l_1 - l_2 - l_3 - 1} \cdot \binom{r}{l_1} \cdot \binom{r - l_1}{l_2} \cdot \binom{r - l_1 - l_2}{l_3}.$$

Hence, the probability of a randomly chosen sequence to have these conditions is;

$$P(r_t = r, r_1 = l_1, r_2 = l_2, r_3 = l_3) = \frac{2 \cdot \binom{n-3r+2l_1+l_2-1}{r-l_1-l_2-l_3-1} \cdot \binom{r}{l_1} \cdot \binom{r-l_1}{l_2} \cdot \binom{r-l_1-l_2}{l_3}}{2^n}.$$

□

We find the number of sequences having r runs, l_1, l_2, l_3 of which are length one, two and three, using formula above. In order to use probabilities in tests we need numbers of sequences with length n and l_3 runs of length two, without depending ob the other variables. Corollary 2.7 enables us to compute the probabilities that are needed for defining the new statistical test.

Corollary 2.7. *Let $N_3(l_3)$ denote the number of runs of sequences with exactly l_3 runs of length three. Clearly, we have maximum $\left\lfloor \frac{n}{3} \right\rfloor$ runs of length three. if $l_3 > \left\lfloor \frac{n}{3} \right\rfloor$ sequence length exceeds n. Then $l_3 = 0, 1, \dots, \left\lfloor \frac{n}{3} \right\rfloor$,,*

$$N_3(l_3) = \sum_{l_2=0}^{n} \sum_{l_1=0}^{n} \sum_{r=1}^{n} 2\binom{n-3r+2l_1+l_2-1}{r-l_1-l_2-l_3-1} \cdot \binom{r}{l_1} \cdot \binom{r-l_1}{l_2} \cdot \binom{r-l_1-l_2}{l_3}. \tag{2.10}$$

Since the number of all sequences of length n is 2^n, probabilities follows immediately;

$$Pr(r_3 = l_3) = \frac{N_3(l_3)}{2^n}$$

Algorithm 3 Calculating $Pr(r_3 = l_3)$ for $l_3 = 1, 2, \dots, \left\lfloor \frac{n}{3} \right\rfloor$

```
$l_3 \leftarrow 1, l_2 \leftarrow 1, l_1 \leftarrow 1, r \leftarrow 1, N_3(l_3) \leftarrow 1.$
while $l_3 \leq \left\lfloor \frac{n}{3} \right\rfloor$ do
  while $l_2 \leq n$ do
    while $l_1 \leq n$ do
      while $r \leq n$ do
        $Pr(r_3 = l_3) \leftarrow Pr(r_3 = l_3) + 2\binom{n-3r+2l_1+l_2-1}{r-l_1-l_2-l_3-1}.\binom{r}{l_1}.\binom{r-l_1}{l_2}.\binom{r-l_1-l_2}{l_3}$
        $r \leftarrow r + 1$
      end while
      $l_1 \leftarrow l_1 + 1$
    end while
    $l_2 \leftarrow l_2 + 1$
  end while
  $l_3 \leftarrow l_3 + 1$
end while
return $N_3$
```

Since the number of all sequences of length n is 2^n, probabilities follows immediately; $Pr(r_3 = l_3) = \frac{N_3(l_3)}{2^n}$. And Algorithm 3 enable the calculations for the number of sequences of length n and l_3 runs of length three and hence subinterval probabilities can be stated in the same way in example 2.2. The subinterval probabilities can be seen in the Table 2.5

2.3 Summary

In this chapter we give the proofs of our main theorems. First, we give the required corollaries and algorithms in order to formulate the exact number of sequences with the given conditions. Hence, probabilities for the conditions on number of runs, number or runs of one, two and three are stated in a combinatorial way.

Table 2.5: Interval and probability values for runs of length three test for 64,128,256 bits blocks.

	n=64		n=128		n=256	
	Interval	Prob.	Interval	Prob.	Interval	Prob.
Box 1	0-2	0.207825	0-5	0.163209	0-13	0.248734
Box 2	3	0.204319	5-7	0.274500	14-15	0.207164
Box 3	4	0.216732	8	0.154854	16-17	0.213743
Box 4	5-6	0.283245	9-10	0.245059	18-20	0.222144
Box 5	7-21	0.087877	11-42	0.162376	20-85	0.108212

Probabilities for runs of length more than three can be calculated theoretically in the same way however, time complexities of algorithms to find exact numbers (and probabilities) grow exponentially. Therefore, they are inconvenient to be used in test suites.

In the Chapter 3 we give the descriptions of the new statistical tests and state the pseudocodes.

CHAPTER 3

TEST DESCRIPTIONS

The Golomb's first postulate is about the weight of a sequence and in many test suites the postulate is implemented with a proper generalization. On the other hand, the second postulate, which is about runs of a sequence, is mostly implemented only by using total number of runs regardless of their length. In this chapter, we define three new statistical test based on the Golomb's second postulate, which are runs of length one test, runs of length two test and runs of length three test. The subjects of new run tests are r_1,r_2 and r_3 as their names state. We use probabilities calculated in previous chapter.

