Contents

Clase 2: Generadores Congruenciales Lineales (GCL) Avanzados y Pruebas de Uniformidad

1. Objetivos Específicos de la Clase:

- Comprender en profundidad la influencia de los parámetros del GCL (a, c, m) en la calidad de la secuencia generada.
- Analizar las propiedades de los GCL de período completo.
- Aplicar la prueba de Chi-cuadrado para evaluar la uniformidad de un GCL.
- Comprender y aplicar otras pruebas de uniformidad como la prueba de Kolmogorov-Smirnov.
- Implementar y comparar diferentes GCL, evaluando su rendimiento y aleatoriedad.

2. Contenido Teórico Detallado:

2.1. Profundizando en el Generador Congruencial Lineal (GCL):

Recordemos que un GCL genera números pseudoaleatorios mediante la fórmula:

$X_{i+1} = (aX_i + c) \bmod m$

Donde:

- X_{i+1} es el siguiente número en la secuencia.
- X_i es el número actual en la secuencia (la semilla inicial es X_0).
- a es el multiplicador.
- c es el incremento.
- m es el módulo.

2.1.1. La Importancia de los Parámetros (a, c, m): La elección adecuada de a, c y m es crucial para obtener una secuencia de números pseudoaleatorios con buenas propiedades. Una mala elección puede resultar en períodos cortos, correlaciones y falta de uniformidad.

- **m (Módulo):** Generalmente, se elige como un número grande. Usualmente es una potencia de 2 ($m = 2^b$) para computadoras binarias, o un número primo grande. Un m grande permite un período máximo mayor.
- **a (Multiplicador):** Afecta la longitud del ciclo y las correlaciones entre los números generados. Debe ser cuidadosamente seleccionado para evitar patrones no aleatorios.
- **c (Incremento):** Un GCL con c 0 se llama *generador congruencial mixto*. Si c = 0, se conoce como *generador congruencial multiplicativo*.

2.1.2. Condiciones para un Período Completo: Un GCL tiene período completo si genera todos los números posibles entre 0 y m-1 antes de repetirse. Para que un GCL tenga un período completo (m), deben cumplirse las siguientes condiciones (Teorema de Hull-Dobell):

1. `c` y `m` deben ser coprimos (su máximo común divisor debe ser 1).
2. `a - 1` debe ser divisible por todos los factores primos de `m`.
3. Si `m` es divisible por 4, entonces `a - 1` también debe ser divisible por 4.

2.2. Pruebas de Uniformidad:

La uniformidad es una propiedad esencial de un buen generador de números aleatorios. Significa que los números generados deben estar distribuidos uniformemente en el intervalo [0, 1). Las pruebas de uniformidad evalúan si la distribución observada de los números generados se ajusta a una distribución uniforme esperada.

2.2.1. Prueba de Chi-Cuadrado (²):

- **Procedimiento:**
  1. Dividir el intervalo [0, 1) en *k* subintervalos de igual longitud.
  2. Generar *n* números aleatorios.

3. Contar el número de observaciones (O_i) que caen en cada subintervalo *i*.
4. Calcular la frecuencia esperada (E_i) para cada subintervalo: $E_i = n/k$.
5. Calcular el estadístico de prueba Chi-cuadrado:

   $^2 = \Sigma\ [(O_i - E_i)^2 / E_i]$ (sumatoria desde i=1 hasta k) 6. Comparar el valor calculado de ^{2} con el valor crítico de la distribución Chi-cuadrado con (k-1) grados de libertad y un nivel de significancia . 7. Si $^2 >\ ^2_{crítico}$, se rechaza la hipótesis nula de uniformidad.

- **Hipótesis:**
  - H_0 (Hipótesis nula): Los números están distribuidos uniformemente.
  - H_1 (Hipótesis alternativa): Los números no están distribuidos uniformemente.

2.2.2. Prueba de Kolmogorov-Smirnov (K-S):

- **Procedimiento:**

1. Ordenar los *n* números aleatorios generados de menor a mayor: $U_{(1)}, U_{(2)}, ..., U_{(n)}$.
2. Calcular la función de distribución empírica: $F_n(x)$ = (número de $U_{(i)}$ x) / n.
3. Calcular la máxima diferencia absoluta entre la función de distribución empírica y la función de distribución uniforme teórica $F(x) = x$:

   $D = \max |F_n(x) - F(x)|$ 4. Calcular los estadísticos $D^+ = \max_i \{i/n - U_{(i)}\}$ y $D^- = \max_i \{U_{(i)} - (i-1)/n\}$ 5. El estadístico K-S es $D = \max(D^+, D^-)$. 6. Comparar el valor calculado de D con el valor crítico de la tabla de Kolmogorov-Smirnov para un tamaño de muestra *n* y un nivel de significancia . 7. Si $D > D_{crítico}$, se rechaza la hipótesis nula de uniformidad.

- **Hipótesis:**
  - H_0 (Hipótesis nula): Los números están distribuidos uniformemente.
  - H_1 (Hipótesis alternativa): Los números no están distribuidos uniformemente.

