

# ¿Cómo juega a los dados el ordenador?

Vicente Trigo Aranda

La simulación es, gracias a los cada vez más potentes ordenadores, una de las herramientas básicas para la resolución de problemas. Pero, ¿en qué consiste eso de la simulación? Según Robert E. Shannon, uno de los mayores estudiosos de esta disciplina, “*la simulación es el proceso de diseñar un modelo de un sistema real y llevar a cabo experiencias con él, con la finalidad de aprender el comportamiento del sistema o de evaluar diversas estrategias para el funcionamiento del mismo*”.

Por ejemplo, supongamos que se pretende repoblar una determinada zona con varias especies de animales o estudiar si las filtraciones de una red pasarán a equis metros del subsuelo o averiguar cómo se desenvolverá un prototipo en unas condiciones extremas o hallar un método para hacer masivos análisis de sangre en el menor tiempo posible y detectar una enfermedad o establecer un nuevo plan hidrográfico nacional o investigar el comportamiento de un determinado virus modificado genéticamente o... ¿Para qué seguir? La lista sería interminable.

¿Y tienen algo en común todos los proyectos anteriores? Pues yo diría que, cuando menos, tres aspectos:

- Todos ellos pueden llevarse a cabo en la práctica, exigiendo, eso sí, unos recursos económicos que normalmente serán muy elevados. Así, una vez

construida una red de tuberías para llevar agua no cuesta mucho controlar las filtraciones; modificar un código genético ya no es asunto de ciencia-ficción; etc.

- Después de realizado el proyecto, ¿qué pasa si resulta una chapuza? Los altos costes que conlleva su puesta a punto pueden dispararse hasta niveles estratosféricos si es preciso remediar algún error catastrófico.
- Si se modelizan los proyectos en el ordenador, lo que sólo obliga a aumentar en un minúsculo porcentaje los gastos del proyecto, puede experimentarse con ellos y evaluar su comportamiento sin arriesgar otra cosa que algo de tiempo.

Y gracias a que el ordenador puede simular el azar es posible modelizar informáticamente todo tipo de experimento científico... además de disfrutar de algunos excelentes juegos.

## EL MÉTODO DE MONTECARLO

John von Neumann (28 Dic 1903 Budapest, 8 Feb 1957 Washington) es uno de los grandes matemáticos del siglo XX y uno de los pioneros en el mundo de los ordenadores.

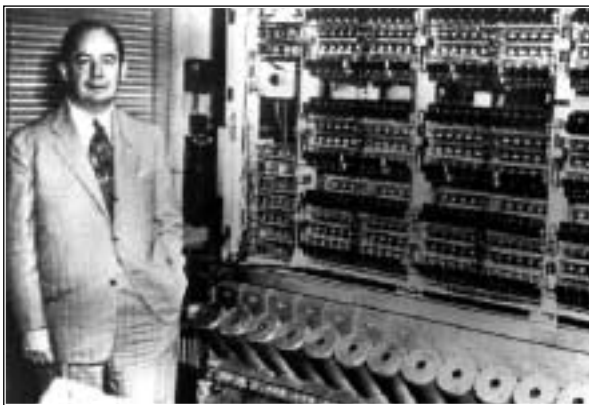


Figura 1. Von Neumann y ENIAC, el primer ordenador

A mediados de los años 30 estudió el problema de las turbulencias hidrodinámicas, cuyo estudio teórico estaba plagado de ecuaciones en derivadas parciales, nada sencillas de manejar, y eso le llevó a interesarse por las primeras máquinas electrónicas, ya que pensó que le podían ser de suma utilidad... Y una cosa llevó a la otra y ésta a su vez a otra, para acabar en EEUU donde fue compañero de Einstein, Goedel y Turing en Princeton y formó parte del laboratorio científico de Los Alamos, de 1943 a 1955.