We use χ^2 as reference distribution and compare the measurements with expected values. In order to to this, we divide number of runs of length one, two and three into subintervals, whose probabilities are approximately the same. That is, tests use the subintervals $(\alpha_0, \alpha_1), (\alpha_1, \alpha_2), \ldots, (\alpha_4, \alpha_5)$ such that, $Pr_i(\alpha_i < R < \alpha_{i+1}) \approx 0.2$. For example; for a 128-bit sequence, runs of length two are divided into 5 parts as follows;

$$\begin{aligned}
Pr_1(1 \leq r_2 \leq 12) &= 0.167075 \\
Pr_2(13 \leq r_2 \leq 14) &= 0.174075 \\
Pr_3(15 \leq r_2 \leq 16) &= 0.209794 \\
Pr_4(17 \leq r_2 \leq 19) &= 0.266590 \\
Pr_5(20 \leq r_2 \leq 32) &= 0.182464
\end{aligned}$$

Then we count the number of runs of length i in the m sequences according to the subintervals and denote the number of sequences in the given subinterval by F_i. Before the last step we calculate the χ^2 by following formula [30].

$$\chi^2 = \sum_{i=1}^{5} \frac{(F_i - m \cdot Pr_i)^2}{m \cdot Pr_i}.$$

Lastly *p-value* is calculated according to the given values;

$$p\text{-}value = \mathtt{igamc}(\frac{5}{2}, \frac{\chi^2}{2}).$$

The next question is; how long sequences should be taken, in order to deducing a reliable conclusion? In NIST test suite it is suggested that sequences should be about 20.000 bits long. But in new statistical tests we suggests that the tested sequences should be about $m \cdot 25$ where m is the block size . This number is a direct consequence of creating subintervals. We need at least 5 block of sequences in order to get true values.

Remark 3.1. **Derivative of a Sequence**

Counting runs of a sequence by using the definition is unpractical. So we use the derivative of a sequence to count the runs. By the definition all 1's in the derivative of a sequence indicates the end of a run. So the number of runs of a sequence can be defined as weight of its derivative.

Let $S = s_0, s_1 \ldots, s_{n-1}$ be a binary sequence of length n then, derivative of S ,denoted by $\Delta S = \Delta s_0, \Delta s_1 \ldots, \Delta s_{n-1}$ is defined as follows;

For $i = 0, 1, \ldots, n-1$

$$\Delta s_i = \begin{cases} s_i + s_{i+1} & \text{if } i = 0, 1, \ldots, n-2 \\ 1 & \text{if } i = n-1 \end{cases} .$$

Also we use a variation of derivative, $\Delta S'$ of length $n+1$ by adding 1's at the beginning of the sequence ΔS. The variation of derivative is an important part of the newly defined run tests, since the number of runs of different lengths is determined by this sequence.

Remark 3.2. Let $S = s_0, s_1 \ldots, s_{n-1}$ be a binary sequence and derivative of S is denoted by $\Delta S = \Delta s_0, \Delta s_1, \ldots, \Delta s_{n-1}$. Then $\Delta S' = \Delta s'_0, \Delta s'_1 \ldots, \Delta s'_n$ is defined as follows;

$$\Delta s'_i = \begin{cases} \Delta s_{i-1} & \text{if } i = 1, \ldots, n \\ 1 & \text{if } i = 0 \end{cases} .$$

Instead of derivative, we use the variation of derivative in order to count the runs at the beginning. Number of runs of length one in a sequence is indicated by the number of overlapping occurrences of *11* . In the same way number of runs of length 2 and 3 in a sequence is indicated by the number of overlapping occurrences of *101* and *1001* respectively. More generally we can say that number of runs of length n is indicated by the overlapping number of occurrences of $1\underbrace{00\ldots0}_{n-1}1$.

Example 3.1. Let S=01100010011111001100011101010000 be a binary sequence length of 32, having 15 runs, 6 runs of length one, 4 runs of length two and 3 runs of length three. Then;

$$\Delta s_0 = s_0 + s_1,\ \Delta s_1 = s_0 + s_2,\ \ldots\ ,\Delta s_{31} = s_{31} + s_{32},\ \Delta s_{32} = 1.$$

$$\begin{aligned}\Delta S &= \quad 10100110100001010100100111100001\\ \Delta S' &= 1\quad 10100110100001010100100111100001\end{aligned}$$

- Weight of ΔS is 15 which corresponds to number of runs.
- Number of overlapping occurrences of 11 is 6 which corresponds to number of runs of length one;

  $$\Delta S' = \underbrace{11}_{1}0100\underbrace{11}_{1}0100001010100100\underbrace{11111}_{4}00001.$$

- Number of overlapping occurrences of 101 is 4 which corresponds to number of runs of length two;
  $\Delta S' = 1\underbrace{101}_{1}001\underbrace{101}_{1}0000\underbrace{10101}_{2}001001111100001.$
- Number of overlapping occurrences of 1001 is 3 which corresponds to number of runs of length three;
  $\Delta S' = 110\underbrace{1001}_{1}10100001010\underbrace{1001001}_{2}111100001.$

Before defining new statistical tests, we give the general idea of the test by following example;

Example 3.2. Let S be a binary sequence of length 2^{21}. Let F_i and Pr_i are the number of sequences in the given subinterval and probability of it respectively.

