Actividad 10: Generador pseudoaleatorio

EDyA 361075 Carlos Eduardo Sánchez Torres

8 de mayo de 2021

1. Descripción de la actividad

GENERAR una tabla de números pseudoaleatorios utilizando el método CONGRUENCIAL MIXTO.
CONSIDERE que los generadores lineales generan una secuencia de números pseudoaleatorios mediante una relación de recurrencia. Es decir, el próximo número pseudoaleatorio es determinado a partir del último número generado, es decir $Xn + 1$ es derivado a partir de X_n.
La relación de recurrencia es la siguiente: X_{n+1} = $(aX_n + c)\ mod\ m$, donde, X_0 = la semilla $(X_0 > 0)$, a = el multiplicador $(a > 0)$, c = constante aditiva $(c > 0)$, m = el módulo ($m > X_0$, $m > a$ y $m > c$). m también representa el intervalo de valores diferentes que pueden ser generados, X_{n+1} = 0,1, 2,3,..., m-1.

PRUEBAS CON VALORES ARBITRARIOS DE a, c y m: Asigne a las variables los siguientes valores $a = 5$; $X_0 = 4$; $c = 7$; $m = 8$; pruebe con un periodo de 8, es decir, genere 8 valores, después genere periodos de 16 y 32. ¿Qué observa?¿Existe algún patrón en los resultados?¿Existe alguna relación entre el periodo y el valor de m?
Asigne a las variables los siguientes valores $a = 7$; $X_0 = 7$; $c = 7$; $m = 10$; pruebe con un periodo de 7, después genere periodos de 16 y 32. ¿Qué observa?¿Existe algún patrón en los resultados?¿Cada cuánto se repite el periodo y el valor de m?
Asigne a las variables los siguientes valores $a = 81$; $X_0 = 5$; $c = 89$; $m = 100$; pruebe con un periodo de 8, es decir, genere 8 valores), después genere periodos de 16 y 32. Compare los resultados para el cálculo con $a = 1$; $X_0 = 5$; $c = 9$; $m = 10$; ¿Qué observa?¿Existe algún patrón en los resultados?¿Existe alguna relación de recurrencia?

DETERMINAR VALORES APROPIADOS PARA a, c y m. Elija los valores apropiados para m, a, c, X_0 de manera que el patrón del generador pseudoaleatorio no sea predecible.

2. Resultados

```python
def lcg(a, x, c, m, i):
    assert(a > 0 and x > 0 and c > 0 and m > x)
    print(x, end=" ")
    for i in range(0,i):
        t = (a*x+c) % m
        print(t, end=" ")
        x = t
    print()

lcg(a=19,x=37,c=33,m=100,i=4)
lcg(a=13,x=6,c=7,m=8,i=8)
lcg(a=12,x=6,c=7,m=8,i=3)
~
~
~
~
~
~
~
~
NORMAL table.py unix | utf-8 | python 8% 1:19
```

Figura 1: Función generadora de números pseudoaleatorios.

```
 cest ~/Workspace/uabc/src/datastructure-simulation  master ± python table.py
37 36 17 56 97
6 5 0 7 2 1 4 3 6
6 7 3 3
 cest ~/Workspace/uabc/src/datastructure-simulation  master ± _
```

Figura 2: Tabla de números pseudoaleatorios.

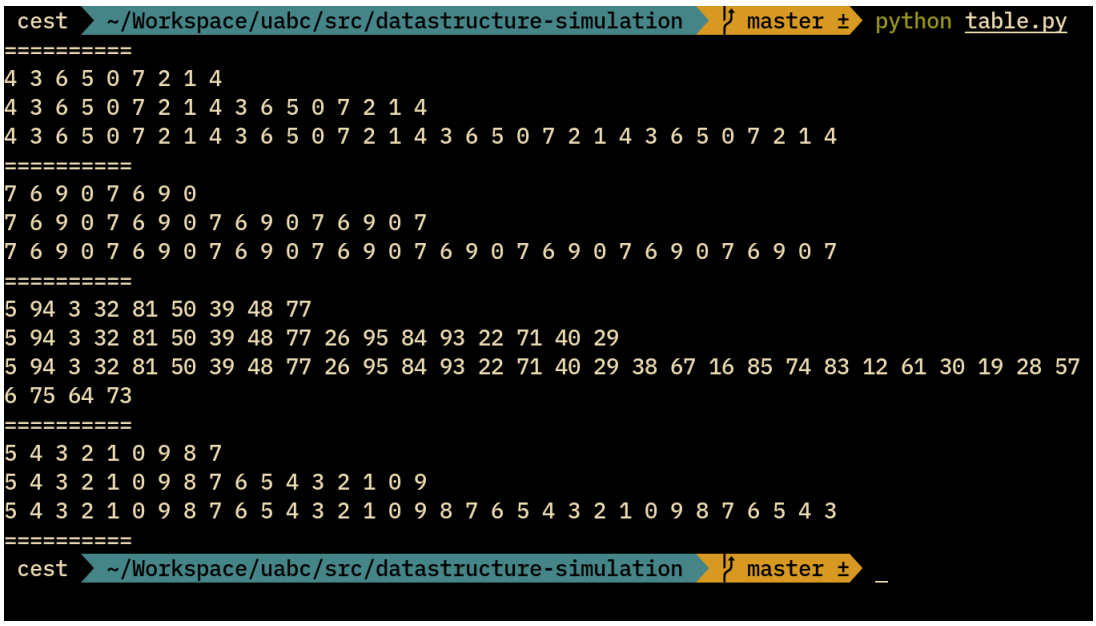

Figura 3: Tabla de valores arbitrarios.

¿Qué observa?¿Existe algún patrón en los resultados?¿Cada cuánto se repite el periodo y el valor de m? En cada uno existe un patrón, sin importar el valor semilla. Por ejemplo, para el primer caso:

$$\underbrace{43650721}_{i=m=8}\cdots\underbrace{43650721}_{m=8,\, i=32=4*m} \tag{1}$$

En el segundo caso, se repite cada $m = 7$ y el periodo $i\,\%7 = 0$.

En el tercer caso y cuarto caso, no hay recurrencia visible y no hay patrón claro para determinar el siguiente número.

3. Conclusiones

Para lograr el máximo periodo m sin repeticiones, se debe cumplir el Teorema Hull-Dobell [2]:

- $mcd(c, m) = 1$
- $a \equiv 1$ mod q, si q es factor primo de m.
- $a \equiv 1$ mod 4, si 4 es un factor de m.

Consecuencias. Dado que trabajamos en máquina binarias, $m = 2^p$, donde p es el número de bits que tiene para una palabra de computadora en el sistema. $a \equiv 1$ mod 4, $a = 2^{p-1} + 1$. c impar y $mcd(c, m) = 1$.
Normalmente, $p = 32$. Entonces, X_0, podría ser un valor fijo o variable. En nuestro caso, usaremos como semilla la hora del reloj.

$$m = 2^{32} = 4294967296 \equiv 0 \text{ mod } 4 \tag{2}$$

$$a = 2^{p-1} + 1 = 2147483649 \tag{3}$$

$$c = 4294967296 + 1 = 4294967297 \tag{4}$$

Figura 4: Posible función aleatoria.

Referencias

[1] Coss B. Raul, “Simulación un enfoque práctico”, Editorial Limusa, S.A. de C.V., México, 2014.

[2] Hull, T. E.; Dobell, A. R. (1 de enero de 1962). «Random Number Generators». SIAM Review 4 (3): 230-254. doi:10.1137/1004061.