FACULTY OF COMPUTER SCIENCE IAȘI

UNIVERSITATEA „ALEXANDRU IOAN CUZA“ din IAȘI

Binary Matrix Rank Test

Nicu Neculache, Vlad-Andrei Petcu

Coordiantor: Prof. Simion Emil

Information Security - Academic Year 2021-2022

Cryptanalysis Techniques and Methods

Agenda

- Introduction
- Binary Matrix Rank Test

  - Mathematical fundamentals

  - Implementation
- Evaluation & observations
- Conclusions

Introduction

Statistical testing - mathematical technique for analysing an algorithm based on some input-output pairs (testing samples)

Run the algorithm (or the system) multiple times, **obtain the results** and **analyze** them in order to **classify/validate** the algorithm

Commonly used in the field of **cryptography**, specifically for the encryption, decryption and the keys or sub-keys generation

Binary Matrix Rank Test

Focus - the rank of disjoint sub-matrices of the entire sequence

Purpose - check for linear dependence among fixed length sub-strings of the original sequence

Main idea

- construct matrices of successive zeroes and ones from the sequence
- check for linear dependence among the rows or columns of the constructed matrices

The statistic of interest - the deviation of the rank from the expected value

Mathematical fundamentals

The rank R of the M × Q random binary matrix takes values r = 0, 1, 2, . . . , m where m ≡ min(M, Q) with probabilities:

$$p_r = 2^{r(Q+M-r)-MQ} \prod_{i=0}^{r-1} \frac{(1-2^{i-Q})(1-2^{i-M})}{1-2^{i-r}}$$

For M = Q = 32:

$$p_M \approx 0.2888..., \ p_{M-1} \approx 0.5776..., \ p_{M-2} \approx 0.1284...$$

Rank frequencies:

$$F_M = \#\{R_l = M\}, \quad F_{M-1} = \#\{R_l = M-1\}$$

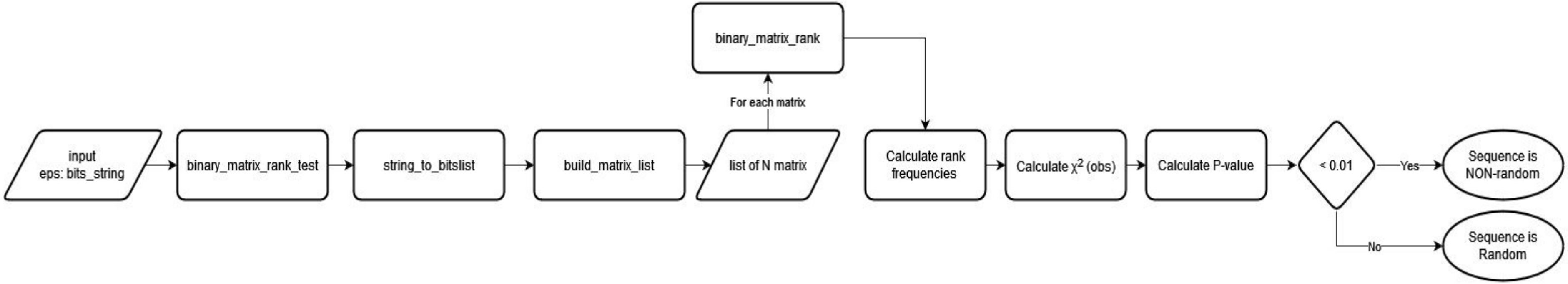
Reference distribution (*chi*):

$$\chi^2 = \frac{(F_M - p_M N)^2}{p_M N} + \frac{(F_{M-1} - p_{M-1} N)^2}{p_{M-1} N} + \frac{(N - F_M - F_{M-1} - p_{M-2} N)^2}{p_{M-2} N}$$