- **Step 1:** Choose a block size *m*. In our example we choose *m* as 128.
- **Step 2:** Then divide the sequence into *m* bit sequences. Then we get the set of sequences as follows, $\mathbf{S} = \{S_1, S_2, \ldots, S_{2^{14}}\}$.
- **Step 3:** For each S_i count the number of runs of length one, two and three. And increment the corresponding boxes by 1.

  $$\begin{aligned} S_1 &= [0,1,0,0,\ldots,1] \longrightarrow r_t{=}65,\ r_1{=}33,\ r_2{=}15,\ r_3{=}8\\ S_2 &= [0,1,1,0,\ldots,0] \longrightarrow r_t{=}64,\ r_1{=}32,\ r_2{=}17,\ r_3{=}9\\ &\vdots\\ S_{2^{14}} &= [0,1,0,0,\ldots,1] \longrightarrow r_t{=}65,\ r_1{=}30,\ r_2{=}16,\ r_3{=}8\end{aligned}$$

- **Step 4:** After that, we get the Table 3.1. *Count* rows in the table correspond to the number of sequences whose number of runs of length one, two or three is in given interval.
- **Step 5:** Then we calculate the χ^2 by the given formula and from χ^2 we compute the *p-value*.

Table 3.1: Number of sequences in the given intervals for number of runs test, runs of length one test, runs of length two test and runs of length three test.

Number of Runs Test		
	Interval	Count
F_1	0-58	3.161
F_2	59-61	2.890
F_3	62-64	3.351
F_4	65-67	3.143
F_5	68-128	3.839

Runs of Length One Test		
	Interval	Count
F_1	0-27	3.699
F_2	28-31	3.744
F_3	32-34	3.016
F_4	35-38	3.155
F_5	39-128	2.770

Runs of Length Two Test		
	Interval	Count
F_1	0-12	2.806
F_2	13-14	2.838
F_3	15-16	3.476
F_4	17-19	4.331
F_5	20-64	2.933

Runs of Length Three Test		
	Interval	Count
F_1	0-5	2.634
F_2	5-7	4.447
F_3	8	2.532
F_4	9-10	4.082
F_5	11-42	2.689

$$\chi^2 = \sum_{i=1}^{5} \frac{(F_i - 2^{20} \cdot Pr_i)^2}{2^{20} \cdot Pr_i} \text{ and } p\text{-}value = \texttt{igamc}(\tfrac{5}{2}, \tfrac{\chi^2}{2}).$$

- **Step 6:** Finally, we conclude the sequence is random or not:
  - Number of runs test: *p-value*=0.175195
  - Number of runs of length one test: *p-value*=0.357056.
  - Number of runs of length two test: *p-value*=0.462207.
  - Number of runs of length three test: *p-value*=0.627001.

3.1 Number of Runs

Number of runs test, is implemented in a similar approach which is stated in [30]. Test uses the probabilities calculated in the previous chapter and counts the runs of sequences, according to the subintervals. First, we collect the algorithm's output and generate the data set **S**. If given sequence of length n is a long binary sequence, the sequence is divided into m bit blocks and get a set of sequences and generate $\mathbf{S} = \{S_1, S_2, \ldots, S_{2^N}\}$ where $N = \left\lfloor \frac{n}{m} \right\rfloor$. In our test m can be 64, 128, 256 or 512. After generating the data set, the test counts the number of runs of length one in each of the sequences, according to the subintervals. In order to find the number of runs of length one, first we find the derivative of the binary sequence ΔS_k, then weight of ΔS_k indicates the number of runs in the sequence. After that we apply χ^2 of Goodness of Fit test to r values. In the last step value of χ^2 is used to find the *p-value*. The pseudocode of the test is given as follows;

Algorithm 4 Number of Runs Test($S_1, S_2, \ldots, S_m$)

```
$\Delta S'_k = s'_{k,0}, s'_{k,1}, \ldots, s'_{k,n-1}$
$i \longleftarrow 0$, $r_{k,t} \longleftarrow 0$
while $i \leq n$ do
    if $s'_{k,i} = 1$ then
        $r_{k,t} \longleftarrow r_{k,t} + 1$
    end if
end while
Apply $\chi^2$ of Goodness of Fit test to $r_t$ values,
return p-value
```

