Original Research Article

International Journal of Silkroad Institute of Research and Training (IJSIRT)

Vol-1. No. 2, Jul-Dec 2023 DOI:10.3126/ijsirt.v1i2.61772 ISSN: 2990-7837 (Online & Print)

A Process based Algorithm for Random Number Generation

Deep Raj Sharma[1]

[1]Department of Engineering, BP Koirala Memorial Cancer Hospital, Bharatpur Chitwan, Nepal.

ABSTRACT

Background: Random number generation plays a crucial role in various scientific, computational, cryptographic applications and stochastic processes. This paper introduces a novel algorithm designed to address the demand for high-quality random numbers with improved statistical properties. The proposed Algorithm uses the elements of hardware-based entropy to generate random numbers for efficiency and unpredictability.

Methods: Pseudo-random numbers are sequences derived from algorithms known as pseudo-random number generators (PRNGs). These generators use an initial value, or seed, to produce sequences that exhibit statistical properties akin to truly random numbers.

Results: z-test statistics provided enough evidence that the number generated using the proposed algorithm are random.

Conclusions: Module U behaves as a Random number generation algorithm. Further, we can test the module U for large samples and we can use module U for simulation of different Random processes to verify it's correctness.

Keywords: random numbers; hardware based random numbers; pseudo random numbers.

Received: 10th July, 2023 Accepted: 4th August, 2023 Published: 31th December, 2023

INTRODUCTION

Random numbers, crucial in various mathematical and computational applications, have a rich history of exploration and application. Early pursuits to understand and utilize randomness are evident in historical games of chance, such as dice rolling.[1] However, the evolution of technology has significantly expanded the scope and importance of random numbers, driving the development of sophisticated algorithms for their generation. In contemporary computing, the term "random number" often refers to pseudo-random numbers, which are generated by algorithms designed to simulate randomness. True randomness, inherently unpredictable and lacking a discernible pattern, remains a challenging but sought-after concept in deterministic systems.[2] The significance of random numbers spans a multitude of fields, from statistics and simulation to cryptography and gaming. In statistical analyses, random numbers serve to model uncertainty and variability, essential for robust experimental design.[3] In the realm of cryptography, the generation of unpredictable sequences is fundamental to securing sensitive information and communications.[4] As we delve into the complexities of randomness, it becomes evident that the reliable generation of random numbers is not only a technological necessity but also a critical component shaping the landscape of contemporary mathematics and computer science.

METHODS

Pseudo-random numbers

Pseudo-random numbers are sequences derived from algorithms known as pseudo-random number generators (PRNGs). These generators use an initial value, or seed, to produce sequences that exhibit statistical properties akin to truly random numbers. However, unlike true randomness, pseudo-random numbers are entirely deterministic and repeatable with the same seed. Pseudo-random numbers find widespread use in diverse applications. In simulations, they allow the modeling of stochastic processes, while in statistical analyses, they facilitate the generation of random samples. Cryptographic systems leverage pseudo-random numbers for key generation and secure communication.[4] Computer graphics and gaming applications also heavily rely on pseudo-random numbers for procedural content generation and gameplay variability. The quality of a pseudo-random number generator is crucial to its effectiveness. A high-quality PRNG must pass rigorous statistical tests for randomness,

Correspondence: Mr. Deep Raj Sharma, Department of Engineering, BP Koirala Memorial Cancer Hospital, Bharatpur Chitwan, Nepal. Email:deep48440@gmail.com, Phone: +977-9841963453.

ensuring that the generated sequence exhibits properties such as uniform distribution and independence. Commonly used PRNGs include the Mersenne Twister (Matsumoto & Nishimura, 1998), linear congruential generators, and XOR-shift algorithms.

