UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Classification of pseudo-random number generators by complex networks and computational geometry analysis

Marcela Lopes Alves

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Marcela Lopes Alves

Classification of pseudo-random number generators by complex networks and computational geometry analysis

Master dissertation submitted to the Institute of
Mathematics and Computer Sciences – ICMC-USP,
in partial fulfillment of the requirements for the
degree of the Master Program in Computer Science
and Computational Mathematics. *EXAMINATION
BOARD PRESENTATION COPY*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Prof^o. Dr. Odemir Martinez Bruno

USP – São Carlos
July 2019

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

L864c Lopes Alves, Marcela
Classification of pseudo-random number
generators by complex networks and computational
geometry analysis / Marcela Lopes Alves; orientador
Odemir Martinez Bruno. -- São Carlos, 2019.
153 p.

Dissertação (Mestrado - Programa de Pós-Graduação
em Ciências de Computação e Matemática
Computacional) -- Instituto de Ciências Matemáticas
e de Computação, Universidade de São Paulo, 2019.

1. Randomness. 2. Complex networks. 3.
Computational geometry. 4. Complex systems. 5.
Pseudorandom number generator. I. Martinez Bruno,
Odemir , orient. II. Título.

Marcela Lopes Alves

**Classificação de geradores de números pseudoaleatórios
aplicando análise de redes complexas e geometria
computacional**

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestra em Ciências – Ciências de Computação e Matemática Computacional. *EXEMPLAR DE DEFESA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof^o. Dr. Odemir Martinez Bruno

USP – São Carlos
Julho de 2019

For those who came from chaos.

ACKNOWLEDGEMENTS

First of all, I thank Prof. Odemir Bruno who gave me the great opportunity to come to USP and achieve the dream of studying in this university. It will always be in my memory our talks about the history of science, women in the exact sciences, and the many debates when we did not agree on a subject. Thank you. I also greatly thank Prof. Jan Baetens who introduced me to Prof. Odemir.

My thanks extend to all the collaborators of the Institute of Computational Mathematics and Computation (ICMC), mainly to the understanding of Professor Adenilso Simão, and of the Institute of Physics of São Carlos (IFSC).

I thank my colleagues of the Scientific Research Group (SCG) who helped me a lot and taught me from the most complex concepts to the good things of everyday life. I thank each one of you for the inspiration you are to me. You are incredibly intelligent, good people and true researchers.

I thank my family who understood that study is my passion! I thank the the family that I built in São Carlos. Several brothers and sisters at heart that I will always have a great affection and always be available for whatever comes and goes. Beside sharing a place to live, we also shared great moments of joy, discoveries, and many laughs. I will take São Carlos and all the people I had the honor of meeting in this city deep in my heart.

I thank the dear Prof. Sérgio Mascarenhas and Guilherme Rosso who helped me immensely and are my heroes of Brazilian science. Guilherme Rosso introduced me to Professor Sérgio and Prof. Sérgio opened the doors of Braincare so that I could carry out many of the experiments described in this dissertation. The consideration that both of you have for me and my personal story is the biggest prize I could win.

Thanks also to my beloved boyfriend, Wouter Naessens, who is the greatest discovery I made in this master's degree: true love. Thanks for the reviews, the explanations and the PCA plots. May we be together in chaos and order, in the complexity and simplicity of every day of our lives.

I thank all my professors, especially Prof. Paulo Benício and Prof. Ricardo Wagner, who have always supported me. My thanks also extend to my teachers of my dear Aratuba city, in Ceará.

Thanks also to Dr. Matthias Stevens and all the Application Prototyping Team of imec - Belgium who received me during my academic exchange with a lot of patience and today I feel

very pleased to work with you all.

So many people helped me on this journey! I feel so grateful for having crossed so many wonderful lives. That dreamy girl from the countryside of Ceará would never have grown up as a person and achieved so many dreams without the support of my co-workers, friends of school and college. To all of you who believe in me and see the sparkle of my eyes when I speak of science, from the bottom of my heart, thank you very much.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code PROEX-9524331/M.

“Chaos is order yet undeciphered.”

(José Saramago)

“Truth is ever to be found in the simplicity, and not in the multiplicity and confusion of things.”

(Isaac Newton)

ABSTRACT

ALVES, M. **Classification of pseudo-random number generators by complex networks and computational geometry analysis**. 2019. 152 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Randomness has stimulated mankind's attention and imagination ever since we began to observe the behavior of nature. It was by understanding the randomness and patterns that humans learned to control crops, for example, which led to the creation of the first communities. In the modern world, with the aim of mimicking the randomness of natural phenomena, computers are used to generate sequences that look as random as possible running pseudo-random number generators (PRNG). PRNGs have diverse applications in information security, digital games, simulations, modeling, gambling, arts, among others. Despite this fact, existing methods of randomness measure do not offer a definitive solution to the need of classifying PRNGs. The main methods of randomness measurement consist of statistical tests through which the sequences generated by the algorithms are analyzed. Once PRNGs are analyzed by these test suits, they are considered (un)satisfactory random. This paper explores an aspect that has been neglected in statistical tests: the spatial distribution of pseudo-random sequences. It is conjectured that this distribution is the source of undisclosed patterns in test suites. One way to study this arrangement more depth is using models that explore the relation of values and iterations. The relation of elements in space has to be the basis of the paradigm. This work applies theory of graphs and computational geometry methods to find patterns in pseudo-random sequences. The analyzed sequences are plotted in a Cartesian plan generating a set of points that are converted into graphs considering the Euclidean distance between the points within a radius. The best combination of descriptors formed by measurements of graphs and geometric properties is selected. When patterns emerge, one can point out flaws in the widely used methods for measuring PRNGs quality. It is intended to suggest a complementary approach for the evaluation of PRNGs, contributing to a better classification of the PRNGs and, consequently, to cause improvements in the studies on information security. This includes identifying patterns in sequences generated by algorithms that are considered pseudo-random by current statistical tests and thus identifying the limitations of these assessments.

Keywords: Randomness, Complex networks, Computational geometry, Complex systems.

RESUMO

ALVES, M. **Classificação de geradores de números pseudoaleatórios aplicando análise de redes complexas e geometria computacional**. 2019. 152 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

A aleatoriedade tem estimulado a atenção e a imaginação da humanidade desde que começamos a observar o comportamento da natureza. Foi entendendo a aleatoriedades e padrões, que os humanos aprenderam a controlar colheitas, por exemplo, o que levou a criação das primeiras comunidades. No mundo moderno, com o objetivo de imitar a aleatoriedade dos fenômenos naturais, os computadores são usados para gerar sequências que sejam as mais aleatórias possível executando os geradores de números pseudoaleatórios (PRNGs). A pseudoaleatoriedade possui diversas aplicações em segurança da informação, jogos digitais, simulações, modelagem, jogos de azar, artes, entre outros. Apesar disso, os métodos existentes de medida de aleatoriedade não oferecem uma solução definitiva para a necessidade de classificar PRNGs. Os principais métodos de medida de aleatoriedade consistem em testes estatísticos através dos quais as sequências geradas pelos algoritmos são analisadas. Uma vez que PRNGs são analisados por estes testes, eles são considerados ou não satisfatoriamente aleatórios. Este trabalho explora um aspecto que tem sido negligenciado nos testes estatísticos: a distribuição espacial das sequências pseudoaleatórias. É conjecturado que essa distribuição seja fonte de padrões não revelados nas suítes de teste. Uma maneira de estudar esse arranjo com mais profundidade é usando modelos que exploram a relação de valores e as iterações. A relação dos elementos no espaço deve ser a base do paradigma. Este trabalho aplica teoria de grafos e métodos de geometria computacional para encontrar padrões em sequências pseudoaleatórias. As sequências analisadas são plotadas em um plano Cartesiano gerando um conjunto de pontos que são convertidos em grafos considerando a distância euclidiana entre os pontos dentro de um raio. A melhor combinação de descritores formados pelas medidas de grafos e propriedades geométricas é selecionada. Quando os padrões emergirem, pode-se apontar falhas nos métodos amplamente utilizados para classificação de PRNGs. Pretende-se sugerir uma abordagem complementar para a avaliação de PRNGs, contribuindo para uma melhor classificação dos PRNGs e, conseqüentemente, causar melhorias nos estudos sobre segurança da informação. Isso inclui identificar padrões em sequências geradas por algoritmos que são considerados pseudoaleatórios pelos testes estatísticos atuais e, assim, identificar as limitações dessas avaliações.

Palavras-chave: Aleatoriedade, Redes complexas, Geometria computacional, Sistemas complexos.

LIST OF FIGURES

Figure 1 – Chinese trigrams	28
Figure 2 – Rule 30	36
Figure 3 – Random Networks	50
Figure 4 – Small-world model in relation to regular and random networks	51
Figure 5 – Scale-free network	53
Figure 6 – Structural networks	54
Figure 7 – Descartes' Solar System and its environs	56
Figure 8 – New Burlington Street, London, England, 1855	57
Figure 9 – Basic Voronoi diagram	58
Figure 10 – Voronoi tessellation application	59
Figure 11 – Basic Delaunay triangulation	60
Figure 12 – Delaunay diagram application	61
Figure 13 – Plot of 1000 points generated by the k-Logistic Map with $K0$	65
Figure 14 – Radius choice	67
Figure 15 – Plot generated by connecting the points in figure 13 with $r = 0.12$	68
Figure 16 – Graph generated using K-logistic map; $K = 0$ and $r = 0.12$	68
Figure 17 – Plots originating from the K-logistic map with $K0$	70
Figure 18 – Plots originating from the K-logistic map with $K4$	70
Figure 19 – Degree distribution	72
Figure 20 – Histogram of the edge lengths studied in Gastner and Newman (2006)	73
Figure 21 – Histogram of the K-logistic Map networks edge lengths	73
Figure 22 – Voronoi diagram originating from K-logistic Map	75
Figure 23 – Voronoi diagram originating from the K-logistic Map colored with 5 colors	77
Figure 24 – Analysis of Voronoi diagram for 5 colors	77
Figure 25 – Delaunay triangulation originating from the K-logistic Map	78
Figure 26 – Delaunay triangulation originating from the K-logistic map colored with 90 colors	80
Figure 27 – Analysis Delaunay triangulation for 5 colors	80
Figure 28 – Complex network-based approach for boundary shape analysis	81
Figure 29 – Complex network-based approach for texture analysis	83
Figure 30 – Summary of the visibility graph method	84
Figure 31 – Degree distribution of K-logistic map by K value	88
Figure 32 – Degree distribution for band	89

Figure 33 – Distribution of isolated and leaves nodes in K-logistic map by band	91
Figure 34 – Analysis of the 1000 points networks descriptors	102
Figure 35 – PCA plot of networks with 1000 points	102
Figure 36 – Analysis of the 2000 points networks descriptors	103
Figure 37 – PCA plot of networks with 2000 points	103
Figure 38 – Analysis of the 3000 points networks descriptors	104
Figure 39 – PCA plot of networks with 3000 points	104
Figure 40 – Analysis of the 3000 points networks with a reduced number of descriptors .	105
Figure 41 – PCA plot of the 3000 points networks with a reduced number of descriptors	105
Figure 42 – Voronoi Diagram analysis of networks with 5000 points	107
Figure 43 – Importance of the first PC within the PCA using Voronoi diagrams	108
Figure 44 – Delauney triangulation analysis of networks with 5000 points - PCAs	108
Figure 45 – Delaunay triangulation analysis	109
Figure 46 – Stages of the applied texture method for colored images	110
Figure 47 – PCA for boundary shape analysis	112
Figure 48 – Mersenne 64bits	135
Figure 49 – ALFG	135
Figure 50 – BlumBlumShub	135
Figure 51 – BC	135
Figure 52 – Java	136
Figure 53 – JDK	136
Figure 54 – Linear Congregational Generator	136
Figure 55 – Linear Congregational Generator	136
Figure 56 – Mersenne Twister	136
Figure 57 – Cellular Automata Rule 30	136
Figure 58 – Degree distribution for band of the k-logistic map PRNG	145
Figure 60 – Leaves and isolated nodes distribution for band	147
Figure 62 – Box-plots of PCA on Delaunay triangulation using different color schemes .	149
Figure 64 – Box-plots of PCA on Voronoi diagrams using different color schemes	151

LIST OF TABLES

Table 1 – Cellular Automaton Rule 30	36
Table 2 – Cellular Automaton Rule 30	94
Table 3 – Complex network attribute evaluation	99
Table 4 – Complex Network measures	100
Table 5 – Voronoi Diagram to K5, K6, K7, K8 and K9	111
Table 6 – Delaunay Triangulation to K5, K6, K7, K8 and K9	111
Table 7 – Visual Graph classification	112
Table 8 – Complex network-based approach for boundary shape analysis	113
Table 9 – Descriptions of the NIST Statistical Test Suite	137
Table 10 – Descriptions of the DIEHARD tests	139
Table 11 – Average number of files that passed DIEHARD tests using the k -logistic map PRNG from 100 file samples. Severely failed tests are shown in bold. All tests passed using the interval $0.0001 < \text{p-value} < 0.9999$	141
Table 12 – Number of files that passed the NIST test suites for the k -logistic map. Failed tests are shown in bold. All the tests passed to the $\alpha = 0.01$ significance level.	142
Table 13 – K-logistic Map networks classification	143

CONTENTS

1	INTRODUCTION	23
1.1	Objectives	24
1.2	Organization	25
2	PSEUDO-RANDOM NUMBERS GENERATORS	27
2.1	Categories of Randomness	30
2.1.0.1	<i>True Random Number Generators</i>	30
2.1.0.2	<i>Pseudo-random Number Generators</i>	31
2.2	PRNG Algorithms	32
2.2.1	<i>k-logistic map</i>	32
2.2.2	<i>Linear Congruential Generator</i>	33
2.2.3	<i>Merssene Twister</i>	33
2.2.4	<i>Lagged Fibonacci generators</i>	34
2.2.5	<i>BlumblumShub</i>	34
2.2.6	<i>Bailey-Crandall</i>	35
2.2.7	<i>Cellular Automaton Rule 30</i>	36
2.3	Applications of randomness	37
3	PRNG TESTING	39
3.1	Randomness requirements	40
3.2	Statistical Tests	40
3.3	Statistical Test Suites	41
3.3.1	<i>NIST</i>	41
3.3.2	<i>DIEHARD</i>	41
3.3.3	<i>TestU01</i>	42
3.3.4	<i>Ent</i>	42
3.3.5	<i>Other approaches</i>	43
3.4	Limitations	43
4	COMPLEX NETWORKS	45
4.1	Basic definitions and measurements	46
4.2	Networks Models	49
4.2.1	<i>Random Networks</i>	49
4.2.2	<i>Small-World Networks</i>	50

4.2.3	<i>Scale-Free Networks</i>	52
4.3	<i>Spatial Networks</i>	53
5	COMPUTATIONAL GEOMETRY	55
5.1	<i>Voronoi diagram</i>	55
5.1.1	<i>Generic Definition of Voronoi Diagram</i>	58
5.2	<i>Delaunay triangulation</i>	59
5.2.1	<i>Generic Definition of Delaunay Triangulation</i>	60
6	METHODOLOGY	63
6.1	<i>Complex Networks Approach</i>	64
6.1.1	<i>Network conception</i>	64
6.1.2	<i>Selection of PRNG for the first experiments</i>	69
6.1.3	<i>Complex Networks visualization</i>	69
6.1.4	<i>Fundamental measurements</i>	71
6.2	<i>Computational Geometry Approach</i>	74
6.2.1	<i>Voronoi Diagrams</i>	75
6.2.2	<i>Delaunay Triangulation</i>	78
6.3	<i>Other inspected techniques</i>	81
6.3.1	<i>Boundary shape analysis</i>	81
6.3.2	<i>Texture analysis</i>	82
6.3.3	<i>The visibility graph</i>	83
6.4	<i>Discussion</i>	85
7	RESULTS	87
7.1	<i>Complex Networks Analysis</i>	87
7.1.1	<i>Main measurements</i>	88
7.1.2	<i>Experiments scope</i>	92
7.1.2.1	<i>Comparing networks</i>	93
7.1.2.2	<i>Connected components</i>	93
7.1.2.3	<i>Classification setup</i>	94
7.1.2.4	<i>Number precision</i>	95
7.1.3	<i>Network measurements</i>	95
7.1.3.1	<i>ML algorithms selection</i>	97
7.1.3.2	<i>Measurement selection</i>	97
7.1.3.3	<i>PRNG analysis as a classification problem</i>	100
7.2	<i>Computational Geometry Analysis</i>	105
7.2.1	<i>Voronoi Diagrams for PRNG testing</i>	106
7.2.2	<i>Delaunay triangulation for PRNG testing</i>	108
7.3	<i>Randomness classification with other techniques</i>	110

7.4	Reproducibility of experiments	114
7.4.1	<i>Classification of descriptors</i>	115
7.4.2	<i>Experiments configuration</i>	115
7.4.2.1	<i>Complex Network analysis selected libraries</i>	116
7.4.3	<i>Technical challenges</i>	116
8	CONCLUSIONS	119
8.0.1	<i>Perspectives for future work</i>	122
BIBLIOGRAPHY		125
ANNEX A	PRNG-BASED GRAPHS	135
ANNEX B	NIST	137
ANNEX C	DIEHARD	139
ANNEX D	K-LOGISTIC MAP TESTING	141
ANNEX E	CLASSIFICATION ALGORITHMS TEST	143
ANNEX F	DEGREE DISTRIBUTION FOR BAND OF THE K-LOGISTIC MAP PRNG	145
ANNEX G	LEAVES AND ISOLATED NODES DISTRIBUTION FOR BAND	147
ANNEX H	BOX-PLOTS OF DELAUNAY TRIANGULATION	149
ANNEX I	BOX-PLOTS OF VORONOI DIAGRAM	151

INTRODUCTION

It is inconceivable to imagine the contemporary world without the application of pseudo-random numbers and the mathematical tools that have arisen around these numbers. Pseudo-random numbers along with prime numbers have become the basis of Internet banking, for example. The use of such numbers in statistics has expanded beyond random sampling or random assignment of treatments to experimental numbers. More common uses now are in simulation studies of stochastic processes (simulation), and analytically intractable mathematical expressions. In engineering and natural sciences, simulations applying pseudo-random sequences are used extensively in studying physical and biological processes ([GENTLE, 2003](#)).

A process is random if the known conditional probability of the next event, given the previous history, is no different from the known unconditional probability. The digital computer cannot generate random numbers, and it is generally not convenient to connect the computer to some external source of random events. Due to technical restrictions of generating random numbers, the use of the so-called pseudo-random sequences has been adopted. The sequences are deterministic but look like they were generated randomly. They are generated by algorithms called pseudo-random numbers generators (PRNGs). Many methods have been suggested for generating such pseudo-random numbers ([GENTLE, 2003](#)).

The good quality of PRNGs is the base of a whole universe created around the use of pseudo-random numbers. An extreme effort is made to ensure that PRNGs produce good sequences, so that sequences can supply the needs of various applications in science and business. The accuracy of the quality of the PRNGs is greater when they are proposed for use in cryptography. ([YASCHENKO, 2002](#)).

There are many different types of testing for the quality measurement, most of them based on the entropy of the sequences. The question raised in this master was whether these PRNGs tests were enough to ensure that the sequences tested do not have patterns. The statistical tests are limited to gather evidence that a generator indeed produces numbers that appear to be

random. Another question is whether these assessments analyze topics such as the temporal and spatial distribution of numbers. It would be of substantial impact if some pattern in sequences generated by PRNGs were identified considering characteristics not contemplated by the tests in large use.

The first step was to identify a set of possible characteristics that are not explored (or are marginally analyzed) in the most well-known PRNG tests. What can be observed is that properties related with spacial distribution of pseudo-random sequences deserves more attention. If one creates a plot of the time (or step) in which each value was generated and all the numbers in the sequence, a set of points in a Cartesian plane is created. From this set of points, this work proposes the PRNG analysis in two manners: (i) converting the points into complex networks by establishing a radius in which all points within that radius are connected to each other. The result is a network whose geometric structure is lost but characteristics related to the number of neighbors and paths in the network can be better exploited and (ii) using the points as the basis for the creation of polygons using computational geometry methodologies. Thus, the topological feature of the network is not lost. A spatial analysis of the points' distribution can be made over the polygons resulting from the meshes creation in computer graphics. For those reasons, the aim of this master degree is finding patterns in the PRNG generated sequences by the analysis of measurements of complex networks and computational geometry.

Complex networks have been useful for several studies covering important issues ranging from the Internet to genetic networks that determine our biological existence ([BARABÁSI, 2016](#)). Many complex problems have been modeled and better understood when simulated in complex networks. The problem addressed in this master's research can also be modeled in complex networks. It has been demonstrated that complex networks measures can be useful to better understand the relationship between the values generated by PRNGs.

1.1 Objectives

- Pointing out limitations of the current tests of PRNGs. Once this work is able to point to patterns in sequences approved in the current pseudo-random test, it can be said that these tests are incomplete because they do not consider the time series of sequences and the network structures created from them.
- Developing complementary tests for the PRNG. The objective is to show that because of the incompleteness of the randomness tests, it is necessary that more tests are added. It is intended to create an additional test that can improve the classification of PRNGs and thus raise the difficulty of predicting the behavior of the sequences generated by the better algorithms.
- Presenting an in-depth study of the characteristics of pseudo-random sequences. This

dissertation thoroughly analyzes the characteristics of a PRNG based in the logistic map. We try not only to present the results of the statistical tests but also to observe how the PRNG generates different groups of sequences with different parameters.

1.2 Organization

The organization of this text is as follows. In chapter 2 there is a succinct resume of randomness study history and emergence of pseudo-randomness. This chapter lists the most important concepts in this field and it describes some PRNGs algorithms. The current methodological approaches for measuring PRNGs performance and some constraints of these tests are in chapter 3. The chapter 4 presents the complex network topic. The complex network distinct models and its main measurements are depicted. Chapter 5 introduces the computational geometry techniques employed in this proposal. This chapter explains and exemplifies the Voronoi Diagram and the Delaunay Triangulation. Chapter 6 announces the proposed methodologies. Chapter 7 describes the results and setup for the experiment reproducibility. Finally, chapter 8 discusses the main results and perspectives for further research development.

PSEUDO-RANDOM NUMBERS GENERATORS

The emergence of apparently random events in nature has instigated the pursuit of the explanation about its origins. Primitive man was certainly appalled at events such as eclipses and meteor showers. Some cultures even associated divine meaning to such events. The mankind has been observing changes in landscapes and has learned to recognize patterns and establish new lifestyles. Over the centuries, many of the events that seemed to happen at random have been mapped and predicted, and events that are actually random are actually hard to detect.

The adoption of the randomness in daily activities begun to be documented approximately in the century III BCE.

According to ancient Chinese mythology, in the beginning of time, there was the great emperor Fu Xi who ruled the world, with a human face and a body of a snake. One day while observing the Heavens and the Earth, he became inspired and began drawing patterns and markings. He called these patterns the Eight Trigrams. According to historians, it is believed that Fu Xi was a human being during the early patriarchal society in China near mid-2800 BCE , and there is evidence of his legend having roots in the Neolithic period in China (8500-2070 BCE) making him one of the first people to think about the concept of randomness in history.[...] The trigrams play a major philosophical role in Chinese culture. A trigram is formed by three lines which are continuous or discontinuous, and are also known as Yin and Yang. Then, the two trigrams can form a hexagram, which contains another 64 possible hexagrams. The hexagrams is a key concept in the I Ching, also known as the Book of Changes, and is one of the oldest and most reviewed oracles even in the present day. This oracle works by randomly generating a hexagram from which interpretations allow for predictions of your own fortune. (CHAN, 2016)

Figure 1 – Chinese trigrams

Source: [CHAN \(2016\)](#).

Note – Patterns and markings of the Chinese trigrams, one of the oldest records of use of randomness in daily life.

After the mystic use cited above, the use of randomness is recorded in games with the use of cubic dice. The dice has its origins in ancient Mesopotamia and in the region that is now Pakistan around 2750 BCE. Around 1320 BCE, the Egyptians, the Greeks and then Romans used dice as well. The effects of randomness fascinated the Greeks and they started assigning god's names to the different combinations of numbers in a throw. "The American Indians had devices such as two-sided dice made out of bones, wood, seeds, beaver teeth, claws, walnut shells, or stones. While the Papago Indians used bison vertebrates as two-sided dice" ([CHAN, 2016](#)).

In the seventeenth century, randomness applied in probability and gambling caught the attention of Girolamo Cardano (1501-1576) who wrote the *Liber de Ludo Aleae*. This book is the first theoretical work on field and contains rules for games and advices about how to play them. "This simple work would later evolve into a broader area of research around the understanding of patterns in the random numbers of the games, propelling the very beginnings of quantitative sciences during the Renaissance" ([BELLHOUSE, 2005](#)). Other renowned scientists such as Galileo Galilei (1564-1642), Blaise Pascal (1623-1662), Pierre de Fermat (1601-1665), Antoine Gombaud (1607–1684), and Christiaan Huygens (1629 – 1695) published very important studies on probability theory within the academic universe ([MELO; FONSECA, 2000](#)).

In the following two centuries, there was a great increase in the number of publications. "In 1713, Jakob Bernoulli and Abraham de Moivre introduced the concepts of sample average and independence. Also, they started the process of systematizing the probability, leaving aside the gambling context" ([VIALI, 2008](#)). By the 1800s, John Venn (1834-1923) and others scholars noted the apparent randomness of the digits of π , but they did not formulate a more accurate study about it ([WOLFRAM, 2002](#)).

In the twentieth century, the study of uncertainty was widespread and began to connect

to the study of physics and chemistry, and developing into a more sophisticated field (CHAN, 2016). In 1933, the Russian mathematician Andrei Kolmogorov(1903-1987) formalized the study of probability theory (KHRENNIKOV, 2009). Randomness was definitely established both in the academic literature and in industry with Claude Shannon's (1916-2001) 1948 paper, being fundamental to the development of information theory (AFTAB *et al.*, 2001).

However, it was with the availability of computers that the PRNGs began to be developed. "The advent of the high-speed computer raised the possibility of generating pseudo-random numbers directly on the computer" (GRINSTEAD; SNELL, 2012). In 1948, Derrick Henry Lehmer (1905 - 1991) proposed a linear congruential generator, also known as Lehmer generator (ECKHARDT, 1987). Another primary pseudo-random algorithm is the middle square method. John von Neumann (1903 - 1957) proposed it in the 1951. Therefore, these generators are also called von Neumann generators in the literature. The middle square method consists of taking the square of the previous random number and to extract the middle digits. This method gives rather poor results since generally sequences tends to get into a short periodic orbit. The implementation of the first pseudo-random algorithms continued during the 1950s.

In the early 1960s, the notion of algorithmic randomness was introduced by Gregory Chaitin(1947), Kolmogorov, and Ray Solomonoff(1926-2009) (WOLFRAM, 2002). In that decade the PRNG RANDU was created. This PRNG fails most randomness tests and is considered a bad PRNG. Despite this, it was one of the most widely used algorithms in the world for many years and the generator used in Microsoft VisualBasic 6.0 (GENTLE, 2003) (L'ECUYER; SIMARD, 2007). In 1965, Tausworthe (1965) created a pseudo-random number generator based on linear recurrence relations. This algorithm started the development of the so-called Feedback Shift Register (FSR). The study of pseudo-random numbers drastically accelerated in the 1970s with the rise of computer science advancing algorithm development (CHAN, 2016). In the same decade, Theodore Gyle Lewis (1941) and W. H. Payne extended the Tausworthe (1965) method and created the Generalized Feedback Shift Register (GFSR). "In the 1980s, however, work on cryptography had led to the study of some slightly weaker definitions of randomness based on inability to do cryptanalysis or make predictions with polynomial-time computations" (WOLFRAM, 2002). Still in the 80's, the BlumBlumShub(BLUM; BLUM; SHUB, 1986) and the cellular automaton(WOLFRAM, 1987) based random number generators were published.