3.2 Runs of Length One Test

The subject of first new run test is runs of length one. Test uses the probabilities calculated in the previous chapter. First, we collect the algorithms output and generate the data set **S**. If the given sequence of length n is a long binary sequence, the sequence is divided into m bit blocks and get a set of sequences and generate $\mathbf{S} = \{S_1, S_2, \ldots, S_{2^N}\}$ where $N = \left\lfloor \frac{n}{m} \right\rfloor$. In our test m can be 64, 128, 256 or 512. After generating the data set, the test counts the number of runs of length one in each of the sequences, according to the subintervals. In order to find the number of runs of length one, first we find the derivative of the binary sequence ΔS_k, then we count the overlapping occurrences 11 in $\Delta S'_k$ for $k = 1, 2, \ldots, m$. After that we apply χ^2 of Goodness of Fit test to l_1 values. In the last step value of χ^2 is used to find the *p-value*. We propose new run test to implement the idea of Golomb's second postulate in statistical randomness test. The pseudocode of the test is given in the Algorithm 5;

Algorithm 5 Runs of Length One Test($S_1, S_2, \ldots, S_m$)

```
$\Delta S'_k == \Delta s'_{k,0}, \Delta s'_{k,1}, \ldots, \Delta s'_{k,n-1}$
$i \longleftarrow 0$, $l_{k,1} \longleftarrow 0$
while $i \leq n$ do
    $temp = \Delta s'_{k,i} \cdot 2^1 + \Delta s'_{k,i+1} \cdot 2^0$
    if $temp = 3$ then
        $l_{k,1} \longleftarrow l_{k,1} + 1$
    end if
end while
Apply $\chi^2$ of Goodness of Fit test to $l_1$ values,
return p-value
```

3.3 Runs of Length Two Test

After giving the first new run test, we define runs of length two test. As its name suggest, the subject of test is runs of length two. Test uses the probabilities calculated previous chapter. As in the runs of length one test first, we collect the algorithms output and generate the data set **S** and if given sequence of length n is a long binary sequence, the sequence is divided into m bit blocks and get a set of sequences and generate $\mathbf{S} = \{S_1, S_2, \ldots, S_{2^N}\}$ where $N = \left\lfloor \frac{n}{m} \right\rfloor$. In our test m can be 64, 128, 256 or 512. After generating the data set test counts the runs of length two in sequences, according to the subintervals. Like in the previous tests we get the derivative of the binary sequence ΔS_k. In order to find the number of runs of length two, we count the overlapping occurrences 101 in $\Delta S'_k$. Then we apply χ^2 of Goodness of Fit test to l_2 values. Then value of χ^2 is used to find the *p-value*. The second new run test constitutes another approach to the Golomb's second postulate. The pseudocode of the test is given as in the Algorithm 5;

Algorithm 6 Runs of Length Two Test($S_1, S_2, \ldots, S_m$)

```
$\Delta S'_k = \Delta s'_{k,0}, \Delta s'_{k,1}, \ldots, \Delta s'_{k,n-1}$
$i \longleftarrow 0,\ l_{k,2} \longleftarrow 0$
**while** $i \leq n - 1$ **do**
    $temp = \Delta s'_{k,i} \cdot 2^2 + \Delta s'_{k,i+1} \cdot 2^1 + \Delta s'_{k,i+2} \cdot 2^0$
    **if** $temp = 5$ **then**
        $l_{k,2} \longleftarrow l_{k,2} + 1$
    **end if**
**end while**
Apply $\chi^2$ of Goodness of Fit test to $l_2$ values,
**return** *p-value*
```

3.4 Runs of Length Three Test

The last new run test is runs of length three test, whose subject is number of runs of length three. Test uses the probabilities calculated in the previous chapter. Data set are created in a same manner, that is, we collect the algorithm's output and generate the data set **S**, if the sequence is a long sequence we just divide into m bit blocks and then generate the data set $\mathbf{S} = \{S_1, S_2, \ldots, S_{2^N}\}$ where $N = \left\lfloor \frac{n}{m} \right\rfloor$. Then, test counts the runs of length three in sequences, according to the subintervals. After getting the derivative of the binary sequence ΔS_k, we count the number of runs of length three by just looking the overlapping occurrences 1001 in in $\Delta S'_k$. Then we apply χ^2 of Goodness of Fit test to l_2 values. Then value of χ^2 is used to find the *p-value*. The pseudocode of the last new run test, runs of length three test, is given as follows;

3.5 Summary

Our aim is to extend the idea of Golomb's postulates in randomness testing. In order to do this, we define three new statistical randomness tests in this chapter. First, Basic definitions and idea for statistical randomness tests are given. New tests are concern with total number of runs and runs of length one, two and three and they are designed to detect deviations of number of runs of different lengths from a random sequence.

Algorithm 7 Runs of Length Three Test($S_1, S_2, \ldots, S_m$)

```
  $\Delta S'_k = \Delta s'_{k,0}, \Delta s'_{k,1}, \ldots, \Delta s'_{k,n-1}$
  $i \longleftarrow 0,\ l_{k,3} \longleftarrow 0$
  **while** $i \leq n-2$ **do**
      temp= $\Delta s'_{k,i} \cdot 2^3 + \Delta s'_{k,i+1} \cdot 2^2 + \Delta s'_{k,i+2} \cdot 2^1 + \Delta s'_{k,i+3} \cdot 2^0$
      **if** $temp = 9$ **then**
          $l_{k,3} \longleftarrow l_{k,3} + 1$
      **end if**
  **end while**
  Apply $\chi^2$ of Goodness of Fit test to $l_3$ values,
  **return** *p-value*
```

In other words, the new tests are not only interested in total number of runs but also distribution of runs. In the next chapter, we focus on the results of the new statistical randomness tests.