P-*value* = $e^{-\chi^2(obs)/2}$

Output: P-*value* < 0.01

Implementation



Evaluation & observations

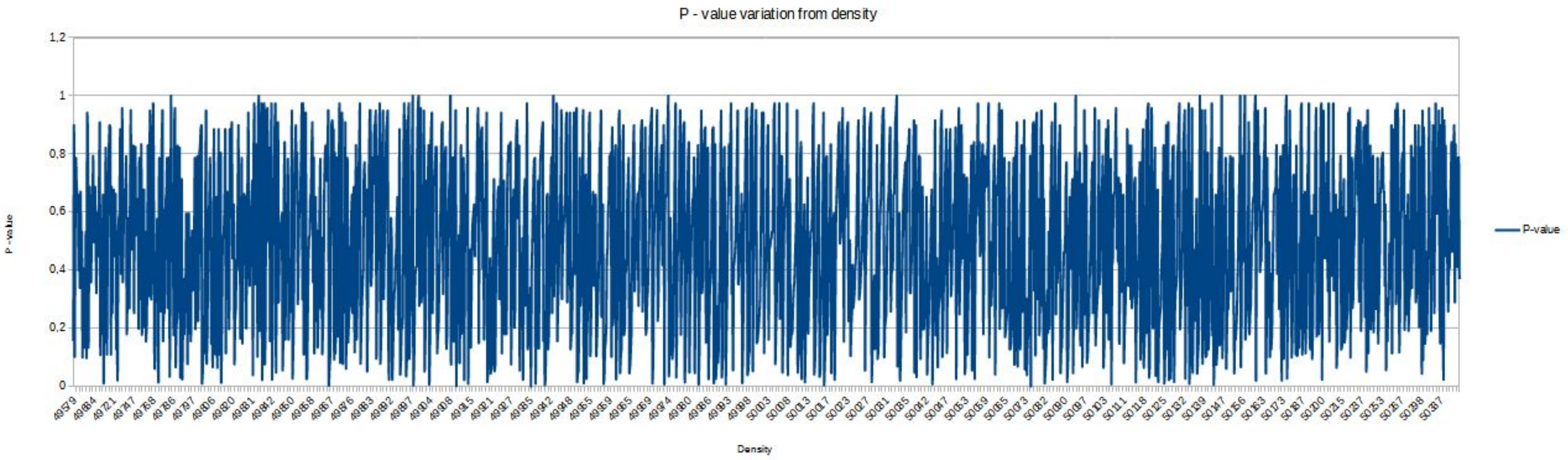
The **variation** of P-*value* from **density** of bits

- Python random library, input length = 100000
- Python random library - density iterations, input length = 50000
- Random/Non-random count from density percentage