In many fields outside of statistics, the idea that block frequencies were somehow the only ultimate tests for randomness persisted until the 1990s (WOLFRAM, 2002). During that decade, very important algorithms such as Mersenne Twister (1997) were published. Other approaches have been used as the creation of PRNGs based on chaotic iterations as the method proposed by Baptista (1998).

These first two decades of the twenty-first century have been very promising for the emergence of increasingly robust PRNGs as L'Ecuyer MRG32k3a Combined Recursive (FCALL, 2013) and Bailey-Crandall both launched in 2002. It is observed a growth in the quantity of

algorithms based on dynamic systems such as the k-logistic map (2016) and the public-key encryption algorithms suggested by [Kocarev *et al.* \(2004\)](#), [Szczepanski *et al.* \(2005\)](#) and [Masuda *et al.* \(2006\)](#).

2.1 Categories of Randomness

The origins of randomness are divided in two large groups: the true random numbers generators and the pseudo-random number generators. The true random numbers have some natural phenomena as source of randomness, however, they are highly complex and expensive to use on a large scale. Computers, on the other hand, can create an approximation of randomness that is useful for various applications. The following subsections briefly describe the main characteristics of the type of random numbers generators.

2.1.0.1 True Random Number Generators

"A true random number generator (TRNG) is a nondeterministic source to produce randomness" ([STALLINGS, 2008](#)). The main characteristics of truly random sequences is no dependencies among consecutive events and known distribution. The implementation details of TRNG solutions are mostly not published because its commercial value ([SCHELLEKENS; PRENEEL; VERBAUWHEDE, 2006](#)).

This kind of generator is constructed over the observation of unpredictable processes like air turbulence of disk drives ([DAVIS; IHAKA; FENSTERMACHER, 1994](#)), pulse detection of ionize radiation events, gas discharge tubes, leaky capacitors ([STALLINGS, 2008](#)), or a substance undergoing atomic decay ([GENTLE, 2003](#)). They are also known as hardware random number generators. Real-world physical processes are believed to be random in nature to create streams of numbers. Decay events from a radioactive source, for example, are random and uncorrelated with each other, atmospheric noise can also be used ([BROCK, 2016](#)). In the following quotation about a TRNG based in particle decay, one can observe the complexity of these generators.

The subatomic particles comprising the decaying substance transmute into other particles at random points in time. At a macro level (that is, given an amount of the substance that contains a very large number of atoms), both theory and empirical observations suggest that there are no dependencies among consecutive events and that the process is constant over sufficiently short time intervals.[...] If we can measure the times between the events, and if the process is stationary, we can form a random variable with a known distribution ([GENTLE, 2003](#)).

Some current TRNGs are based on quantum theory. Quantum theory is intrinsically random being thus the ideal base for a physical random number source ([STEFANOV *et al.*, 2000](#)). The scientific community has already witnessed the implementation of a quantum random

number generator in 2018 at the NIST's labs. These numbers are binary sequence that were generated by entangled photons using a laser (BIERHORST *et al.*, 2018).

The true random number generators using hardware are likely to be slow. In addition, they require the user involvement, they are difficult to implement, and they make assumptions about the hardware that are not guaranteed. For some applications, it is not necessary to have true random number sequences. In this case, computers are used to generate sequences by software that is approximation of the true random numbers (JUNOD, 1999).

2.1.0.2 Pseudo-random Number Generators

PRNGs are deterministic generators that can be implemented in ordinary computer programs (SIDORENKO; SCHOENMAKERS, 2005). The output of these algorithms is sequence of numbers that approximates the properties of true random numbers. It is expected that pseudo-random sequences be very difficult for an attacker to distinguish from true random numbers. Such deterministic generator uses the previous k numbers (often just the single previous number) to determine the next number. The number of previous numbers used to generate a new one, k , is called the **order** of the generator. The set of values at the start of the recursion is called **seed** (SIDORENKO; SCHOENMAKERS, 2005). Each time the recursion is begun with the same seed, the same sequence is generated. Because the set of numbers directly representable in the computer is finite, the sequence will repeat. The maximum length of the sequence before it begins to repeat is called **period or cycle length** (GENTLE, 2003).

The sequences generated by a PRNG cannot be discriminated from random sequences of the same length by a polynomial time probabilistic algorithm (YASCHENKO, 2002). In computational simulation, digital games, for example, these numbers merely need to be reasonably random and have good statistical properties. On the other hand, in cryptography applications, they must be indistinguishable from real random numbers, even for observers with large amount of computational resources (SIDORENKO; SCHOENMAKERS, 2005).

PRNGs are important in practice for simulations, and are central in the practice of cryptography. Common classes of these algorithms are Linear Congruential Generators (LCG), Lagged Fibonacci Generators, and Generalized Feedback Shift Registers. The PRNGs are largely used in computer systems because their efficiency in terms of producing many number in short time, their technological inexpensiveness and their facility to be implemented (MACHICAO; BRUNO, 2017).

Among pseudo-random sequence applications, it is important to mention the **quasi-random sequences**. "Quasi-random sequences take advantage of a group of mathematical algorithms producing low-discrepancy sequences" (SHAHBAZI; TAPPENDEN; MILLER, 2013). Low discrepancy (quasirandom) sequences are used in numerical integration, simulation and optimization. Like pseudo-random numbers they are uniformly distributed but they are not statistically independent, rather they are designed to give more even distribution in multidimensional

space (uniformity) (MASUDA *et al.*, 2006). They are a finite collection of numbers that are meant to be representative of a sample space. In statistical applications there a requirement that the analyzed samples be representative of the whole sequence (GENTLE, 2003).

2.2 PRNG Algorithms

Since pseudo-random number generator algorithms became part of the production of academic research, many types of algorithms have emerged. Some PRNGs are based on modular reduction, feedback shift register, cellular automaton, chaotic systems, among others. The literature review pointed out some models of these types of algorithms. They are explained below.

2.2.1 *k*-logistic map

The *k*-logistic map (MACHICAO; BRUNO, 2017) is a PRNG based on chaotic maps. This algorithm takes advantage of the digits precision of a chaotic system equation. It was proven that the *k*-digits to the right from the decimal separator of a given point from the chaotic map result in a rapid randomization. This algorithm is based on the logistic map equation.

$$x_{t+1} = f(x_t) = rx_t(1 - x_t) \quad (2.1)$$

where $r \in [0, 4]$, $x \in [0, 1]$ and t is the discrete time step. The chaotic behavior of the logistic map emerges when its parameters are configured in specific ranges. The r parameter must to be 3.56995 so the equation reaches the onset of chaos. The *k*-logistic map adopted $r = 3.99999$. As the logistic map goes through iterations, the number of decimal places of x grows. The position of these decimal places is called k . Thus, it is possible to create a generalized approach to analyze the *k*-right digits of precision of the logistic map. This is the basic definition of the *k*-logistic map.

$$x_t^K = \frac{\lfloor x_t 10^{K+L} \rfloor}{10^L} - \lfloor x_t 10^K \rfloor \quad (2.2)$$

where L is the length of digits of precision of the new point x_t^K . It is necessary to specify which position of the decimal places will be used. For example, to construct the sequence with the first digit of the decimal places (from left to right), then K must equal 0. To use the second digit, then $K = 1$, and so on. The *k*-logistic map with the $K = 4$ has passed successfully the DIEHARD and NIST tests.

This paper explores the deep-zoom properties of the chaotic *k*-logistic map, in order to propose an improved chaos-based cryptosystem. This map was shown to enhance the random features of the Logistic map, while at the same time reducing the predictability about its orbits. We incorporate its strengths to security into a previously published cryptosystem to provide an optimal pseudo-random number generator (PRNG) as its core operation. The result is a reliable method that does not have the weaknesses previously reported about the original cryptosystem.

The logistic map has increased the randomness of a published PRNG that uses the logistic map as its basis (MACHICAO *et al.*, 2019). This because the logistic map based algorithms often use the first decimal places of numbers. The k-logistic map proves that randomness is actually in the least significant decimal places of the number in the chaos set.

From now on, the text adopts K in capital letter to refer to the k position of the logistic map to not be confused with the k of node degree in complex networks.

2.2.2 Linear Congruential Generator

The linear congruential generator (LCG) is a very common class of PRNG used by many programming languages. The following recurrence relation defines it:

$$X_{n+1} = (aX_n + c) \mod m \quad (2.3)$$

where X is the number in a pseudo-random sequence. The m parameter is the modulus, a is the multiplier and c is the increment. The quality of the sequence randomness is based on the mathematical relation of m , a , and c .

Derrick Lehmer (1905-1991) made the first description of this method in 1948. The behavior of the LCG depends greatly on the exact choice of a parameter (WOLFRAM, 2002). "LCGs are fast and easy to implement. Unfortunately, this class of generator is subject to a number of defects making it unsuitable for simulations or cryptography" (PERSOHN; POVINELLI, 2012). A well-known LCG example is the program RANDU. This software was once considered the most extensively used PRNG in the world (GENTLE, 2003). The **java.util.Random** class of the Java programming language uses an adaptation of the LCG. This class was also analyzed over the initial period of the experimentation phase of this master.

2.2.3 Merssene Twister

Makoto Matsumoto and Takuji Nishimura published the Mersenne Twister algorithm in 1997. Mersenne twister random numbers have the colossal period of $2^{19937} - 1$. "They are proven to be equidistributed in (up to) 623 dimensions (for 32-bit values), and can be generated faster than other statistically reasonable generators" (TIAN; BENKRID, 2009). The Mersenne Twister algorithm is based on the following recurrence:

$$x_i = x_{i-p} \oplus Ax_{i-p+q} \quad (2.4)$$

where $\oplus$ represents the *exclusive – or* operation. "The Merssene Twister requires a rather complicated initialization procedure" (GENTLE, 2003). It is also considered in this research the **Merssene Twister64**. The Merssene Twister64 is similar to the original one, but implemented using 64-bit registers.

2.2.4 Lagged Fibonacci generators

The Lagged Fibonacci generators are algorithms based on Fibonacci sequences. The algorithms are specified by the recurrence:

$$X_k = X_{k-p} \otimes X_{k-p+q} \mod m \quad (2.5)$$

where $\otimes$ denotes the operation which could be any of $+$, $-$, $*$, or $\oplus$. $m = 2^l$ (machine word size) for generating l bit random numbers. p is called the lag of the generator. The seed for these generators is the first p random numbers. For the multiplicative generator, the random numbers are odd numbers mod m . For the other generators, the random numbers are integers mod m (PETERSEN, 1994).

All the lagged Fibonacci generators have numerous advantages over the linear congruential generators. One distinguishing feature is their relatively large period when compared to the linear congruential generators. The period can be made large by choosing an appropriately large value of p , with the same word size. Therefore, the period is not limited by the word size of the machine on which the generator is implemented. With extremely long periods, they are generally faster than linear congruential generators and have excellent statistical properties (ALURU, 1997). The **Additive Lagged Fibonacci Generators (ALFG)** (the $+$ operation is choose) and the **Multiplicative Lagged Fibonacci Generators (MLFG)** (the $*$ operations is choose) were included in the initial experiments of this project.

2.2.5 BlumblumShub

BlumBlumShub was proposed in 1986 by Lenore Blum (1942), Manuel Blum (1938) and Michael Shub (1943). This PRNG is cryptographically secure in which the prediction of the next bit is computationally equivalent to integer factorization. This generator is unpredictable under certain assumptions (GENTLE, 2003). The first concept towards the BlumblumShub algorithm understanding is the Blum prime number definition:

Definition 1: A prime number p with $p \equiv 3 \pmod{4}$ is called a Blum prime number (JUNOD, 1999).

In the light of this concept, it is possible to characterize the BlumblumShub algorithm adapted from Junod (1999):

- I. Generations of two big Blum prime numbers, p and q
- II. Defines $n = p * q$
- III. Choose the random seed S , $s \in [1, n - 1]$
- IV. $x_0 = s^2 \mod n$

$$\text{V. } x_i = x_{i-1} \bmod n$$

$$\text{VI. } z_i = \text{parity}(x_i)$$

The output of *parity()* function is 1 if the number is odd and 0 if the number is even. The algorithm output is the binary sequence defined by z .

Despite being considered the low speed generator, this algorithm continues to be applied in different authentication and encryption processes. The BlumBlumShub has been applied in tasks where the generation rate is crucial. For example, the generation of keys in asymmetric systems and message authentication code (VYBORNOVA, 2017).

2.2.6 Bailey-Crandall

The Bailey-Crandall algorithm uses high-precision floating-point calculations to generate random bits. It was proposed by David H. Bailey (1948) and Richard E. Crandall (1947-2012) in 2002. Bailey and Crandall (2002) formalized a class of number called "Normal numbers". The Normal numbers definition is:

$$\alpha_{a,b} = \sum_{k=1}^{\infty} \frac{1}{c^k b^k} \quad (2.6)$$

where b is integer and greater than 1. c is odd and co-prime to b . For the proposed algorithm, it is necessary exactly one member of this class: the $\alpha_{2,3}$. This normal number has the property that its binary expansion contains every binary string in the exactly same frequency as a "truly random sequence". This property is very convenient to the PRNG formulation.

The following steps are the procedures of the Bailey-Crandall algorithm. The k parameter is 53 because the deviation of the IEEE 64-bit floating-point numbers in $(0,1)$ contain, in their mantissas, successive 53-bit segments of the binary expansion of $\alpha_{2,3}$ (BAILEY; CRANDALL, 2002).

- I. Select the seed s in the range $[3^{33}, 2^{53}]$
- II. Compute $x = 2^{s-(3^{33})} * \lfloor \frac{3^{33}}{2} \rfloor \bmod 3^{33}$
- III. Compute a "random" 64-bit IEEE double value with:
 - Compute $x = 2^{53}x \bmod 3^{33}$
 - Return $x3^{-33}$

Each sequence value generated by this algorithm can be calculated on only one processor since each output has its starting value. Thus, the algorithm has great performance in parallel processing (BAILEY, 2004). This algorithm was also used in the early stages of this work.

2.2.7 Cellular Automaton Rule 30

In 1985, Stephen Wolfram designed the rule 30 random number generator ([WOLFRAM, 2002](#)) and patented it in 1987([WOLFRAM, 1987](#)). This PRNG is based on the one dimensional cellular automaton rule 30. In this algorithm, the rule 30 is executed in a circular array of 192 cells. It has been considered a high-speed algorithm and high-quality generator. Since its publication, the generator has become quite widely used for a variety of applications.

This projects used a Rule30 algorithm implemented in Java programming language. In this code, each cell is updated, in parallel, based on the three values: the left neighbor, itself, and the right neighbor.

Figure 2 – Rule 30

Source: [Wolfram \(2002\)](#).

Note – Description of rule 30 one dimensional cellular automaton evolution.

In numbers:

Table 1 – Cellular Automaton Rule 30

Configuration	Middle cell value
111	0
110	0
101	0
100	1
011	1
010	1
001	1
000	0

Source: [Wolfram \(2002\)](#).

The central cell is the random value. The Java implementation of this algorithm take the seed of $1L \ll 32$ and subsequent calls to `java.util.Random.next(1)` function produces a sequence of random bits.

Experiments proved that with a circular array - also called window - of 200 cell is adequate to generate good pseudo-random sequences. A large range of array sizes of is this

PRNG fail in statistics tests, but the Rule 30 is at least as good as some currently accepted PRNGs. However, this algorithm is not considered secure for encryption. It is only approved by statistical tests when it uses certain values of even window sizes ([GAGE; LAUB; MCGARRY, 2005](#)).

2.3 Applications of randomness

Pseudo-random numbers are critical in many aspects of business and scientific research. Such numbers are essential to encrypt e-mails, to digitally sign documents and for electronic payment systems ([SIDORENKO; SCHOENMAKERS, 2005](#)). The PRNGs are needed also for simulation of stochastic systems, numerical analysis and probabilistic algorithms ([L'ECUYER; SIMARD, 2007](#)) ([JUNOD, 1999](#)).

The quality of pseudo-random generators is the base to the construction of secure cryptosystems ([YASCHENKO, 2002](#)). Cryptography is generally acknowledged as the best method of data protection against passive and active fraud. Three most common cryptographic objects are: block-encryption algorithms (private-key algorithms), pseudo-random number generators (additive stream ciphers), and public-key algorithms ([MASUDA *et al.*, 2006](#)). All of these algorithms use pseudo-random numbers at some stage. A summary of the main applications of randomness can be seen below:

The notion of taking random samples as a basis for making unbiased deductions has been common since the early 1900s, notably in polling and market research. [...] Such methods became increasingly popular, especially for simulating systems like telephone networks and particle detectors that have many heterogeneous elements as well as in statistical physics.[...] And in the 1980 many such randomized algorithms were invented, but by the mid-1990 it was realized that most did not require any kind of true randomness, and could readily be derandomized and made more predictable. [...]. In cryptography randomness is used to make messages look typical of all possibilities. It is also used in roughly the same way in hashing. Such randomness must be repeatable. But for cryptographic keys it should not be. Randomness is in effect also used in a similar way in the shotgun method for DNA sequencing, as well as in creating radar pulses that are difficult to forge. The unpredictability of randomness is often useful, say for animals or military vehicles avoiding predators. Such unpredictability can also be used in simulating human or natural processes, say for computer graphics, video-games, or mock handwriting. Random patterns are often used as a way to hide regularities as in camouflage, security envelopes, and many forms of texturing and distressing. ([WOLFRAM, 2002](#))

Many Machine Learning algorithms randomly separate samples so that the algorithm can detect patterns in descriptors. Thus, random numbers are also used in selecting random samples from data sets and creating a useful test/training configuration ([HALL *et al.*, 2009](#)).

Another application of PRNGs is in the billionaire digital games market. Game designers are looking to create games that are more than just fun. Games should provide emotions and experiences for their players. The crucial factor is to make users to be engaged and always return for more challenges in the games(KUMARI; POWER; CAIRNS, 2017). An important factor in the quality of digital games is the feeling of uncertainty provided by the changing conditions in games that are the result of game rules and the application of PRNG.

Still related to games, there are gambling that have been registered since antiquity and today are also a very popular game mode on the internet and in places with electronic machines. This is where PRNGs are applied. The randomness evident in chance games attracts and challenges the cognitive abilities of players(KARLSEN, 2010). Designers of these types of games often lead the player to believe that the randomness of the game can be reproduced leading the player to believe that they can discover the "secret" of the game the more games they play.

With so many important applications, PRNGs must be thoroughly analyzed before they can be used in industry. The following chapter discusses the main PRNG quality measurement algorithms.

PRNG TESTING

Over the past 80 years, the society has witnessed the evolution and growth of information technology. The computers, which were exclusive pieces of large companies and universities, became popular and today is fully inserted in the day to day of the modern world. In the last decades, there was a great technological revolution with the creation of the World Wide Web (WWW) and the use of the internet for non-military purposes.

The number of devices connected to the Internet in the present days has increase data in transit and data storage. One concern about this huge amount of data is security. There are several applications with serious privacy restrictions. Some examples are computer programs such as clinical online history, bank account data, and systems that must guarantee the right of confidentiality. On the other hand, the development of applications that access sensitive data is not accompanied by the capacity to ensure maximum security of such data. Along with the electronic systems, the challenge of protecting the data that transit in these systems has arisen. The challenge is to secure this huge data flow, which makes evaluating the algorithms that protect them a demanding task (OLIVEIRA *et al.*, 2015). In this scenario, PRNGs are critical.

According L'Ecuyer and Simard (2007), the evaluation of the PRNGs has to mainly test the following points:

- Successive output values of the RNG, say $u_0, u_1, u_2, \dots$, are independent random variables from the uniform distribution over the interval $(0, 1)$.
- Verify if the sequence being tested is independent random bits, each taking the value 0 or 1 with equal probabilities independently of the others.

In the case of PRNGs to cryptology-related applications and for gambling machines in casinos, a third requirement must be tested:

- Guarantee the unpredictability of the forthcoming numbers in a sequence.

Generally, two types of test are addressed for PRNGs: theoretical tests and empirical (statistical) tests. Theoretical tests analyze the mathematical properties of the sequences. The empirical tests use statistical techniques to evaluate if the distribution of a set of experimental data adheres to a uniform distribution. The objectives of these tests are to analyze if the sequences generated by PRNGs generate standards or any regular behavior. The scientific literature highlights three suites of statistical tests: NIST, DIEHARD and TestU01. These tests formulate hypotheses to verify if the distribution of the random sequence of entry is adhered to some known distribution. The following section briefly describes these test suites.

3.1 Randomness requirements

Randomness tests (or tests for randomness) are used to analyze the distribution of a set of data to see if it is random (patternless). The primordial objective of these tests is to scrutinize if the sequences generated by PRNGs exhibit patterns or any regular behavior.

3.2 Statistical Tests

Randomness is a probabilistic property; that is, the properties of a random sequence can be characterized and described in terms of probability.

Statistical tests can be applied to a sequence of bits to analyze if it behaves randomly. Such tests can be used on PRNGs as an initial step in determining whether a generator is suitable for a particular application. There are an infinite number of possible statistical tests, each assessing the presence or absence of a “pattern” which, if detected, would indicate that the sequence is nonrandom(RUKHIN *et al.*, 2001).

A statistical test is formulated to test a null hypothesis (H_0) and an alternative hypothesis(H_a):

- H_0 : *The sequence being tested is random.*
- H_a : *The sequence is not random.*

In case of a strong PRNG, the null hypothesis will be reject a few times. The rejection of the correct hypothesis is called a Type I error. The chances of a Type I error is often called the level of significance of the test. If the PRNG is weak, then the null hypothesis will be rejected in most of the tests. In the cases where the conclusion point it out that a weak sequence is is random then it is called a Type II error. Its probability is called the critical point.

Some premises are made with respect to random binary sequences to be tested. In the case of binary sequences, the probability of finding 0 or 1 has to be $1/2$, ensuring uniformity of distribution. The selected statistical test has to be applicable also to the to sub-sequences

extracted at random from the main sequence in test. If a sequence is considered pseudo-random, its subs-sequences are pseudo-random as well. The tested PRNG must be tested with sequences generated from different seeds. If the PRNG is strong, most of the sequences will present good results in the statistical tests (RUKHIN *et al.*, 2001)

3.3 Statistical Test Suites

Statistical test batteries can consist of various individual tests with specific configurations depending on the aims of their developers. It is important to make sure that the configuration of the tests are correct. One example is prevent the sequences return to their first values after reaching the end of the file submitted to test. This situation may easily introduce Type I errors (HURLEY-SMITH; HERNANDEZ-CASTRO, 2018). Statistical tests can be applied to a sequence of bits to analyze if it behaves randomly. Such tests can be used on pseudo-random number generators as an initial step in determining whether a generator is suitable for a particular cryptography application.

3.3.1 NIST

The National Institute of Standards and Technology (NIST) - part of the U.S. Commerce Department - issues standards and guidelines for use by U.S. government departments and agencies (STALLINGS, 2008). It is "concerned with the advancement of measurement science, standards and technology (HURLEY-SMITH; HERNANDEZ-CASTRO, 2018).

The NIST Statistical Test Suite consists of 15 tests. This suite was released by the Information Technology Laboratory Presentation, in 2000. The tests focus on a variety of different types of non-randomness that could exist in a sequence. Some tests are a combination of another variety of sub-tests. The NIST test suite has been passed through some changes in reason of some weak points, as can be seen in Kim, Umeno and Hasegawa (2004) Hamano (2005) Bahi *et al.* (2011). The complete description of the NIST suite of tests is in the Annex B.

3.3.2 DIEHARD

The DIEHARD is a battery of eighteen tests. Most of tests are performed on integers on the interval $(0, 2^{31} - 1)$ that are hypothesized to be realizations of a discrete uniform distribution with mass points being in the integers in that range (GENTLE, 2003).

The DIEHARD tests produce 215 results called *p-values*. Each *p-value* is obtained by applying a function $[p-value = F_i(X)(i = 1 - 215)]$, where the function F_i seeks to establish a distribution function of the sample random variable X as uniform between 0 and 1. In addition all of the functions F_i are an asymptotic approximation, for which the fit will be worst in the tails. Thus only rarely will one find *p-values*

near 0 or 1, such as 0.0012 or 0.9983, if the sequence is random. When a sequence is decidedly non-random numerous p – values of 0 or 1 to six or more decimal places (i.e. 1.000000) will be present. Thus for a random sequence, only a small number of p – values should have a value near zero or one, although the presence of occasional p – values of 1.000000 or 0.000000 is insufficient. (EPSTEIN *et al.*, 2003)

DIEHARD contains several statistical tests but has drawbacks and limitations. The sequence of tests as well as the parameters of these tests (sample size, etc.) are fixed in the implementation code package. The sample sizes are not very large: the entire test suite runs in a few seconds of CPU time on a standard desktop computer. As a result, they are not very stringent and the user has little flexibility for changing that. The package also requires that the random numbers to be tested are in a binary file in the form of 32-bit (exactly) integers. The setup is quite restrictive. For instance, many PRNGs produce numbers with less than 32 bits of accuracy (e.g., 31 bits is frequent) and DIEHARD does not process that as an input (L'ECUYER; SIMARD, 2007). A short description of the DIEHARD suite of tests is in the Annex C.

3.3.3 TestU01

TestU01 is a very extensive test suite that includes the tests from DIEHARD and NIST. The statistical suite also has additional tests to focus on analyze the unveiled problems in some PRNGs that pass DIEHARD and NIST. TestU01, written in C, is much easier to use than DIEHARD or the NIST suite, according to Gentle (2003). The input to this set may be sequences whose vary between and 0 and 1 or binary sequences.

In this test suite tools are also offered to perform systematic studies of the interaction between a specific test and the structure of the point sets produced by a given family of PRNGs. "That is, for a given kind of test and a given class of PRNGs, to determine how large should be the sample size of the test, as a function of the generator's period length, before the generator starts to fail the test systematically" (L'ECUYER; SIMARD, 2007).

Some of the distribution of the nearest pairs of points in various dimensions are the base of interesting tests in TestU01. It needs to be calculated how large should be the sample size of the test as a function of the generator's period length. This estimate also has to consider which test will be performed and which class of the PRNG under test. If this value is not specified, PRNGs may begin to fail the test systematically (L'ECUYER; SIMARD, 2007).

3.3.4 Ent

Ent is a battery of tests applied to test PRNGs for for encryption and statistical sampling applications, compression algorithms, among others. Battery tests include entropy test, chi-square test, a serial correlation test, a Monte-Carlo estimate of π test, and an arithmetic mean calculation

test. This battery can be performed over bits or bytes, and runs for the whole length of any given sample file ([HURLEY-SMITH; HERNANDEZ-CASTRO, 2018](#)).

3.3.5 Other approaches

The approaches cited above have the inconvenience of requiring too much time to complete all the tests. In addition, some tests are difficult to interpret and may present conflicting results (the same sequence may pass one test and be disregarded in another).