CHAPTER 4

APPLICATIONS

In Chapter 4 new statistical randomness tests are implemented new statistical randomness tests on some well-known algorithms and binary expansion of the irrational numbers e, π and $\sqrt{2}$. The experiments are done in order to show both the performance and the sensitivity of new statistical tests. Also, some statistical randomness tests, included in NIST test package, are applied on each data file in order to compare the results and results are listed in the following tables.

New statistical tests are designed to detect deviations of number of various length from a random sequence. Therefore we generate some non-random data to show the efficiency of new statistical randomness tests. Data generation methods are explained briefly in the following sections.

4.1 Application on AES Finalists Algorithms

As stated in Section 1.1 outputs of block ciphers should give no information in the absence of input or in other words should be indistinguishable from random mapping and outputs of block ciphers should be evaluated carefully. Therefore, we choose the application on block ciphers as our first implementation.

In order to check the validity of tests stated in the previous section, they are applied to random data with some of tests included in NIST test suite. In the first implementation we select 5 known algorithms, which are Advanced Encryption Algorithms finalists, MARS [5], RC6 [22], Rijndael [7], Serpent [4] and Twofish [26]. AES finalist algorithms are used to generate the 2^{14} pseudorandom sequences of length 128 by encrypting non-correlated data. In the same way we generate 2^{13} pseudorandom sequences of length 256.

Tests from NIST test suite and new statistical tests are implemented in order to compare the results. For the tests, which included in NIST test suite, each algorithms' outputs are concatenated and a long sequence of length 2^{21} is generated. This method is also used in evaluation of AES finalists [28]. On the other hand, new statistical randomness tests take each binary sequences of length 128 and 256 individually. After analyzing 2^{14} and 2^{13} sequences, tests are implemented as they stated in previous chapter. The results can be seen in Table 4.1 and Table 4.2. It can be seen from the results that, if there is a deviation in number of runs, with the new statistical randomness tests we can indicate cause of this deviation.

Table 4.1: Test results for the 128-bit outputs of AES finalists.

Statistical Tests	Rijndeal	Serpent	Mars	RC6	Twofish
Frequency Test	0.877073	0.385771	0.100285	0.813306	0.667550
Block Freq test	0.722551	0.159257	0.801489	0.475342	0.199609
Run test	0.703085	0.000651	0.003002	0.006542	0.006737
Longest Run of Ones in a Block	0.031990	0.661453	0.229015	0.338937	0.308989
Universal Statistical Test	0.006504	0.048462	0.007328	0.108877	0.023687
Linear Complexity Test	0.308490	0.231002	0.159494	0.662083	0.452449
Serial Test^1	0.016532	0.249989	0.748831	0.307892	0.629330
Serial Test^2	0.444775	0.504040	0.226215	0.602572	0.923866
Approximate Entropy Test	0.001276	0.070437	0.322856	0.053931	0.220444
Cumulative Sums Test Backward	0.271617	0.627426	0.152360	0.822441	0.838133
Cumulative Sums Test Forward	0.362406	0.501622	0.057094	0.971814	0.877082
Random Excursion Test	0.949243	0.143578	0.455967	0.307333	0.409744
Random Excursions Variant Test	0.816055	0.042998	0.515433	0.160018	0.041629
Number of Runs Test	0.820133	0.030861	0.062100	0.043231	0.060107
Runs of Length One Test	0.535513	0.076538	0.021622	0.055930	0.008255
Runs of Length Two Test	0.095602	0.339466	0.051861	0.057043	0.309454
Runs of Length Three Test	0.359483	0.213636	0.388663	0.318248	0.081348

4.2 Application on Binary Expansions

In the second part of our experiments, we use binary expansion of irrational numbers e, π and $\sqrt{2}$. The data can be found in the NIST package. As in the first part we also use some test that are included in NIST test suite. We collect first 2^{19} bits of the binary expansions and tested. For our tests we divide them into 128 bit blocks, hence we get 2^{12} sequences each of length 128. By this implementation we try to show the performance of new statistical tests. The test results can be seen in Table 4.3.

4.3 Applications on Non-Random Data

Following implementation involves a random number generator, whose outputs is between 0 and 1. From each generated random number, we construct our sequence. We also use an important definition stated in [12].

Definition 4.1. Let S be a binary sequence of length n and i^{th} element of it is represented as s_i, then bias q is defined as follows,

$$Pr(s_i = 1) = \frac{1}{2} + q$$
$$Pr(s_i = 0) = \frac{1}{2} - q$$

Table 4.2: Test results for the 256-bit outputs of AES finalists.