¿Le suena verdad? En efecto, durante la Segunda (¡y ojalá que también última!) Guerra Mundial allí se llevó a cabo el proyecto Manhattan, nombre en clave con el que se denominaba a la construcción de la bomba atómica.



Figura 2. La historia de Los Alamos: <http://www.lanl.gov/worldview/welcome/history.html>

Un día John von Neumann se entretuvo hablando con unos físicos del laboratorio, lo que no era nada extraño puesto que fue un hombre sumamente socia-

ble (eran famosas las fiestas que organizaba en su casa) Los físicos estaban analizando la capacidad de diversos materiales para absorber neutrones y andaban un tanto liados con el tema.

Por lo visto, al encontrarse con una plancha de material el neutrón puede ser absorbido por ella, pero también puede traspasarla e incluso ser rechazado, todo ello de acuerdo con unas determinadas probabilidades. Los cálculos a efectuar para realizar un estudio teórico eran, como es lógico, muy complicados y von Neumann les echó una mano sugiriendo un sorprendente y novedoso enfoque del problema.

En lugar de resolver la cuestión al modo tradicional, es decir, obteniendo un número exacto para el grosor de la plancha de material, ¿por qué no simular el proceso en el ordenador y encontrar un valor aproximado? ¿Para qué hallar millones de decimales exactos si los aparatos de medida se bastan con media docena?

Aquel fue el primer caso conocido en que se programó un ordenador para resolver un problema mediante simulación y, como estaban en medio de un proyecto secreto, también se le asignó un nombre en clave a esa modalidad de trabajo: método de Montecarlo... en recuerdo del célebre casino<sup>1</sup>.

## ¿EXISTEN LOS NÚMEROS ALEATORIOS?

Si nos movemos en el terreno de las probabilidades es evidente que los números que intervengan en los procesos deben ser completamente aleatorios, ya que de otra forma los resultados no serían nada fiables... lo mismo que si son aleatorios pero presentan algún tipo de sesgo.

¿Y cómo se consiguen números aleatorios? La cuestión, que en principio parece sin importancia, es realmente complicada desde el punto de vista técnico (aunque espero que este artículo introductorio le dé una visión de por donde van los tiros) y también conlleva aspectos filosóficos: "Una sucesión de dígitos absolutamente aleatoria es un concepto lógicamente contradictorio" (Martin Gardner)

No negaré que la afirmación anterior sea válida, pero en la práctica de algún modo hay que agenciarse

<sup>1</sup> Encontrará muchísimo material sobre la generación de números aleatorios, método de Montecarlo incluido, en la Web de Computational Science Education Project: <http://csepl.phy.ornl.gov/rn/rm.html>



números aleatorios y D. H. Lehmer propuso que se considerase aleatoria cualquier sucesión de números que cumpliera las dos condiciones siguientes:

- Cualquier persona que desconozca cómo se ha obtenido la sucesión debe ser incapaz de predecir el siguiente término.
- La sucesión de números debe pasar con nota todos los tests de aleatoriedad a que sea sometida: rachas, gaps, serial, etc.



Figura 3. Página sobre tests de aleatoriedad: <http://random.mat.sbg.ac.at/tests/>

Hay una célebre frase de R. Coveyou, atribuida también a Donald Knuth, que aparece muchas veces por Internet en páginas dedicadas a chistes científicos... aunque es real como la vida misma: “La generación de números aleatorios es una cuestión demasiado importante para dejarla en manos del azar”.

## ¿QUÉ TIPO DE NÚMEROS ALEATORIOS SE BUSCAN?

Bueno, pues en la práctica ya sabemos que unos números se considerarán aleatorios cuando sean más o menos impredecibles y no sean declarados no aleatorios por algún test. Como ve, la definición no es muy allá pero es útil, que es lo que importa.