3. Ejemplos o Casos de Estudio:

Ejemplo 1: Análisis de un GCL con parámetros inapropiados:

Consideremos un GCL con a = 2, c = 3, m = 16, y semilla $X_0 = 1$.

La secuencia generada es: 1, 5, 13, 1, 5, 13, ... (período = 3).

Claramente, este GCL tiene un período muy corto y no es adecuado. Esto se debe a que no cumple las condiciones del teorema de Hull-Dobell. Por ejemplo, m es divisible por 4, pero a-1 no es divisible por 4.

Ejemplo 2: Implementación y prueba de un GCL de Período Completo:

Consideremos un GCL con a = 65539, c = 0, m = 2^{31} (un generador multiplicativo común). $X_0 = 12345$.

Este GCL cumple (aproximadamente) las condiciones para un período completo. Implementarlo en Python y aplicar la prueba de Chi-cuadrado para k = 10 subintervalos con n = 10000 números generados. Se calcula el valor de ^{2} y se compara con el valor crítico de la tabla Chi-cuadrado con 9 grados de libertad y = 0.05.

4. Problemas Prácticos o Ejercicios con Soluciones:

Problema 1:

Implementar un GCL con a = 1664525, c = 1013904223, m = 2^{32} y semilla $X_0 = 0$.

a) Generar 1000 números aleatorios y graficar un histograma de los valores generados. b) Aplicar la prueba de Chi-cuadrado con k = 20 subintervalos y = 0.05. ¿Se rechaza la hipótesis de uniformidad? c) Aplicar la prueba de Kolmogorov-Smirnov con =0.05. ¿Se rechaza la hipótesis de uniformidad?

Solución:

a) Implementación en Python:

“‘python def gcl(a, c, m, x0, n): resultados = [] x = x0 for _ in range(n): x = (a * x + c) % m resultados.append(x / m) # Normalizar a [0, 1) return resultados

import matplotlib.pyplot as plt import numpy as np from scipy.stats import chi2, kstest

a = 1664525 c = 1013904223 m = 2**32 x0 = 0 n = 1000 numeros_aleatorios = gcl(a, c, m, x0, n)

plt.hist(numeros_aleatorios, bins=20) plt.title(”Histograma de Números Aleatorios (GCL)”) plt.xlabel(”Valor”) plt.ylabel(”Frecuencia”) plt.show() “‘

b) Aplicación de la prueba de Chi-cuadrado:

```
python k = 20 frecuencias_esperadas = n / k frecuencias_observadas, _ = np.histogram(numeros_aleatorios
bins=k, range=(0, 1)) chi_cuadrado = np.sum((frecuencias_observadas - frecuencias_esperadas)**2
/ frecuencias_esperadas) grados_libertad = k - 1 valor_critico = chi2.ppf(1 - 0.05,
grados_libertad) print(f"Estadístico Chi-cuadrado: {chi_cuadrado}") print(f"Valor crítico
Chi-cuadrado: {valor_critico}") if chi_cuadrado > valor_critico:     print("Se rechaza
la hipótesis nula de uniformidad.") else:     print("No se rechaza la hipótesis nula de
uniformidad.")
```

c) Aplicación de la prueba de Kolmogorov-Smirnov:

“‘python estadistico_ks, p_valor = kstest(numeros_aleatorios, ’uniform’)

print(f”Estadístico K-S: {estadistico_ks}”) print(f”P-valor: {p_valor}”)

alfa = 0.05 if p_valor < alfa: print(”Se rechaza la hipótesis nula de uniformidad.”) else: print(”No se rechaza la hipótesis nula de uniformidad.”) “‘

Problema 2:

Investigar diferentes GCL y comparar sus parámetros (a, c, m) y períodos. Implementar al menos dos GCL diferentes y aplicarles la prueba de Chi-cuadrado y Kolmogorov-Smirnov. ¿Cuál GCL parece ser mejor en términos de uniformidad?

Solución: (Requiere investigación y experimentación con diferentes GCL y sus parámetros). Se espera que el estudiante investigue y compare diferentes GCL comunes, como el MINSTD de Lehmer, y evalúe su rendimiento usando las pruebas mencionadas.

5. Materiales Complementarios Recomendados:

- **Libros:**
  - ”Simulation Modeling and Analysis” by Averill M. Law
  - ”Discrete-Event System Simulation” by Jerry Banks, John S. Carson II, Barry L. Nelson, David M. Nicol
- **Artículos:**
  - Artículos sobre pruebas estadísticas de aleatoriedad (Chi-cuadrado, Kolmogorov-Smirnov)
  - Documentación de bibliotecas de Python para generación de números aleatorios (numpy.random) y pruebas estadísticas (scipy.stats).

Esta clase profundiza en los aspectos prácticos de la generación de números aleatorios con GCL, incluyendo la selección de parámetros, las pruebas de uniformidad y su implementación. La clase siguiente se centrará en las pruebas de independencia.