Statistical Tests	Rijndeal	Serpent	Mars	RC6	Twofish
Frequency Test	0.706150	0.564679	0.543406	0.238226	0.986777
Block Freq test	0.026959	0.378371	0.499770	0.389093	0.014817
Run test	0.012096	0.024743	0.015844	0.004712	0.333670
Longest Run of Ones in a Block	0.485537	0.709083	0.241949	0.354484	0.096190
Universal Statistical Test	0.022617	2.022754	0.197799	0.003131	0.011990
Linear Complexity Test	0.736271	0.755534	0.693168	0.162651	0.506699
Serial Test[1]	0.097033	0.254846	0.236358	0.442786	0.664981
Serial Test[2]	0.024441	0.524554	0.320521	0.594446	0.602472
Approximate Entropy Test	0.008027	0.056474	0.036288	0.137759	0.351455
Cumulative Sums Test - Backward	0.660527	0.690829	0.800794	0.315770	0.590035
Cumulative Sums Test - Forward	0.358374	0.795147	0.549565	0.275989	0.575370
Random Excursion Test(+1)	0.347825	0.177147	0.335655	0.115141	0.024925
Random Excursions Variant Test(-1)	0.019553	0.680250	0.330268	0.182713	0.546645
Number of runs	0.032656	0.031279	0.067175	0.056588	0.907728
Number of Runs Test	0.820133	0.381061	0.062100	0.043231	0.907728
Runs of Length One Test	0.077425	0.038299	0.069153	0.035235	0.554352
Runs of Length Two Test	0.263405	0.361392	0.030149	0.115706	0.784606
Runs of Length Three Test	0.285480	0.346626	0.861162	0.345888	0.108004

As stated in the Section 1.1 random sequences should have some properties, from the three basic properties we can say that in a true random sequence we expect bias as 0. that is, $Pr(s_i = 1) = Pr(s_i = 0) = \frac{1}{2}$

If we increase the bias and generate the sequence, it is clear that the sequence become non-random. The Algorithm 8 shows the generation of biased sequence.

Algorithm 8 Generation of Biased Sequence $S^q = s_1^q, s_2^q, \ldots, s_n^q$

```
Let $R = r_0, r_1, \ldots, r_{n-1}$ be the outputs of a random number generator and $0 \leq r_i \leq 1$ for $i = 1, 2, \ldots, n-1$
i ← 0
while i < n do
  if $r_i \leq 0.5 + q$ then
    $s_i \leftarrow 0$
  else
    $s_i \leftarrow 1$
  end if
  i ← i+1
end while
return $S^q$
```

Example 4.1. Let $R = r_0, r_1, \ldots, r_{n-1}$ be a random sequence with $0 \leq r_i \leq 1$ for $i = 1, 2, \ldots, n-1$, from this sequence we construct a binary sequence with bias 0.05. The sequence with bias is constructed as follows,

$$s_j^q = \begin{cases} 0 \text{ if } r_i \leq 0.5 + 0.05 \\ 1 \text{ if } r_i > 0.5 + 0.05 \end{cases} .$$

Table 4.3: Test results for the binary expansion of e, π and $\sqrt{2}$.

Statistical Test	e	π	$\sqrt{2}$
Frequency Test	0.818668	0.393382	0.820816
Block Freq test	0.069195	0.191721	0.578760
Run test	0.489904	0.409869	0.894467
Longest Run of Ones in a Block	0.328344	0.048248	0.537307
Universal Statistical Test	0.930374	0.915310	0.462562
Linear Complexity Test	0.927809	0.208269	0.396546
Serial Test[1]	0.924970	0.232328	0.247445
Serial Test[2]	0.719054	0.221747	0.037551
Approximate Entropy Test	0.707174	0.085060	0.837672
Cumulative Sums Test - Backward	0.373319	0.333600	0.629320
Cumulative Sums Test - Forward	0.242488	0.313745	0.838133
Random Excursion Test	0.892831	0.844143	0.270246
Random Excursions Variant Test	0.388323	0.760966	0.461287
Number of Runs Test	0.225035	0.086202	0.272067
Runs of Length One Test	0.241279	0.097072	0.138194
Runs of Length Two Test	0.092391	0.129520	0.158537
Runs of Length Three Test	0.215721	0.114384	0.076582

$$Pr(s_j^q = 1) = 0.55 \text{ and } Pr(s_j^q = 0) = 0.45$$

Table 4.4: Test results for non-random data sets.

Statistical Test	$q = 0.0$	$q = 0.01$	$q = 0.03$
Frequency Test	0.375269	0.040143	0.000475
Block Frequency test	0.760739	0.802281	0.777309
Run test	0.794303	0.903035	0.859454
Longest Run of Ones in a Block	0.562918	0.257811	0.093295
NonOverlapping template test(M=9, B=000000001)	0.436359	0.377016	0.328182
Overlapping template test(M=9)	0.746164	0.642254	0.714769
Linear Complexity Test	0.693577	0.703492	0.670893
Serial Test[1]	0.680524	0.681398	0.549883
Serial Test[2]	0.538842	0.746869	0.615192
Approximate Entropy Test	0.372373	0.239482	0.308904
Cumulative Sums Test- Backward	0.372373	0.032272	0.000333
Cumulative Sums Test - Forward	0.429406	0.073315	0.000857
Number of Runs Test	0.912964	0.893689	0.941435
Runs of Length One Test	0.818485	0.832025	0.809327
Runs of Length Two Test	0.944299	0.790180	0.852354
Runs of Length Three Test	0.574298	0.782597	0.891987

In this implementation we create data sets with different biases with above construction and therefore we observe the behavior of tests with respect to the randomness of a sequence. The test results can be seen in Table 4.4.