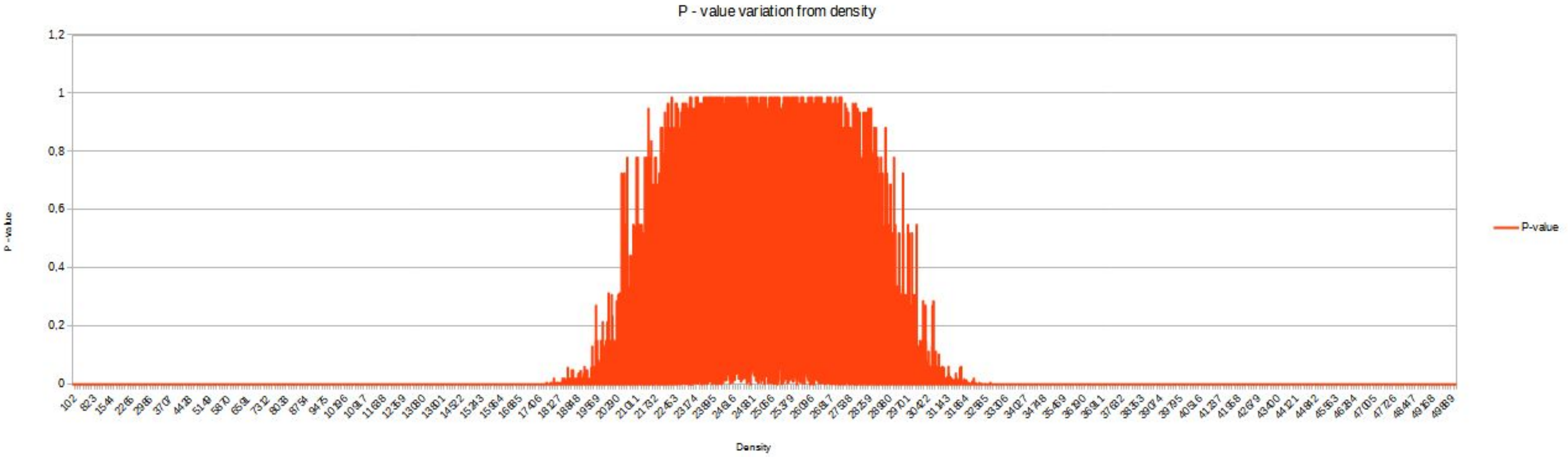
FACULTY OF COMPUTER SCIENCE IAȘI

UNIVERSITATEA „ALEXANDRU IOAN CUZA“ din IAȘI

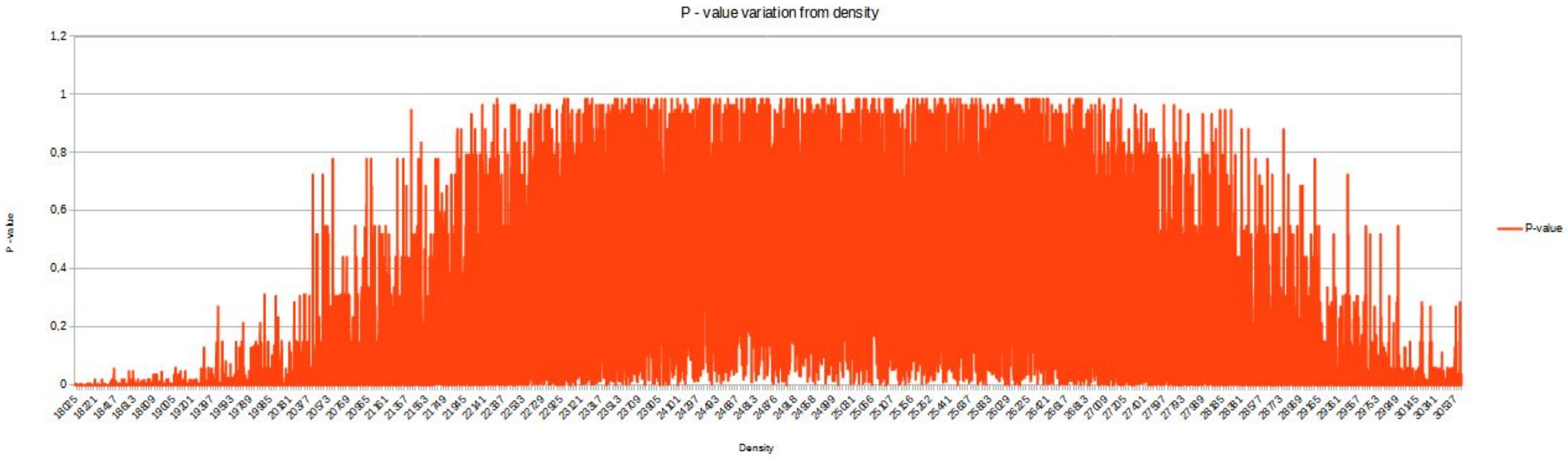
Python random library, input length = 100000



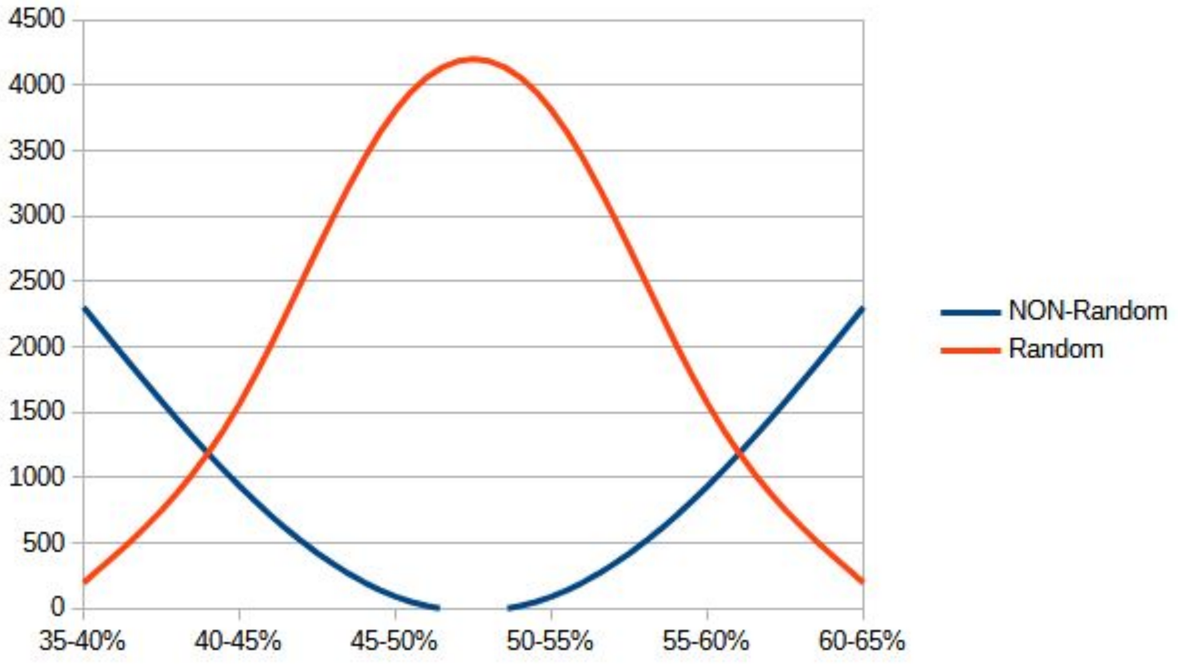
Python random library - density iterations, input length = 50000



Python random library - density iterations 40-60%, input length = 50000



Random/Non-random count from density percentage



Conclusions

Bits density target: ≅ 50%