Quantum Computing

Random Number Generation

Siddhant Rathi

TABLE OF CONTENTS

INTRODUCTION

INTRODUCTION

Random Numbers

Number sequences can be categorized based on their randomness as follows[2]:

• Eurandom - A sequence of truly random numbers, which is unpredictable, unbiased, and its numbers are uncorrelated.

• Pseudorandom – A sequence of falsely random numbers, which is predictable, but locally unbiased and has locally uncorrelated numbers.

• Quasirandom - A sequence of apparently random numbers, which is predictable and has correlations among its numbers, but is unbiased. Also known as a low-discrepancy sequence.

• Nonrandom - A sequence of non-random numbers, which is predictable, biased, and its numbers are correlated.

INTRODUCTION

Random Number Generators

A random number generator generates a sequence of numbers or symbols that cannot be reasonably predicted better than by a random chance.

The categories into which random number generators can be grouped into are[1][2]:

• Hardware Random Number Generators (HRNGs) - They generate random numbers as a function of current value of some physical environment attribute that is constantly changing in a manner that is practically impossible to model. They are also known as True Random Number Generators.

• Pseudo-Random Number Generators (PRNGs) - They follow a particular process to generate numbers that look random, but are actually deterministic, and can be reproduced if the state (or seed) of the PRNG is known.

• Quasi-Random Number Generators (QRNGs) – Similar to PRNGs, they follow a particular scheme and generate quasi-random numbers.

INTRODUCTION

HRNGs

Hardware RNGs can be based on random processes such as a coin flip, a dice roll, atmospheric noise, radioactive decay, avalanche noise in Zener diodes, hard-disk read-write timing, etc. These processes are usually rate restricted, and are therefore called ‘blocking ’, since they require time to generate enough random numbers.

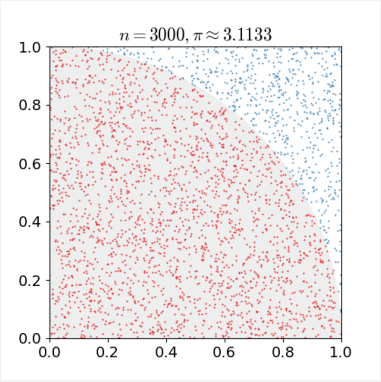
Though the numbers generated are ‘truly’ random, the tools used to measure these phenomena might introduce their own systematic biases. These biases are usually predictable and can be dealt with in the software.

To counter the effect of blocking, there have been attempts at faster HRNGs. One notable approach among them employs photonic devices to generate entropy[4].

INTRODUCTION

Pseudo-random MC Methods

Monte Carlo methods are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. The key principle is that a random sample tends to exhibit the same properties as the population from which it is drawn. Monte Carlo simulations find use in many fields.



Source: File:Pi 30K.gif - Wikipedia

Here is a simple MC simulation that estimates the value of π by estimating the ratio of the area of the quarter-circle to the area of the square.

The time taken for processing the high number of samples required for a Monte Carlo simulation can be reduced by running these calculations in parallel.

INTRODUCTION

Quasi-random MC Methods

Quasi-random Monte-Carlo methods have a faster convergence than the pseudorandom Monte-Carlo methods. The convergence rate of both the MC methods is given as[5]

•Quasi-random MC $O\left(\frac{(\log N)^s}{N}\right)$ --(1)

•Pseudorandom-MC $O\left(\frac{1}{\sqrt{N}}\right)$ --(2)

where **s** denotes the dimensionality of the problem.

Quasi-random MC methods converge asymptotically faster than the pseudo-random MC methods. However, this advantage is only apparent when **s** is small and **N** is large, and the Hardy-Kraus variation of the function is finite[5].

A quite comprehensive comparison of the performance of quasi-random MC methods and pseudorandom MC methods is provided in [3].

INTRODUCTION

Markov Chains

A Markov chain describes a sequence of possible events in which the probability of each event depends only on the state attained in the previous event[6]. They can be modeled by a directed graph, or a Finite State Machine.

A matrix representation of a Markov Chain can be given by the matrix **M**, with its **(i,j)**th element given as (**X** denotes the state, and **t** denotes the time)

$$M_t(i,j) = P(X_{t+1} = j \mid X_t = i)$$

An important property to keep in mind while dealing with these matrices is

$$(M_t \cdot M_{t+1})(i,j) = P(X_{t+2} = j \mid X_t = i)$$

CLASSICAL CASE

CLASSICAL CASE

Linear Congruential Generator

The generator is defined by the recurrence relation

$$X_{n+1} = (aX_n + b) \bmod(m) \quad \text{--(3)} \qquad \begin{matrix} 0 \leq c, X_0 < m \\ 0 < a < m \end{matrix}$$

where $\mathbf{X_0}$ is the seed.

