ПРОГРАММНОЕ СРЕДСТВО ВИЗУАЛИЗАЦИИ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ ИСТОЧНИКОВ СЛУЧАЙНЫХ ЧИСЕЛ НА ЖИВУЧЕСТЬ

Бурко Л.А.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Иванюк А.А. – доктор технических наук

Аннотация. В данной работе представлена разработка программного средства для визуализации результатов тестирования источников случайных чисел на живучесть, реализованного при помощи языка программирования Python, а именно библиотеки PyQt5. Приведен анализ тестирования случайных последовательностей от разных источников на обнаружение различного рода зависимостей.

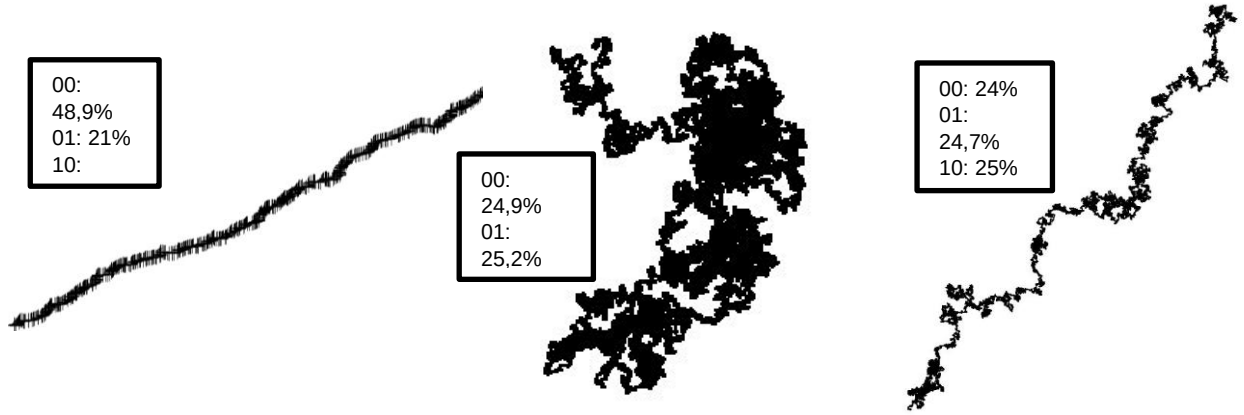
В современных цифровых системах случайные числа применяются повсеместно. Они играют ключевую роль в обеспечении безопасности, точности и надежности вычислительных процессов. Поэтому, вопросы, связанные с эффективной генерацией случайных чисел, остаются актуальным предметом многих научных исследований.

Системы, которые генерируют случайные числа могут быть подвержены сбоям и взломам. Для быстрого обнаружения проблем существуют различные методы тестирования источников случайных чисел как в режиме реального времени, так и для постобработки. Например, специальные рекомендации NIST 800-90B [1], BSI AIS20/31 [2] предлагаются варианты тестов для источников случайных чисел. Если основные тесты пройдены успешно – можно считать, что источник работает корректно. В прошлых исследованиях уже были изучены основные тесты из NIST 800-90B [3], а именно Adaptive Proportion test (T1) и Repetition Count Test (T2).

Цель данной работы – создать удобное для пользователя приложение, которое наглядно демонстрирует состояние источника случайных данных, а также позволяет провести анализ битовых последовательностей на различного рода зависимости.

Приложение написано на языке Python при помощи библиотеки PyQt5 [4]. Источник случайных чисел по запросу генерирует однобитный символ. Реализована программная функция, которая получает символ от указанного источника и формирует битовую последовательность. В дальнейшем для обработки последовательность делится на слова по N бит. В приложении можно регулировать параметр N. При запуске показаны $K = k * N$ текущих бит последовательности, где $k = 1,2$ ...; результат прохождения тестов T1 и T2; текущее значение энтропии Шеннона $H(x) = -\sum_{i=1}^{n} P_i \log_2 P_i$, где P_i – вероятность появление символа от источника случайных чисел, и график ее изменения; оценка по Маркову; графический тест Random Walk; графический тест распределения на плоскости (динамический, статический).

