9. ANALYSIS OF PSEUDO-RANDOM NUMBER GENERATION METHODS

Vorotnitskaya M.V.

Belarusian State University of Informatics and Radioelectronics
Minsk, Republic of Belarus

Kaspiarovich N.G. – Senior Lecturer

The information about the analysis of pseudo-random number generation methods is provided. The qualitative characteristics of the corresponding algorithms are presented in this paper.

There are different methods of generating random numbers. One of the basic and most understandable is the dictionary, where we first save a set of numbers and take them in turn. The first technical methods of obtaining random numbers were generators using entropy. These are devices that are based on physical properties such as capacitance of a capacitor, radio wave noise, etc. However, this method lacks one of the important criteria which is repeatability [1].

All current random number generators can be divided into two types: true random number generator (TRNG) and pseudo-random number generator (PRNG). These two types of generators differ in the way they get a random number. The TRNG uses a physical process to produce a random number. The PRNG also uses mathematical algorithms that are completely produced by a computer [2].

You can use four different algorithms to generate random numbers: The Lehmer's algorithm, the linear congruent algorithm, the Wichmann-Hill algorithm, and the Fibonacci algorithm with delays. However, none of the algorithms presented has cryptographic level of reliability and does not require static strictness.

One of the simplest algorithms for generating pseudo-random numbers is **the Lehmer's algorithm**. The definition of this algorithm can be formulated in the following way: each new random number is an old random number multiplied by the constant *a*, after which an operation is performed on the result modulo the constant *m* [3]. Figure 1 displays the distribution of this algorithm.

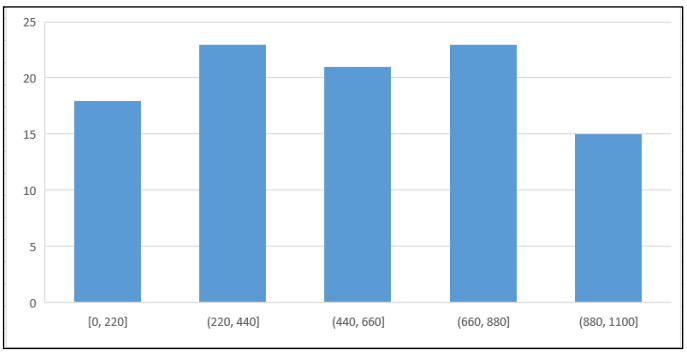

Figure 1 – Histogram of the distribution of the Lehmer's algorithm

The Wichman-Hill algorithm dates back to 1982. The idea of Wichman-Hill is to generate three preliminary results and then combine them into one final result. Since the Wichman-Hill algorithm uses three different generating equations, it requires three initial values. The Wichman-Hill algorithm is a more difficult to implement than the Lehmer's algorithm. The advantage of the Wichman-Hill algorithm is that it generates a longer sequence (more than 6,000,000,000,000 values) before it starts repeating [3]. Figure 2 features the distribution of this algorithm.

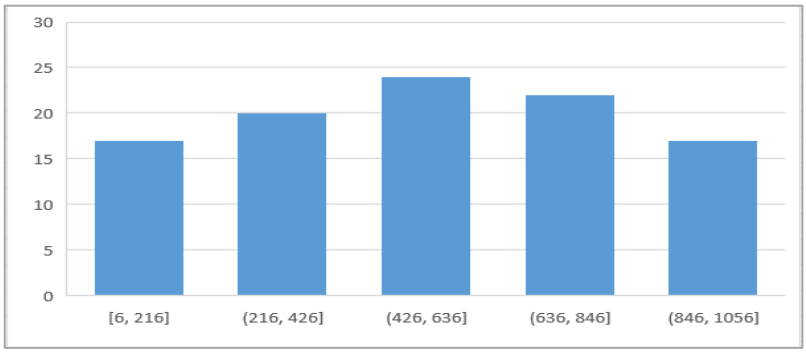

Figure 2 – Histogram of the distribution of the Wichman-Hill algorithm

It proves that both the Lehmer's algorithm and the Wichman-Hill algorithm can be considered special cases of the so-called **linear congruent (linear congruential, LC) algorithm**. It exactly corresponds to the Lehmer's algorithm with the addition of an extra constant *c*. The inclusion of *c* gives the universal LC algorithm slightly better statistical properties compared to the Lehmer's algorithm. Unlike the Lemaire's and Wichman-Hill generators, the generator of a linear congruent algorithm can take an initial value of 0. Many common implementations of this algorithm perform preliminary manipulations on the input initial value in order to avoid generating well-known series of initial values [3]. Figure 3 illustrates the distribution of this algorithm.

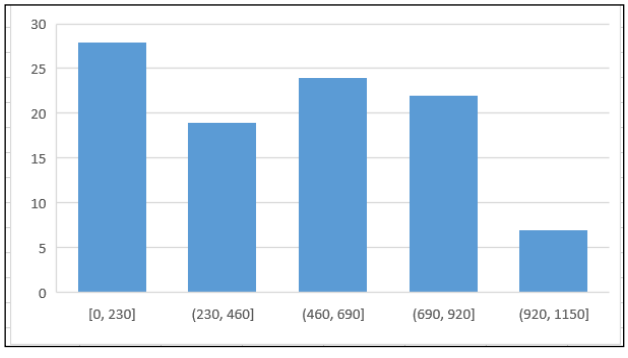

Figure 3 – Histogram of the distribution of the linear congruent algorithm

In **the Lagged Fibonacci algorithm**, a new random number is the sum of numbers *m*, where the first random number *a* was generated 7 times ago, the second random number *b* was generated 10 times ago, and the resulting sum *m* is divisible by *c*. The key task in the Lagged Fibonacci algorithm is to generate the initial values needed to start the process [3]. Figure 4 presents the distribution of this algorithm.

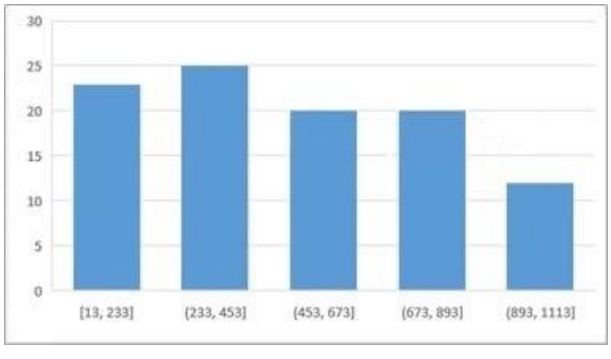

Figure 4 – Histogram of the distribution of the Lagged Fibonacci algorithm

All four pseudo-random algorithms are intended only for non-critical use cases. But even in this case, the insidiousness of pseudo-random algorithms is well known, and from time to time, even in standard pseudo-random number generators, defects are found, sometimes only after years of their use.

Based on the generated numbers and the histograms constructed for them, it can be concluded that the Lehmer's algorithm is the most effective for a small set of numbers, since it is the closest to even distribution.

References:

1. Analysis of pseudorandom number generation algorithms [Electronic resource]. — Mode of access: http://habr.com/ru/company/vk/blog/574414/. — Date of access: 24.03.2023.

2. Methods of generating random numbers [Electronic resource]. — Mode of access: http://moluch.ru/archive/142/40025/?ysclid=lfsmp1m8w635706637. — Date of access: 23.03.2023.

3. Simplified random number generation [Electronic resource]. — Mode of access: https://learn.microsoft.com/ru-ru/archive/msdn-magazine/2016/august/test-run-lightweight-random-number-generation/. — Date of acsess: 22.03.2023.