The numbers **a**, **b**, and **m** are large integers. The maximum number of numbers that can be generated by this generator is **m-1**, after which repetition begins. The commonly used parameters in software using this generator can be found at **[7]**.

The parameters must be chosen carefully, since the quality of the randomness depends on the selected values of the parameters.

CLASSICAL CASE

Blum Blum Shub

The generator is defined by the recurrence relation

$$X_{n+1} = {X_n}^2 \bmod(m) \quad \text{--(4)}$$

where **m** is the product of two large primes, **p** and **q**. The seed must be co-prime to **m**, and not be 1 or 0. Further, there are also some conditions on the selection of **p** and **q**[8].

A number of other pseudo-random number generation algorithms exist that either combine the simpler algorithms to increase randomness, or use an entirely different approach. Some other popular pseudo-random number generators are Yarrow, Fortuna, Mersenne Twister, AEM-CTR DRBG, ChaCha20, Blum-Micali, and ANSI X9.17.

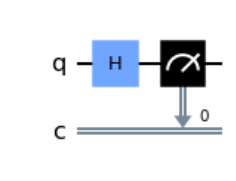
Similar to PRNGs, there exist multiple schemes for QRNGs, such as van der Corput sequence, Sobol sequence, Poisson disk sampling, etc.

QUANTUM CASE

QUANTUM CASE

Simple Quantum Computer based RNG*

One of the simplest RNG possible using a quantum computer, our RNG will use a Hadamard gate to prepare a superposition where both the outcomes are equiprobable, and then perform a measurement to generate a bit. The circuit is as shown below.



We can also extend this to multiple qubits.

Due to the noise on NISQ devices, the quality of output of this RNG is affected. The main causes of error are readout error, gate error, and decoherence[9]. To mitigate these errors, machine learning schemes which compensate for noise, as well as other circuit topologies are suggested[10].

*We are using the term Quantum Computing based RNG rather than Quantum RNG to distinguish these circuits from other experimental random number generators which usually exploit the quantum nature of photons

QUANTUM CASE

While simulating this RNG on Qiskit's Aer simulator, the results we obtained were similar to a TRNG. However, running the circuit on an actual IBM device, IBM Santiago, revealed the effect of noise on our observations. For the case of 1 qubit, for each of the 10 trials (each consisting of 8192 shots), 1 was observed more frequently than 0.

From our observations, we can say that the probability of obtaining 1 on the IBM Santiago computer is 0.5219 +/- 0.015288067242133666 with a confidence of 95%, based on our 10 runs.

For the case of four qubits, the probabilities were fairly spread out over the ten trials (meaning that there was no definite trend). However, some numbers were usually more likely to occur than others. We have provided the data for the simulations in the notebook for further analysis.

APPLICATIONS

APPLICATIONS

Random Number Generators find use in a wide range of applications. Some of these are:

- Cryptographic encryption schemes
- Monte Carlo Simulations, which are widely used in all domains of science and mathematics
- NPC decision-making and environment generation in computer games
- Generation of random patterns for art

REFERENCES

[1] Random number generation - Wikipedia

[2] Brian Hayes: Orderly Randomness - Quasirandom Numbers and Quasi-Monte Carlo | IACS Seminar - YouTube

[3] Morokoff and Caflisch - Quasirandom MC Integration (1995) (ucla.edu)

[4] 4.5 Gbps high-speed real-time physical random bit generator

[5] Quasi-Monte Carlo method – Wikipedia

[6] Markov Chains | Brilliant Math & Science Wiki

[7] Linear congruential generator – Wikipedia

[8] A Simple Unpredictable Pseudo-Random Number Generator | SIAM Journal on Computing | Vol. 15, No. 2 | Society for Industrial and Applied Mathematics

[9] Improving Reliability of Quantum True Random Number Generator using Machine Learning | IEEE Conference Publication | IEEE Xplore

[10] (PDF) Real-Time Source-Independent Quantum Random Number Generator Based on a Cloud Superconducting Quantum Computer (researchgate.net)

THANKS