Because of the limitations of statistical testing, other approaches have been published. [Shahbazi, Tappenden and Miller \(2013\)](#) proposes the using of Centroidal Voronoi Tessellations (CVT) called Random Border CVT (RBCVT). The proposed method is capable of enhance the previous pseudo-random generators test methods and improve their coverage of the input space. The algorithm is able to produce a superiorly distributed set of test cases when compared to widely used pseudo-random generators test. "This randomness requirement is investigated using Kolmogorov complexity, which provides a new class of distances appropriate for measuring similarity relations between sequences" ([SHAHBAZI; TAPPENDEN; MILLER, 2013](#)). In the work of [Oliveira et al. \(2015\)](#) it proposed the use of information theory as PRNG evaluation approach. In order to assess the quality of any PRNG, it was developed a non-parametric hypothesis test to measure the quality of the sequence generated by Mersenne Twister e LCG through the position of the point observed in the entropy-complexity plane(HC).

3.4 Limitations

All of the above tests have been well accepted in the literature, but passing these tests does not guarantee the efficacy of the PRNG. The NIST undergoes several updates because the scientific community finds problems or incompleteness in its test suite. The DIEHARD has limitations because it admits only numbers with exactly 32 bit of accuracy. As stated by [Soto \(1999\)](#), the statistical testing is employed to gather evidence that a generator indeed produces numbers that appear to be random.

However, passing these tests does not guarantee the efficacy of the cryptographic system to cryptanalysis, both for mathematical limitations and for test implementation limitations. "Unfortunately, no universal test or battery of tests can guarantee that a given generator is fully reliable for all kinds of simulations" ([CASTRO et al., 2005](#)). Although there are many pseudo-randomness tests, there is no description of the full set of tests that a PRNG must pass to be considered a strong algorithm ([CHANG et al., 2010](#)). The test suites are not considered enough for PRNG testing. "Few resources are readily available to researchers in academia and industry who wish to analyze their newly developed PRNG" ([SOTO, 1999](#)). Even with the limitations of PRNG testing, being approved in NIST and DIEHARD test suits is the supporting

evidence of cryptography security commonly used by TRNGs manufacturers([HURLEY-SMITH; HERNANDEZ-CASTRO, 2018](#)).

The limitations of randomness testing need to be discussed because of the impact the results can have. The fact that there is no unanimity as to the tests to be performed, contradictory situations may occur. For instances, sequences that were generated by devices that theoretically generate true random sequences passed in NIST and DIEHARD but fail in a simple chi-square test ([HURLEY-SMITH; HERNANDEZ-CASTRO, 2018](#)).

The main limitation perceived in the the description of the test suites above, is that the temporal and spatial distribution of numbers are not contemplated (or at least are not deeply analyzed). It can be a promising source of weakness of the PRNGs. The next chapters describe the fields that are the bases of an improvement proposal to identify some pattern in sequences generated by PRNGs. This proposal involves the study of measurements of complex networks and the analysis of the polygons drawn by algorithms of computational geometry.

COMPLEX NETWORKS

The study of networks has its origin in mathematics. The graph theory emerged in 1736 with the solution of the famous Königsberg's seven bridge problem by Leonard Euler (1707-1783). After Euler, renowned scientists such as Augustin-Louis Cauchy (1789-1857), Sir William Rowan Hamilton (1805-1865), Arthur Cayley (1821 - 1895), and Gustav Kirchhoff (1824 – 1887) dedicated themselves to the studies of graphs ([BARABASI, 2003](#)). These scientists contributed profoundly to the study of large and ordered graphs.

Until half of the last century, the theory of graphs was summed up to find measures. In the second half of the twentieth century, graph theory expanded to explain new questions. "The origin of real networks and the laws that govern their appearance and structure was first explain by two Hungarian mathematicians: Paul Erdős and Alfréd Rényi" ([BARABASI, 2003](#)). Paul Erdős (1913-1996) and Alfréd Rényi (1921-1970) have studied deeply about the simplest model of complex networks: random networks. This model was fundamental for the other models to emerge. However, these new revolutionary models only emerged in the late 1990s with Duncan J. Watts, Steven Strogatz, Albert-László Barabási and Réka Albert.

According to [Figueiredo \(2011\)](#) network is "an abstraction that allows to code some kind of relationship between pairs of objects". A common example of networks present in everyday life are social networks, where individuals relate (are connected) to friendships, groups, pages and feeds. The study of the topological structure of networks has a fundamental role as it directly influences the functionality and processes that operate on them. This description is of paramount importance since complex networks have been applied in highly relevant studies such as the spread of epidemics([BOGUNÁ; PASTOR-SATORRAS, 2002](#)), neural networks([BULLMORE; SPORNS, 2009](#))([MILO *et al.*, 2002](#)), protein networks([IDEKER; SHARAN, 2008](#)), security, and power supply networks([MOTTER; LAI, 2002](#)).

4.1 Basic definitions and measurements

"A network is defined in graph theory as a set of nodes or vertices and the edges or lines between them" (BULLMORE; SPORNS, 2009). Network theory uses the mathematical formalism of graph theory but also statistical methods to characterize the structure of network connections, to create models and analyze the structure and dynamics of networks (CASANOVA, 2013). The term complex networks refers to a graph that presents a non-trivial topographic structure. The measurements collected from networks are essential related to the types of dynamics that may happen in them.

A network $G(V, E)$ corresponds to a graph, which is represented by a non-empty set of nodes V connected by edges E . Node or vertex is a local characteristic of a network. It may correspond to a document (in the context of the Web), a computer (in a network), a gene (in biology), for instance. The edge or link is the line joining two nodes. They establish some kind of relationship between two vertices according to the modeled phenomenon (METZ *et al.*, 2007). When these connections are considered only in one direction, the network is so called undirected, and directed otherwise. The edge and vertices can also have extra information: weight. This characteristic represents strength, actual weight, cost, or value and it is usually a real number (NEWMAN, 2010).

In a directed graph (digraph), each edge has a direction that connects a source vertex to a destination vertex. Examples of digraphs are those used to represent phone calls and e-mail messages, in which messages are directed from one person to another. The digraphs can be cyclic, when there is a vertex path for itself, or acyclic when there is no such path (METZ *et al.*, 2007)

The main network measurements considered in this work are described bellow:

- Node degree is the number of neighbors of a node. "The number of connections that link it to the rest of the network — this is the most fundamental network measure" (BULLMORE; SPORNS, 2009).

$$k$$

- Degree distribution: fraction of vertices of the network that has a certain degree (FIGUEIREDO, 2011).

$$P(k)$$

- Average degree: arithmetic mean of the degree of all nodes (FIGUEIREDO, 2011).

$$k_{\mu} = \frac{1}{n} \sum_i k_i$$

- Second moment of degree distribution: measure of connectivity distribution fluctuations (ALMEIDA, 2013).

- Shannon's entropy of degree distribution: indicates the measure of probabilistic uncertainty in a given probability distribution (NASCIMENTO, 2013).

$$k_h = - \sum_i P(k_i) \log(P(k_i))$$

- Average of the local agglomeration coefficient: mean of the cluster coefficient for the graph (NASCIMENTO, 2013).

$$\bar{C} = \frac{1}{n} \sum_{i=1}^n C_i$$

- Coefficient of agglomeration of the whole network by the formula of the transitivity (Average Clustering): the local agglomeration coefficient is a measure that allows to identify the transitivity of the neighborhood of a given node. For transitivity it is understood the level of cohesion between the neighbors of a given node (PRESTES, 2016).

$$C_i = \frac{\lambda_G(v)}{\tau_G(v)}$$

where $\lambda_G(v)$ is the number of triangles on $v \in V(G)$, or is the number of sub-graphs with 3 edges and 3 vertices, one of which is v . $\tau_G(v)$ is the number of triples on $v \in G$, or is the number of sub-graphs (not necessarily induced) with 2 edges and 3 vertices, one of which is v and such that v is incident to both edges.

- Average of the smallest paths: average number of steps along the shortest paths for all possible network node pairs (PRESTES, 2016).
- Spearman coefficient: it assesses how well an arbitrary monotonic function can describe the relationship between two variables, without making any assumptions about the frequency distribution of the variables (HAUKE; KOSSOWSKI, 2011).
- Modularity of the edge betweenness centrality: in this algorithm, the edges that are located between the communities are favored and those that are within the communities are disadvantaged. Thus, one can identify if two communities are united by only a few edges (NEWMAN; GIRVAN, 2004).
- Fast-greedy modularity: this method is essentially a rapid implementation of the technique based on the edge centrality proposed by Newman (NEWMAN; GIRVAN, 2004). Starting from a set of isolated nodes, the links in the original graph are added iteratively to produce the largest possible increase of Newman and Girvan's modularity at each step.
- Randomwalk betweenness centrality: the expected number of times a random walk passes through vertex i on its way from a source s to a target t , averaged over all s and t (NEWMAN, 2005).

- Rich club phenomenon: it is characterized when the hubs (nodes with high degrees) are on average more intensely interconnected than the nodes with smaller degrees (MCAULEY; COSTA; CAETANO, 2007).
- Hierarchical degree: the total number of virtual edges established along successive transitions (COSTA, 2004).
- Degree distribution: fraction of vertices of the network that has a certain degree (FIGUEIREDO, 2011).
- Density: the average strength per edge (CHENG *et al.*, 2012). It can be defined also as the fraction of present connections to all possible connections (HOSSEINI; HOEFT; KESLER, 2012).

$$\delta = \frac{m}{\binom{n}{2}}$$

where n is the number of vertices and m is the number of edges

- Efficiency: measure of information exchange efficiency. This concept encompasses the study of information exchange across the network, where information is simultaneously exchanged and network resistance to failure on a small scale (PRESTES, 2016).

$$E = \frac{1}{N(N-1)} \sum_{i \neq j} \frac{1}{d_{ij}}$$

- Assortativity coefficient: a correlation coefficient between the degrees of all nodes on two opposite ends of a link (CHENG *et al.*, 2012).
- Pearson correlation coefficient: a measure of the strength of the linear relationship between two such variables (HAUKE; KOSSOWSKI, 2011).
- Betweenness centrality: evaluates the importance of a node in the transmission of messages and events among others. This measure can be used to assess the importance of both vertex and network connections (RONQUI, 2014).

$$g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

where σ_{st} is the total number of shortest paths from node s to node t and $\sigma_{st}(v)$ is the number of those paths that pass through v

- Eigenvector centrality: the eigenvector approach is an effort to find the most central nodes (that is, the ones with the least distance from other nodes) in terms of the "global" network structure, and to pay less attention to patterns that are more "local" (HANNEMAN; RIDDLE, 2005).

In addition to the measures cited above, the following measures were also analyzed over the bibliographical research: greater degree, vulnerability and diameter. The measures to detect communities were modularity maximization improvement, modularity of the walktrap, modularity of the leading eigenvector, modularity of the label propagation, modularity of the multilevel, and modularity of the spinglass. The centrality measures were the closeness centrality, PageRank centrality, information centrality, harmonic centrality, and load centrality. Other additional measures were Estrada index, Burt's score, authority score, hub score, and eccentricity.

4.2 Networks Models

Over the second half of the last century, the empirical studies in social psychology and sociology started the probabilistic models. This models led to an extensive publication of scientific literature in the 1970s and 1980s. Since then, statistical models have been found in statistical physics and computer science(GOLDENBERG *et al.*, 2010). The growth of the World Wide Web, the emergence of online “networking communities” such as Facebook, Twitter, and LinkedIn, and the growing interest of the scientific community has intensified interest in the study of networks, network data and the models that they are organized.

4.2.1 Random Networks

The random network model was introduced by Ray Solomonoff(1926-2009) and Anatol Rapoport (1911-2007) in 1951 (NEWMAN; WATTS; BARABÁSI, 2006). This network model was extensively studied by Paul Erdős (1913-1996) and Alfréd Rényi (1921-1970) what makes the model be also known as Erdős-Rényi. It is often referred to by its mathematical name $G(n, m)$. The random graph model is not defined in terms of a single randomly generated network, but an ensemble of networks, i.e., a probability distribution over possible networks.

In general, a random graph is a model network in which some specific set of parameters take fixed values, but the network is random in other respects. One of the simplest examples of a random graph is the network in which we fix only the number of vertices and the number of edges in. That is, we take n vertices and place in edges among them at random. More precisely, we choose m pairs of vertices uniformly at random from all possible pairs and connect them with an edge. (NEWMAN, 2010)

The proposed model consists of N nodes, which, in pairs, are connected to each other with probability p and k is the total of edges that are associated with a certain vertex. Thus, above a certain threshold of the probability of connection between any two nodes, the set of random graphs undergoes a phase transition. In this period, the graph is made up of separate small sub-graphs, which unite to form a giant component (ALMEIDA, 2013).

Figure 3 – Random Networks

Source: [Barabási \(2016\)](#).

Note – These three networks were generated from the same parameters: $p = 1/6$ and $N = 12$. Even though they have the same value for the parameters, the networks do not only look different, but they have a different number of links as well.

Large random graphs have Gaussian degree distributions. It is now known that most graphs describing real-world networks significantly deviate from the simple random-graph model ([BULLMORE; SPORNS, 2009](#)). The random model guided the thinking about networks until the late 1990s. From then on, random networks consolidated as a versatile tool to explore and offer new horizons in the study of several real systems ([ALMEIDA, 2013](#)). It is believed that the network construction process is random in the sense that vertices are randomly aggregated. Based on this premise, Erdős and Rény concluded that all the vertices of a given network have approximately the same number of connections and the same chances of receiving new connections ([METZ *et al.*, 2007](#)).

4.2.2 Small-World Networks

The small-world model was proposed in 1998 by Duncan Watts (1971) and Steven Strogatz (1959) as a simple model of social networks. In this model, most pairs of nodes are not directly connected to each other, but they can be reached by a small number of steps ([CASASNOVAS, 2012](#)). This model presents also a high transitivity.

The property of short paths in the network is called six degrees of separation. In 1967, the Harvard professor Stanley Milgram (1933-1984) conducted an experiment to find the distance between two people in The United States. For this, he sent letters to residents of the cities of Wichita (Kansas) and Omaha (Nebraska) asking if they would like to participate in an academic experiment. In the letter, there were instructions from the experiment, which included sending

the letter to a particular person, the target-person. If the person who received this invitation letter knows the target-person, that person should send the letter directly to that person. Otherwise, the person should send it to someone who might know the target-person. 42 of 160 letters return successfully to Harvard. Milgram then calculated that the average number of people needed to send letters from the person initially contacted to the person-target was 5.5. It was roundup to 6 and so the name 6 degrees of separation ([BARABASI, 2003](#)).

The second main feature of this model is the high clustering. This means that there is a high probability that two nodes are directly connected if they have a neighbor in common ([NEWMAN; WATTS; BARABÁSI, 2006](#)). This measure can be calculated by the transitivity, a measure suggested by Watts and Strogatz.

The small-world model interpolates between a regular graph and a random graph, depending on a single parameter $p \in [0, 1]$, without altering neither the number of nodes nor the number of connections per node of the original graph. In the case of $p = 0$, the network would be regular. In the case of $p = 1$, the network would be random. Therefore, there is an intermediate region of where it is possible to generate networks with both features: the high value for the clustering coefficient and the short average path length ([CASASNOVAS, 2012](#)).

Figure 4 – Small-world model in relation to regular and random networks

Source: Adapted from [Barabási \(2016\)](#).

Note – The network type is directly related to the value of p . Small-world networks are between regular networks and random networks.

Some examples of small world networks are social networks, gene networks, the Internet ([CASASNOVAS, 2012](#)), networks of brain neurons ([BULLMORE; SPORNS, 2009](#)), electric power grids, word co-occurrence, food web, metabolite networks, biology collaboration, and company directors ([NEWMAN; WATTS; BARABÁSI, 2006](#)).

4.2.3 Scale-Free Networks

The scale-free networks were described by Albert Barabási (1967) and Réka Albert (1972) in 1999 while they studied the WWW degree distribution. The degree distribution observed followed a inspected power law. From this identification, several academic papers were published on this model (NEWMAN; WATTS; BARABÁSI, 2006).

The scale-free network is characterized by its power law degree distribution Barabási (2016). Networks with this characteristic are called free of scale due to their mathematical representation. The scale-free property is related to the hubs in the network. In a network with a power-law degree distribution most nodes have only a few links. These numerous small nodes are held together by a few highly connected hubs.

A scale-free network looks like the air-traffic network, whose nodes are airports and links are the direct flights between them. Most airports are tiny, with only a few flights. Yet, we have a few very large airports, like Chicago or Los Angeles, that act as major hubs, connecting many smaller airports. Once hubs are present, they change the way we navigate the network. For example, if we travel from Boston to Los Angeles by car, we must drive through many cities. On the airplane network, however, we can reach most destinations via a single hub, like Chicago. (BARABÁSI, 2016)

The building scale-free networks is based on preferential attachment and growth. The preferential attachment, also called preferred connection, is the assumption that the tendency of receiving new edges increases with the node's degree (CASASNOVAS, 2012). This feature implies networks with few highly connected vertices and many vertices with few connections. The grown is basically the step of adding nodes to the network for each given period of time (BARABASI, 2003). These networks have been observed in several systems, for example, in networks of metabolisms and in networks of citations of scientific articles. Figure 5 shows a scale-free network of earthquakes to study the correlation of time interval between events, the spatial distance between epicenters and magnitude of the earthquakes(BAIESI; PACZUSKI, 2004).

Figure 5 – Scale-free network

Source: Adapted from [Baiesi and Paczuski \(2004\)](#).

Note – Scale-free network of earthquakes that occurred between the years 1984 to 2000 in different areas of the Earth. These networks supports the investigation of the correlation of time interval between events, the spatial distance between epicenters and magnitude of the earthquakes

4.3 Spatial Networks

In the last few years, there has been considerable interest within the physics community in the analysis and modeling of network systems. Some of these networks, such as biochemical networks and citation networks, exist only in an abstract “network space” where the precise positions of the network nodes have no particular meaning. But many others, such as the Internet, live in the real space of everyday experience, with nodes (e.g., computers in the case of the Internet) having well-defined positions ([GASTNER; NEWMAN, 2006](#)).

When applied in real-world networks, most of the complex network models mentioned above have ignored the geography relevance the system ([CRUCITTI; LATORA; PORTA, 2006](#)). A particular class of complex networks that concentrates in phenomena embedded in the real space are the structural networks. In structural networks nodes occupy a precise position in two or three-dimensional Euclidean space, and whose edges are real physical connections. In this model, the geography matters and it represents the most important characteristic of the system. The spatial constraints may have a strong effect on structural connectivity patterns.

Interest in the applications of structural networks dates back to the era of the economic geography movement of the 1960s. However, the limitations of data and computing resource made the field only re-exploited recently. "The effect of space on network topology has recently come under the spotlight because of the emergence of pervasive technologies based on geolocal-

ization, which constantly fill databases with people's movements and thus reveal their trajectories and spatial behavior" (EXPERT *et al.*, 2011). Some practical examples are the roads or railway lines in transportation networks, the Internet, power grid, or oil pipelines. In transportation networks, flights between airports and federal and state highways, for example, are the edges of a network that connects cities. "A crucial difference stems from the cost associated to long-distance links, which restricts the existence of hubs (i.e., high-degree nodes), and thus the observation of fat-tailed degree distributions in spatial networks" (BARTHÉLEMY, 2003).

The work of Crucitti, Latora and Porta (2006) criticizes previous publications where even dealing with urban street patterns, the network centrality was studied in relational networks and they neglected the geography, a fundamental aspect of streets. In the case of the study of cities as structural networks, also known as information city network, a city is transformed into a topological graph by mapping the streets as nodes and the intersections between streets into edges. In Crucitti, Latora and Porta (2006), the cities are modeled in spatial terms (meters, miles). The results is a visual characterization of the structural properties of a city, as seen in figure 6a.

One of the most evident applications of structural networks in recent years is in the study of the brain as a network (BULLMORE; SPORNS, 2009), Hosseini, Hoeft and Kesler (2012). Brain structural and functional connectivity plays an important role in neuroanatomy, neurodevelopment, electrophysiology, functional brain imaging, and neural basis of cognition. Modeling the brain connection as seen in figure 6 helps to understand the relation of the connection patterns with normal aging, multiple sclerosis, Alzheimer's disease, schizophrenia, depression, and epilepsy. Two planar versions of the structural networks will be detailed in the next chapter.

Figure 6 – Structural networks

Note – Structural networks representing: (a) the city of Venice. The streets are the nodes and the intersections between streets are the networks' edges. The four sub-images are closeness, betweenness, straightness, information centralities. (b) is the representation of the brain in order to understand the relation of neurons connection patterns with normal aging and brain diseases.

COMPUTATIONAL GEOMETRY

"Computational geometry is a computer-based, analysis and syntheses (design) and computer control manufacture of two- and three-dimensional shapes" (FORREST, 1971). Aurenhammer (1991) says that computational geometry is "concerned with the design and analysis of algorithms for geometrical problems". This field of study focus on geometric information - points and their associated data, or polygons that describes spatial objects. The main objective is find suitable information about the geometric shapes based on a mathematical model by using computers (BU-QING, 1989).

Computational geometry has been a key problem-solving tool for years principally in ship-building, aircraft and car industries(BU-QING, 1989). The most known computational geometry algorithms are the Voronoi Diagram and Delaunay Triangulation (GUIBAS; KNUTH; SHARIR, 1992). Because of their relevance and wide-ranging utilization, there is a need for faster implementations of algorithms for these techniques (HELD, 2001). In addition, there are challenges in the modeling step because these mechanisms are limited to points and lines manipulations. For this reason, there have been tentatives to expand the classes of geometrical elements they can handle (HELD; HUBER, 2009). Computational geometry has been established due to its wide application in computer graphics, computer-aided design, robotics, pattern recognition, and operations research (AURENHAMMER, 1991).

5.1 Voronoi diagram

It is possible that natural structures that resembles Voronoi diagrams has been observed before the publication of *Le Monde* and Part III of *Principia Philosophiae*, both published in 1644 by René Descartes. The disposition of matter in the solar system and its surrounding area are described in this book by structures that resemble Voronoi structures. Notwithstanding, there is no mention to the mathematical statement of the polygons what led Okabe *et al.* (2000) supposes that this diagrams were already known by the academic community.

Figure 8 – New Burlington Street, London, England, 1855

Note – John Snow's Voronoi diagram created in 1854 where he highlighted Cholera cases.

The term Voronoi Diagram was coined by the work of Georgy Fedoseevich Voronoy (1868 - 1908) and Peter Gustav Lejeune Dirichlet (1805-1859). Their studies reached the positive definite quadratic forms - a special form of the Voronoi diagram.

Their concern was with the distribution of points with integer coordinates that give minima of the values of a given quadratic form. In that context, they considered the set of points regularly placed in the m -dimensional space generated by linear combinations of m linearly independent vectors with integer coefficients. This set contains infinitely many points, and the Voronoi diagram generated by this set of points gives the partition of the space into mutually congruent polyhedra. (OKABE *et al.*, 2000)

Its first formal applications were in crystallography in the late 19th and early 20th centuries. At this time, it appears that Voronoi diagram was developed independently in meteorology and estimation of ore reserves in a deposit using information obtained from bore holes. Over the years there have been many rediscoveries of the Voronoi Diagram in different fields: chemistry, the study of codes, ecology, astronomy (VAVREK *et al.*, 2008), physiology, linguistic variations and geography.

In 1975, Shamos and Hoey (1975) published the paper *Closest-point Problems* where it is presented an algorithm for constructing Voronoi diagrams, illustrate how they could be used to solve a series of what were then seen as essentially independent problems relation to a finite set of distinct points in Euclidean space, whose center is inside the convex hull of of the point. This publication, the advances of computer science in the 1970s (FORREST, 1971), and the creating

of other algorithms to construct the Voronoi diagram(OKABE *et al.*, 2000) resulted caused a fast expansion of computational geometry.

5.1.1 Generic Definition of Voronoi Diagram

"Given some number of points in the plane, their Voronoi diagram divides the plane according to the nearest-neighbor rule: each point is associated with the region of the plane closest to it" (AURENHAMMER, 1991). This definition can be better explained by the following relation:

$$V_p = \{x \in \mathbb{R}^2 \mid \|x - p\| \leq \|x - q\|, \forall q \in S\} \quad (5.1)$$

where $S \subseteq \mathbb{R}^2$ is a set of points, the Voronoi region of $p \in S$ is a set of points where $x \in \mathbb{R}^2$ and are at least as close to p as to any point in S (EDELSBRUNNER, 2001).

Figure 9 – Basic Voronoi diagram

Source: Aurenhammer (1991).

Voronoi diagram has been useful in the study of resource allocation. An example is the work of Mureddu *et al.* (2016) where the the electricity network of Germany is splitted into regions according to their internal power demand, represented in figure 10. The Voronoi algorithm is used in order to create a division where the electricity would have continuity in case of critical events.

Figure 10 – Voronoi tessellation application

Source: [Mureddu et al. \(2016\)](#).

Note – Voronoi tessellation of the German power system, where red nodes indicate transmission node

5.2 Delaunay triangulation

The concept of Delaunay triangulation (or tessellation) is originated with Voronoi in 1908 who defined it by the way of the neighbor relations in the Voronoi diagrams. However it was the Russian mathematician Boris Delone (Delaunay) who first defined the tessellation using empty sphere method. Delaunay initial paper in this topic was included in the international Mathematical Congress held in Toronto, Canada, in 1924 ([OKABE et al., 2000](#)).

One of the properties of the Delaunay triangulation in two dimensions is that the individual triangle are as equilateral as possible ([PITERI; JUNIOR, 2007](#)). "Delaunay triangulation maximizes the minimum angle and minimizes the maximum angle of all triangles in the triangulation." ([MOURA, 2006](#)). Given a set of distributed points randomly in a plane, a virtually unlimited number of meshes of triangles can be generated. However, there is a single triangulation that maximizes the sum of the small angles of each triangle in the mesh. In other words, given a cloud of points, the Delaunay triangulation is that which results in a set of triangles as close as possible to triangles ([MAGALHÃES; PASSARO; ABE, 2000](#)).

First, most polygons that describe real-world objects have irregular shapes and regions belonging to different domains of interest. In this context, the Delaunay triangulation, conceptually, can be seen as a strategy to decompose a domain into triangles, respecting its geometric characteristics, as an initial step of the discretization process.[...] Second, the Delaunay triangulation contributes to the quality of the final mesh, given that given a set of vertices, it maximizes the minimum angle among all possible ways of triangulating ([MOURA, 2006](#)).

5.2.1 Generic Definition of Delaunay Triangulation

"The definition of a Delaunay triangle will serve to ensure that the set of Delaunay edges of a set of vertices collectively form a triangulation. The circle-surrounding of a triangle is the only circle that passes through all its three vertices. A triangle is said to be of Delaunay if only if its circle-surrounding has its interior empty "(MOURA, 2006). This is the empty circle-surrounding property. The triangulation $DT(V)$ of a set of points $V \subseteq \mathbb{R}^2$ is called Delaunay triangulation if, and only if, the circumference that circumscribes each of the triangles $\triangle(p_i p_j p_k) \in DT(V)$ is empty (PITERI; JUNIOR, 2007).

Figure 11 – Basic Delaunay triangulation

Source: Piteri and JUNIOR (2007).

Over the last two decades, Voronoi diagram and Delaunay Triangulation has been used in face aging(MONTILLO; LING, 2009), information visualization (GANSNER *et al.*, 2009), Computer Graphics (DEUSSEN *et al.*, 2000) , network architecture (XU; TAKAHARA; HASSANEIN, 2006), pathology (SUDBØ; MARCELPOIL; REITH, 2000),power and energy engineering (FENG; GE; LIU, 2012), robot (BHATTACHARYA; GAVRILOVA, 2007), spatial databases(CHENG *et al.*, 2010), graph theory(ERWIG; HAGEN, 2000) and geophysics(HOPKINS; FRANKENSTEIN; THORNDIKE, 2004).