Графический тест Random Walk (RW) описывает путь, состоящий из последовательности шагов в математическом пространстве. Последовательность разбивается по два символа, в зависимости от комбинации символов производится шаг в определенном направлении: ‘00’ – вправо, ‘01’ – вверх, ‘10’ – влево, ‘11’ – вниз. Для исследования были взяты три источника случайных чисел (рисунок 1): встроенная функция *«random»* с заведомо плохим распределением (а), где $P(1) = 0.3$; генератор псевдослучайных последовательностей LFSR32 (б) и ФНФ типа Арбитр на ПЛИС фирмы Xilinx (в).

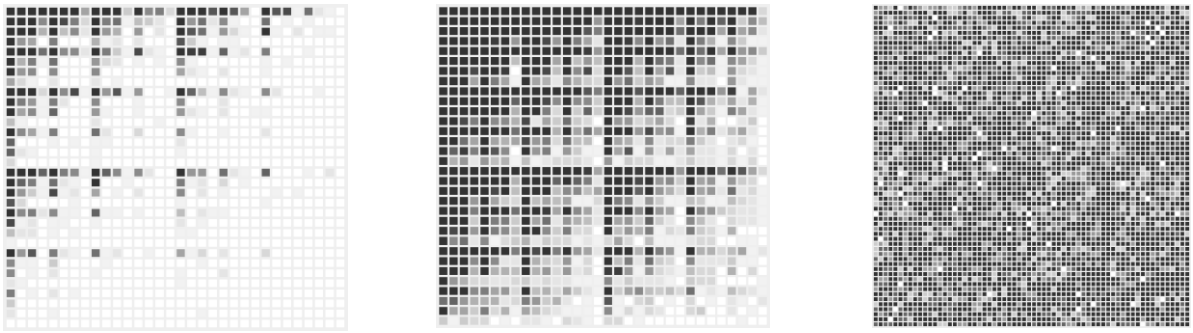


(а) (б) (в)

Рисунок 1 – Результаты теста Random Walk

Для графического теста битовая последовательность разбивается по N бит, и формируется числовая последовательность $x_0,\ x_1, \dots, x_i, \dots, x_k$, из которой составляются точки $(x_0,\ x_1), (x_1,\ x_2) \dots (x_{k-1},\ x_k)$. Если тест динамический, точка отрисовывается черным и при повторной встрече цвет точки инвертируется, а если тест статический, то в качестве начального цвета задается серый и при каждом последующем появлении точки увеличивается глубина цвета.

На рисунках 2а и 2б – результат теста для функции *«random»,* где $P(1) = 0.1$ и $P(1) = 0.3$ соответственно. Очевидно, из-за преобладающего количества нулей, числа расположены ближе к осям $x = 0, y = 0$ и на осях, значения которых кратны степеням двойки. На рисунке 2в пример «хорошего» источника случайных чисел, последовательность от LFSR32.



(а) (б) (в)

Рисунок 2 – Результаты графического теста

Для анализа последовательностей важной составляющей является обнаружение зависимостей. В битовых последовательностях для этого хорошо подходят тест «покер» из стандарта BSI AIS20/31 (отдельно из-за смещения он позволяет обнаружить зависимости, которые не охватываются стохастическим анализом) и оценка энтропии по Маркову из NIST 800-90B. Как известно, в марковском процессе следующее значение зависит только от текущего. Оценка Маркова дает оценку минимальной энтропии путем измерения зависимости между последовательными значениями из входного набора данных. По этим данным определяются вероятности как начального состояния, так и переходов между любыми двумя состояниями.