Ahora bien, ¿qué clase de números aleatorios se necesitan? Al menos en este aspecto la cosa está clara, gracias a las Matemáticas. Basta con obtener números que estén comprendidos en el intervalo  $[0,1)$  ya que, si están distribuidos uniformemente, a partir de ellos pueden generarse valores de variables normales, binomia-

<sup>2</sup> Observe que para obtener un número aleatorio del intervalo  $[0,1)$  con, por ejemplo, cinco decimales, basta con encontrar cinco dígitos aleatorios y considerar que son la parte decimal del número buscado.

les, de Poisson, exponenciales, etc.

¿Y cómo se consiguen esos números? Pues básicamente hay tres formas (sin contar las páginas que hay en Internet que ofrecen números aleatorios):

- Obtenerlos de tablas de números aleatorios, que son utilizadas en simulaciones manuales y que ya están muy en desuso, salvo en trabajos escolares.
- Generar números pseudoaleatorios mediante una serie de métodos aritméticos, lo que resulta muy conveniente a la hora de hacer simulaciones por ordenador.
- Utilizar medios electrónicos para producir números aleatorios puros, necesarios para criptografía.



Figura 4. También puede generar números aleatorios on-line. Por ejemplo, aquí: <http://www.randomizer.org/form.htm>

## LAS TABLAS DE NÚMEROS ALEATORIOS

A principios de siglo XX se comenzó a vislumbrar la utilidad de los números aleatorios y rápidamente comenzaron los trabajos para generarlos. Como entonces todavía no había ordenadores, se utilizaron métodos mecánicos para obtenerlos (ruletas, dados, etc.) pero pronto se descubrió su inutilidad, ya que presentaban sesgos de todo tipo.

El primer trabajo serio lo publicó Leonard H.C. Tippett en 1927; su tabla contenía 41.600 dígitos... que había conseguido tomando las cifras centrales de la superficie de las parroquias inglesas. Años después, en 1939, Maurice George Kendall y Bernard Babington Smith publicaron *Tables of Random Sampling Numbers*, que ofrecía cien mil dígitos<sup>2</sup>.



Figura 5. Maurice George Kendall (1907-1983)

Pero, aunque parezca sorprendente, cien mil dígitos resultan insuficientes para muchas simulaciones y por este motivo prosiguió la búsqueda de más y más números aleatorios, que finalizó cuando Rand Corporation publicó su *A Million Random Digits with 100,000 Normal Deviates*, en 1955. Los datos se obtuvieron mediante dispositivos entre mayo y junio de 1947 pero después se sometieron a un proceso de depurado, para eliminar al máximo posibles sesgos<sup>3</sup>.

¡Cómo! ¡Que le apetecería echarle un vistazo a esa publicación! Cosas más raras se han leído, si lo sabré yo. Pues nada, sólo tiene que acudir a la siguiente dirección... y esperar un poquito a que se cargue la extensa página:

[http://ftp.rand.org/software\\_and\\_data/random/digits.txt](http://ftp.rand.org/software_and_data/random/digits.txt)

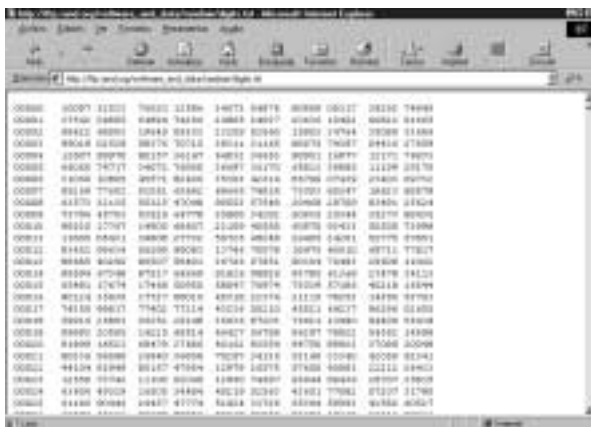


Figura 6. Una lectura muy poco estimulante, la verdad.