A more recent example is seen in figure 12 where the Delauney triangulation is used to forecasts areas of possible flooding in the downstream water of Irkutsk hydroelectric station, in Russia (GACHENKO; HMELNOV, 2019).

Figure 12 – Delaunay diagram application

Source: [Gachenko and Hmelnov \(2019\)](#).

Note – Development of Complex GIS Monitoring of the Angara River

METHODOLOGY

The application of PRNGs is of extreme relevance for science and business. It is inconceivable to imagine the contemporary world without the employment of pseudo-random sequences and the mathematics that have arisen around these numbers. Pseudo-random numbers, along with prime numbers, have become the basis of the connected world as in Internet banking functionalities, for instance. A more common use now-days is the simulation of stochastic processes and analytically intractable mathematical expressions. In engineering and natural sciences, these numbers are used extensively in simulation of physical and biological processes ([GENTLE, 2003](#)). In addition, pseudo-randomness has diverse applications in information security, digital games, simulations, modeling, gambling, arts, among others.

Despite the relevance of research about PRNGs, existing methods of randomness measurement do not offer a definitive solution to the need of classifying PRNGs. As mentioned in the sections above, NIST and DIEHARD, the main methods of randomness measurement today, are statistical tests suites in which the sequences generated by the algorithms pass through a battery of verification. The number of tests on which the sequences fail indicates the level of weakness of the algorithm that generated them. Once these algorithms fulfill all the requirements, they are considered satisfactory random, but there is no way to inform which of the algorithms is the strongest or which one will be decrypted first.

Although the prevailing randomness tests are accepted to make algorithm selections, one can not say that they scan all possible characteristics of the sequences to ensure that no patterns in the sequences have emerged. For instance, in both statistic test suites there is no inspection of the importance of values in a global context or, in a 2D plot, the relation of the value with the space around it. This omission from other analytic perspectives may embody weaknesses in those extensively applied methodologies.

In order to better understand the properties of pseudo-random sequences, they have already been studied with the use of statistics, information security and genetic programming, for

example. In the present work, another approach is added : modeling in graphs. The bibliographical research on the use of graphs and complex networks for PRNG testing revealed that this approach is marginally exploited and not conclusive. Seeing the successful research on complex networks' peculiarities have been useful for a considerable number of fields, it is conjectured that an ample investigation applying graph theory can reveal some disguised properties in PRNG sequences.

Complex networks have already been widely used in the creation of descriptors in pattern recognition. Generally, the object of analysis is adapted into a network. After this stage, complex network characteristics are selected. These characteristics are transformed into descriptors and then used in the pattern recognition phase. This phase also contributes to the screening of the best network measurements, thus, it is possible to reduce the number of measures collected from the networks, reducing the time of the experiments.

This proposed methodology is based on the use of graph theory as a tool of analyzing pseudo-random algorithms. In this methodology, the 2D plot of PRNG output sequences are converted into structural networks. The main complex networks measurements and computational geometry meshes are applied in order to extract important properties of the pseudo-random sequences. It is hypothesized that these sets of measures are important descriptors of the algorithms that generated the sequences. This work focuses on identifying patterns in the properties of graphs generated by algorithms that are considered random by PRNG tests and thus to pinpoint improvements to the randomness evaluation.

6.1 Complex Networks Approach

The theory of complex networks is a multidisciplinary domain adopting graph theory, mathematics, and statistical mechanics. All these areas support the interpretation of real events through community structure identification, connection degree distributions, and information propagation, among others. In the past two decades, the scientific community has witnessed the application of networks in a wide list of fields. This is partially explained by the fact that many phenomena can be represented as special cases of graphs as well as the advance of computational processing power and data storage capacity. In this section, the modeling of PRNG sequences as graphs are introduced, along with the consideration of initial tests for validation of this modeling.

6.1.1 Network conception

The representation of events in complex networks requires the indication of the elements that will be mapped as nodes as well the identification of which entities will compose the set of edges and their properties. These attributes should be defined according to what one wants to inspect. Also, it is important to have a connection criterion between nodes. This subsection illustrates the conversion of PRNG-generated sequences into structural graphs following these specifications.

Since the output of a PRNG is a sequence of pseudo-random numbers, one can say that these numbers were generated by a function f and each element of the sequence was generated at a different iteration, for convenience, it will be called time t . In view of this, it is possible to design a tuple (t, y) to express the iteration at which the value was generated and the value itself. If t is renamed x , then there is a representation that can be illustrated in a Cartesian plane. As an example, figure 13 shows the plot of the first 1000 k-Logistic Map values for $K0$.

Figure 13 – Plot of 1000 points generated by the k-Logistic Map with $K0$

Source: Elaborated by the author.

Applying this mechanism, it is stipulated that the set of points represented by (x, y) represents a collection of nodes of the network that will be created. The list of resulting points then passes through the procedure of connection, resulting in creating the edges. However, there is an x axis formed by integers representing the number of iterations. The y -axis contains real numbers ranging from 0 to 1. To prevent the connections from becoming too long and horizontal, the x values were all multiplied by 0.005. Thus, viewing mainly of clusters is facilitated.

Before elaborating the connection requirement between two points, the main characteristics expected from a good PRNG must be reviewed. By doing this, one can conjecture which properties might reveal weaknesses in the PRNG sequences once they are converted in graphs. The main points to be considered are listed as follows:

- Only uniformly distributed sequences must originate from a PRNG. Thus, an analysis of the plot of points must reveal a balance between the values of y .
- The 2D-plot of the PRNG must not point out clusters in its topology. The existence of many clusters in a plot would show that within short periods of time there was a large occurrence of approximate values.

- The period of an good PRNG must be large. The creation of a network covering all points of a period of a PRNG is computationally heavy. Thus, the graph representing such a sequence must be much smaller than the PRNG period and still have relevant characteristics.
- In an good PRNG, it is not possible to find patterns in the temporal distribution of its output within its period. In this case, it is expected that no one can identify identical structures that are repeated throughout the graph.

The selected approach was the use of a radius r ; all the neighboring points within the radius of a certain point are connected to it. For instance, one could consider the points A and B represented by (x_a, y_a) and (x_b, y_b) respectively. The task is to analyze whether B will be connected to A. In this way, the Euclidean distance, d , from point A to B is calculated.

$$d = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2} \quad (6.1)$$

If the distance is less than or equal to the radius, then B is connected to A. Otherwise, B is not connected to A.

$$\overleftrightarrow{AB} = \begin{cases} 1 & \text{if } d \leq R \\ 0 & \text{if } d > R \end{cases} \quad (6.2)$$

An important issue to be settled is the value of r . This value should be selected so that the networks to be formed have the following attributes:

- An acceptable average grade so that the nodes are no more connected than necessary. If the average degree of the network is very high, the computational cost of the calculation of measures related to paths in the graph increases considerably.
- The graphs have similar number of points so that there is a better compilation of networks. Networks with very discrepant number of points could precipitate a technical complexity to determine the better normalization of the measurements' values. Different values of average degree generate very different network, hampering the feature extraction and the process of pattern recognition.
- The highest possible number of points in the largest connected component. Several measures of complex networks only manage calculations with the largest connected component.

The most appropriate r -value was empirically defined. For this decision, the algorithm used for the tests was the K-Logistic map, with the parameter K ranging from 0 to 4 and the

sequences of random numbers had a maximum of 300 values. The relation of r and the number of points in the largest connected component can be seen in figure 14.

Figure 14 – Radius choice

Source: Elaborated by the author.

Note – Result of testing different radius sizes. Sets of 300 nodes each were connected using those different radius and the one that connects most of the points in the final networks was the radius size selected.

It can be seen that for radii of 0.12 and larger, the networks for all values of the parameter K show almost complete connectivity, but high connectivity is not a desired feature. Therefore, the points are connected with all neighboring points within a radius of 0.12.

The connection processes is repeated until all points are processed. Once all points are connected to their respective neighbors, the points that do not have neighbors are eliminated. This removal of isolated points means that the resulting networks are only presenting points that are part of some connected component, facilitating the calculation of network measurements. The result is a network whose nodes exhibit different degrees as seen in figure 15.

Figure 15 – Plot generated by connecting the points in figure 13 with $r = 0.12$

Source: Elaborated by the author.

It should be mentioned that the graph plot gives only an overview of how these networks look. PRNGs such as those based on chaos theory are very sensitive to their initial conditions. Thus, a small change in the value of x_0 of the K-logistic map generates a completely different sequence, and consequently, graphs with different arrangements. For example, the graph in figure 16 can only be generated if the K-logistic map uses the same values of K and x_0 . The axes are no needed anymore since the Cartesian position is now part of the node's characteristics.

Figure 16 – Graph generated using K-logistic map; $K = 0$ and $r = 0.12$

Source: Elaborated by the author.

Therefore, an algorithm is selected, in the case of the examples above it was the K-logistic map, and different seeds are used as x_0 in this algorithm, resulting in a set of different sequences. Then each of these outputs is converted into a structural network. Nodes are the values formed by the Cartesian representation of the iteration vs. output at that specific iteration, and the edges are the connections made between nodes within a radius. In the following subsections, the PRNGs and their resulting graphs are investigated in detail.

6.1.2 Selection of PRNG for the first experiments

In the PRNGs survey phase, a very relevant aspect was observed. There is a bisection within the PRNGs that is very advantageous for research: some algorithms generate sequences that might be weakly or strongly random depending of specific parameters. Based on this criterion, PRNGs can be divided into two different classes. The first class includes the algorithms whose output randomness quality can be predicted. Besides the seed that almost all PRNGs have as an input, those PRNGs also require other parameters. For instance, the LCG, in addition to the seed, receives the parameters m , c and a . Depending on the mathematical relation between these values, one can foretell the quality of the output. The second group covers the methods that rely upon the seed and therefore do not depend on additional parameters. For example, the LFSR depends only on the seed and no inferences can be made about its sequences. Since one of the objectives of this research is finding relevant information in PRNG sequences, the first class provides an excellent opportunity to correlate the collected observations with the PRNG parameter types. For instance, the K-logistic map with $K0$ originates a very weak sequence that does not pass the NIST test. Nevertheless, the same PRNG with $K4$ generates a sequence that is approved both by NIST and Diehard.

In addition to belonging to the first group of PRNGs, the choice of a first algorithm for experimentation considered the ease of implementation, license to use the code and low computational cost. For these requirements, the K-logistic map was chosen. Since it is a recently published method, this research may turn out to be a first in-depth study of more intrinsic features of that algorithm.

6.1.3 Complex Networks visualization

In contemplation of a better understanding of how the networks produced by the K -logistic map sequences look, visualization is a suitable tool. Although it is impracticable to reproduce here a graph with millions of points, the reflection in a sub-image of the graph might speed up the selection of appropriate measurements and expose fingerprints of the network.

Figure 17 shows the points and two graphs originating from the K-logistic Map with $K0$ while figure 18 shows those generated by the k -logistic map with $K4$. The two sets of images are formed by 5000 points. The nodes were colored considering their degrees; the intensity of the red color is proportional to the degree of the node. Only the leaf nodes are painted green so it is easier to locate them.

Figure 17 – Plots originating from the K-logistic map with $K0$ (a) $K = 0$; 5000 points (b) $K = 0$; 5000 points and $R = 0.12$ (c) $K = 0$; 5000 points and $R = 0.16$

Source: Elaborated by the author.

Figure 18 – Plots originating from the K-logistic map with $K4$ (a) $K = 4$; 5000 points (b) $K = 4$; 5000 points and $R = 0.12$ (c) $K = 4$; 500 points and $R = 0.16$

Source: Elaborated by the author.

As mentioned earlier, $K0$ does not generate pseudo-random sequences. This is evident in figure 17a: there is a high concentration of points on the upper and lower sides of the image making it conclusive that this sequence has many values close to 0 and 1, thus not exhibiting a uniform distribution. Consequently, in the graph of figure 17b the nodes in the bottom and top sections have the larger degree than those in the middle. As a consequence, the image does not show visible leaves in the horizontal borders. Furthermore, the middle region of the graphic does not present a lot of hubs, which are more prominent in figure 17c. The increase in the radius of connection causes nodes that already had a high degree to become even more connected but the

large empty spaces within the network remain. The number of leaves decreases as the radius increases.

The images of $K4$ display an different trend. Although presenting clusters, as seen in figure 18a, they are not concentrated in the lateral regions of the image. When connected, the nodes do not generate the large holes in the image as in the case of $K0$ and many leaf nodes are identified on the sides of figure 18b. The hubs of this graph are distributed across all regions of the image. In figure 18c it is possible to notice that the horizontal sides of the image present elevations and some small holes, whereas in the image of $K0$, the edges form smooth borders in the graph evidencing the non-uniform distribution of the sequence that originated them. Another feature that draws attention to images in figure 18 is the fact that visually, leaves are not concentrated in the center, but found more distributed in the network. Apparently, the leaf nodes appear more distributed along the network. The list of images with radius = 0.12 generated by other PRNGs can be seen in Annex A.

Some of the observations from the previous paragraphs could be made with a simple visual analysis of the sequence that generated these images. However, while the sequence increases its level of entropy, the graphs become more and more similar and a simple analysis is not able to identify which parameter generates the sequence. This first inspection shows that in addition to the simpler features, such as a uniform distribution, these networks have a rich range of information. Therefore it is necessary to focus on the measures that show the small differences of these networks. The inspection of these measures is described below.

6.1.4 Fundamental measurements

The network models seen in the previous chapters present features that define them and are intrinsically related with the type of network. These features are fundamental for the analysis, discrimination, and synthesis of those models for this extraction of local and global characteristics (COSTA *et al.*, 2007).

Structural networks are a type of graph whose measures related to distances are very important. Three of the fundamental properties of structural networks are node degrees, edge lengths, and network diameter (GASTNER; NEWMAN, 2006).

Figure 19 – Degree distribution

Source: Elaborated by the author.

The degree distribution of this type of network is a Gaussian distribution as seen in figure 19. All the graphs show the *small world* phenomenon since the average degree varies just from 7.18 to 7.37. The most deviating distribution is the one of $K0$. Since this class of networks is the one with the most clusters, its degree distribution has a small increase in relation to the others beginning at approximately degree 13. All networks have approximately the same number of nodes. Since $K0$ is the class that has a larger range in the distribution, the peak of this is at a lower value than the others. The asymmetric format of this distribution also shows that the number of clusters in these networks is higher than in the other classes.

Another interesting characteristic is the edge length of real structural networks, such as highways and airports have constraints because they are generally implemented by physical means. For example, the construction of a highway takes into account the monetary cost. Therefore, in this type of network, only some highways have great mileage since long railways might be more expensive to construct and manage (GASTNER; NEWMAN, 2006). Most of the highways are of small size, as seen in the section about structural networks. The airplane flights network, even though it is a structural network, is organized into hubs to facilitate transport and also to make a single place connected to several other places, reducing the number of connection flights - the degrees of separation, in the context of networks. The edge lengths distribution of these networks are shown in figure 20.

Figure 20 – Histogram of the edge lengths studied in [Gastner and Newman \(2006\)](#)

Source: [Gastner and Newman \(2006\)](#).

Note – Edge lengths of the networks representing the USA interstates highways, Internet, and flights of Delta Airlines for February 2003

The networks generated by PRNGs are structural networks as well, but they have opposite edge length distribution properties as shown in figure 21 for $K0$ to $K4$. The reason is that in the PRNG sequence plot, each value y is a number of a random sequence generated in iteration x , then it is expected that the numbers generated in approximate iterations are not close to each other. For this reason, the histogram has a small number of short connections while the number of big connections is larger due to the radius value.

Figure 21 – Histogram of the K-logistic Map networks edge lengths

Source: Elaborated by the author.

The similar distribution of the edges length among the K -logistic maps networks has direct impact on the number of edges in average to walk from one vertex to another. For this reason, the diameter of the graphs generated by different values of K do not varies considerably. The measurements based on the graph diameter, however, will be explored further in the next chapter because the small decrease in the diameter value as K increases may have an impact on measures more sensitive to millesimal differences.

Although these measures are very important for the description of structural networks they are not enough. They should be seen as the basis for selecting more descriptive measures, particularly in the case discussed here, the randomness test of PRNGs. These sequences are programmed to be indistinguishable, so separating the networks according to the parameter K is a task that requires a series of more rigorous tests and many other measures.

This research analyzes which measures of complex networks are important for the description of networks created by PRNGs. This analysis is done considering the importance of nodes and their neighbors with measures of networks. The Euclidean distance of points is integrated as weights on edges in the networks, therefore weighted network measurements are also applied. Visualization of networks and the basic measures discussed above are the guide to the results chapter.

6.2 Computational Geometry Approach

The spacial information is one of the most relevant attributes of the type of network that is created after the points connection. However, the intrinsic peculiarities of the network geometry are not considered when the network is converted into an unweighted network. In unweighted networks, no matter how far one point is from the other in Euclidean distance, it is only significant if one point is connected to another. In weighted networks, the distance can be a characteristic of the edges, but still, this is not enough for a deep spatial view of the networks. Although the relationship of nodes and their neighbors are relevant measurements of networks, the challenge of scrutinizing PRNGs also requires a study of how points are distributed in space and how space around points can provide details of global sequence information.

Computational geometry is adopted in this research because of its mechanism of creating meshes and their analysis in a plane space. The methods in computational geometry are centered on the distribution of points in space and the relation with their direct neighbors. The Delaunay triangulation considers all points as vertices, highlighting the importance of the point in the format of geometric figures. Another technique is the division of space into polygons in which the original points are not part of the vertices, but are located within the polygons. That is the case for the Voronoi diagram algorithm. This section explores how these tools can be useful for visualization and composition of images for the study of PRNGs randomness.

6.2.1 Voronoi Diagrams

The Voronoi diagram algorithm is capable of evidencing the distance between points. Diagrams with large areas denote isolated points in space whereas the agglomerated points occupy a small portion of the space. It is expected that the correct selection of the diagrams' attribute create good descriptors of the structure of the spatial network.

Given a PRNG sequence in the plane, its Voronoi diagram divides the plane according to the nearest-neighbor rule: each point is associated with the region of the plane closest to it. In the radius connection approach in complex network, points are eliminated from the network if they do not have any neighbors and it is possible to estimate the maximum number of vertices that a graph can have. In the case of the Voronoi diagram, all points are considered. The maximum number of vertices is not known before the algorithm is executed as the points are not the vertices of the resulting polygons. The number of polygons in the diagram is the number of points because each point has a polygons assigned to itself.

Image 23 is a Voronoi diagram originating from the K -logistic map algorithm with 600 points, consequently, it consists of 600 polygons. In 22a, generated with $K0$, the polygons in the central region are noticeably larger than those in the central region of 22b, generated with $K4$. There are many values in the regions near 0 and 1, making the polygons in this region very small and with little variation of area. These polygons on the lateral edges of the first image have almost parallel edges while in the second figure these polygons have much more inclined edges forming them.

Figure 22 – Voronoi diagram originating from K-logistic Map

(a) $K0$; 600 points

(b) $K4$; 600 points

Source: Elaborated by the author.

The characteristics of the polygons that change according to the value K are indications that this approach might deliver important information about the sequences. At first, a distribution of the areas by location in the image would give more details about these polygons. But since the vertices are located in the most diverse regions, there would be a great loss of information because many polygons would be cut between two or more regions. The challenge is studying the spatial distribution of these polygons without compromising the information contained in them.

A possible solution is painting the polygons according to their area and from the whole figure observe if they are enough information for some basic classification. The ideal scenario would be having all the polygons originated from the same value of K parameter having the same color. Then, if the experiment has Voronoi diagrams originated from $K0$ to $K4$, five colors would be the minimum to perform a classification. The other decision to be taken is the maximum number of colors. Since the parameter has 5 values, a multiple of 5 should be, at least, a reasonable value. In the worst case, each of these polygons can have a different area which would make the task of analyzing these images very complex. In the scenario of a 1000 points sequences and 40 sequences per K , each polygon colored differently would need $5 * 40 * 1000 = 200,000$ colors. This would not result in a good classification in the pattern recognition phase. The ideal maximum number is the number of colors that turns difficult to classify without the need of implement the iterations until the real worse case. An extremely complicated case would be if a diagram presented all its polygons with similar areas, making them all have the same color. If this behavior is repeated within each parameter K , then there are 40 different colors. That was the worse case scenario tested. Image 23 shows an example of resulting colored Voronoi diagram for $K0$ until $K4$ using 5 colors.

In figure 23a the region with the purple color is much more continuous than for the case of figure 23b. In the initial part of the image related to $K0$, there are large purple regions while in the figure for $K4$, the same region has the predominance of yellow and green colors. In the upper region in the case of $K0$ it is possible to see much more yellow and green than on $K4$ image and the red color is more identified in the lower part of the $K0$.

Before making a general statement, it is necessary to analyze whether the color setting in the images is not just a coincidence. A more deep analysis is described in the results session. At this stage, one only seeks to understand if the colors at have some important information. A color histogram per image is tested and this histogram is subjected to in a PCA (Principal Component Analysis). As each K has 40 images and each image has three channels, each channel was divided in 12 bins, resulting in 36 values per image. The PCA as a preliminary analysis can be seen in figure 24.

Both the PCA and score plot of the colors histogram per channel show that $K0$ is the easiest to be separated from the other groups as K increase they are more similar to each other however only this color histogram analysis is enough to point out which is the weakest sequence.

Figure 23 – Voronoi diagram originating from the K-logistic Map colored with 5 colors

(a) $k = 0$; 1000 points (b) $k = 4$; 1000 points

Source: Elaborated by the author.

Figure 24 – Analysis of Voronoi diagram for 5 colors

(a) PCA

(b) Box-plot

Source: Elaborated by the author.

In the box plot we observe $K0$ is well separated from the other groups. $K1$, $K2$, $K3$, and $K4$ have a large degree of overlap. $K3$ and $K4$ are in closer regions but $K4$ has fewer values within the second and third quartiles. The PCA score plot and the box plot for the other tested numbers of colors used can be found in Annex I.

This set of experiments was performed with a small number of images for K and with a

reduced amount of possible values of K just to see if the color actually carries some information about the PRNG sequences. Since the PCA showed that the color histogram already presents agglomerations of the networks per K , the experiments with Voronoi diagram were used to generate part of the results of this research. The complete experiments are described in the results section.

6.2.2 Delaunay Triangulation

The Delaunay triangulation frames points and their neighbors in triangles' vertices. Thus the more neighbors a point has, the greater the number of triangles with vertices at that point and the smaller the areas of those objects. Due to this characteristic, Delaunay triangulation was used to stress the clusters in the graphs and to analyze the distribution of the size of the edges. It is known in advance how many vertices the tessellation will have before the execution of the algorithm, but the number of triangles is not known.

Figure 25 shows an example of Delaunay triangulation for $K0$ and $K4$. Figure 25b, the points were more distributed, this causes the edges of the triangles at the horizontal edges of the image to be much larger than the edges at the same location at figure 25a, where many values are close to 0 and 1. At the vertical borders of the image, the edges of the triangles are larger in $K0$ because the accumulation of points near 1 and 0 and the central region of the image does not have as many points. The borders of the image to $K0$ are more convex because the connections are made between values not so close in value but generated in close iterations.

Figure 25 – Delaunay triangulation originating from the K-logistic Map

(a) $k = 0$; 1000 points

(b) $k = 4$; 1000 points

Source: Elaborated by the author.

The study of the distribution of triangles by area would have the same problem as in the case of Voronoi diagram: loss of information because many triangles would be divided between two or more regions. So the solution was also to color the triangles according to their areas. For the definition of number of colors in the Delaunay triangulation, it can not be done analogously as in the previous case. In the case of Voronoi diagram, each polygon is associated with a point, being possible to know how many polygons the diagram has before executing the algorithm. In the case of triangulation, the points are the vertices of the polygons. So the maximum number of colors has dependency not only with the number of points but also with the spatial relation among them. The number of triangles of a triangulation of n vertices depends on the number of vertex h in the convex envelope. As shown previously, the networks presented have an average degree close to 7 and one characteristic of the Delaunay triangulation is that in average, each vertex is connected to more 6 vertices. Since there are 5 classes - 5 values of K - this is the initial number of colors. But the maximum number is defined considering that each network has 1000 points, these points have in average 6 connections and each connection is counted twice. This was rounded to the number 3000, the maximum number of colors.

Figure 26 shows the image of the Delaunay triangulation for $K0$ and $K4$. The color distribution in the figure 26a and figure 26b are far more varied than the Voronoi images. In images 26a, there are continuous regions of the same color and it is possible, just by looking at the image, to indicate the predominant color. In the images of the tessellation, it is more difficult for two triangles, sharing two vertices and one edge, to have the same area. In figure 26a, a greater presence of red, the color associated to the triangles of smaller area, is dominant than in the image 26a. In this scenario it becomes even more necessary to analyze if the color in the images is important even if they are in such different regions.

As in the case of the Voronoi diagram, the PCA was also applied in the Delaunay triangulation. As seen in images 27, Delaunay triangulation can identify the weaker class, the points in 27a. This information is also shown in the 27b, where one can see the similarity of the areas of $K2$ and $K4$, although $K4$ has the highest mean.

Figure 26 – Delaunay triangulation originating from the K-logistic map colored with 90 colors

(a) $k = 0$; 1000 points (b) $k = 4$; 1000 points

Source: Elaborated by the author.

Figure 27 – Analysis Delaunay triangulation for 5 colors

(a) PCA

(b) Box-plot

Source: Elaborated by the author.

Even with a reduced number of images per class and a classification based only on the color histogram for RGB channels, the computational geometry approaches were able to indicate which sequence was generated by the K parameter that generates the least random sequences. Because these images present polygons in the most diverse regions of the image, the method used for the final experiments takes into account not only the color of the polygon but its position in the plane and the color of the polygons.

6.3 Other inspected techniques

The previous section proposed the use of complex networks and computational geometry to create descriptors for PRNGs. Some related work proposing other ways to create descriptors can be found in literature and will briefly be discussed in this section. The boundary shape analysis(BACKES; CASANOVA; BRUNO, 2009) and texture analysis(BACKES; CASANOVA; BRUNO, 2013) apply complex networks for image analysis modeling image's pixels into networks. In both approaches, points are connected using different radius values. In this way several networks are created with the same set of nodes. The characteristics of these networks are documented in a descriptor for classification. The third method, the visibility graph(LACASA *et al.*, 2008), presents a method for converting time series into graphs. The rule for the conversion is based on the "vision" that the points have in the time series plot. The details of the methods are described below.

6.3.1 Boundary shape analysis

An interesting use of networks consists of measuring the structural properties of evolving networks in order to characterize how the connectivity of the investigated structures changes along the process. In the method proposed by Backes, Casanova and Bruno (2009), a boundary shape is modeled into a small-world complex network. This view is coherent since the literature presents various tools to analyze images and objects using boundary shape, and the basis of most of them considers the shape as a set of connected points. The sequence of pixels that are part of the boundary in an image are considered the nodes. The edges are the connections between the pixels and they dependent on the distance between points within a radius. The radius values increase forming different and increasingly connected networks as seen in detail in figure 28.

Figure 28 – Complex network-based approach for boundary shape analysis

Source: Backes, Casanova and Bruno (2009).

The peculiarity of this method is that it captures only average degree and maximum degree for each network created over the evolution. Consequently, these measurements can be used to compose a shape descriptor. The following is the representation of these measures as a set of descriptors:

$$\varphi = [(< k > T)_{j-(j-1)}, (< k > T)_{j-(j-2)}, \dots, (< k > T)_{j-1}, (< k > T)_j]$$

Where $< k >$ is the average degree of the network created when used a specific radius, T is the max degree of the respective network and j is the number of radii used.

