Ingeniería en Sistemas y Ciencias de la Computación

Análisis de Datos

Tarea #2

Fundamentos Matemáticos

1. LCG

Un generador de congruencia lineal es un algoritmo que produce una secuencia de números pseudo randomizados calculados mediante una ecuación lineal discontinua. El generador se define por la relación de recurrencia:

$Xn_{+1} = (aXn + c) \bmod m$

donde:

- X es la secuencia de valores pseudo aleatorios.
- m, con $0 < m$ — el módulo.
- a, con $0 < a < m$ — el multiplicador.
- c, con $0 \leq c < m$ — el incremento.
- X_0, con $0 \leq X_0 < m$ — la semilla o valor inicial.

Estos valores son constantes enteras que especifican el generador. Si $c = 0$, el generador suele llamarse generador congruencial multiplicativo (MCG), o RNG de Lehmer. Si $c \neq 0$, el método se denomina generador congruencial mixto.

2. Blum Blum Shub

Blum Blum Shub es un generador de números pseudo aleatorios propuesto en 1986 por Lenore Blum, Manuel Blum y Michael Shub.

El algoritmo tiene la forma:

$\mathbf{xn_{+1} = (xn)^2 \bmod M}$

donde $\mathbf{M = \rho \cdot q}$ es el producto de dos primos grandes **ρ** y **q**. En cada paso del algoritmo, la salida se obtiene a partir de $\mathbf{xn_{+1}}$, comúnmente se utiliza la paridad del bit de $\mathbf{xn_{+1}}$ o uno o más de los bits menos significativos de $\mathbf{xn_{+1}}$.

La **semilla** $\mathbf{x_0}$ debe ser un número entero que sea coprimo con **M** (es decir, que no tenga como factores a **ρ** ni a **q**) y que no sea ni 1 ni 0.

Los dos primos **ρ** y **q** deben cumplir que:

- Sean congruentes con 3 (módulo 4), esto garantiza que cada residuo cuadrático tenga una raíz cuadrada que también es un residuo cuadrático.

- Sean primos seguros, con un máximo común divisor pequeño en **gcd((p−3)/2, (q−3)/2)**, esto asegura que la longitud del ciclo sea grande.

3. RANDU

RANDU es un generador pseudo aleatorio de números congruenciales lineales.

Está definido por la recurrencia:

$$Vj_{+1} = 65539 \cdot Vj \bmod 2^{31}$$

con la semilla inicial V_0 como un número impar. Genera enteros pseudo aleatorios Vj, que están distribuidos uniformemente en el intervalo $[1, 2^{31} - 1]$. En aplicaciones prácticas, estos valores suelen transformarse en números racionales pseudo aleatorios **Xj** en el intervalo (0, 1) mediante la fórmula:

$$Xj = Vj / 2^{31}$$

La razón por la que se eligieron esos valores específicos para el multiplicador y el módulo fue que, con un tamaño de palabra de 32 bits, la aritmética con módulo 2^{31} y con el número $65539 = 2^{16} + 3$ podía realizarse rápidamente mediante operadores binarios. Sin embargo, dichos valores fueron seleccionados por conveniencia computacional y no por calidad estadística.

4. Middle Square

Para generar una secuencia de números pseudoaleatorios de n dígitos, se crea un valor inicial de n dígitos y se eleva al cuadrado, lo que produce un número de 2n dígitos. Si el resultado tiene menos de 2n dígitos, se añaden ceros a la izquierda para compensar. Los n dígitos centrales del resultado serían el siguiente número de la secuencia y se devolverían como resultado. Este proceso se repite para generar más números.

El valor de n debe ser par para que el método funcione, si el valor de n es impar, no necesariamente habrá un número central de n dígitos definido de forma única para seleccionar. Si se eleva al cuadrado un número de 3 dígitos, puede obtenerse un número de 6 dígitos (ej., 5402 = 291600). Si hubiera 3 dígitos centrales, quedarían 6 − 3 = 3 dígitos para distribuir a la izquierda y a la derecha del número central. Es imposible distribuir estos dígitos equitativamente a

ambos lados del número central, por lo que no existen "dígitos centrales". Es aceptable rellenar las semillas con ceros a la izquierda para crear un número par de n dígitos (ej., 540 → 0540).

Para un generador de números de n dígitos, el periodo no puede ser mayor que 8n. Si los n dígitos centrales son todos ceros, el generador genera ceros indefinidamente. Si la primera mitad de un número en la secuencia son ceros, los números subsiguientes serán decrecientes hasta cero. Aunque estas secuencias de ceros son fáciles de detectar, ocurren con demasiada frecuencia para que este método sea de utilidad práctica. Este método también puede bloquearse en un número distinto de cero. Para n = 4, esto ocurre con los valores 0100, 2500, 3792 y 7600. Otros valores de semilla forman ciclos repetitivos muy cortos, por ejemplo, 0540 → 2916 → 5030 → 3009.