4.4 Summary

In this chapter new statistical randomness tests are implemented on some well-known algorithms and binary expansion on three irrational numbers. Experiments shows the performance and sensitivity of the tests. Moreover, we implement other tests, which are also included in NIST test suite, on the same data in order to compare the results. The results show that, the cause of a deviation in number of runs can be detected only by new statistical randomness tests.

We generate some sequences with bias from a pseudorandom sequence to show the sensitivity of the tests. Also implementation results show the efficiency of the tests. New tests can detect the deviations in distributions of runs while other tests cannot detect.

Last implementation results show the efficiency of the new tests and detecting the deviations in distributions of runs while other tests cannot detect.

CHAPTER 5

CONCLUSION

Random numbers and random sequences are essential in many different areas. In cryptography, need for random values emerge in almost all protocols and most importantly in key generation. Therefore, randomness is one of the most important issue for cryptographic algorithms. In fact, using weak random values enable us an adversary to break the whole system. For all applications, randomness and size of used values is important issues. Accordingly, the probability of any chosen quantity should be so small that, an adversary shouldn't get any specific information. Thus, sequences used in cryptographic algorithms should be pseudorandom and should have good statistical properties. Statistical tests are designed to detect deficiencies that a generator can have. Therefore, statistical randomness tests are stated as an essential part of evaluating security of cryptographic algorithms.

In this thesis, we propose three new statistical tests based the Golomb's second postulate. Finding the real probabilities related to number of runs of length one, two and three enable us to compare the observed values accordingly. New run tests can be used in test suites to test security of algorithms so that Golomb's second postulate is implemented in a proper way. Moreover, these tests can be used as an evaluation tool for short sequences such as, outputs of block ciphers and hash functions. These tests can detect the deviation in distribution runs which cannot be detected by other tests. Also, we use them on some standard algorithms that behave pseudorandom number generator and random sequences such as binary expansion of e, π and $\sqrt{2}$. Implementations shows the consistency of new statistical test with other well-known statistical tests. It is shown that in order to detect some deviation from randomness, new statistical tests are more efficient than other statistical tests.

The contribution of the thesis and future works can be stated as follows.

- We have pointed out the importance of randomness in cryptography, and we have focused on statistical test suites and Golomb's postulates.
- According to the Golomb's second postulate we propose three new statistical randomness tests.
- We develop a notation to illustrate runs of a sequence. With the notation and combinatorial formulas, exact probabilities are calculated. Calculated probabilities have some independence. Therefore, the corollaries and the algorithms that

are needed to find the desired probabilities are given. These probabilities can be summarized as follows; for an n-bit binary sequence $Pr(r_t = r)$, $Pr(r_1 = l_1)$, $Pr(r_2 = l_2)$ and $Pr(r_3 = l_3)$.

- Probabilities associated to total number of runs, runs of length one, two and three probabilities are divided into five intervals, in such a manner that, each interval has nearly equal probabilities. Each interval called as a subintervals.

- According to the subinterval probabilities the new testes are defined. New test are called as number of runs test, runs of length one test, runs of length two test and runs of length three test.

- We give exact probabilities for different block length which are 64, 128, 256 and 512. Therefore, the new tests can be applied on short sequences and hence can be used for evaluating block ciphers and hash functions.

- Implementation on some well-known algorithms and binary expansion on irrational number are done in order to check the validity of the new tests. Also, we give the implementation of other tests on the same data in order to compare the results.

- We propose a new method for generating a biased sequence and using this method we generate non-random sequences. We show the sensitivity of the new tests.

- We developed a software in order to do the implementation. This software includes the test in NIST package and new statistical randomness tests. The source code and executable file can be downloaded from: `https://www.dropbox.com/s/8ggjndn7tzc8kdh/SRT_GUI.rar`

- As a future work, correlations between new statistical tests and also with other statistical tests can be examined.

REFERENCES

[1] D. M. M. Alani, Testing randomness in ciphertext of block-ciphers using diehard tests, International Journal of Computer Science and Network Security, 10(4), 2010.

[2] P. M. Alcover, A. Guillamon, and M. d. C. Ruiz, New randomness test for bit sequences., Informatica, 24(3), pp. 339–356, 2013.

[3] L. E. Bassham, III, A. L. Rukhin, J. Soto, J. R. Nechvatal, M. E. Smid, E. B. Barker, S. D. Leigh, M. Levenson, M. Vangel, D. L. Banks, N. A. Heckert, J. F. Dray, and S. Vo, Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications, Technical report, NIST, Gaithersburg, MD, United States, 2010.