This technique is quite interesting because it facilitates the investigation of how the same set of nodes can create different networks and how only two measures are able to compose efficient descriptors. Another feature is that the method presents good results with few variations of the radius (13 variations). No library for graph modeling is necessary since the algorithm only uses two simple measures. Networks are represented in weight matrices, making the method easy to parallelize and low-cost storage in RAM. It has been shown that the technique is robust, noise, scale and rotation invariant.

This method was adapted to the test scenario of PRNGs randomness. Instead of pixels, pseudo-random sequence values and their iterations are the nodes. The variation of the radius was made in several versions: the same variation proposed by [Backes, Casanova and Bruno \(2009\)](#), bigger ranges, bigger steps, smaller steps, and without normalization of node size, different from the original algorithm. These changes are useful to understand if only the analysis of two measures in networks generated by the same PRNG is enough for the recognition of different values of K .

6.3.2 Texture analysis

A similar concept is applied in [Backes, Casanova and Bruno \(2013\)](#) for texture analysis. The method uses degree measurements in a dynamic evolution network to extract information an image. In texture analysis, it is fundamental to recognize the relation of a pixel and its neighbors, and its influence in a global perspective in an image. Therefore, a mechanism that associates pixels with its neighbors by means of a connection might be a plausible approach for texture since it takes into account the complex visual patterns such as entities, bright, color, deepness, size and the relation among pixels.

The complex network-based approach for texture analysis investigated how nodes and edges can carry relevant information a texture image. In order to do that, the texture image pixels are modeled in nodes and the edges represent similarities between pixels. The similarity between the pixels is a measure that considers the Euclidean distance between the pixels, the connection radius and the maximum value of the intensity that a pixel can have. The set of features extracted from a set of nodes connected by a crescent radius, as shown in figure 29, is used in the classification process.

Figure 29 – Complex network-based approach for texture analysis

Source: [Backes, Casanova and Bruno \(2013\)](#).

Note – Image pixels are converted to nodes while edges are the connections between pixels. The value used to calculate the connection radius is derived from a threshold T that changes and generates new networks.

This method only uses the average degree and the information originated from the difference between the feature vectors generated with a radius over the original network. The node degree distributions are good representations for different textures but these parameters do not have a great influence in the final results. The research shows that the most of texture information lays in the beginning of the dynamic evolution process. The method is also very robust, invariant to rotation, the pattern recognition presents a great classification for the specified classes and great capacity to work with both micro and macro texture.

In this work an adaptation of this method was also made. Instead of pixels, points were used on a 2D plot and the edges are distance relations between the points. The adopted radius are the same as those of the previous subsection. Since the node in a random sequence has no intensity as in the pixels it has, the connection condition is dependent on the maximum and minimum values that the PRNG can generate.

6.3.3 The visibility graph

A PRNG output can be seen as a time series if one notices that each iteration of a PRNG happens at a time x and each time x generates a value y . This interpretation required a bibliographical research on methods of converting time series into graphs. A set of papers propose a remodeling using the values of the time series and the time as information for the

creation of the points and edges related to the distance between the points.

An example is the work of [Lacasa *et al.* \(2008\)](#). It describes a straightforward computational method that converts a time series into a graph called visibility graph. The main idea is to investigate if graph theory can be useful as a way to investigate time series. The resulting graph inherits several properties of the time series, and its study reveals nontrivial information about the series itself.

As seen in figure 30, a time series is converted into vertical bars. The time is the x axis and on the y axis are the values of the time series. The algorithm links every bar (every point of the time series) with all those that can be seen from the top of the considered one, obtaining the associated graph. In this graph, every node corresponds to data of the time series and two nodes are connected if visibility exists between the corresponding data, that is to say, if there is a straight line that connects two bars providing a "visibility line" that does not intersect any intermediate bar.

Figure 30 – Summary of the visibility graph method

Source: [Lacasa *et al.* \(2008\)](#).

The authors establish the following visibility criteria: two arbitrary data values (t_a, y_a) and (t_b, y_b) will have visibility, and consequently will become two connected nodes of the associated graph, if any other data (t_c, y_c) placed between them fulfills:

$$y_c = y_b + (y_a - y_b) \frac{(t_b - t_c)}{(t_b - t_a)} \quad (6.3)$$

The kind of time series considered in the aforementioned work is limited to periodical series. However, a time series that is originating from PRNGs would need an immense number of points until it reaches its period. In the experiments described in the next chapter, a suited version was implemented. This adaptation runs on time series without concern about their period, but their number of points. These experiments are carried out to determine if this technique is also useful for constructing graphs that contain information about the PRNG that generates the series.

6.4 Discussion

The two approaches in this chapter provide a new insight into the problem of classifying PRNGs. The use of graph theory in randomness tests is not explored. Although the most popular test suites today, NIST and DIEHARD, make a very relevant test battery, when the objectives of these tests are analyzed, they do not contemplate the spatial relationship of the points. There is a scrutiny of the mathematical properties of the sequences but there is a test where each point is studied taking into account both the generated values in iterations near this point and what information can be deduced when these points are connected by means of neighborhood relation. This observation opens up many possibilities for suggesting improvements to the PRNG testing.

The modeling of PRNG sequences in complex networks is an innovation in the visualization and analysis of these sequences. When the values of this type of sequence are connected in a graph, characteristics, like distribution, are evident. The edges of the graphs are the source of interesting information about how values close to 0 and 1, the PRNG boundaries, behave. The measurements of complex networks of both unweighted and weighted networks give a more global view of the resulting graph. Measures involving walks can make one point be related to all others in the sequence, evidencing the influence of the number of edges in the network. Community measures relate how points are clustered even though the pseudo-random sequences are evenly distributed. The conversion of PRNG sequences into networks and subsequent extraction of measurements is a new way of trying to understand the peculiarities of the algorithms that generate such sequences.

PRNG sequences are also studied in the light of computational geometry, a field of research where the algorithms are specialized in making relations of points with space. The proposed methodology showed that is indispensable the exploration of geometric relations of the PRNG sequences focusing on the area of the resulting polygons. The Voronoi diagram is used to better understand the relation of points in a 2D space and Delaunay Triangulation is used for more local analysis since the triangles are formed by adjacent points. The triangulation makes the hubs more evident, giving information about where nodes with more degrees are, in terms of randomness study, where the points are not uniformly distributed. These two points of view are complementary and are promising given the PCA charts seen in the section above. These algorithms are fundamental to the final results of this research.

The other inspected techniques are also complex network applications. These approaches were proposed for the study of images and time series but with small adaptations, they were applied in the study of randomness of PRNGs. The first two approaches highlight the importance of studying a set of points in a dynamic network. With few measures, these approaches have proven quite promising in their original applications. The method of temporal series conversion in a graph makes a transformation that, according to the original publication, results in a network that brings characteristics of the sequence that generated it. This approach was used to test whether this conversion can also evidence information from PRNGs.

All suggested approaches were tested with various adaptations. The changes were necessary for the classification step to have the best descriptors. In network generation, each configuration of an algorithm is considered a class. Thus a set of sequences of a PRNG forms the samples of a class. After modeling, there is the selection of the most relevant measures at the same time that the classification algorithms that best solve the problem are selected. This entire pipeline is discussed in the next chapter.

RESULTS

Graphs are structures without defined space, and depending on the modeled phenomenon, they can be examined as complex networks. An event in a physical space represented in a complex network has its information, Euclidean distance or maintenance cost, for instance, converted into characteristics of the resulting network's edges. In complex networks, the relation of the nodes is the most important characteristic for the understanding of the considered phenomenon. When inserted into a Cartesian plan, graphs become more space-dependent. The graphs become structural networks and their geometric representations are additional characteristics, since the graph format is related to the constraints in the space. The nodes are described by the variables that defines their positions and the edges are responsible for the structure of the graphs. The experiments of this research are executed considering two forms of modeling phenomena in graphs: complex network theory and computational geometry.

The investigation of patterns in graphs generated from pseudo-random sequences is the challenge of this work. The presented methodology showed that the K-logistic map graphs originating from $K0$, that is, non pseudo-random sequences, can be separated from the graphs originating from larger K s. The following experiments give focus on separating the networks by K s values above 0. The intention is to inspect whether the proposed methods can also be applied in classic PRNGs analysis. In this chapter, methodologies are evaluated using Machine Learning (ML) classifiers. For this, the PRNGs are divided into classes. The methodologies will be compared according to the correct separation percentage of these classes. This chapter characterizes the experiments, details the experiments setup for reproducibility of the results, and presents the discussion of the final results.

7.1 Complex Networks Analysis

The K-logistic map is an interesting case study because the change to suitable parameters of K generates pseudo-random sequences. Starting from the premise that there is a relation of

the characteristic of the sequences with the parameter K , these sequences can be separated into classes: one class by K . This suggestion makes the separation of sequences by K a classification problem.

Theoretically, the K -logistic map can be executed using infinite values of K s. However, when implemented by a computer, K has a limit set by machine accuracy. For the purpose of this work, K values ranging from 0 to 9 are sufficient. This range is chosen also considering that the randomness tests of this interval in NIST and DIEHARD are already documented. The K -logistic map randomness test table is reproduced in appendix D.

7.1.1 Main measurements

As in the methodology, the start point of the experiments is the analysis of the degree distribution. One of the main characteristics of structural networks is their Gaussian degree distribution. Comparing the distributions between classes K in figure 31, one can see that they are in the same format and become symmetrical from $K3$. The asymmetry of the distributions of $K0$, $K1$ and $K2$ occurs because the networks of these classes have hubs along their structure. Since the networks above $K3$ have basically the same degree distribution, the measures that have this distribution as a basis may not be efficient measures for group separation.

Figure 31 – Degree distribution of K -logistic map by K value

Source: Elaborated by the author.

In structural networks, the position of nodes in space is of extreme importance. In many geometric networks, the distance between points is the main characteristic considered for the establishment of connections. To understand the distribution of these points in the Cartesian plan, the space where the networks are drawn was divided into 11 horizontal bands. This number of bands has been set so that the 6^o band is exactly in the central region of the image. Thus, taking

the top and bottom of the images are divided into 5 bands, placing the 1^o band in the region where the values are close to 0 on the y axis and the 11^o band in the region with values close to 1.

Figure 32 shows the violin plots of the degree distribution per band from $K0$ to $K5$. The complete distribution with K ranging up to $K9$ can be seen in Annex F. These distributions were made by analyzing 100 networks per K , each network originating from 1000 points.

Figure 32 – Degree distribution for band

Source: Elaborated by the author.

The non-uniformity of the value distributions is sufficient for the recognition of the networks generated from $K0$. In distribution 32a the 1^a band is the one with the occurrence of the highest possible degree between all classes of networks. In the context of PRNG testing, if the network has hubs, this means that in a short space of time, many approximate values have been found. In the 2^a band there are many clusters as well. The nodes in all the upper bands of the networks present degrees greater than the average. In the central region are the nodes with the lowest degrees. The curvature connecting mean values in $K0$ is a network-type signature created by non-uniform sequences with highest occurrence of values close to the PRNG minimum and maximum.

In distribution 32b, it is also in the 1^a band where the highest degree of this set of networks occurs. The maximum degree per band decreases until it reaches the three bands closest to the center of the distribution. There is a curvature at the end of the distribution. Note that the closest bands to the outermost band, 2^a and 10^a bands, have the highest average degree, with the peaks of the curvatures. Figure 32c shows a more uniform distribution than $K0$ and $K1$. There is only a small variation between the highest degrees per band. The two edge bands have, on average, lower degrees, but this is understandable since the nodes that are close to 0 and 1 can only be connected to values greater or smaller than them, respectively. In other bands, the only restriction of space is the radius established in the creation of graphs. This time, the highest degree between the networks is in the upper bands of the Cartesian plan.

Figures 32d, 32e and figure32f have very similar distributions. In the three images, the highest degree per band forms a curvature from the 1^a band to the central band. It is in these two sets of networks that the challenge of identifying classes begins: $K4$ was approved for all NIST and DIEHARD tests. $K3$ was approved only in NIST but failed only 1 test of the eighteen DIEHARD tests.

As shown in the previous chapter, the points are connected by a radius of 0.12. According to initial experiments, this radius connects most points of the initial pseudo-random sequence without resulting in a super connected network. Nevertheless, this is not a guarantee that all points will be connected. The description of the experiments record the size of the initial sequence but not the number of points in the final graph. Another feature related to the connection radius is the existence of the leaf nodes. These nodes bring information about numbers that have only an approximate value within the number of iterations covered in the radius. Figure 33 shows the distribution of the number of points that were eliminated from the networks after the connection operation. The figure also exposes the position of the leaf nodes after the connection.

Figure 33 – Distribution of isolated and leaves nodes in K-logistic map by band

Source: Elaborated by the author.

The distribution of the eliminated nodes in the edge regions in image 33a is much smaller than in the other classes network. In addition to the border regions having high degree nodes, as seen in the degree distribution per band, the radius around the points of the those regions does not cover values around the number. This is because PRNG has a limitation of values between 0 and 1. Even with this restriction, in $K0$, few points are eliminated from the network, as few are leaf nodes. The radius is also sufficient so that there are few leaf nodes in these regions. The

regions between the edges and the center have many leaves. The graph is divided into two peaks by the central region as well as the border regions did not eliminate many nodes but has leaf nodes.

Figure 33b presents another trend in the edge region close to 0. The change from $K0$ to $K1$ is important enough so in that region, the number of leaf nodes is already much larger than $K0$. More numbers are also eliminated precisely because this region does not have as many clusters as $K0$. The growth of leaves and isolated points is due to the issue discussed in the previous paragraph of the node having less "space" for radius connection. However, at the top edge of these networks, just as in $K0$, there are few nodes eliminated from the final networks. Compared to the lower border, the upper edge of $K1$ exhibits far fewer leaves but still more leaves than the same region in $K0$.

From $K2$, the distribution of the number of isolated points is similar. The eliminated points are close to 0, so the networks in these classes have similar numbers of nodes. The distribution of the number of leaf nodes presents a behavior quite different from $K0$ in the border regions. One can notice that in figures 33c, 33d and 33f as well as the plot representing $K0$ have peaks separated by the central band, although they are low peaks.

Figure 33e has a peak in the central band, highlighting it of the others. $K4$ also differs from the others because the other networks originated from more distributed sequences have more leaf nodes in the upper region of the plan. The lower region of $K4$ resembles the same region in $K5$, but $K5$ has much more leaves than $K4$ in the upper region of the structural networks. Analyzing this same distribution for all K in Annex G, it is noticed that this shape of the $K4$ border regions resembles $K6$, $K7$, and $K8$. The same region in $K9$ resembles $K5$, but $K5$ has more leaves. The distribution of the isolated numbers is very similar in format, changing only the amount of points.

These distributions demonstrate that the networks bring relevant information about the PRNGS sequences. K parameters that do not generate random sequences are identified by their characteristic distributions. These distributions give a better understanding of the sequences but are not sufficient to separate the networks by K parameter. The search for suitable measures is further explored in the following section.

7.1.2 Experiments scope

In addition to the method of converting PNRG sequences into points in the plan and the condition of creating edges, there are other variables that need to be specified. This delimitation has impact on processing time of networks, hardware configurations of the used computers, percentages of accuracy, among others. A key part of the research is to test which parameters best fit the problem and how each parameter impacts the final result.

7.1.2.1 Comparing networks

Network measurements are generally in two formats: a number, when it is a global measurement, or a dictionary, when there is a measure calculated per node. In this second format, the dictionary key is the node and the value is a feature of the node. Forming descriptors only with global measures is simpler since it is a number for each network, independent of the number of nodes. However, a descriptor with only global measures is not sufficient for the level of difficulty of the problem addressed in this work. For measures be used in the classification step, all networks must have the same number of descriptors. In order to ensure this regulation, the characteristics of nodes are summarized in their statistical moments or in frequencies. In the case of statistical moments, each network is analyzed individually and for each measurement, 5 moments degree describes the sequence, independent of its size. In the second case, it is necessary to analyze all the networks in order to be able to point the highest and the lowest value among all the values of the networks, thus having a range to create the frequency distribution. Better results are obtained when calculating the statistical moments.

Two Python libraries calculated the degree moments: *numpy* and *scipy*. Reinforcing that the focus of this project is the networks, the two libraries were tested and the one with the best classification was applied in the later experiments. For this calculation, we used the *moments*, *skew*, *kurtosis*, and *entropy* methods of the *scipy* library and the *mean* method of the *numpy* library.

7.1.2.2 Connected components

Some measurements are only calculated using the largest connected component. The concern is again the measurements that calculate paths in the graphs. The applied libraries do not perform their calculations if the input graph is disconnected. Early experiments carried the networks into memory, extracted the largest component of the networks, and used them when there was such a restriction in the implementation of the measure. With the first problems of memory overflow, it was considered to work only with the largest connected components. For this, it was analyzed whether the use of only the largest component would be disadvantageous to the classification of networks.

A basic set of measures was used as descriptors of 100 networks for K network, with K ranging from 0 to 5. The networks were created from sequences of 1000 points. Table 2 shows the obtained results. The notations *fn* and *gc* are represent *full network* and *giant component* respectively. *_met* is used for the results obtained from the *scipy* library, *_mon* is used for *numpy*.

Table 2 – Cellular Automaton Rule 30

Model	SMO	NaiveBayes	MLP	LDA	Logistic	Regression	MultiClass	Decision Table
<i>fn_met</i>	58.93	56.27	54.73	59.80	58.21	60.85	61.05	55.19
<i>fn_mon</i>	57.13	52.13	52.60	56.73	58.25	60.56	60.57	54.69
<i>gc_met</i>	61.53	58.00	58.67	59	58.25	60.56	60.49	54.69
<i>gc_mon</i>	57.73	53.93	54.60	55.53	56.13	58.22	58.71	54.18

Source: Research data.

Thus, subsequent experiments were done using the largest connected component and *scipy* library methods.

7.1.2.3 Classification setup

In ML it is important that a set of descriptors be able to correctly describe not only the data used in the experiments. The classification step should ensure that the descriptors can be used in data never seen during modeling and are still able to classify them correctly.

The software applied in the ML analysis is Weka¹. The experiments configured in Weka separate the data into two sets: training samples and test samples. The algorithms implemented in Weka use the set of training samples to learn how to classify the classes. The test set is not used during training the classifier. To evaluate whether the algorithm has learned how to routinely sort a sequence of descriptors, the test set is used. The algorithm receives the data from the test suite without knowing which class that data belongs to. The accuracy of the algorithm is percentage of correctly predicted classes.

Another method used to train classification algorithms is cross validation. The dataset can be randomized or kept in the same order and divided into N (almost) equal size partitions. One of these partitions is chosen for testing and the other N-1 partitions are used in the training phase of the classifier. The process is repeated N times to get an average of the metric. To diversify learning, the process of dividing into N folders and doing training and test groups is repeated M times and take average of the metric. Weka has the configuration to set how the algorithm should try to learn from the data: cross-validation procedure, where you must specify the number of folders, or training/test procedure, where the scientist must specify the percentage of data for training and if the data should be randomized or kept in the order that it is. By default, the cross validation procedure is repeated 10 times (BROWNLEE, 2014).

Tests were performed to pinpoint which approach best ranks the networks. The methodology that presented the best results was that using cross validation.

¹ <https://www.cs.waikato.ac.nz/ml/weka/>

7.1.2.4 Number precision

Another point raised during the preparation of the experiments was the importance of the decimal places of the descriptors. The values returned in Python were float with 16 decimal places. This accuracy could cause a low percentage of classification by segregating the values into many small and specific sets. Therefore, another comparison needs to be made to choose the most appropriate decimal precision. The test is done with networks of classes with K varying from 0 to 5, 300 networks by K . The results improve as the accuracy increases. Thus the experiments are performed with the measurements having 16 digits of accuracy.

7.1.3 Network measurements

The first measures selected were based on bibliographical research. Measures of spatial networks have been listed, particularly the measures that differentiate the structural networks from other topologies. Most previous studies have ignored the geography of networks, concentrating instead on other issues. To the graphs applied, the geometry matters greatly, and to ignore it is to miss some of these systems' most interesting features. The remainder of the measures were included after some tests because they have proven to be efficient for the pattern recognition step.

Many measures have been eliminated because they are correlated with other measures, are computationally very costly or are not widely used measures in the academic literature. Some measures are only applied to unweighted network for its computational cost. All the selected networks measures are compiled to compose a descriptor. It is expected that these measurements may well describe the networks characteristic with the objective of facilitating the classification phase (CASTRO; PRADO, 2002). These descriptors may be used to identify and distinguish classes. It is presupposed that in case that the characteristics that originate some pattern are identified, they are extended to algorithms whose sequences may not be inferred. The following are the measurements used for the initial tests. They are separated according to their characteristics. This separation is done only in order to organize the measures, it is not a study of grouping of measures using some mathematical criterion.

1. Centrality

- Closeness
- Betweenness
- PageRank
- Eigenvector
- Katz
- Harmonic
- Information

- Random walk betweenness
- Load

2. Degrees

- Average Degree
- Average nearest neighbor
- Average Weighted
- Hierarchical
- Occurrences
- Variance
- Maximun
- Distribution

3. Statistics

- Eccentricity
- Average clustering
- Transitivity
- Diameter
- Global efficiency
- Average shortest path

4. Modularities

- Fastreedy
- Walktrap
- Leading eigenvector
- Infomap
- Label propagation
- Multilevel
- Spinglass

5. Coefficients

- Pearson
- Pearson degree
- Rich club

- Local efficiency
- Shannon
- Spearman
- Burts constraint
- Kleinber hub
- Kleinber authority
- Vulnerability

7.1.3.1 ML algorithms selection

There is an extensive list of classification algorithms in ML. Each can be applied to many categories of problems. In order to point out the most appropriate algorithms for classifying the K-logistic map classes, a list of algorithms is tested with the each ML algorithm having different parameters specification. The classes ranged from $K0$ to $K4$. The selected algorithms were those that even with a list of weak descriptors, achieved a percentage greater than 50%. The selected algorithms are: Naive Bayes, LDA, Simple Logistic, SVM, Classification via Regression, MultiClass Classifier, Decision Table, Random Tree and Random Forest. The Bayes Net, Logistic, and Vote algorithms were eliminated because the tests were not performed completely, given the computational cost, measured here in time. The list of all tested algorithms and their percentages are listed in Annex E.

7.1.3.2 Measurement selection

All the measures cited above were used as descriptors of the networks. However, it is necessary to indicate which of those measures has more significance, that is, producing the best separation of classes. It is expected that the identification of the most important measures will reduce the execution time of the experiment with the elimination of the least significant measures.

Attribute evaluation algorithms are used to point out the most significant measures (HALL *et al.*, 2009).

- Subset Eval(SE): evaluates the worth of an attribute subset by considering the individual predictive ability of each feature along with the degree of redundancy between them.
- Correlation Filter (CF): evaluates the worth of an attribute by measuring the correlation (Pearson's) between it and the class.
- Wrapper Subset Eval (WSE): evaluates attribute sets by using a learning scheme.
- Info gain attribute Eval(IGR): evaluates the worth of an attribute by measuring the information gain with respect to the class.

- Gain ratio attribute eval (GR): evaluates the worth of an attribute by measuring the gain ratio with respect to the class.
- One R attribute eval (OR): uses a ruled-bases system to select the most relevant attributes.
- ReliefF(RE): evaluates the worth of an attribute by repeatedly sampling an instance and considering the value of the given attribute for the nearest instance of the same and different class.
- Symmetrical Uncertainty (SU): evaluates the worth of an attribute by measuring the symmetrical uncertainty with respect to the class

These attribute evaluation algorithms have been configured with the following search methods:

- Best First: searches the space of attribute subsets by greedy hill-climbing augmented with a backtracking facility.
- Greedy Stepwise: performs a greedy forward or backward search through the space of attribute subsets.
- Ranker: ranks attributes by their individual evaluations.

For each measure whose method output is a dictionary, 5 variables are created to store the 5 statistical moments of the dictionary values. For this reason, some variables appear more than once but with the end identifying which statistical moment it is representing. Each algorithm analyzed the same set of descriptors. Table 3 lists the eleven elements with the highest score by algorithms.

Table 3 – Complex network attribute evaluation

SE	SE	WSE	CF	GR	IGR	OR	RE	SU
Best First	Greedy Stepwise	Ranker	Ranker	Ranker	Ranker	Ranker	Ranker	Ranker
entropy	entropy	mod_sp	info_centralty1	clustering_transitivity	rw_betweenness1	rw_betweenness1	Entropy	harmonic1
clustering_transitivity	clustering_transitivity	harmonic2	mod_sp	harmonic1	clustering_transitivity	clustering_transitivity	shanon	clustering_transitivity
load5	load5	betweenness1	mod_sp	knn_average2	harmonic1	harmonic1	shanon	rw_betweenness1
harmonic1	harmonic1	betweenness2	ent	rw_betweenness1	burts_constraint1	burts_constraint1	degree_pearson_cor	burts_constraint1
betweenness5	betweenness5	betweenness3	shanon	degree_pearson_cor	clousness2	clousness2	assort_coefficient	degree_pearson_cor
knn_average2	knn_average2	eigenvector4	knn_average1	assort_coefficient	load5	load5	mod_ml	assort_coefficient
info_centralty1	info_centralty1	eigenvector3	knn_average2	betweenness5	betweenness5	betweenness5	betweenness2	betweenness5
rw_betweenness1	c	eigenvector2	knn_ml	burts_constraint1	load5	burts_constraint5	rw_betweenness1	load5
burts_constraint1	burts_constraint1	harmonic4	load2	load5	clousness2	shanon	load2	clousness2
burts_constraint5	burts_constraint5	harmonic5	betweenness2	clousness3	ent	ent	mod_betweenness	variance
mod_sp	mod_sp	eigenvector1	mod_fastgreedy	mod_sp	burts_constraint5	averageDegreeCentrality_ig	info_centralty1	shanon

Source: Research data.

The measures used in the following experiments are the measures that were most pointed out by the nine attribute evaluation algorithms configurations. The experiments were also done with weighted networks created from the K-logistic map sequences.

Table 4 – Complex Network measures

Measurement	Unweighted networks	Weighted networks
Average degree	X	X
Degree Variance	X	X
Second Moment Degree	X	X
Shannon Entropy	X	X
Transitivity	X	X
Average Clustering	X	X
Node connectivity	X	X
Degree centrality	X	X
Assortativity coefficient	X	X
Pearson coefficient	X	X
Spearman coefficient	X	X
Degree Pearson correlation	X	X
Fast-greedy Modularity	X	
Betweenness centrality	X	X
Random Walk Betweenness centrality	X	
Eigenvector centrality	X	X
Rich club coefficient	X	X
Average degree connectivity	X	X
Number of nodes	X	X
Hierarchical degree	X	
Density		X
Average weighted degree		X

Source: Research data.

7.1.3.3 PRNG analysis as a classification problem

PRNG test suites require a large amount of data to perform randomness tests. These data, pseudo-random sequences, are organized in a variety of ways depending on the purpose of the test.

The experiments carried out in this work create a relation between close values generated within a range of iterations by specifying a radius. When a sequence generated by the same

K is created, the hypothesis is that the graphs resulting from these sequence have information that might lead to unmask K . It is assumed that this information will be repeated over iteration windows. Nevertheless, unlike the test suites, once a sequence is subdivided, it loses information. The values at the end of the sub-sequences are not connected to all the values that were actually created within an interval of interactions but ended up in the beginning of the next sub-sequence. Because those values stay in separate sub-sequences, consequently, they will be in separate networks.