True-random numbers

Various physical processes serve as sources for true random numbers. Quantum phenomena, such as the decay of radioactive particles or the measurement of quantum states, offer inherently unpredictable events (Stipčević, 2005). Other sources include atmospheric noise, electronic noise in semiconductors, and the chaotic behavior of mechanical systems. True random numbers play a pivotal role in cryptography, particularly in scenarios where the predictability of pseudo-random numbers poses security risks. Cryptographic applications, such as key generation and nonce creation, benefit from the intrinsic unpredictability of true randomness (NIST SP 800-90B, 2018). Additionally, true random numbers find use in secure communication protocols and cryptographic protocols.

Proposed Algorithm

Proposed algorithm is based on the assumption that a random event is capable to produce further random events. Let's consider a scenario where there exist a powerful machine (commonly termed as a server) which executes different processes following some rules, most commonly a queue system. In Operating Systems, CPU Scheduling is a process that allows one process to use the CPU while another process is delayed (in standby) due to unavailability of any resources such as I / O etc., thus making full use of the CPU. Whenever the CPU becomes idle, the operating system must select one of the processes in the line ready for launch. The selection process is done by a temporary (CPU) scheduler. The Scheduler selects between memory processes ready to launch and assigns the CPU to one of them.

Here, the arrival of processes doesn't follow any rule, they are random. For instance, consider a mail server. Number of users who logins to the mail server at any instance of time do not follow any rule. Let's consider this event (a user logins to the mail server) as a process, P. At a particular time instant, let the number of processes in Queue to the mail server be n. So we have processes P_1, P_2, ……., P_n in the queue. We add another Process U in the queue. Operating System scheduler manage to provide resources to the processes such that starvation doesn't occur for any process.

RESULTS

Process U is a small module defined as follows:

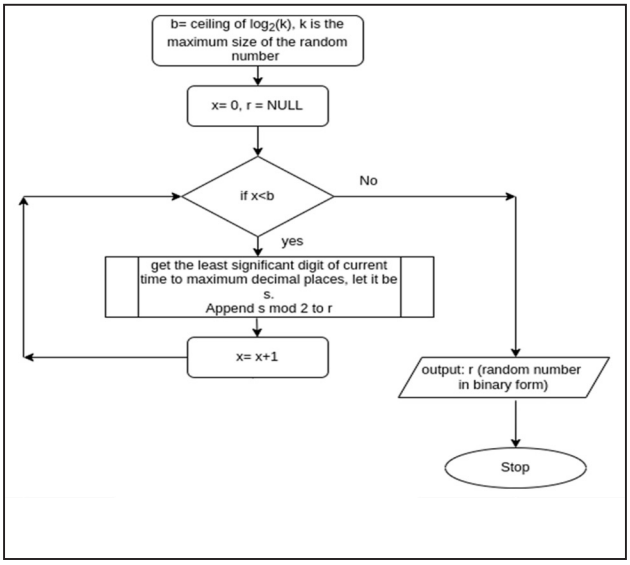

Figure 1. Proposed algorithm.

Runs test over the proposed algorithm

Runs test has been applied to the data generated using module U. The five samples with their z values and p values are presented in the table below. Each sample consists of 20 numbers generated using the module U:

For the z-test statistic and the corresponding p-value in the above five samples we found that p-value is not less than $\alpha = 0.05$, therefore we fail to reject the null hypothesis. We have sufficient evidence to say that the data was produced in a random manner. Similarly, the table below sample consists of 50 numbers generated using the module U:

For these sample too, z-test statistic and the corresponding p-value we found that p-value is not less than $\alpha = .05$, therefore we fail to reject

the null hypothesis. We have sufficient evidence to say that the data was produced in a random manner.

Table 1. Data samples of size 20.