5. Mersenne Twister

El Mersenne Twister es un generador de números pseudoaleatorios desarrollado en 1997 por Makoto Matsumoto y Takuji Nishimura. Su nombre deriva de la elección de un primo de Mersenne como la duración de su período. El periodo es específicamente 2^19937−1.

Sus fundamentos principales son:

1. **Campo finito:**

- Todas las operaciones se hacen como si los bits fueran módulo 2, es decir, operaciones lógicas como XOR equivalen a sumas mod 2.

2. **Recurrencia lineal de gran dimensión:**

- El núcleo del algoritmo es una recurrencia lineal de orden 624.

- Cada nuevo número depende linealmente de los anteriores, pero con un diseño que evita ciclos cortos y patrones obvios.

3. **Tempering:**

- Después de generar un número, se le aplican operaciones adicionales de XOR y shifts para mejorar la uniformidad estadística.

- Esto reduce correlaciones y hace que los números parezcan más aleatorios.

Tabla de Comparación entre Algoritmos

Algoritmo	Velocidad	Periodo	Seguridad	Casos de Uso
LCG	Es un algoritmo muy rápido ya que cada valor se genera con una multiplicación, suma y módulo.	El máximo posible periodo depende del valor m seleccionado.	Tiene una baja seguridad ya que conociendo pocos valores consecutivos se pueden reconstruir los parámetros de entrada.	Es un algoritmo que se usa en simulaciones sencillas, videojuegos y aplicaciones donde importa la rapidez pero no tanto la aleatoriedad total.
Middle Square	Es un algoritmo rápido ya que se basa en una sola multiplicación y en la extracción de dígitos.	Tiene un periodo muy malo que tiende a colapsar muy rápido.	Los periodos son muy fáciles de detectar y muy predecibles, por lo que no es un algoritmo seguro.	El algoritmo middle square está sobre todo obsoleto, tuvo un uso histórico a partir de su presentación por Von Neuman como un ejemplo práctico de generación de números pseudoaleatorios.
Blum Blum Shub	En general es un algoritmo más lento ya que requiere cálculos grandes utilizando módulo.	El periodo depende de la multiplicación de los 2 primos utilizados, puede llegar a ser un periodo bastante grande.	Su seguridad se basa en la dificultad de factorizar enteros bastante grandes.	Debido a su mayor seguridad el algoritmo se puede utilizar en contextos criptográficos tal como en la generación de llaves, y sistemas de imprevisibilidad.
RANDU	Al igual que LCG es muy rápido, debido a las	El periodo de RANDU máximo es de 2^{31}.	Mala seguridad debido a la poca cantidad de planos en la	Se utilizaba al principio de su creación en los años 60-70s en

	operaciones simple de multiplicación, suma y módulo.		que el algoritmo genera números.	dispositivos IBM. Sin embargo debido a lo poco representativo que representa el algoritmo se dejó de utilizar.
Mersenne Twister	Es rápido debido a la utilización de operaciones binarias, sin embargo la adición de otros pasos lo hace más lento que LCG.	El periodo máximo es muy grande, ~2^{19937} - 1. Muy buena distribución en 623 dimensiones.	No es criptográficamente seguro, hay operaciones reversibles y los valores pueden ser reconstruidos a partir de ciertos valores conocidos.	Mersenne twister es de los algoritmos más utilizados, teniendo aplicaciones en simulaciones de Monte Carlo, modelado estocástico e incluso machine learning.

Comparación de Python Random con LCG

El algoritmo LCG es un generador de números pseudoaleatorios bastante simple. Su funcionamiento se basa en una fórmula lineal que combina multiplicación, suma y módulo lo que lo hace rápido y fácil de implementar, pero limita su calidad, el período es relativamente corto y los números generados pueden llegar a mostrar patrones o correlaciones, lo que podría ser problema en aplicaciones que requieren aleatoriedad estadísticamente sólida.

El generador random de Python utiliza por detrás el algoritmo de Mersenne Twister. El algoritmo mantiene un estado interno mucho más grande y aplica operaciones binarias para producir números con buena uniformidad estadística y un período bastante largo, lo que significa que los números generados no muestran patrones evidentes y son adecuados para simulaciones, análisis numérico y otras aplicaciones donde la calidad de la aleatoriedad es importante. Sin embargo al igual que LCG el algoritmo de Mersenne Twister no está diseñado para aplicaciones criptográficas.

Referencias

Middle-square method. https://en.wikipedia.org/wiki/Middle-square_method.

RANDU. https://en.wikipedia.org/wiki/RANDU.

Linear congruential generator. https://en.wikipedia.org/wiki/Linear_congruential_generator.

Blum Blum Shub. https://en.wikipedia.org/wiki/Blum_Blum_Shub.

Mersenne Twister. https://en.wikipedia.org/wiki/Mersenne_Twister.

Monte Carlo Method. https://en.wikipedia.org/wiki/Monte_Carlo_method.