Many measures have a high computational cost, and mainly measures related to walking consume a lot of memory. It is very simple to create random sequences and store them, the problem is to load all those values into memory, model them as networks and calculate the network measurements. Experiments in this research have frequently reported memory overflow errors. That is why some strategies were used to overcome this challenge. The selected strategy was the selection of the size of cut windows and analysis of the classification change of the networks according to the size of the cut window. The windows ranged from 1000 to 10,000. The maximum size that did not show a memory overflow error when running in multiprocessing was 3000. So, tests with the 3000, 2000, and 1000 iterations window were documented.

The networks were separated into two groups. The first consists of the classes with K varying from 0 to 4. The other group has the variations of K between 5 and 9. These separations are due to the fact that the lowest values of K are the simplest to identify. In the first group, only with basic experiments, it is possible to point out the class of PRNGs stronger and weaker. In the second group the challenge is more difficult. These sequences present very similar degree distributions, number of leaves and number of points not included in the network. That fact that these networks have such correlated measures is one of the characteristics that guarantee the quality of the PRNG.

The values of the measurements extracted in the networks generated by the same K are the descriptors. These descriptors may be sufficient to classify the networks. But before these descriptors are passed to the classification algorithms, it is important to observe how these values relate to the principal components analysis and transformation of the data.

The figures 34 and 35 show that 1000 points make a reasonable separation of the groups between K_0 and K_4 . However, in groups between K_5 and K_6 , the box-plots are in very similar locations and all classes are clustered. These measures were analyzed by PCA but no component was shown to be most representative.

Figure 34 – Analysis of the 1000 points networks descriptors

Source: Elaborated by the author.

Note – Eigenvector- importance of the measurements in the separation of classes - and the box-plot of this separation using the detected principal component from K0 to K9. The descriptors originated from complex network measurements.

Figure 35 – PCA plot of networks with 1000 points

Source: Elaborated by the author.

Figures from 36 to 39 represent networks generated by sequences of 2000 and 3000 points. The amount of all points together in the two groups are the same: 300,000. Thus the graphs in 37 have less nodes than the graphs in 39 but each K has 150 networks and in figure 39 every K has 100. The networks of the classes in 37 have more samples but the classes are visually best sorted out in 39, which evidences the loss of information when the sequences are divided - the 39 sequences were less divided. The images in 36 and 38 show that just as in the case of networks created with sets of 1000 points, the problem becomes more complex from K4.

Figure 36 – Analysis of the 2000 points networks descriptors

Source: Elaborated by the author.

Note – Eigenvector and the box-plot of the classes separation using the detected principal component from K0 to K4. The descriptors originated from complex network measurements.

Figure 37 – PCA plot of networks with 2000 points

Source: Elaborated by the author.

While in 37a, K0 and K1 are very close, in 39a K0 is more widely known and K1, although less concentrated in one region, is better separated from the other classes. In the case of 37b, the varying classes of K0 and K5 are all mixed while in 39b, K9 is highlighted as well as the agglomeration of K8. Thus, even if early experiments led to the assumption that randomness grows as K increases, this is not confirmed by this experiment.

As no measure stood out as the most significant, experiments were done with a smaller set of descriptors. From this observation it is possible to say that the withdrawal of some measures should not have a very large impact. The PCA was also used to analyze networks but the

Figure 38 – Analysis of the 3000 points networks descriptors

Source: Elaborated by the author.

Note – Eigenvector and the box-plot of the classes separation using the detected principal component from K0 to K4. The descriptors originated from complex network measurements.

Figure 39 – PCA plot of networks with 3000 points

Source: Elaborated by the author.

computationally costly descriptors were eliminated from the set. The analysis can be seen in figures 40 and 41.

The viewpoint exemplified in image 41a shows K0 more isolated from the rest of the other classes than in the other images. K3 begins to stand out more while K2 remains clustered with K3 and K4. In image 41b, K9 seems to be more mixed with the other groups, which remain merged as in image 39.

The PCA of the complex network descriptors does not present class clusters, but some considerations can be made. K0, the class whose sequences are weakly random, is the one that is highlighted the most, as K grows up to 4, the classes become more grouped. This may lead one

Figure 40 – Analysis of the 3000 points networks with a reduced number of descriptors

Source: Elaborated by the author.

Figure 41 – PCA plot of the 3000 points networks with a reduced number of descriptors

Source: Elaborated by the author.

to think that randomness increases as K increases, but the second group of networks brings an unexpected cluster. $K9$ is the class that is better visualized despite being the largest studied K . $K5$ and $K8$ networks are clustered while $K9$ is distinct almost as well as $K0$. Therefore, although the classes are not separated by K , the measures of complex networks let imply that randomness may not be related to the value of K .

7.2 Computational Geometry Analysis

Computational geometry is a powerful research area for studying objects in space. The list of computational geometry applications includes graphics, computer vision, image processing, robotics, computer-aided designing, geographic information systems, among others. An objective of this work is to analyze if this field of study is also useful for the study of PRNG randomness

quality.

In order to select the computational geometry techniques that would be useful in the analysis of PRNGs the algorithms were analyzed according to some criterion. The procedures selected were those that highlight the position of the point in the plan, analyze of the space according to the distribution of points, and create of meshes with the nearest neighbors resulting in a set of polygons. The Voronoi Diagram, and the Delaunay Triangulation were applied in sequences generated by PRNGs. The study of pseudo-random sequences is converted into a problem of straight or flat objects (lines, line segments, polygons) that are the focus of computational geometry.

In the analysis of complex networks, the plan were divided into 11 bands to better understand the distribution of points by region. Considering that the degree is a characteristic of the node, this division does not entail loss of information. In the case of computational geometry, the presentation section of this methodology explained the problem of geometric object analysis by area. The objects are formed by a set of points of several regions of the plan. In the case of the Voronoi diagram and the Delaunay triangulation, the size of the edges can vary greatly as they depend on the position of the points and not on an established radius, as in the case of creating polygons. Because of this, the objects in the plan were colored according to the area, giving an overview of how the areas are related to the values of K , in the K -logistic map.

Histograms of images formed by only 1000 points and 12bins per RGB channel separated K_0 from the others. K_0 does not generate random sequences but rather concentrates values in the neighboring regions 0 and 1. However, a cloud of points of different colors is observed in the PCAs of the upper K s, even increasing the number of colors. One may want to understand how this variation in color distribution is related to K greater than 4. For this, a set of histograms was made for K values ranging from 5 to 9 and with images originating from sequences of 5000 points.

7.2.1 Voronoi Diagrams for PRNG testing

The collection of PCAs and box-plots for different number of color is available in Annex I. Fig 45 shows the box-plot and PCA obtained with 30 colors. Visually the PCAs for polygons of different number of colors do not vary much, so the observations of this image can be generalized.

Figure 42 – Voronoi Diagram analysis of networks with 5000 points

(a) Box-plot of Voronoi Diagram

(b) PCA

Source: Elaborated by the author.

When the analyzes of $K0$ to $K4$ networks were presented, an increase in randomness of the sequences was observed. $K0$ represents the class of non-random sequences with clusters in the regions near 0 and 1, but as K increases, the points are distributed to the point that $K4$ becomes difficult to separate from the other classes. Contrary to what was intuitively expected, classes $K5$ and $K7$ have colors, i.e. areas, more related than networks $K6$, $K8$ and $K9$. The two horizontal groups seen in figure 42b are retained even when the polygons are divided into 270 colors. Groups overlap as the number of colors increases, however, it is still possible to see most of the 5 separate groups.

Figure 43 shows the relative importance of the first principal component within the PCA for the different color schemes, using its eigenvalue. It can be seen that the variance from the data-set that is captured by this principal component varies between approximately 64% and 76%. The best result is obtained by using the scheme with only 10 different colors, although it can be seen that most of the color schemes provide similar performances in the PCA. It can thus be concluded that the choice of color scheme slightly affects the classification, but especially

rising above the minimum number of 5 colors is important.

Figure 43 – Importance of the first PC within the PCA using Voronoi diagrams

Source: Elaborated by the author.

7.2.2 Delaunay triangulation for PRNG testing

The PCAs of Delaunay triangulation of different colors are not as similar as PCAs of Voronoi diagrams. PCAs pick up clouds scattered vertically from the plan, more than one cluster to represent a class and distribution of agglomerations diagonally to the plan. These two results are represented in figure 44. Despite the differences, these PCAs have some common characteristics. In both graphs, K6 is separated from the others. The K5 and K7 networks are close together as are the K8 and K9 networks. As in Voronoi, the K6 networks are closer to K9 and K8. The box plot set for different colors is found in Annex H.

Figure 44 – Delaunay triangulation analysis of networks with 5000 points - PCAs

Source: Elaborated by the author.

Note – The two most common PCA plots from Delaunay triangulation.

Figure 45b shows the relative importance of the first principal component within the PCA

for the different color schemes. The variance captured by this PC varies less than in the case for Voronoi diagram: only about 4%. In this case, the best result is obtained by using the scheme with only 5 different colors, while all other schemes result in a similar but slightly lower performance in the PCA. It can be concluded that the used color scheme in the case of Delaunay triangulation is of minor importance, where the more simple color scheme would probably provide marginally better classification.

Figure 45 – Delaunay triangulation analysis

Source: Elaborated by the author.

Note – Box-plot and importance of the first principal component of the measures originated from Delaunay triangulation.

The color, representing the area, brings information about the network, but only the color histogram is not enough. It requires an approach that studies colors at a pixel level, relates the color of a pixel to neighboring pixels, and considers the location of colors in the image. For this, texture analysis algorithms are applied.

For this, the images of the Delaunay triangulation and the Voronoi polygons were separated by K . For each K , 1000-point meshes were generated in a large variety of colors. Each set had 40 images. These images were analyzed as textures by the Compound Local Binary Pattern (CLBP), Fourier transform, Gabor transform, Grey Level Co-occurrence Matrix (GLCM), and Local Binary Patterns (LBP) methods. All images have been converted to grayscale to satisfy the input condition for these methods. For K ranging from 0 to 4, the results had only 60% of accuracy.

The best results are from a color–texture analysis that the image is modeled as a multilayer complex network where each color channel is a layer, and each pixel (in each color channel) is represented as a network vertex (SCABINI *et al.*, 2019). It relies on the creation of multi-layer networks connecting sub-nets formed by the RGB channels of the images. In this way, this approach has a view of the dynamics of how channels create different networks when the number

of edges changes. The algorithm pipeline is reproduced in figure 46.

Figure 46 – Stages of the applied texture method for colored images

Source: Scabini *et al.* (2019).

Note – An image is modeled as a multilayer complex network. The pixels are connected among them and connected crossing the RGB channels. Different radius are applied creating different networks. For each network, measurements are taken. The result is a list of image descriptors.

This multilayer complex network-based method was used to classify images generated from $K5$ to $K9$. In this algorithm each K had 200 images created from 5000 points each. The results are shown in tables 5 and 6.

7.3 Randomness classification with other techniques

The challenge of converting a sequence into a graph has been explored in many other academic works. The purpose of these works is to model in a graph the information of time series and using the graph to understand the phenomenon in more detail. There are also graph applications for border detection and texture analysis. In this context, instead of values generated in a determining time or iteration, the pixels of the images are converted into nodes and the number of edges changes generating several networks with the same number of points with different numbers of edges, giving an in-depth view of the dynamics of the network. Many of

Table 5 – Voronoi Diagram to K5, K6, K7, K8 and K9

r	M1	M1	M1	M1	M1
1	90.30	99.20	98.90	98.90	98.70
2	90.90	98.60	99.10	98.10	98.00
3	90.50	98.50	98.40	97.80	95.30
4	90.60	98.10	98.20	96.30	93.50
5	91.70	97.60	98.00	94.60	87.10
6	92.20	97.80	97.80	93.60	74.40

Source: Research data.

Table 6 – Delaunay Triangulation to K5, K6, K7, K8 and K9

r	M1	M1	M1	M1	M1
1	84.90	84.00	85.10	84.30	84.00
2	85.40	84.00	84.80	83.70	83.90
3	84.50	83.60	83.60	83.50	80.20
4	84.60	83.10	84.10	82.10	76.80
5	83.40	82.80	82.40	79.70	69.30
6	83.00	82.40	82.60	78.70	48.70

Source: Research data.

these approaches were analyzed, and some were tested. The most interesting for the discussion are the Visibility Graph ([LACASA et al., 2008](#)), complex network-based approach for boundary shape analysis ([BACKES; CASANOVA; BRUNO, 2009](#)) and the complex network-based approach for texture analysis ([BACKES; CASANOVA; BRUNO, 2013](#)).

The Visual Graph ([LACASA et al., 2008](#)) was part of the initial experiment set was the method of converting time-series into graphs. The networks generated by the process was analyzed considering the same measurements cited in section 7.1. It is important to note that in this approach there was no remodeling in the original method. It is implemented as described in the article cited above. To test this method, 100 networks were created for each K parameter. The classification was executed by the knn algorithm. The result is shown in table 7.

The Visual Graph result was modest and quite peculiar. It was expected that insertion of the weight in the algorithm would result in improvement in classification, but this was not the case attested. The low classification probability shows that even with the adaptation for the type of network explored in the research, the problem explored in this master's degree requires more than only convert a time series in graph and analyze basic measurements. Considering this, this

Table 7 – Visual Graph classification

Method	Correctly Classified Instances Percentages - knn(1)
Unweighted network	46.20%
Weighted network	45.20%

Source: Research data.

algorithm was no longer used in the advanced stages of experimentation.

The complex network-based approach for boundary shape analysis (BACKES; CASANOVA; BRUNO, 2009) brings another point of view to the problem. In Visibility Graph, the connection occurs automatically since it is based on the visible points that can be seen by a point. In practical terms, the points are connected to all other points of lower value, since a point of greater value on the y-axis blocks the connections to the next iterations. In the case of boundary shape analysis algorithms, the connection radius changes over time. The radius varies between values that are chosen empirically, opening the possibility of several adaptations depending on the phenomenon one wants to understand. The PCA resulting from the experiments using this methodology is shown in figure 47.

Figure 47 – PCA for boundary shape analysis

Source: Elaborated by the author.

Table 8 – Complex network-based approach for boundary shape analysis

Model	NaiveBayes	LDA	Logistic	MP	SMO	Regression	MultiClass	DecisionTable	RandomForest	SimpleLogistic
BCB-k04	83.40	84.88	84.14	67.56	85.24	86.82 v	85.24	86.62 v	84.56	89.4
BCB-multi-k04	83.44	68.78	79.98	78.54	86.18	86.98	82.54	86.08	85.38	84.36
BCB-nn-k04	88.10	66.84	65.72	61.72	65.48	86.92	66.16	86.98	87.08	67.22
BCB-c-k09	43.86	43.70	42.56	31.07	44.82	43.59	44.07	45.58	45.58	45.58
BCB-rad-k09	43.15	33.88	41.81	37.66	45.33	44.40	42.88	43.65	44.39	50.04
BCB-nonorm-k09	45.78	33.55	33.36	29.45	33.04	44.71	32.90	44.19	45.16	33.62

Source: Research data.

Table 8 summarizes the results of complex network-based approach for boundary shape analysis as a descriptor extractor. The method was first used to classify the sequences with K varying from 0 to 4. The result is interesting since the method creates a list of descriptors from only the maximum degree and the average degree of each network formed when the connection radius changes. An adaptation of the method is to use more radii of connection than the 13 radii proposed in the original work. In this case, either the percentage improved very little or worsened enough, reinforcing that a greater number of descriptors does not necessarily result in a better classification. The original method also makes a normalization of the largest distance presents and here the classification result is described without this normalization. As in the previous case, either the percentage improved very little or worsened greatly, hence the importance of selecting the classification algorithm according to the problem.

When the algorithm extracts the descriptors for K ranging from 0 to 9, described in the second part of the table, the correct percentage of classification decreases drastically. The method is promising when there are few sequences that are not pseudo-random. When the classification has K values ranging from 5 to 9, the differentiation of these sequences becomes a very complex task to be done only with the analysis of two measures, even in a dynamic context.

7.4 Reproducibility of experiments

The challenge of this research is to propose a new view on PRNGs in the light of graphs. A large list of measures has been selected and also a good visualization has been very important since the beginning of the development of this work. For coding for measurement extraction and network visualization, two libraries were used for network analysis. It is important to cite this information because in many papers, simpler measures can be quickly calculated with the representation of the network in an adjacency list. Other measures require more data to be saved in memory so these libraries implement a "network" object and the measures are optimized to be calculated in this framework. Therefore, in most of the implementation time, this project had a limited condition related to the RAM required by the measurement calculation and the available machines.

These implementations were made using the spatial algorithms and data structures of the SciPy² library and the Shapely³ library.

For the visualization of the networks, the points were stored in a file so the reading methods of the specific libraries could access them. This file included the list of all points and their respective neighbors. This information is sufficient for the construction and exploration of unweighted networks. Since this project also uses the weighted networks, besides the points and its neighbors, the Euclidean distances have been saved. These distances are the edge weights

² <https://www.scipy.org/>

³ <https://pypi.python.org/pypi/Shapely>

in the weighted networks approach. In later implementations, the weights were calculated in execution time.

The next stage was to produce Cartesian points and calculate the neighbors of sequences generated by the algorithms cited in the section about PRNGs. The value of the coordinates of the points and their respective neighbors were saved in .json⁴ files. This resolution resulted in a set of files that have been being vital to the reproducibility of the experiments considering that all sequences are different each time that a PNRG algorithm is executed (for different seeds).

From the use of OpenGL it became feasible to observe the evolution of large sequences. This visualization was possible with Java's Swing⁵ library. Also, applying this library, it was easiest to see the distribution of the points degree over the space. It was possible to observe the appearance of clusters, polygons, dense areas in the central horizontal region and the appearance of nodes with only one neighbor does not occur in the edge region. Because the Java library generates images with static pixels, code execution was very costly. In the second part of this paper, when Python entirely replaced Java, the matplotlib library was used. It has been shown to be much more efficient in terms of runtime and image quality since this library generates vector graphics so the images do not lose quality when resized or transformed.

7.4.1 Classification of descriptors

The classification algorithms selected are those that presented the best performance in the problem addressed. Tests with known classification methods were performed to select those most appropriate for the classification of networks originating from PRNGs sequences. The selection of the algorithms considered only those that are implemented in the tool Weka 3.9. This simplified the selection and reproduction of experiments since Weka is a tool freely available for download. The evaluated algorithms were: Bayes Net, Naive Bayes, Multilayer Perceptron, LDA, QDA, Simple Logistic, SVM, knn, Classification via Regression, Multiclass classifier, Logistic, Vote, Decision Table, Random Tree and Random Forest. 150 networks per K ranging from 0 to 4 generated by sequences of 1000 points were used in this selection. The table with all the results obtained in this experiments then in annex E.

7.4.2 Experiments configuration

The Python⁶ programming language is extensively used in the analysis of complex networks. Python 2 is the version used in the codification. There is a wide range of academic papers using the NetworkX and igraph libraries in Python (MULLER *et al.*, 2009), (PEREZ; GRANGER; HUNTER, 2011), (SMEDT; DAELEMANS, 2012), (MOYA-GARCÍA; RANEA, 2013), (SADOVSKY; MACLEAN, 2014), (LUIZI *et al.*, 2015), (PADILLA, 2016), (GIECOLD

⁴ <http://www.json.org/>

⁵ <https://docs.oracle.com/javase/7/docs/api/javafx/swing/package-summary.html>

⁶ <https://www.python.org/>

et al., 2016), (BOEING, 2017). These libraries create graphs objects both from files and adjacent matrices and have the implementation of the main complex networks measures. These implementations were made using the spatial algorithms and data structures of the SciPy⁷ library and the Shapely⁸ library.

7.4.2.1 Complex Network analysis selected libraries

The NetworkX⁹ package that implements the basic data structures and the computation of network properties and structure measures: shortest paths, betweenness centrality, clustering, and degree distribution and many more. Also, NetworkX provides generators for many classic graphs and popular graph models.(HAGBERG; SWART; CHULT, 2008). NetworkX is a powerful tool for scientific computations and one of the most used network analysis packages(WANDELT; SUN, 2016).

Igraph¹⁰ is a free software package for creating and manipulating graphs. It has the ability to handle large graphs efficiently. "It has a quite efficient data structure for representing graphs [...]. In fact these data structures evolved along the implementation of the classic and non-classic graph algorithms which make up the major part of the igraph library"(CSARDI; NEPUSZ, 2006). igraph additionally comprises implementations for classic graph theory problems like minimum spanning trees and network flow, and also implements algorithms like community structure search. The efficient implementation of igraph allows it to handle graphs with millions of nodes and edges.(AKHTAR, 2014)

In order for this visual inspection to be possible, the networks started to have 1000 to 5000 points. For such large time series to be observed, it would not be computationally feasible to plot the points using a drawing approach with pixels. OpenGL¹¹ was chosen for this task by virtue of its powerful graphical rendering capability so that such large-sized images could be plotted and analyzed in a timely manner. For the reason that OpenGL is written in C/C++, a low level language, it results a faster processing. Java language, used initially, uses code interpretation. OpenGL uses the graphics board directly and Java uses the GPU instead. The Java binding used was the JOGL¹².

7.4.3 Technical challenges

The maximum number of nodes per network in this work is 10,000. However, the maximum number of values within the period of a PRNG generally reaches 2^{31} . Thus, the number 10,000 represents a very small percentage within the possibility of a PRNG output, even

⁷ <https://www.scipy.org/>

⁸ <https://pypi.python.org/pypi/Shapely>

⁹ <https://networkx.github.io/>

¹⁰ <http://igraph.org/redirect.html>

¹¹ <https://www.opengl.org/>

¹² <http://jogamp.org/jogl/www/>

in the setup of some experiments in this work that created 200 networks of 10,000 points. A considerable number of points are produced for the creation of sets of networks of the same class. Networks originating from these values are large, and as networks are made up of more values, the measures consume more memory. This fact is relevant in this context because for structural networks, measures related to walking are very important. These walks require that the graph be copied many times in memory mainly when the execution is done in multiprocessing.

The K-logistic map requires in average 1,000 networks for experiment: 100 networks for K, with K varying from 0 to 9. For each network, several measures are taken and many are time-consuming measures. These measures require not only a RAM memory but also processing speed. Powerful computers were required to perform the tests as they depended on several small modifications and analysis of how these modifications altered the percentage of correct classifications.

CONCLUSIONS

The first mathematical concepts about randomness reach most students in elementary school. The teacher explains how the consecutive launches of coins or tossing a dice, even though they are independent events, have a known distribution. Then the students understand that board games, and later in life, gambling tables, are forms of producing numbers which can not be predicted from the preceding ones. Students also learn from the probability of someone winning the Mega Sena federal lottery to the probability of removing balls of certain colors from within a box. At some point in learning, mathematics presents its tools to help understand these events.

When universities and large companies started to have computers, researchers started to adapt mathematical tools into algorithms. PRNGs are one of those algorithms where the computer generates values based on an equation or mathematical property. These algorithms implemented in a computer program are completely pseudo-random. PRNGs are expected to be unpredictable and independent, so they are also used to enhanced security for the internet and for encrypted messages. Some algorithms that are initialized using a seed add a layer of security. They employ hardware and software protection to ensure that the seed is never observed by outsiders and consequently ensure that output can not be predicted. Although deterministic, these sequences are largely used in many applications ranging from online banking to computer games.

Today, mechanisms that generate secure PRNG for encryption or devices that generate TRN are available to sign official documents with a smart card and registration of residence and opening bank accounts. The work of [Hurley-Smith and Hernandez-Castro \(2018\)](#) lists a series of security issues in computer systems that were caused by algorithms that are not random enough but are nevertheless passed in statistical tests. They describe the 2008 publication of the Mifare Classic, an RFID card which weak sequence of random numbers facilitated attacks against its supposedly secure communications channel. Even attacks were made in a real-world payment system showing that because of its weak random number generator the system is flawed. Another case reported in [Hurley-Smith and Hernandez-Castro \(2018\)](#) is the attack against Taiwan's

national Citizen Digital Certificate database, caused by a poor TRNG implementation in the BSI certified.

Given the wide range of applications, it is expected that the randomness test of PRNG is very strict since these algorithms are used even in the protection of sensitive data. Although the power of processing, storage and data transmission has grown substantially, the tests of PRNGs randomness have changed very little since their first implementations. These test suites have been well accepted in the literature, but passing these tests does not guarantee the efficacy of the PRNG. For example, the NIST test suite has gone through several updates because the scientific community finds problems or incompleteness in its tests. The DIEHARD is quite rigid in its configuration, not allowing testing with sequences that do not fit into bit size and accuracy. However, with the description of the test suites above, the temporal and spatial distribution of numbers are not contemplated (or at least are not deeply analyzed). It can be a promising source of weakness of the PRNGs.

The problem of measuring randomness has gained a new approach in this dissertation. Rather than testing whether sequences obey a list of requirements specified in tests, a classification problem has been modeled. Each parameter K generated a set of sequences that were converted into graphs. Each set was considered a class. Classical measures of complex networks and measures of texture became the descriptors of the classes. If K classes could be separated, then they would not be random because there would be a Machine Learning algorithm capable of grasping the particularities of each class. The focus of statistical tests is the number of tests that an algorithm passes with high precision. The point of view used in this research changed the concern for which PRNGs produce sequences whose descriptors are not sufficient for a classification algorithm to generate high percentages of correctness.

The first way to model the sequences generated by PRNGs was complex networks. Complex networks have been shown to be satisfactory in image border detection, texturing, time series, among others. This work has hypothesized that measurements taken from complex networks created from PRNGs sequences can reveal patterns in those sequences, and consequently, point to at least one weakness in the algorithm. The networks were then created using the iteration value and the value generated in that iteration as points in a Cartesian plane. The edges were staggered between us within a given radius. After this step, a comprehensive list of measurements was extracted and classification algorithms used these measures as descriptors. Extraction of measurements from the network was important for the separation of sequences that are actually pseudo-random sequences from those that are not. However, as K increases, the problem of classifying K becomes more complex. However, one curious finding is that $K9$, the highest value tested, seems to be highlighted in the PCA of K 's from 5 to 9. This leads to the belief that the randomness of these sequences is not increasing.

The experiments in this paper also tested whether academic articles on the application of complex networks in other fields would be useful for the study of randomness of PRNGs.

Small adaptations were done in algorithms based on network dynamics. Another algorithm that underwent adaptations was an algorithm of temporal series conversion into a graph. Only the approaches based on network dynamics were able to separate the pseudo-random sequences from the non-uniform sequences. However, they did not prove useful in separating the classes of K . These algorithms are easy to implement and execute, but do not generate good descriptors for randomness classification of PRNGs.

It was found that the most used test suites do not exploit the relation of the values of the sequences under test in a spatial context. Therefore, the PRNGs were also studied using computational geometry. In computational geometry, robust and efficient algorithms are applied for the studying of practical problems such as finding proximity between points or intersections between sets of segments (QUEIROZ, 2003). Two computational geometry algorithms were used to analyze the PRNGs: the Voronoi Diagram and the Delauney Triangulation. The Voronoi diagram was applied in order to have a better view of the incorporation of sequence values in a 2D space. The Delaunay Triangulation was important to put the network's hubs in evidence, giving information about in what interval of iteration many numbers with approximate values were generated. In this way, the visualization of non-uniformly distributed sequences was facilitated.

These two techniques were promising given the PCA charts seen in the sections above. They initially did not show promise. Early experiments relied on images generated by 1000-point networks in a wide variety of colors. For K ranging from 0 to 4, the results had only 60% of accuracy. In addition to the size of the sequences, another constraint was the execution speed of the code that generated the images. The result was classifications based on learning with only 40 images per class. It is important to note also that the texture algorithms used to generate the descriptors were not designed for colored images, i.e. the images generated by coloring the polygons according to their area.