## MÉTODOS ARITMÉTICOS

La llegada de los ordenadores significó el fin de las tablas de números aleatorios, ya que eran muy costosas de generar y, además, implementarlas en los sistemas informáticos de aquellos años era una tarea prácticamente imposible. Encima, un millón de dígitos, como ofrecía el libro de RAND Corporation, resulta una cantidad todavía pequeña para muchas simulaciones.

En resumen, se buscó algún método para que el ordenador generara rápidamente millones y millones de números, siguiendo sencillos algoritmos que no consumiesen grandes recursos del sistema. Claro está que estos números no serían realmente aleatorios, y por eso se les suele llamar **pseudoaleatorios**, pero si satisfacen las dos condiciones fijadas por Lehmer resultan útiles para las simulaciones, que al fin y al cabo es lo que interesa.

Pronto se vio que los algoritmos más eficaces para implementar en el ordenador son los aritméticos. Partiendo de un término inicial, que se acostumbra denominar **semilla**, se obtiene otro, que a su vez genera otro y así sucesivamente.

$$x_{n+1} = f(x_n)$$

Lógicamente un mismo método con una misma semilla siempre generará la misma sucesión de números aleatorios... lo que es una ventaja nada desdeñable cuando se trabaja con el azar en informática y se está en la fase de puesta a punto de un programa. Supongamos que durante las pruebas aparece un error, que presuntamente se corrige tras analizar el código del programa. ¿Cómo tener la seguridad de que se ha corregido realmente?

Si el proceso fuese aleatorio puro podría suceder que el error siguiese estando ahí y no volviese a aparecer otra vez durante la depuración... aunque sí cuando pudiese causar más daño (las leyes de Murphy son inexorables) Colocando la misma semilla se repite exactamente la misma secuencia y así se comprueba si el error ha desaparecido o no.

Pero, además, la semilla también facilita que la pseudoaleatoriedad se aproxime más a la verdadera aleatoriedad. Si introducimos como semilla un valor que, por ejemplo, depende del tiempo que el ordenador lleva encendido, centésimas de segundo incluidas,

<sup>3</sup> Si le interesa saber más sobre cómo se obtuvo ese millón de dígitos aleatorios puede visitar la página que dedica a ese tema RAND Corporation: <http://www.rand.org/publications/classics/randomdigits/>



en la práctica es casi imposible averiguar su valor y, por tanto, el de los números que genera. Algo así es lo que hace el procedimiento Randomize que está implementado en múltiples lenguajes de programación.

Un último detalle antes de pasar a comentarle el método más difundido para generar números aleatorios. Los métodos aritméticos obtienen números naturales comprendidos entre 0 y  $m-1$ , siendo  $m$  un número fijado de antemano en función de las características del procesador. Basta dividir el número pseudoaleatorio entre  $m$  para disponer de un número del intervalo  $[0,1)$  que, recuerde, es lo que realmente interesa.

## MÉTODO CONGRUENCIAL LINEAL

Este método, propuesto por Lehmer en 1948, permite obtener una sucesión de números pseudoaleatorios mediante una sencilla fórmula aritmética:

$$x_{n+1} = (ax_n + c) \text{ mod } m$$

donde  $a$  (multiplicador),  $c$  (incremento) y  $x_0$  (semilla) son números naturales inferiores a  $m$ .

Si no domina las Matemáticas no se asuste por la expresión anterior, que sólo formalmente parece complicada<sup>4</sup>. Únicamente hay que multiplicar y dividir... y para el ordenador todavía es más sencillo, pues sólo debe quedarse con los bits representativos resultantes de cada operación.

Veamos, para acabar de coger la idea, un ejemplo elemental que puede seguir haciendo las operaciones a mano... aunque, ¿para qué está el ordenador? ¡Que trabaje él! Si no le apetece ponerse a programar, siempre puede utilizar una hoja de cálculo.

Así, supongamos que introduce los cuatro parámetros que determinarán la sucesión de números pseudoaleatorios en la segunda columna de la hoja.