Исследуется битовая последовательность размером M. Необходимо найти значения $V(00), V(01), V(10), V(11)$, где $V(\#\#)$ – это количество комбинаций ## с учетом перекрытия на всей последовательности. Вероятности $P_{i,j}$ появления бита j после бита i находится по формулам:

$$P_{0,0} = \left(1 + \frac{V(01)}{V(00)}\right)^{-1}, P_{0,1} = \left(1 + \frac{V(00)}{V(01)}\right)^{-1}, P_{1,0} = \left(1 + \frac{V(11)}{V(10)}\right)^{-1}, P_{1,1} = \left(1 + \frac{V(10)}{V(11)}\right)^{-1}.$$

Далее необходимо найти $\alpha 1, \dots, \alpha 6$ – вероятности появления определенной 128-битной подпоследовательности (A) среди исследуемой последовательности размером M [1]:

$$\alpha 1 = P_0 \cdot {P_{0,0}}^{127}\ (000 \dots 0), \quad \alpha 2 = P_0 \cdot {P_{0,1}}^{64} \cdot {P_{1,0}}^{63}\ (010 \dots 1), \quad \alpha 3 = P_0 \cdot P_{0,1} \cdot {P_{1,1}}^{126}\ (011 \dots 1),$$
$$\alpha 4 = P_1 \cdot P_{1,0} \cdot {P_{0,0}}^{126} (100 \dots 0), \quad \alpha 5 = P_1 \cdot {P_{1,0}}^{64} \cdot {P_{0,1}}^{63}\ (101 \dots 0), \quad \alpha 6 = P_1 \cdot {P_{1,1}}^{127} (111 \dots 1),$$

где $P_0 = \frac{V(0)}{M}$, $P_1 = 1 - P_0$, ${P_{i,j}}^r$ – вероятность, что комбинация битов ij встретится r раз на A.

Энтропия находится по формуле: $\min\left(-\log_2 \frac{\alpha_{max}}{128}, 1\right)$, где $\alpha_{max} = \max(\alpha 1, \alpha 2, \alpha 3, \alpha 4, \alpha 5, \alpha 6)$. В таблице 1 представлены результаты по четырем исследуемым наборам данных.

Таблица 1 – Марковский анализ последовательности различных источников случайных чисел

Вероятность	Random (0.1)	Random (0.3)	ФНФ	LFSR32
$P_0,\ P_1$	0.9, 0.1	0.7, 0.3	0.489, 0.511	0.49967, 0.50033
$P_{0,0},\ P_{0,1}$	0.825, 0,175	0.576, 0.424	0.392, 0.608	0.39934, 0.60066
$P_{1,0},\ P_{1,1}$	0.905, 0.095	0.752, 0.248	0.590, 0.410	0.60009, 0.39990
Энтропия, α_{max}	0.273, $\alpha 1$	0.786, $\alpha 1$	0.741, $\alpha 5$	0.738, $\alpha 5$

В результате разработки программного средства были исследованы различные методы визуализации основных характеристик случайных последовательностей. Дальнейшая работа будет заключаться в расширении перечня тестов и добавлении 3D-визуализации результатов.

Список использованных источников:

1. NIST 800-90B, Recommendation for the Entropy Sources Used for Random Bit Generation / https://doi.org/10.6028/NIST.SP.800-90B

2. A Proposal for Functionality Classes for Random Number Generators

3. Information Tehnologies and Systems 2023 (ITS 2023) : материалы международной научной конференции, Минск, Беларусь, 22 ноября / Л. Ю. Шилин [и др.]. – Минск : БГУИР, 2023.

4. https://www.riverbankcomputing.com/static/Docs/PyQt5/

A SOFTWARE TOOL FOR VISUALIZING THE RESULTS OF TESTING RANDOM NUMBER SOURCES FOR SURVIVABILITY

Burko L.A.

Belarusian State University of Informatics and Radioelectronics[1], Minsk, Republic of Belarus

Republican Scientific and Practical Center of Traumatology and Orthopedics[2], Minsk, Republic of Belarus

Ivanyuk A.A. – PhD

Annotation. This paper describes hardware and software implementation of a coordination and rehabilitation complex for diagnosing and preventing disorders of the musculoskeletal system. The game aspect of using the complex is examined individually.

Keywords. Coordination and rehabilitation complex, musculoskeletal disorders, game methods of data collection, rehabilitation, coordinate transformations, Euler angles, polar coordinate system, window coordinate system, frequency diagram.