In the second part of the experiments, the algorithms of image generation were replaced allowing each class to have 200 images created from 5000 points each. In addition, the texture algorithm used was a newly published algorithm that apprehends excellent percentages compared to other algorithms that are state of the art in the area of pattern recognition in texture. This algorithm is also specialized in color images. It relies on the creation of multi-layer networks connecting sub-nets formed by the RGB channels of the images. In this way, this approach has a view of the dynamics of how channels create different networks when the number of edges changes. This texture algorithm was also efficient in detecting patterns in the images generated by PRNGs. The best percentages of network classification were achieved with this algorithm. They were very important so that along with a texture algorithm for colored images it was possible to point patterns in an algorithm that is considered pseudo-random by two test suites.

It is already possible to generate sequences with a quantum random number generator. These sequences are true random numbers thanks to the statistical nature of quantum mechanics (HAYES, 2001). Sequences of this type would guarantee total security in applications where

this is the basic requirement. Although NIST researchers are creating the integration of the quantum procedure in order to share it with the public, this mechanism is still not used on a large scale by companies. For some years to come, many applications will continue to depend on the quality of PRNGs.

The work developed during this master incorporated the use of graphs as a test tool for PRNGs. The work analyzes the test suites, pointing out their limitations. Recent work is cited to state that, in fact, these test suites have many limitations. These same test suites are used as a foundation by companies that sell devices that theoretically generate True Random Number sequences. These limitations can lead to serious security problems. It can be said that statistical tests are based on randomness naivety, the use of a reductionist definition of randomness: "accepting a simple pass or fail outcome instead of characterizing tested devices as suitably random for a given application" (HURLEY-SMITH; HERNANDEZ-CASTRO, 2018). Therefore, it is necessary that the randomness tests suites are continuously reviewed and incorporate new approaches. It is expected that this work will contribute to new ways of analyzing this aspect that is so crucial in the globalized world: the quality of PRNGs.

8.0.1 Perspectives for future work

This work made a first analysis of the use of graphs as an auxiliary tool to measure the randomness of PRNGs. Many papers may originate from this research because it involved complex networks, computational geometry, texture, analysis of statistical test suites, and PRNG based on chaos. The work also applies machine learning classification algorithms with the aim of having another view on the PRNGs. Therefore, a wide range of adaptations can be made with the objective of understanding how each of these tools is related to the pseudo-randomness presented by PRNGs.

The PCA of the measures of the complex networks did not highlight any measure, so a first task is eliminating all the computationally costly measures. However, it has not been testing whether the removal of these measures has a major impact on the classification. The ideal would be to have a set of descriptors that can be generated quickly. The measures used in this work are classic measures but there are several surveys listing new measures. It is important to take the test with increasingly modern measures since these can bring information that could not be contemplated in this work.

The Computational Geometry algorithms already give very satisfactory results when the areas are distributed in 5 groups, that is, the images are painted with 5 colors. Therefore, gray-scale images are probably sufficient for the classification of PRNGs. One can say that if this is true, the initial tests with texture algorithms for gray-scale images would already have given good results. But this first test was very limited with only 40 networks per class and images made with 1000-point networks. It might be that only this first test lacked good results. Thus, algorithms such as CLBP, FOURIER, GABOR, GLCM, and LBP, for instance, might be

sufficient to do correct classification.

The Voronoi diagram and the Delaunay Triangulation are very powerful tools. A future work is to predict if creating simple polygons with the establishment of a connection radius brings information about the sequences. The procedure is the same as that used for the conversion of the sequence into a graph. But instead of analyzing the result as a graph and the relation between the points of this graph, the problem can be studied again by the distribution of the areas of the resulting polygons.

One improvement can be the using of large cut windows. The experiments described in this work have used 1000 and 5000-point windows for computational geometry algorithms and windows ranging from 500 to 3000 for complex networks. Experiments were done with complex networks with windows of size 4000 to 10000, increasing 1000 points in each change. These experiments could not be finalized as the available computers had memory overflow. Complex networks have good increasing results with windows varying from 1000 to 3000. An analysis to be made is when classification improves with increasing sequence cut window.

It is also interesting to analyze how randomness changes in higher values of K . It was observed that $K = 9$ presented a clustering in the PCA of the measurements of complex networks. This may mean that randomness is not increasing as K grows. It may also mean that randomness does not become excitable and that even high values of K may generate non uniformly distributed sequences. These interpretations can be explained by experimenting with networks generated with K from 10.

The PRNG analysis with graph theory should be extended to classical PRNGs. This work was limited to investigating a PRNG successfully tested in statistical suites but it was not addressed to classical algorithms such as LCG, Rule 30, LFSR, among others. It is already known that many of these algorithms are not approved in statistical tests. Nevertheless, it is interesting to test if the same methodology that resulted in a good classification in this work - computational geometry and texture - can also classify the sequences of different weak PRNGs.

Finally, the proposed methodology can be applied in testing TRN sequences. The result of this test would be very interesting work because it can point out which features separate the PRNGs that are random enough from the sequences generated by TRN devices. This experiment would find a great challenge in obtaining large sequences of TRN but the result would be of value since it is not yet known if the modeling of TRN in spatial approaches are useful for detecting patterns in these sequences.

The study of chaotic systems gained prominence in the last century. Chaos-based PRNGs only began to be implemented in the late 1990s. The type of algorithm analyzed in this paper as well as the theory of complex networks are relatively new subjects in the context of the history of science. The origin of randomness in chaos, which mathematical phenomena can be modeled in networks, and which approaches should be added to ensure increasingly safe randomness tests

are examples of the many questions that will be investigated in the field of chaos and randomness. It is not known if one day these questions will be answered, but it is the curiosity to answer them that puts us on the shoulders of giants to make progress in science.

BIBLIOGRAPHY

AFTAB, O.; CHEUNG, P.; KIM, A.; THAKKAR, S.; YEDDANAPUDI, N. Information theory and the digital age. **Bandwagon, pgs**, p. 9–11, 2001. Citation on page [29](#).

AKHTAR, N. Social network analysis tools. In: IEEE. **Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on**. [S.l.], 2014. p. 388–392. Citation on page [116](#).

ALMEIDA, M. L. d. Análises estatísticas em redes complexas: propriedades topológicas, críticas e dinâmicas. Universidade Federal do Rio Grande do Norte, 2013. Citations on pages [46](#), [49](#), and [50](#).

ALURU, S. Lagged fibonacci random number generators for distributed memory parallel computers. **Journal of Parallel and Distributed Computing**, Elsevier, v. 45, n. 1, p. 1–12, 1997. Citation on page [34](#).

AURENHAMMER, F. Voronoi diagrams—a survey of a fundamental geometric data structure. **ACM Computing Surveys (CSUR)**, ACM, v. 23, n. 3, p. 345–405, 1991. Citations on pages [55](#) and [58](#).

BACKES, A. R.; CASANOVA, D.; BRUNO, O. M. A complex network-based approach for boundary shape analysis. **Pattern Recognition**, Elsevier, v. 42, n. 1, p. 54–67, 2009. Citations on pages [81](#), [82](#), [111](#), and [112](#).

_____. Texture analysis and classification: A complex network-based approach. **Inf. Sci.**, v. 219, p. 168–180, 2013. Citations on pages [81](#), [82](#), [83](#), and [111](#).

BAHI, J. M.; FANG, X.; GUYEUX, C.; WANG, Q. Evaluating quality of chaotic pseudo-random generators: Application to information hiding. **arXiv preprint arXiv:1112.1201**, 2011. Citation on page [41](#).

BAIESI, M.; PACZUSKI, M. Scale-free networks of earthquakes and aftershocks. **Physical review E**, APS, v. 69, n. 6, p. 066106, 2004. Citations on pages [52](#) and [53](#).

BAILEY, D. H. **A Pseudo-Random Number Generator Based on Normal Numbers**. [S.l.], 2004. Citation on page [35](#).

BAILEY, D. H.; CRANDALL, R. E. Random generators and normal numbers. **Experimental Mathematics**, Taylor & Francis, v. 11, n. 4, p. 527–546, 2002. Citation on page [35](#).

BAPTISTA, M. Cryptography with chaos. **Physics Letters A**, Elsevier, v. 240, n. 1-2, p. 50–54, 1998. Citation on page [29](#).

BARABASI, A.-L. **Linked : how everything is connected to everything else and what it means for business, science, and everyday life**. New York: Plume, 2003. ISBN 978-0-452-28-439-5. Citations on pages [45](#), [51](#), and [52](#).

BARABÁSI, A.-L. **Network science**. [S.l.]: Cambridge university press, 2016. Citations on pages 24, 50, 51, and 52.

BARTHÉLEMY, M. Crossover from scale-free to spatial networks. **EPL (Europhysics Letters)**, IOP Publishing, v. 63, n. 6, p. 915, 2003. Citation on page 54.

BELLHOUSE, D. Decoding cardano's liber de ludo aleae. **Historia Mathematica**, Elsevier, v. 32, n. 2, p. 180–202, 2005. Citation on page 28.

BHATTACHARYA, P.; GAVRILOVA, M. L. Voronoi diagram in optimal path planning. In: IEEE. **Voronoi Diagrams in Science and Engineering, 2007. ISVD'07. 4th International Symposium on**. [S.l.], 2007. p. 38–47. Citation on page 60.

BIERHORST, P.; KNILL, E.; GLANCY, S.; ZHANG, Y.; MINK, A.; JORDAN, S.; ROMMAL, A.; LIU, Y.-K.; CHRISTENSEN, B.; NAM, S. W. *et al.* Experimentally generated randomness certified by the impossibility of superluminal signals. **Nature**, Nature Publishing Group, v. 556, n. 7700, p. 223, 2018. Citation on page 31.

BLUM, L.; BLUM, M.; SHUB, M. A simple unpredictable pseudo-random number generator. **SIAM Journal on computing**, SIAM, v. 15, n. 2, p. 364–383, 1986. Citation on page 29.

BOEING, G. Osmnx: A python package to work with graph-theoretic openstreetmap street networks. 2017. Citation on page 116.

BOGUNÁ, M.; PASTOR-SATORRAS, R. Epidemic spreading in correlated complex networks. **Physical Review E**, APS, v. 66, n. 4, p. 047104, 2002. Citation on page 45.

BROCK, T. **Foundations of Random Number Generation in JavaScript - Evangelism - Infragistics.com Blog**. 2016. Available: <<https://www.infragistics.com/community/blogs/tim{ }brock/archive/2016/07/14/foundations-of-random-number-generation-in-javascript.aspx>>. Citation on page 30.

BROWNLEE, J. Machine learning mastery. URL: <http://machinelearningmastery.com/discover-feature-engineering-howtoengineer-features-and-how-to-getgood-at-it>, 2014. Citation on page 94.

BU-QING, D.-Y. **Computational geometry—curve and surface modeling**. Boston: Academic Press, 1989. ISBN 0-12-675610-4. Citation on page 55.

BULLMORE, E.; SPORNS, O. Complex brain networks: graph theoretical analysis of structural and functional systems. **Nature reviews. Neuroscience**, Nature Publishing Group, v. 10, n. 3, p. 186, 2009. Citations on pages 45, 46, 50, 51, and 54.

CASANOVA, D. **Redes complexas em visão computacional com aplicações em bioinformática**. Phd Thesis (PhD Thesis) — Universidade de São Paulo, 2013. Citation on page 46.

CASASNOVAS, J. P. **Evolutionary Games in Complex Topologies: Interplay Between Structure and Dynamics**. [S.l.]: Springer Science & Business Media, 2012. Citations on pages 50, 51, and 52.

CASTRO, A. A. M. de; PRADO, P. P. L. do. Algoritmos para reconhecimento de padrões. **Revista Ciências Exatas**, v. 8, n. 2002, 2002. Citation on page 95.

CASTRO, J. C. H.; SIERRA, J. M.; SEZNEC, A.; IZQUIERDO, A.; RIBAGORDA, A. The strict avalanche criterion randomness test. **Mathematics and Computers in Simulation**, Elsevier, v. 68, n. 1, p. 1–7, 2005. Citation on page 43.

CHAN, J. J. M. **True random number generation using genetic algorithms on high performance architectures**. Phd Thesis (PhD Thesis) — Department of Electrical and Computer Engineering. The University of Manitoba, <http://hdl.handle.net/1993/31619>, 10 2016. Citations on pages 27, 28, and 29.

CHANG, W.; FANG, B.; YUN, X.; WANG, S.; YU, X. Randomness testing of compressed data. **arXiv preprint arXiv:1001.3485**, 2010. Citation on page 43.

CHENG, H.; WANG, Y.; SHENG, J.; KRONENBERGER, W. G.; MATHEWS, V. P.; HUMMER, T. A.; SAYKIN, A. J. Characteristics and variability of structural networks derived from diffusion tensor imaging. **Neuroimage**, Elsevier, v. 61, n. 4, p. 1153–1164, 2012. Citation on page 48.

CHENG, R.; XIE, X.; YIU, M. L.; CHEN, J.; SUN, L. Uv-diagram: A voronoi diagram for uncertain data. In: IEEE. **Data Engineering (ICDE), 2010 IEEE 26th International Conference on**. [S.l.], 2010. p. 796–807. Citation on page 60.

COSTA, L. d. F.; RODRIGUES, F. A.; TRAVIESO, G.; BOAS, P. R. V. Characterization of complex networks: A survey of measurements. **Advances in physics**, Taylor & Francis, v. 56, n. 1, p. 167–242, 2007. Citation on page 71.

COSTA, L. da F. The hierarchical backbone of complex networks. **Physical review letters**, APS, v. 93, n. 9, p. 098702, 2004. Citation on page 48.

CRUCITTI, P.; LATORA, V.; PORTA, S. Centrality measures in spatial networks of urban streets. **Physical Review E**, APS, v. 73, n. 3, p. 036125, 2006. Citations on pages 53 and 54.

CSARDI, G.; NEPUSZ, T. The igraph software package for complex network research. **Inter-Journal, Complex Systems**, v. 1695, n. 5, p. 1–9, 2006. Citation on page 116.

DAVIS, D.; IHAKA, R.; FENSTERMACHER, P. Cryptographic randomness from air turbulence in disk drives. In: **Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings**. [S.l.]: Springer, 1994. (Lecture Notes in Computer Science, v. 839), p. 114–120. Citation on page 30.

DEUSSEN, O.; HILLER, S.; OVERVELD, C. V.; STROTHOTTE, T. Floating points: A method for computing stipple drawings. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. [S.l.], 2000. v. 19, n. 3, p. 41–50. Citation on page 60.

ECKHARDT, R. Stan ulam, john von neumann, and the monte carlo method. **Los Alamos Science**, v. 15, n. 131-136, p. 30, 1987. Citation on page 29.

EDELSBRUNNER, H. **Geometry and topology for mesh generation**. [S.l.]: Cambridge University Press, 2001. Citation on page 58.

EPSTEIN, M.; HARS, L.; KRASINSKI, R.; ROSNER, M.; ZHENG, H. Design and implementation of a true random number generator based on digital circuit artifacts. **Cryptographic Hardware and Embedded Systems-CHES 2003**, Springer, p. 152–165, 2003. Citation on page 42.

- ERWIG, M.; HAGEN, F. The graph voronoi diagram with applications. **Networks**, v. 36, n. 3, p. 156–163, 2000. Citation on page 60.
- EXPERT, P.; EVANS, T. S.; BLONDEL, V. D.; LAMBIOTTE, R. Uncovering space-independent communities in spatial networks. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 108, n. 19, p. 7663–7668, 2011. Citation on page 54.
- FCALL, L. Nag fortran library routine document g05yaf. The Numerical Algorithms Group Limited, 2013. Citation on page 29.
- FENG, L.; GE, S.; LIU, H. Electric vehicle charging station planning based on weighted voronoi diagram. In: IEEE. **Power and Energy Engineering Conference (APPEEC), 2012 Asia-Pacific**. [S.l.], 2012. p. 1–5. Citation on page 60.
- FIGUEIREDO, D. R. Introduçao a redes complexas. **Atualizações em Informática**, p. 303–358, 2011. Citations on pages 45, 46, and 48.
- FORREST, A. Computational geometry. In: THE ROYAL SOCIETY. **Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences**. [S.l.], 1971. v. 321, n. 1545, p. 187–195. Citations on pages 55 and 57.
- GACHENKO, A. S.; HMELNOV, A. E. Development of complex gis monitoring of the angara river. In: **Information Technologies in the Research of Biodiversity**. [S.l.]: Springer, 2019. p. 181–186. Citations on pages 60 and 61.
- GAGE, D.; LAUB, E.; MCGARRY, B. Cellular automata: Is rule 30 random. In: **Proceedings of the Midwest NKS Conference, Indiana University**. [S.l.: s.n.], 2005. Citation on page 37.
- GANSNER, E.; HU, Y.; KOBOUROV, S.; VOLINSKY, C. Putting recommendations on the map: visualizing clusters and relations. In: ACM. **Proceedings of the third ACM conference on Recommender systems**. [S.l.], 2009. p. 345–348. Citation on page 60.
- GASTNER, M. T.; NEWMAN, M. E. The spatial structure of networks. **The European Physical Journal B**, v. 49, n. 2, p. 247–252, 2006. Available: <<http://www.springerlink.com/index/10.1140/epjb/e2006-00046-8>>. Citations on pages 15, 53, 71, 72, and 73.
- GENTLE, J. E. **Random Number Generation and Monte Carlo Methods**. [S.l.]: Springer, 2003. ISBN 0387001786. Citations on pages 23, 29, 30, 31, 32, 33, 34, 41, 42, 63, and 140.
- GIECOLD, G.; MARCO, E.; GARCIA, S. P.; TRIPPA, L.; YUAN, G.-C. Robust lineage reconstruction from high-dimensional single-cell data. **Nucleic acids research**, Oxford University Press, v. 44, n. 14, p. e122–e122, 2016. Citation on page 116.
- GOLDENBERG, A.; ZHENG, A. X.; FIENBERG, S. E.; AIROLDI, E. M. *et al.* A survey of statistical network models. **Foundations and Trends® in Machine Learning**, Now Publishers, Inc., v. 2, n. 2, p. 129–233, 2010. Citation on page 49.
- GRINSTEAD, C. M.; SNELL, J. L. **Introduction to probability**. [S.l.]: American Mathematical Soc., 2012. Citation on page 29.
- GUIBAS, L. J.; KNUTH, D. E.; SHARIR, M. Randomized Incremental Construction of Delaunay and Voronoi Diagrams 1. **Algorithmica**, v. 7, p. 381–13, 1992. Available: <<https://link.springer.com/content/pdf/10.1007/%7B2FBF01758770.pdf>>. Citation on page 55.

HAGBERG, A.; SWART, P.; CHULT, D. S. **Exploring network structure, dynamics, and function using NetworkX**. [S.l.], 2008. Citation on page [116](#).

HALL, M.; FRANK, E.; HOLMES, G.; PFAHRINGER, B.; REUTEMANN, P.; WITTEN, I. H. The weka data mining software: an update. **ACM SIGKDD explorations newsletter**, ACM, v. 11, n. 1, p. 10–18, 2009. Citations on pages [37](#) and [97](#).

HAMANO, K. The distribution of the spectrum for the discrete fourier transform test included in sp800-22. **IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences**, The Institute of Electronics, Information and Communication Engineers, v. 88, n. 1, p. 67–73, 2005. Citation on page [41](#).

HANNEMAN, R. A.; RIDDLE, M. **Introduction to social network methods**. [S.l.]: University of California Riverside, 2005. Citation on page [48](#).

HAUKE, J.; KOSSOWSKI, T. Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. **Quaestiones geographicae**, De Gruyter Open Sp. z oo, v. 30, n. 2, p. 87, 2011. Citations on pages [47](#) and [48](#).

HAYES, B. Computing science: Randomness as a resource. **American Scientist**, JSTOR, v. 89, n. 4, p. 300–304, 2001. Citation on page [121](#).

HELD, M. Vroni: An engineering approach to the reliable and efficient computation of voronoi diagrams of points and line segments. **Computational Geometry**, Elsevier, v. 18, n. 2, p. 95–123, 2001. Citation on page [55](#).

HELD, M.; HUBER, S. Topology-oriented incremental computation of voronoi diagrams of circular arcs and straight-line segments. **Computer-Aided Design**, Elsevier, v. 41, n. 5, p. 327–338, 2009. Citation on page [55](#).

HOPKINS, M. A.; FRANKENSTEIN, S.; THORNDIKE, A. S. Formation of an aggregate scale in arctic sea ice. **Journal of Geophysical Research: Oceans**, Wiley Online Library, v. 109, n. C1, 2004. Citation on page [60](#).

HOSSEINI, S. H.; HOEFT, F.; KESLER, S. R. Gat: a graph-theoretical analysis toolbox for analyzing between-group differences in large-scale structural and functional brain networks. **PloS one**, Public Library of Science, v. 7, n. 7, p. e40709, 2012. Citations on pages [48](#) and [54](#).

HURLEY-SMITH, D.; HERNANDEZ-CASTRO, J. Great expectations: A critique of current approaches to random number generation testing & certification. In: SPRINGER. **International Conference on Research in Security Standardisation**. [S.l.], 2018. p. 143–163. Citations on pages [41](#), [43](#), [44](#), [119](#), and [122](#).

IDEKER, T.; SHARAN, R. Protein networks in disease. **Genome research**, Cold Spring Harbor Lab, v. 18, n. 4, p. 644–652, 2008. Citation on page [45](#).

JUNOD, P. **Cryptographic secure pseudo-random bits generation: The Blum-Blum-Shub generator**. [S.l.]: August, 1999. Citations on pages [31](#), [34](#), and [37](#).

KARLSEN, F. Addiction and randomness. **Teoksessa Children and Youth in the Digital Media Culture. Toim. Ulla Carlsson**. Göteborg: Nordicom, p. 157–172, 2010. Citation on page [38](#).

KHRENNIKOV, A. **Interpretations of probability**. [S.l.]: Walter de Gruyter, 2009. Citation on page 29.

KIM, S.-J.; UMENO, K.; HASEGAWA, A. Corrections of the nist statistical test suite for randomness. **arXiv preprint nlin/0401040**, 2004. Citation on page 41.

KOCAREV, L.; STERJEV, M.; FEKETE, A.; VATTAY, G. Public-key encryption with chaos. **Chaos: An Interdisciplinary Journal of Nonlinear Science**, AIP, v. 14, n. 4, p. 1078–1082, 2004. Citation on page 30.

KUMARI, S.; POWER, C.; CAIRNS, P. Investigating uncertainty in digital games and its impact on player immersion. In: ACM. **Extended abstracts publication of the annual symposium on computer-human interaction in play**. [S.l.], 2017. p. 503–509. Citation on page 38.

LACASA, L.; LUQUE, B.; BALLESTEROS, F.; LUQUE, J.; NUNO, J. C. From time series to complex networks: The visibility graph. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 105, n. 13, p. 4972–4975, 2008. Citations on pages 81, 84, and 111.

L'ECUYER, P.; SIMARD, R. Testu01: A library for empirical testing of random number generators. **ACM Transactions on Mathematical Software (TOMS)**, ACM, v. 33, n. 4, p. 22, 2007. Citations on pages 29, 37, 39, and 42.

LUISI, P.; ALVAREZ-PONCE, D.; PYBUS, M.; FARES, M. A.; BERTRANPETIT, J.; LAAYOUNI, H. Recent positive selection has acted on genes encoding proteins with more interactions within the whole human interactome. **Genome Biology and Evolution**, v. 7, n. 4, p. 1141–1154, 2015. Available: <<http://dx.doi.org/10.1093/gbe/evv055>>. Citation on page 115.

MACHICAO, J.; ALVES, M.; BAPTISTA, M. S.; BRUNO, O. M. Exploiting ergodicity of the logistic map using deep-zoom to improve security of chaos-based cryptosystems. **International Journal of Modern Physics C**, v. 30, n. 05, p. 1950033, 2019. Available: <<https://doi.org/10.1142/S0129183119500335>>. Citation on page 33.

MACHICAO, J.; BRUNO, O. M. Improving the pseudo-randomness properties of chaotic maps using deep-zoom. **Chaos: An Interdisciplinary Journal of Nonlinear Science**, AIP Publishing, v. 27, n. 5, p. 053116, 2017. Citations on pages 31, 32, 141, and 142.

MAGALHÃES, G. M.; PASSARO, A.; ABE, N. M. GERAÇÃO DE MALHA DE DELAUNAY ORIENTADA A OBJETOS. n. 10, p. 17–18, 2000. Available: <<https://www.researchgate.net/profile/Angelo{ }Passaro/publication/228604478{ }Geracao{ }de{ }malha{ }de{ }Delaunay{ }orientada{ }a{ }objetos/links/0deec5165fd3b0d8cd000000/Geracao-de-malha-de-Delaunay-orientada-a-objetos.pdf>>. Citation on page 59.

MASUDA, N.; JAKIMOSKI, G.; AIHARA, K.; KOCAREV, L. Chaotic block ciphers: from theory to practical algorithms. **IEEE Transactions on Circuits and Systems I: Regular Papers**, IEEE, v. 53, n. 6, p. 1341–1352, 2006. Citations on pages 30, 32, and 37.

MCAULEY, J. J.; COSTA, L. da F.; CAETANO, T. S. Rich-club phenomenon across complex network hierarchies. **Applied Physics Letters**, AIP, v. 91, n. 8, p. 084103, 2007. Citation on page 48.

MELO, T. de; FONSECA da. Relações históricas entre os jogos de azar e a probabilidade. 2000. Citation on page 28.

METZ, J.; CALVO, R.; SENO, E. R.; ROMERO, R. A.; LIANG, Z. Redes complexas: conceitos e aplicações. **Relatórios Técnicos do Instituto de Ciências Matemáticas e de Computação, USP, Janeiro**, 2007. Citations on pages 46 and 50.

MILO, R.; SHEN-ORR, S.; ITZKOVITZ, S.; KASHTAN, N.; CHKLOVSKII, D.; ALON, U. Network motifs: simple building blocks of complex networks. **Science**, American Association for the Advancement of Science, v. 298, n. 5594, p. 824–827, 2002. Citation on page 45.

MONTILLO, A.; LING, H. Age regression from faces using random forests. In: IEEE. **Image Processing (ICIP), 2009 16th IEEE International Conference on**. [S.l.], 2009. p. 2465–2468. Citation on page 60.

MOTTER, A. E.; LAI, Y.-C. Cascade-based attacks on complex networks. **Physical Review E, APS**, v. 66, n. 6, p. 065102, 2002. Citation on page 45.

MOURA, A. L. Uma proposta para a triangulação de delaunay 2d e localização planar de pontos em ocaml. Universidade Federal de Uberlândia, 2006. Citations on pages 59 and 60.

MOYA-GARCÍA, A. A.; RANEA, J. A. Insights into polypharmacology from drug-domain associations. **Bioinformatics**, Oxford University Press, v. 29, n. 16, p. 1934–1937, 2013. Citation on page 115.

MULLER, B.; RICHARDS, A. J.; JIN, B.; LU, X. Gographer: A python library for go graph representation and analysis. **BMC research notes**, BioMed Central, v. 2, n. 1, p. 122, 2009. Citation on page 115.

MUREDDU, M.; CALDARELLI, G.; DAMIANO, A.; SCALA, A.; MEYER-ORTMANN, H. Islanding the power grid on the transmission level: less connections for more security. **Scientific reports**, Nature Publishing Group, v. 6, p. 34797, 2016. Citations on pages 58 and 59.

