# Resumen del paper "Software for Uniform Random Number Generation: Distinguishing the Good and the Bad"

Carlos M. Martinez

Mayo de 2022

**Abstract**

Los autores repasan los requerimientos y principios de diseño del software para generación de números aleatorios (RNG). Describen algunos métodos de verificación estadística para distinguir "lo bueno de lo malo" en un RNG.

## 1 Objetivo del trabajo

Los autores buscan identificar y repasar los requerimientos y principios de diseño necesarios para la construcción de generadores de números aleatorios de buena calidad.

Establecen un marco conceptual para verificar la calidad de estos RNGs. Luego aplican estos tests a los RNGs en uso más común en el momento en que el paper fue publicado (año 2001).

Los RNGs valorados fueron los incluidos en Java 1.3, Visual Basic y Microsoft Excel.

## 2 Introducción y definiciones

Los autores establecen una definición para un RNG, "El proceso de imitar el concepto matemático de múltiples variables aleatorias distribuidas uniformemente en [0,1], mutualmente independientes"

Luego proponen una definición más formal en estos términos:

Un RNG es una estructura $(S, \mu, f, U, g)$ donde:

- S es una serie finita de estados

- $\mu$ es una distribución de probabilidad sobre S utilizada para seleccionar el estado inicial, lo que llamamos la semilla $s_0$
- $f : S- > S$ es una función de transición
- $U = [0, 1]$ es el conjunto de salida, en este caso el internvalo [0,1]
- $g : S- > U$ es la función de salida, que mapea los estados S en puntos del intervalo [0,1]

Los estados en un RNG evolucionan en forma de recurrencia, $s_i = f(s_{i-1})$.

Los puntos en $u_i$ son los números aleatorios generados por nuestro RNG. Al ser S un conjunto finito en algún momento los estados se van a repetir, es decir que para cada i habrá un j tal que $s_{i+j} = s_i$.

Llamamos *período* periodo del RNG al j más pequeño para el que esto ocurre.

# 3 Uniformidad y Tests para RNGs

Los $u_i$ *"deben parecer"* iid en [0,1]. Para todo entero $t$ las $u_i$ se distribuyen uniformemente en el hipercubo t-dimensional $[0, 1]^t$.

Esta constituye nuestra "hipótesis nula" ($H_0$), lo que será importante a la hora de definir tests para los RNGs.

Un *testeo estadístico empírico* es una estadística $T$, en forma de una función que parte de un conjunto finito de $u_i$ cuya distribución bajo $H_0$ es conocida.

Los autores proponen dos tests para RNGs en este trabajo:

## 3.1 Tests de Colisiones

En este test particionamos el hipercubo $[0, 1]^t$ mediante un cierto entero positivo $d$, lo cual crea $k = d^t$ particiones.

Definimos una variable aleatoria C como la cantidad de veces que un nuevo punto cae en una partición que ya tenía un punto asignado.

Este problema entonces parte de la estiimación de cual es la esperanza de C, $E[C]$.

### 3.2 Birthday Spacings Test

Este test se basa en el problema clásico de los cumpleaños coincidentes. La distribución es bien conocida y podemos plantear este test el estudiar los cumpleaños de $n$ personas en un planeta donde los años tienen $k$ días.

## 4 Familias populares de RNGs

Los autores discuten fundamentalmente RNGs lineales basados en recurrencias.

(2) $x_i = (a_i x_{i-1} + ... + a_k x_{i-k}) \, mod \, m$

- $(m, k)$ son enteros positivos
- Los estados son $s_i = (x_{i-k+1}, ...., x_i)$
- $a_i$ son enteros en $0, ..., m-1$

Estos RNGs se denominan RNGs, Multiple Recursive Generators. Cuando $k = 1$ tenemos los LCG, Linear Congruent Generators.

Otra familia de RNGs es la de los LSFR, Linear Feedback Shift Register.

## 5 Recomendaciones de los autores

Los autores remarcan la importancia de garantizar la independencia mutua de diferentes instancias de cada RNG.

También destacan que es necesario ser cuidadosos a la hora de utilizar RNGs disponibles en paquetes de software de uso común. Marcan que existen alternativas muy buenas que pueden ser utilizadas.