Data Sample	z value	p-value
14, 9, 1, 12, 9, 8, 12, 11, 5, 1, 10, 13, 7, 5, 7, 11, 12, 9, 12, 13	0	1
2, 13, 9, 3, 14, 13, 13, 12, 7, 6, 9, 3, 1, 11, 6, 5, 13, 3, 12, 5	0.9158	0.35812
7, 1, 10, 4, 1, 8, 11, 4, 6, 5, 5, 9, 4, 3, 14, 2, 9, 6, 3, 11	1.15133	0.24959
12, 10, 5, 7, 9, 2, 11, 6, 3, 12, 7, 12, 14, 11, 3, 6, 10, 6, 2, 4	0.510807	0.249593
0, 3, 3, 9, 4, 12, 10, 12, 2, 4, 1, 2, 11, 4, 9, 4, 2, 8, 6, 14	-0.4179	0.67599

CONCLUSIONS

Hence, we found that for a small sample, Module U behaves as a Random number generation algorithm. Further, we can test the module U for large samples and we can use module U for simulation of different Random processes to verify it's correctness.

Table 2. Data samples of size 50.

Data Sample	z value	p-value
4, 3, 3, 12, 2, 6, 9, 6, 11, 3, 7, 9, 2, 11, 12, 3, 1, 0, 12, 8, 7, 13, 5, 9, 12, 10, 14, 10, 6, 11, 5, 1, 12, 14, 10, 14, 4, 5, 9, 6, 15, 6, 9, 12, 4, 13, 7, 11, 2, 9	1.728898	0.083827
8, 7, 1, 6, 13, 2, 8, 6, 7, 7, 8, 2, 2, 4, 6, 11, 4, 2, 13, 10, 8, 6, 3, 4, 6, 1, 6, 7, 4, 6, 11, 6, 4, 5, 4, 7, 10, 11, 10, 6, 3, 0, 4, 9, 12, 9, 9, 2, 8, 14	-1.39224	0.16384
5, 0, 1, 14, 12, 9, 4, 3, 13, 14, 8, 5, 13, 6, 11, 12, 11, 13, 6, 14, 3, 8, 10, 10, 12, 6, 14, 7, 4, 10, 12, 6, 9, 12, 6, 9, 5, 5, 7, 11, 9, 6, 3, 11, 2, 9, 4, 3, 5, 6	0.29769	0.0764
14, 4, 15, 10, 9, 5, 8, 11, 12, 3, 5, 2, 11, 6, 1, 4, 10, 6, 9, 8, 2, 9, 6, 12, 7, 9, 1, 14, 12, 1, 14, 9, 6, 1, 7, 1, 3, 12, 5, 12, 3, 10, 3, 4, 5, 6, 14, 7, 9, 3	0.510807	0.765939
7, 1, 5, 1, 12, 10, 4, 6, 13, 2, 8, 6, 11, 14, 10, 12, 6, 1, 5, 8, 6, 8, 13, 2, 1, 13, 1, 5, 9, 14, 11, 8, 6, 9, 13, 6, 14, 3, 8, 6, 10, 12, 2, 1, 5, 9, 14, 8, 4, 6	0.28577	0.77505

REFERENCES

1. Smith G. An Analog History of Procedural Content Generation. InFDG 2015 Jun 22.
2. Knuth DE. Art of computer programming, volume 2: Seminumerical algorithms. Addison-Wesley Professional; 2014 May 6.
3. Box GE, Hunter JS, Hunter WG. Statistics for experimenters. InWiley series in probability and statistics 2005. Hoboken, NJ: Wiley.
4. Menezes AJ, Van Oorschot PC, Vanstone SA. Handbook of applied cryptography. CRC press; 2018 Dec 7.
5. Atsumoto M, Nishimura T. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Transactions on Modeling and Computer Simulation (TOMACS). 1998 Jan 1;8(1):3-0
6. Tuna M, Fidan CB. A Study on the importance of chaotic oscillators based on FPGA for true random number generating (TRNG) and chaotic systems. Journal of the Faculty of Engineering and Architecture of Gazi University. 2018 Jan 1;33(2):469-86.
7. Sönmez Turan M, Barker E, Kelsey J, McKay K, Baish M, Boyle M. Recommendation for the entropy sources used for random bit generation. National Institute of Standards and Technology; 2016 Jan 27.

Citation: Sharma DR. A Process based Algorithm for Random Number Generation. IJSIRT. 2023; 1(2): 115-17.