NASCIMENTO, W. S. **Sobre algumas características da entropia de Shannon para sistemas atômicos confinados**. Master's Thesis (Master's Thesis) — UNIVERSIDADE FEDERAL DA BAHIA, 2013. Citation on page 47.

NEWMAN, M.; WATTS, D. J.; BARABÁSI, A.-L. **The structure and dynamics of networks**. Princeton, N.J. Oxford: Princeton University Press, 2006. ISBN 978-0-691-11356-2. Citations on pages 49, 51, and 52.

NEWMAN, M. E. A measure of betweenness centrality based on random walks. **Social networks**, Elsevier, v. 27, n. 1, p. 39–54, 2005. Citation on page 47.

NEWMAN, M. E.; GIRVAN, M. Finding and evaluating community structure in networks. **Physical review E, APS**, v. 69, n. 2, p. 026113, 2004. Citation on page 47.

NEWMAN, M. E. J. **Networks : an introduction**. Oxford New York: Oxford University Press, 2010. ISBN 978-0-19-920665-0. Citations on pages 46 and 49.

OKABE, A.; BOOTS, B.; SUGIHARA, K.; CHIU, S. N. **Spatial Tessellations: Concepts and Applications of Voronoi Diagrams**. 2. ed. Wiley, 2000. ISBN 9780471986355. Available: <<http://amazon.com/o/ASIN/0471986356/>>. Citations on pages 55, 56, 57, 58, and 59.

_____. **Spatial tessellations: concepts and applications of Voronoi diagrams**. [S.l.]: John Wiley & Sons, 2009. Citation on page 56.

OLIVEIRA, M. Q. A.; CAVALCANTE, T. S. G.; RAMOS, H. S.; ROSSO, O.; FRERY, A. C. Avalia ao de Geradores de umeros Pseudoaleatórios Através de Técnicas da Teoria da Informa ao. In: **Terceira Escola Regional de Alto Desempenho (ERAD) - Região Nordeste**. Maceió-Alagoas, Brazil: [s.n.], 2015. Available: <<http://erad.lccv.ufal.br/Downloads/Trabalhos/Paper{ }Avaliacao{ }de{ }Geradores{ }de{ }Numeros{ }Pseudo{ }Aleatorios.pdf>>. Citations on pages 39 and 43.

PADILLA, T. Humanities data in the library: Integrity, form, access. **D-Lib Magazine**, v. 22, n. 3/4, 2016. Citation on page 115.

PEREZ, F.; GRANGER, B. E.; HUNTER, J. D. Python: an ecosystem for scientific computing. **Computing in Science & Engineering**, IEEE, v. 13, n. 2, p. 13–21, 2011. Citation on page 115.

PERSOHN, K.; POVINELLI, R. J. Analyzing logistic map pseudorandom number generators for periodicity induced by finite precision floating-point representation. **Chaos, Solitons & Fractals**, Elsevier, v. 45, n. 3, p. 238–245, 2012. Citation on page 33.

PETERSEN, W. Lagged fibonacci series random number generators for the nec sx-3. **International Journal of High Speed Computing**, World Scientific, v. 6, n. 03, p. 387–398, 1994. Citation on page 34.

PITERI, M. A.; JUNIOR, M. M. Triangulação de delaunay e o princípio de inserção randomizado. **II Simpósio Brasileiro de Geomática-V Colóquio Brasileiro de Ciências Geodésicas, Presidente Prudente-SP**, p. 9, 2007. Citations on pages 59 and 60.

PRESTES, E. **Introdução a Teoria dos Grafos**. [s.n.], 2016. 122 p. Available: <<http://www.inf.ufrgs.br/{ }prestes/Courses/GraphTheory/Livro/ParteLivroGrafos.pdf>>. Citations on pages 47 and 48.

QUEIROZ, G. **Algoritmos Geométricos para Banco de Dados geográficos: da teoria à prática na TerraLib**. São José dos Campos, SP: INPE-Instituto Nacional de Pesquisas Espaciais, 2003. Phd Thesis (PhD Thesis) — Dissertação de Mestrado, Computação Aplicada, 2003. Citation on page 121.

RONQUI, J. R. F. **Medidas de centralidade em redes complexas: correlações, efetividade e caracterização de sistemas**. Phd Thesis (PhD Thesis) — Universidade de São Paulo, 2014. Citation on page 48.

RUKHIN, A.; SOTO, J.; NECHVATAL, J.; SMID, M.; BARKER, E. **A statistical test suite for random and pseudorandom number generators for cryptographic applications**. [S.l.], 2001. Citations on pages 40 and 41.

SADOVSKY, A. J.; MACLEAN, J. N. Mouse visual neocortex supports multiple stereotyped patterns of microcircuit activity. **Journal of Neuroscience**, Soc Neuroscience, v. 34, n. 23, p. 7769–7777, 2014. Citation on page 115.

SCABINI, L. F.; CONDORI, R. H.; GONÇALVES, W. N.; BRUNO, O. M. Multilayer complex network descriptors for color–texture characterization. **Information Sciences**, v. 491, p. 30 – 47, 2019. ISSN 0020-0255. Available: <<http://www.sciencedirect.com/science/article/pii/S0020025519301847>>. Citations on pages 109 and 110.

SCHELLEKENS, D.; PRENEEL, B.; VERBAUWHEDE, I. Fpga vendor agnostic true random number generator. In: IEEE. **Field Programmable Logic and Applications, 2006. FPL'06. International Conference on**. [S.l.], 2006. p. 1–6. Citation on page [30](#).

SHAHBAZI, A.; TAPPENDEN, A. F.; MILLER, J. Centroidal voronoi tessellations-a new approach to random testing. **IEEE Transactions on Software Engineering**, IEEE, v. 39, n. 2, p. 163–183, 2013. Citations on pages [31](#) and [43](#).

SHAMOS, M. I.; HOEY, D. Closest-point problems. In: IEEE. **Foundations of Computer Science, 1975., 16th Annual Symposium on**. [S.l.], 1975. p. 151–162. Citation on page [57](#).

SIDORENKO, A.; SCHOENMAKERS, B. Concrete security of the blum-blum-shub pseudo-random generator. In: SPRINGER. **IMA International Conference on Cryptography and Coding**. [S.l.], 2005. p. 355–375. Citations on pages [31](#) and [37](#).

SMEDT, T. D.; DAELEMANS, W. Pattern for python. **Journal of Machine Learning Research**, v. 13, n. Jun, p. 2063–2067, 2012. Citation on page [115](#).

SOTO, J. Statistical testing of random number generators. In: NIST GAITHERSBURG, MD. **Proceedings of the 22nd National Information Systems Security Conference**. [S.l.], 1999. v. 10, n. 99, p. 12. Citations on pages [43](#) and [138](#).

STALLINGS, W. **Computer security : principles and practice**. Upper Saddle River, N.J: Prentice Hall, 2008. ISBN 0-13-600424-5. Citations on pages [30](#) and [41](#).

STEFANOV, A.; GISIN, N.; GUINNARD, O.; GUINNARD, L.; ZBINDEN, H. Optical quantum random number generator. **Journal of Modern Optics**, Taylor & Francis, v. 47, n. 4, p. 595–598, 2000. Citation on page [30](#).

SUDBØ, J.; MARCELPOIL, R.; REITH, A. New algorithms based on the voronoi diagram applied in a pilot study on normal mucosa and carcinomas. **Analytical Cellular Pathology**, IOS Press, v. 21, n. 2, p. 71–86, 2000. Citation on page [60](#).

SZCZEPANSKI, J.; AMIGÓ, J. M.; MICHALEK, T.; KOCAREV, L. Cryptographically secure substitutions based on the approximation of mixing maps. **IEEE Transactions on Circuits and Systems I: Regular Papers**, IEEE, v. 52, n. 2, p. 443–453, 2005. Citation on page [30](#).

TAUSWORTHE, R. C. Random numbers generated by linear recurrence modulo two. **Mathematics of Computation**, JSTOR, v. 19, n. 90, p. 201–209, 1965. Citation on page [29](#).

TIAN, X.; BENKRID, K. Mersenne twister random number generation on fpga, cpu and gpu. In: IEEE. **Adaptive Hardware and Systems, 2009. AHS 2009. NASA/ESA Conference on**. [S.l.], 2009. p. 460–464. Citation on page [33](#).

VAVREK, R.; BALÁZS, L.; MÉSZÁROS, A.; HORVÁTH, I.; BAGOLY, Z. Testing the randomness in the sky-distribution of gamma-ray bursts. **Monthly Notices of the Royal Astronomical Society**, Blackwell Publishing Ltd Oxford, UK, v. 391, n. 4, p. 1741–1748, 2008. Citation on page [57](#).

VIALI, L. Algumas considerações sobre a origem da teoria da probabilidade. **Revista Brasileira de História da Matemática**, v. 8, n. 16, p. 143–53, 2008. Citation on page [28](#).

VYBORNOVA, Y. Password-based key derivation function as one of blum-blum-shub pseudo-random generator applications. **Procedia engineering**, Elsevier, v. 201, p. 428–435, 2017. Citation on page 35.

WANDELT, S.; SUN, X. Complex network analysis: The need for speed. In: IEEE. **Control Conference (CCC), 2016 35th Chinese**. [S.l.], 2016. p. 1213–1218. Citation on page 116.

WOLFRAM, S. **Random sequence generators**. Google Patents, 1987. US Patent 4,691,291. Available: <<https://www.google.com/patents/US4691291>>. Citations on pages 29 and 36.

_____. **A new kind of science**. [S.l.]: Wolfram media Champaign, 2002. Citations on pages 28, 29, 33, 36, and 37.

XU, K.; TAKAHARA, G.; HASSANEIN, H. On the robustness of grid-based deployment in wireless sensor networks. In: ACM. **Proceedings of the 2006 international conference on Wireless communications and mobile computing**. [S.l.], 2006. p. 1183–1188. Citation on page 60.

YASCHENKO, V. V. **Cryptography : an introduction**. Providence, RI: American Mathematical Society, 2002. ISBN 0-8218-2986-6. Citations on pages 23, 31, and 37.

PRNG-BASED GRAPHS

Figure 48 – Mersenne 64bits

Figure 49 – ALFG

Figure 50 – BlumBlumShub

Figure 51 – BC

Figure 52 – Java

Figure 53 – JDK

Figure 54 – Linear Congregational Generator

Figure 55 – Linear Congregational Generator

Figure 56 – Mersenne Twister

Figure 57 – Cellular Automata Rule 30

Table 9 – Descriptions of the NIST Statistical Test Suite

Statistical Test	Defect Detected
Frequency	Too many zeros or ones.
Cumulative Sums	Too many zeros or ones at the beginning of the sequence.
Longest Runs Of Ones	Deviation of the distribution of long runs of ones.
Runs	Deviation of the distribution of long runs of ones. Large (small) total number of runs indicates that the oscillation in the bit stream is too fast (too slow).
Rank	Deviation of the rank distribution from a corresponding random sequence, due to periodicity.
Spectral	Periodic features in the bit stream.
Non-overlapping Template Matchings	Too many occurrences of non-periodic templates.
Overlapping Template Matchings	Too many occurrences of m-bit runs of ones.
Universal Statistical	Compressibility (regularity).
Random Excursions	Deviation from the distribution of the number of visits of a random walk to a certain state.
Random Excursion Variant	Deviation from the distribution of the total number of visits (across many random walks) to a certain state.
Approximate Entropy	Non-uniform distribution of m-length words. Small values of $ApEn(m)$ imply strong regularity.
Serial	Non-uniform distribution of m-length words. Similar to Approximate Entropy
Lempel-Ziv Complexity	More compressed than a truly random sequence.

Linear Complexity	Deviation from the distribution of the linear complexity for finite length (sub)strings.
-------------------	--

Source: Adapted from [Soto \(1999\)](#).

DIEHARD

Table 10 – Descriptions of the DIEHARD tests

Good-of-fit Test	Test Short Description
Birthday Spacings	Various groups of bits in the binary representation are used to simulate birthdays spacings.
Overlapping 5-Permutations	Observed numbers of states and transitions frequencies in the 120 states for 5! possible ordering of 5 numbers
Binary Rank for 31x31 matrices	A 31 x 31 binary matrix over the a finite field with 2 elements is formed. The rank of matrix is determined. Each time that this is done, counts are accumulated for matrices with ranks of 31,30,29,and 28 or less.
Binary Rank for 32x32 matrices	Similar to the test above.
Binary Rank for 6x8 matrices	Similar to the test above.
Bit stream	20-bit overlap words are created with the stream of bits in test. It is compared the observed number of missing 20-bit words in a string of 2^{21} overlapping 20-bit words with the approximate distribution of that number.
Overlapping-pairs-sparse-occupancy	2 words are formed. The letters in his words are determined by a specific ten bits from a 32-bit integer in the sequence to be tested.The test counts the number of missing words.
Overlapping-quadruples-sparse-occupancy	Similar to the test above.
DNA	Similar to the test above excepted that it uses ten-letter words built on a four-letter alphabet.

Count-the-1s test on a stream of bytes	This test is based on the binomial (8, 0.5) distribution of 1s in uniform random bytes. Rather than testing this directly, however, counts of 1s are grouped into five sets: 0,1,2, 3, 4, 5, and 6, 7, 8; that is if a byte has no 1s, exactly one 1, or exactly two 1s, it is counted in the first group. Next, overlapping sequences of length 5 are formed, and the counts of each of the 5^5 combinations are obtained. A chi-squared test is performed on the counts.
Parking lot	Based on the results of randomly packing circles of radius 1 about a center located randomly within a square of side 100.
Minimum distance	based in the minimum distance between a large number of random numbers in a square. If the points are independent uniform, then the This test is based on the minimum distance between a large exponentially square of the minimum distance should be approximately distributed with mean dependent on the length of the side of the square and the number of points. A chi-squared goodness-of-fit test is perform on the CDF values of a large number of tests for exponentially.
3-D spheres	4000 points are chosen randomly in a cube of edge 1000. At each point, a sphere is centered that is large enough to reach the next closest point. The volume of the smallest such sphere is approximately exponentially distributed with mean . Thus, the radius cubed is exponential with mean 30.
Squeeze test	Starting with $k = 2^{21}$, the test finds j , the number of iterations necessary to reduce k to 1, using the reduction $k = [k * U]$, with U from the sequence in test. Such j s are found 100,000 times and then counts for the number of times j was $\leq 6, 7, \dots, 47, \geq 48$ are used to provide a chi-squared test.
Overlapping sums test	Sums of overlapping subsequences of 100 uniforms are tested for multivariate normally by chi-squared test several times and then a chi-squared goodness-of-fit is performed on the CDF values of the chi-squared tests.
Craps	Simulates games pf craps and counts the number of wins and number of throws necessary to end each game and then .
Runs	It runs up and downs tests.

Source: Adapted from [Gentle \(2003\)](#).

K-LOGISTIC MAP TESTING

Table 11 – Average number of files that passed DIEHARD tests using the k -logistic map PRNG from 100 file samples. Severely failed tests are shown in bold. All tests passed using the interval $0.0001 < p\text{-value} < 0.9999$.

DIEHARD tests	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$
BirthdaySpacings [KS]	100	100	100	100	100	100	100	100	100	100
OverlappingPermutations	99	97	98	95	98	96	98	98	99	100
Ranks31x31 matrices	100	100	100	100	100	100	100	100	100	100
Ranks32x32 matrices	100	100	100	100	100	100	100	100	100	100
Ranks6x8 matrices [KS]	0	0	25	99	100	100	100	100	100	100
Monkey20bitsWords [KS]	0	99	100	100	100	100	100	100	100	100
OPSO [KS]	98	99	100	100	100	100	100	100	100	100
OQSO [KS]	98	100	100	100	100	100	100	100	100	100
DNA [KS]	100	100	100	100	100	100	100	100	100	100
Count1sStream	0	0	0	98	100	100	100	100	100	100
Count1sSpecific [KS]	0	0	0	0	94	100	100	100	100	100
ParkingLot [KS]	100	100	100	100	100	100	100	100	100	100
MinimumDistance [KS]	96	100	100	100	100	100	100	100	99	100
RandomSpheres [KS]	100	100	100	100	100	100	100	100	100	100
Squeeze [KS]	100	100	100	100	100	100	100	100	100	100
OverlappingSums [KS]	100	100	100	100	100	100	100	100	100	100
Runs (up)	100	100	100	100	100	100	100	100	100	100
Runs (down)	100	100	100	100	100	100	100	100	100	100
Craps (wins)	100	100	100	100	100	100	100	100	100	100
Craps (throws/game)	100	100	100	100	100	100	100	100	100	100

Source: Adapted from [Machicao and Bruno \(2017\)](#).

Table 12 – Number of files that passed the NIST test suites for the k -logistic map. Failed tests are shown in bold. All the tests passed to the $\alpha = 0.01$ significance level.

NIST test	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$
Frequency	98	99	99	99	99	99	99	99	99	99
BlockFrequency ($m = 128$)	0	1	66	95	98	98	99	100	99	99
CumulativeSums										
Forward sums	97	98	99	99	98	99	99	99	99	99
Reverse sums	97	99	99	99	99	99	99	99	99	99
Runs	0	0	14	91	98	99	99	99	99	100
LongestRun	0	0	15	89	98	99	98	100	99	99
Rank	99	100	99	99	99	99	99	99	99	99
FFT	77	98	99	99	99	99	99	99	99	99
Non-overlappingTemplate										
000000001	0	0	48	97	99	99	99	100	99	99
000000011	0	3	87	98	99	99	99	99	99	99
000000101	0	41	94	98	98	99	99	99	98	99
OverlappingTemplate	0	0	11	93	98	99	98	99	99	99
Universal	0	68	97	98	99	99	99	99	99	99
ApproxEntropy ($m = 10$)	0	0	64	98	99	99	99	100	99	99
RandomExcursions										
$x = -4$	90	98	99	99	98	99	99	99	99	100
$x = -3$	91	97	99	99	99	99	99	99	99	99
$x = -2$	94	99	98	99	99	98	99	99	99	99
$x = -1$	95	99	99	98	99	99	99	99	99	100
RandExcursVar										
$x = -9$	99	100	99	99	100	99	99	99	99	100
$x = -8$	99	99	99	99	99	99	99	99	99	100
$x = -7$	100	99	99	99	99	99	99	99	99	100
$x = -6$	100	99	99	99	99	99	98	99	99	99
$x = -5$	100	99	99	99	99	99	99	99	99	99
$x = -4$	99	100	99	99	99	99	99	99	99	99
$x = -3$	99	100	99	98	99	99	99	99	99	99
$x = -2$	99	99	99	98	99	99	99	100	99	99
$x = -1$	99	99	99	99	99	99	99	99	99	99
Serial ($m = 16$)										
Serial 1	0	1	82	96	98	99	99	98	99	99
Serial 2	10	81	95	98	98	99	99	99	98	99
LinearComplexity ($M = 500$)	99	98	99	99	99	99	99	98	99	99

Source: Adapted from Machicao and Bruno (2017).

CLASSIFICATION ALGORITHMS TEST

Table 13 – K-logistic Map networks classification

Algorithm	Parameters	16 cross	16 split	8 cross	8 split	4 cross	4 split	2 cross	2 split
BayesNet (ABORTED)	classifier = Simple Estimator search = K2	52.99	52.11						
	classifier = BMAEstimator search = K2	52.99	51.82						
	classifier = SimpleEstimator search = Generic Search								
	classifier = BMAEstimator search = Generic Search								
Naive Bayes	userKernalEstimator=false useSuperisedDiscratization=false	37.89	36.89	34.72	33.87	34.27	33.20	32.39	31.98
	userKernalEstimator=false useSuperisedDiscratization=true	43.66	40.71	40.39	39.8	39.85	39.27	52.19	51.82
Logistic ABORTED	conjugateGradient = false	58.73	57.56	58.77	57.64				
	conjugateGradient = true	58.84	57.42	58.81	57.18				
	conjugateGradient = false maxIts = 20	54.01	55.00	53.99	55.11				
	conjugateGradient = true maxIts = 20	51.18	51.29	51.22	51.78				
Multilayer perceptron	decay = false	52.76	52.20	52.53	52.09	53.30	52.78	50.49	50.82
	decay = true	54.45	54.51	54.45	54.51	52.67	52.91	52.54	51.82
	decay = true altobuild= false	22.16	20.84	22.16	20.84	21.66	21.49	21.63	21.56
	decay = true altobuild= true	22.16	20.84	22.16	20.84	21.66	21.49	21.63	21.56
	traning time 700 treshhold = 20	22.07	20.84	22.07	20.84	21.66	21.40	21.68	21.53
	traning time 700 treshhold = 30	22.07	20.84	22.07	20.84	21.66	21.40	21.68	21.53
LDA	Default Batch and debug	55.94	55.31	55.94	55.31	55.91	55.44	55.36	55.44
QDA	Batch 200/100/50	41.58	42.02	41.58	42.02	41.59	41.98	41.07	42.22
Simple Logistic	heuristic Stop=50 Error On Probabilities=false	53.75	53.42	53.56	53.62	54.99	56.47	54.40	53.89
	heuristic Stop=50 Error On Probabilities=true	56.42	57.09	56.47	56.91	56.53	56.47	54.40	53.89
	heuristic Stop=60 Error On Probabilities=false	53.98	54.13	54.05	54.13	56.47	56.64	54.47	54.11
	heuristic Stop=60 Error On Probabilities=false	54.05	53.80	56.46	57.02	56.47	56.64	54.47	54.11
	use cross validation = true								
	calibration = true Normalized data	54.35	54.38	56.47	54.49	55.07	54.38	53.01	53.07

	calibration = true Standart Training data	56.76	56.84	56.57	57.09	57.01	56.51	54.89	54.53
	c = 1	56.37	56.22	56.57	57.09	57.01	56.51	52.78	54.53
	c = 2	56.51	56.98	56.57	57.09	57.70	56.91	52.91	54.76
	c = 3	56.81	57.58	56.83	56.78	57.96	57.02	21.49	55.09
	c = 4	56.45	57.20	56.51	57.18	57.44	57.29	21.49	54.80
	c = 5	56.93	57.24	56.83	57.16	58.21	57.31	21.40	54.98
	c = 6	56.96	57.20	56.63	57.20	58.14	57.47	21.40	55.09
	c = 1 kernel = Normalized Poly Kernel	47.57	47.87	46.39	45.62	46.07	45.11	45.45	44.76
	c = 1 kernel = Puk	48.02	48.11	45.05	44.80	44.03	43.87	44.06	43.36
	c = 1 kernel = RBF	39.45	38.93	49.42	48.78	49.70	49.27	49.82	48.22
Knn	1	40.44	40.96	40.41	40.96	40.73	40.82	42.89	42.89
Classification Regression	classifier = m5	55.79	55.29	55.70	55.38	55.41	55.69	54.07	54.53
	classifier = Random Forest	58.43	58.11	58.87	57.80	58.23	57.80	57.15	56.18
	classifier = Random Tree	39.06	40.58	40.59	40.1	39.83	40.89	38.46	38.49
MultiClass Classifier	Logistic - one against all	55.65	55.91	55.57	55.78	56.49	57.44	52.15	53.53
	Logistic - Random Correction code	54.08	53.80	54.36	53.53	53.85	54.02	50.73	51.13
	Logistic - Exhaustive correction code	55.93	56.20	55.90	56.20	56.68	57.56	53.13	53.29
	Logistic - One - against - one	57.72	57.29	57.70	57.11	58.40	58.58	55.09	54.80
	SMO - one against all	29.59	29.53	29.61	29.60	29.33	29.62	26.75	26.64
	SMO- Random Correction code	49.95	48.33	50.23	48.40	50.08	48.33	47.13	46.24
	SMO- Exhaustive correction code	48.98	49.04	49.00	49.00	49.72	49.78	47.59	47.38
	SMO- One - against - one	55.03	54.82	54.94	54.87	55.87	55.31	53.00	52.96
Vote ABORTED	Average of Probabilities	51.16	53.44						
	Majority voting	54.53	54.67						
	Average of Probabilities	45.67	54.67						
	Majority voting	58.35	54.67						
Decision Table	Best First	56.86	52.20	53.86	52.22	55.15	53.22	51.99	49.36
	Greedy Stepwise	53.86	52.20	53.86	52.22	55.17	53.18	51.95	49.36
	Best First - Accuracy	53.86	52.20	55.72	52.22	55.15	53.18	51.99	49.36
	Best First - RMSE	55.72	53.29	53.51	53.29	55.61	53.78	53.51	51.02
	Best First - MAE	55.26	52.58	55.26	52.58	55.05	53.69	52.37	50.73
	Best First - AUC	53.64	51.67	53.64	51.67	53.93	52.29	52.33	49.33
	Greedy - Accuracy	53.86	52.20	53.86	52.22	55.17	53.22	51.95	49.36
	Greedy - RMSE	55.72	53.29	55.72	53.29	55.61	53.78	53.51	51.02
	Greedy - MAE	55.26	52.58	55.26	52.58	55.05	53.69	52.37	50.73
	Greedy - AUC	53.64	51.67	53.64	51.67	53.93	52.29	52.33	49.33
	Best First - Accuracy	54.16	52.16	54.16	52.18	53.47	52.00	52.11	49.40
	Best First - RMSE	56.06	53.42	56.06	53.42	55.21	53.40	53.67	51.07
	Best First - MAE	55.56	52.62	55.56	52.62	54.06	52.07	52.41	50.76
	Best First - AUC	53.94	51.69	53.94	51.69	53.83	50.44	52.71	49.78
	Greedy - Accuracy	54.16	52.16	54.16	52.18	53.43	52.00	52.07	49.40
	Greedy - RMSE	56.06	53.42	56.06	53.42	55.21	53.47	53.51	51.02
	Greedy - MAE	55.56	52.62	55.56	52.62	54.06	52.07	52.41	50.76
	Greedy - AUC	53.94	51.69	53.94	51.69	53.83	50.44	52.71	49.78
REPTree	Weka default	53.76	52.20	58.87	57.8	53.06	52.73	52.33	51.67
Random Tree	Weka default	47.47	46.22	40.59	40.11	58.04	56.49	45.91	44.56
Random Forest	Weka default	58.39	57.89	52.68	51.44	46.72	46.22	57.35	56.20

DEGREE DISTRIBUTION FOR BAND OF THE K-LOGISTIC MAP PRNG

Figure 58 – Degree distribution for band of the k-logistic map PRNG

(a) K0

(b) K1

(c) K2

(d) K3

(e) K4

(f) K5

(g) K6

(h) K7

(i) K8

(j) K9

Source: Elaborated by the author.

LEAVES AND ISOLATED NODES DISTRIBUTION FOR BAND

Figure 60 – Leaves and isolated nodes distribution for band

(e) K4

(f) K5

(g) K6

(h) K7

(i) K8

(j) K9

Source: Elaborated by the author.

BOX-PLOTS OF DELAUNAY TRIANGULATION

Figure 62 – Box-plots of PCA on Delaunay triangulation using different color schemes

(e) 120 colors

(f) 150 colors

(g) 240 colors

(h) 500 colors

(i) 1500 colors

(j) 3000 colors

Source: Elaborated by the author.

BOX-PLOTS OF VORONOI DIAGRAM

Figure 64 – Box-plots of PCA on Voronoi diagrams using different color schemes

(a) 5 colors

(b) 10 colors

(c) 20 colors

(d) 30 colors

(e) 60 colors

(f) 120 colors

(g) 180 colors

(h) 210 colors

(i) 240 colors

(j) 270 colors

Source: Elaborated by the author.