[4] E. Biham, R. J. Anderson, and L. R. Knudsen, Serpent: A new block cipher proposal, in *Fast Software Encryption, 5th International Workshop, FSE '98, Paris, France, March 23-25, 1998, Proceedings*, volume 1372 of *Lecture Notes in Computer Science*, pp. 222–238, Springer, 1998.

[5] C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S. M. M. Jr, L. O'Connor, M. Peyravian, J. Luke, O. M. Peyravian, D. Stafford, and N. Zunic, Mars - a candidate cipher for AES, NIST AES Proposal, 1999.

[6] W. Caelli, Crypt x package documentation, Technical report, Information Security Research, 1992.

[7] J. Daemen and V. Rijmen, *The Design of Rijndael*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002, ISBN 3540425802.

[8] S. W. Golomb, *Shift Register Sequences*, Aegean Park Press, Laguna Hills, CA, USA, 1982, ISBN 978-0894120480.

[9] K. Hamano and T. Kaneko, Correction of overlapping template matching test included in nist randomness test suite, IEICE Trans. Fundam. Electron. Commun. Comput. Sci., E90-A(9), pp. 1788–1792, 2007, ISSN 0916-8508.

[10] K. Hamano and H. Yamamoto, A randomness test based on t-codes, in *Information Theory and Its Applications, 2008. ISITA 2008. International Symposium on*, pp. 1–6, Dec 2008.

[11] J. Hernandez, J. Sierra, and A. Seznec, The sac test: A new randomness test, with some applications to prng analysis, in A. Laganá, M. Gavrilova, V. Kumar, Y. Mun, C. Tan, and O. Gervasi, editors, *Computational Science and Its Applications – ICCSA 2004*, volume 3043 of *Lecture Notes in Computer Science*, pp. 960–967, Springer Berlin Heidelberg, 2004, ISBN 978-3-540-22054-1.

[12] H. M. Heys, A tutorial on linear and differential cryptanalysis, Cryptologia, 26(3), pp. 189–221, July 2002, ISSN 0161-1194.

[13] V. Katos, A randomness test for block ciphers., Applied Mathematics and Computation, 162, pp. 29–35, 2005.

[14] A. Kerckhoffs, La cryptographie militaire, Journal des Sciences Militaires, pp. 161–191, 1883.

[15] S.-J. Kim, K. Umeno, and A. Hasegawa, Corrections of the nist statistical test suite for randomness., IACR Cryptology ePrint Archive, 2004, p. 18, 2004.

[16] D. E. Knuth, *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997, ISBN 0-201-89684-2.

[17] A. N. Kolmogorov, Three approaches to the quantitative definition of information, International Journal of Computer Mathematics, 2(1-4), pp. 157–168, 1968.

[18] P. L'Ecuyer and R. Simard, Testu01: A c library for empirical testing of random number generators, ACM Trans. Math. Softw., 33(4), August 2007, ISSN 0098-3500.

[19] G. Marsaglia, The marsaglia random number CDROM including the diehard battery of tests of randomness, http://www.stat.fsu.edu/pub/diehard/, 1995.

[20] U. Maurer, A universal statistical test for random bit generators, Journal of cryptology, 5, pp. 89–105, 1992.

[21] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*, CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996, ISBN 0849385237.

[22] R. L. Rivest, M. J. B. Robshaw, Y. Yin, and R. Sidney, The rc6 block cipher, 1998.

[23] S. Ross, *A First course in probability*, Prentice Hall, 6. ed edition, 2002, ISBN 0131218026.

[24] A. Ruhkin, Testing randomness: A suite of statistical procedures, Theory of Probability & Its Applications, 45(1), pp. 111–132, 2001.

[25] B. Ryabko, V. Stognienko, and Y. Shokin, A new test for randomness and its application to some cryptographic problems, Journal of Statistical Planning and Inference, 123(2), pp. 365 – 376, 2004, ISSN 0378-3758.

[26] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson, Twofish: A 128-bit block cipher, in *First Advanced Encryption Standard (AES) Conference*, 1998.

[27] M. Sönmez Turan, *On statistical analysis of synchronous stream ciphers, PhD Thesis Supervisor Assoc. Prof. Dr. Ali Doğanaksoy*, Ankara : METU, 2008.

[28] J. Soto and L. Bassham, Randomness testing of the advanced encryption standard finalist candidates, in *NIST IR 6483, National Institute of Standards and Technology*, 1999.

[29] F. Sulak, *Statistical analysis of block ciphers and hash functions, PhD Thesis Supervisor Assoc. Prof. Dr. Ali Doğanaksoy*, Ankara : METU, 2011.

[30] F. Sulak, A. Doğanaksoy, B. Ege, and O. Koçak, Evaluation of randomness test results for short sequences, in *Proceedings of the 6th International Conference on Sequences and Their Applications*, SETA'10, pp. 309–319, Springer-Verlag, Berlin, Heidelberg, 2010.