	A	B
1	$m$	16
2	$a$	5
3	$c$	1
4	semilla	6

Figura 7. Introduzca en la segunda columna los parámetros que desee.

En la columna donde desea que aparezcan los números pseudoaleatorios (en mi caso la D) escriba las siguientes expresiones:

Celda	Fórmula
D1	= $\$B\$4$
D2	=RESIDUO(D1* $\$B\$2$ + $\$B\$3$ ; $\$B\$1$ )

Luego sólo tiene que arrastrar la celda D2 hasta donde desee y ya tendrá la serie de números en pantalla.

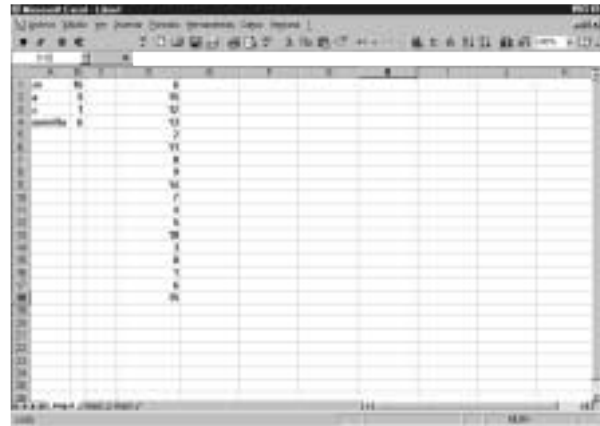


Figura 8. Una serie de números pseudoaleatorios cualesquiera.

Si dedica unos pocos minutos a probar con diversos parámetros observará que la calidad de la serie deja mucho que desear en algunos casos y que siempre, antes o después, uno de los números se repetirá y la misma secuencia de números volverá a salir una y otra vez. Esto es lógico, ya que al dividir cualquier número por  $m$  sólo hay  $m$  posibles restos (desde 0 hasta  $m-1$ ), por lo que como máximo se generarán  $m$  valores distintos.

Por tanto, lo deseable sería que en cualquier sucesión su **periodo** (número de valores distintos) sea el máximo posible. Por ejemplo, al trabajar con ordenadores de 32 bits el valor habitual de  $m$  es  $2^{32}$  y, por tanto, podrían generarse hasta 4.294.967.296 números pseudoaleatorios diferentes... ¡Ve como el millón de la tabla de RAND Corporation era una minucia!

Afortunadamente para quienes se dedican a buscar parámetros adecuados para obtener buenos resultados pseudoaleatorios, existe un resultado teórico que facilita su búsqueda. Así, la sucesión generada tendrá

<sup>4</sup> Recuerde que dados dos números,  $p$  y  $q$ ,  $p \text{ mod } q$  es el resto que resulta al dividir  $p$  entre  $q$ .



## ¿Cómo juega a los dados el ordenador?

el periodo máximo,  $m$ , si y sólo si se cumplen las siguientes tres condiciones:

- i.  $c$  y  $m$  son primos entre sí
- ii.  $a-1$  es múltiplo de todos los primos que dividen a  $m$
- iii. si  $m$  es múltiplo de 4,  $a-1$  también lo es

Tan sencillo de codificar resulta este método congruencial lineal, que además genera sucesiones de bastante calidad, que muchos programas simulan con él el azar. Por ejemplo, la rutina MTH\$RANDOM de la librería VAX/VMS lo hace con los siguientes parámetros:

$$m = 2^{32} \quad c = 1 \quad a = 69069$$

El generador de números aleatorios de Turbo Pascal también lo sigue. En este caso los parámetros son:

$$m = 2^{32} \quad c = 1 \quad a = 134775813$$

Eso sí, encontrar los valores de  $a$  y  $c$  que generan series de números pseudoaleatorios con calidad estadística es todo un arte, por llamarlo de alguna manera. Los resultados teóricos, aún siendo muchos, no ofrecen algoritmos claros para hallar  $a$  y  $c$  sino sólo orientaciones, por lo que la única forma de buscarlos es ir probando y tener algo de suerte. Curioso, ¿verdad? El azar también interviene en la generación de números pseudoaleatorios.

