Random Number Generators

Design and Applications

Student's Name
100517029
Supervisor: Prof First Name, Last Name
MSc Mathematics for Applications
Royal Holloway University of London
Day, Month, Year

Table of Contents

1. Introduction

A random number generator, or RNG, is a device intended to create a sequence of numbers that does not have a discernable or calculable pattern. Random number generators can be physical, generated by the flip of a coin or a toss of the dice. The i-Ching's yarrow sticks or a deck of cards. Many random number generators are computational. Some generators do not generate truly random numbers, but generate statistically or perceptually random numbers. These are known as pseudo-random number generators. Most computational random number generators are actually pseudo-random number generators, due to the deterministic nature of computers and the difficulty of providing a truly random number from them.

Random number generators require an input to produce a number as an output. Computerized number generators generally use one of two sources for these inputs, which can either be used directly to create the random number or can be used as a seed to algorithmically determine it (Schneier, 2000, p 99). One group uses seemingly random physical observations to create its numbers. These can include Geiger counters, microphone input, white noise receivers, visual input, air turbulence across the disk drives, gyroscope input or network packet arrival. The second group turns user movements, such as keystroke sequence or timing, mouse movements or microphone input to generate its numbers. Depending on the implementation, the random number generator may use either group's input either directly or as a seed.

Computer-based pseudorandom number generators are simpler, and often use facilities such as the computer's system clock or the date to generate a seemingly random, but reproducible, number. This is less intensive in terms of both system resources and hardware reliability than the true random number generator, but it is also less secure – the algorithm, given the same seed should produce the same output unless there is an additional random input or error included…

…2. History of Random Numbers

Gentle (2003, p 1) discussed the history of random numbers. He remarked that before the

advent of computer-based random number generators, statistics texts and other sources that required random numbers were often supplied with reference tables of "random" numbers that could be used for analysis. The largest of these, published around 1955, had over one million numbers listed (Bewersdorff, 2005, p 97). While some statisticians still continue to use computerized versions of these charts, most statistics programs now implement a direct random or pseudorandom number generation routine.

One of the earliest discussions of random number generators in the literature of computing was in 1968, at a time when random number generators were already an established procedure. D.H. Lehmer described a method to generate a new random integer (I') by multiplying the current random integer (I) by a constant multiplier (K) and keeping the remainder after overflow. This method was called the multiplicative congruential generator and was described by the following equation:

$$I' = K \times I \text{ modulo } M$$

Marsaglia noted that this naïve method of generating random numbers, although seemingly effective for many applications, was not suitable for Monte Carlo simulations because the results fell in a crystalline pattern among a small number of parallel hyper planes – in other words, the results weren't random at all, but were perfectly arrayed in a crystalline structure. "Furthermore," Marsaglia noted (1968, p25), "there are many systems of parallel hyper planes which contain all of the points; the points are about as randomly spaced in the unit *n*-cube as the atoms in a perfect crystal at absolute zero." Marsaglia demonstrated that there were a very limited number of planes containing all n-tuples, leading to a very small chance of actually obtaining a random number set from one of the above equations. Marsaglia pointed out that there are many Monte Carlo simulations which do not work with *n*-space regular "random" numbers; but more insidiously, this algorithm had been in use for twenty years without detection of this flaw, leading to the possibility that many Monte Carlo simulations had run and been accepted without understanding that the results were badly flawed due to the *n*-space regularity of the seemingly random numbers. As

Gentle (2003, p 2) noted, both the randomness of the random numbers generated by an algorithm and the distribution of the numbers generated must be understood in order to make the numbers useful.

Modern pseudorandom number generators use a number of techniques to generate numbers. These include pseudorandom functions, pseudorandom permutations and other pseudorandom objects that can be used to generate a computationally indistinguishable sequence of numbers. Some pseudorandom number generators use a complex approach of external and internal random systems, with the external "shell" systems using the output from the internal systems as input for their own calculations (Maurer, 2002, 110). These systems are among the most secure and computationally infeasible of the generation methods available…

…References

Beauchemin, Pierre, Brassard, Gilles, Crepeau, Claude, Goutier, Claude & Pomerance, Carl (1999). *The Generation of Random Numbers That are Probably Prime.*

Bellare, Mihir, Goldwasser, Shafi & Miccaiancio (1997). *"Pseudo-Random" Number Generation within Cryptographic Algorithms: the DSS Case."* Advances in Cryptography – Crypto 97 Proceedings. Lecture Notes in Computer Science, 1294.

Bellare, Mihir, Krovetz, Ted & Rogaway, Phillip (1998). *Luby-Rackoff Backwards: Increasing Security by Making Block Ciphers Non-invertible.* Advances in Cryptology-Eurocrypt 98 Proceedings, Lecture Notes in Computer Science Vol. 1403, K. Nyberg ed, Springer-Verlag, 1998.

Bewersdorff, Jorg (2005). *Luck, Logic and White Lies: The Mathematics of Games.* Wellesley, MA (USA): AK Peters, Ltd.

Chaitin, G. J.: 2001, Exploring Randomness. London: Springer-Verlag.

Dodis, Yevgeniy, Yampolskiy, Aleksandr, & Yung, Moti (2006). *"Threshold and Proactive Pseudo-Random Permutations".* Presented at TCC 2006, Columbia University, New York, NY (USA). 10 September, 2007 < http://www.informatik.uni-trier.de/~ley/db/conf/tcc/tcc2006.html >

Ferguson, Niels & Schneier, Bruce (2003). *Practical Cryptography.* Indianapolis, IN (USA): Wiley Publishing.

Fishman, George (1996). *Monte Carlo: Concepts, Algorithms and Applications.* New York, NY (USA): Springer Publishing.

Gentle, James E. (2003). *Random Number Generation and Monte Carlo Methods.* New York, NY (USA): Springer Publishing.

Giuliani, Kenneth (1998). *Randomness, Pseudorandomness and its applications to Cryptography…*