## MÉTODO MULTIPLICATIVO PURO

Un caso particular del método congruencial lineal es aquel en que  $c$  es nulo. En este caso, que se conoce por el nombre de método multiplicativo puro, la expresión generadora adopta la siguiente forma:

$$x_{n+1} = a^n x_0 \text{ mod } m$$

La ventaja de la expresión anterior es que resulta más rápida de computar, aunque por el contrario el periodo de la sucesión nunca podrá ser máximo (recuerde lo que decía el teorema antecedente) De todas formas, siempre puede conseguirse un periodo de longitud  $m-1$ , que en la práctica viene a ser lo mismo que un periodo  $m$ , sin más que elegir los parámetros adecuados:

Todo divisor primo  $q$  de  $m-1$  debe verificar que el resto de dividir  $a^{(m-1)/q}$  entre  $m$  es distinto de 1.

Por ejemplo este método multiplicativo puro está implementado en la librería de rutinas estadísticas IMSL, con los siguientes parámetros:

$$m = 2^{31} - 1 = 2147483647 \quad a = 950706376$$

¿Verdad que recuerda el simpático Spectrum? Su generador de números aleatorios también era multiplicativo puro y sus parámetros eran:

$$m = 2^{16} + 1 = 65537 \quad a = 75$$



Figura 9. ¡Cuanta gente comenzamos con él!

## NÚMEROS VERDADERAMENTE ALEATORIOS

Uno de los algoritmos más utilizados para encriptar mensajes es el RSA... cuyas siglas son las iniciales de sus creadores: Rivest, Shamir y Adleman.



Figura 10. Página de RSA Security:  
<http://www.rsasecurity.com/>

El algoritmo RSA está basado en la descomposición factorial de enormes números. ¿Le suena de algo eso de las claves públicas y privadas? Por si acaso quiere documentarse más sobre este tema, le remito a los excelentes artículos *Una aproximación a la seguridad en las comunicaciones* y *Protección de la información a*



través del cifrado de José A. Labodía Bonastre, que encontrará en el Manual Formativo nº 10 y 16, respectivamente.

Para hacer todavía más seguro el sistema, en lugar de encriptar mensajes mediante claves públicas pueden utilizarse algoritmos de clave simétrica, donde ésta es un número completamente aleatorio. Por ejemplo, en esto se basa el programa PGP, uno de los más conocidos.



Figura 11. Web de PGP: <http://www.pgpi.org/>

En otras palabras, si para la simulación de experimentos científicos basta con números pseudoaleatorios, para temas criptográficos es imprescindible que los números sean completamente impredecibles; es decir, que sean aleatorios puros.

¿Y cómo se consigue eso? Generalmente mediante hardware; así, puede leerse la señal generada por un diodo, conectar un tubo Geiger-Müller para medir algún tipo de radioactividad, determinar las turbulencias de aire en la disquetera, etc.



Figura 12. ¿Le interesa comprar un contador Geiger para su PC?

Por ejemplo, en la presentación del nuevo chipset Intel® 820 se indica: “El generador de números aleatorios de Intel proporciona hardware que genera números verdaderamente aleatorios a través del uso de ruido térmico para permitir cifrado más sólido, firma digital y protocolos de seguridad”

Seguidamente le indico unas cuantas direcciones de Internet donde podrá encontrar más información sobre este tema:

- <http://users.javanet.com/~othello/random.htm>
- <http://www.random.org/essay.html>
- [http://webnz.com/robert/true\\_rng.html](http://webnz.com/robert/true_rng.html)
- [http://webnz.com/robert/rng\\_links.htm](http://webnz.com/robert/rng_links.htm)
- <http://www.fourmilab.ch/hotbits/>



Figura 13. ¿Cómo generar números aleatorios puros?