# Simulación - Segunda tarea

Sergio Arnaud Gómez 159189

10 de septiembre del 2018

1. Probar por inducción que para un GLC:

$$Z_i \equiv \left[ a^i Z_0 + c \frac{a^i - 1}{a - 1} \right] \mod m$$

<u>Demostración</u>: (Por inducción sobre i)

(Base de inducción) si i = 0 tenemos que:

$$a^{i}Z_{0} + c\frac{a^{i} - 1}{a - 1} = a^{0}Z_{0} + c\frac{a^{0} - 1}{a - 1}$$
$$= Z_{0} + c\frac{1 - 1}{a - 1}$$
$$= Z_{0}$$
$$\equiv Z_{0} \mod m$$

De forma que para i = 0 tendremos

(Hipótesis de inducción) Ahora supongamos que el resultado válido para i=n y probemos la afirmación para n+1.

Por un lado, por la definición de los generadores lineales congruenciales tendemos que :

$$Z_{n+1} \equiv (aZ_n + c) \bmod m \tag{1}$$

Por otro lado, por la hipótesis de inducción tenemos que:

$$Z_n \equiv \left[ a^n Z_0 + c \frac{a^n - 1}{a - 1} \right] \bmod m$$

Trabajando con esta última expresión obtenemos:

$$Z_{n} \equiv \left[ a^{n} Z_{0} + c \frac{a^{n} - 1}{a - 1} \right] \mod m$$

$$\Rightarrow \qquad aZ_{n} \equiv a \left[ a^{n} Z_{0} + c \frac{a^{n} - 1}{a - 1} \right] \mod m$$

$$\Rightarrow \qquad aZ_{n} + c \equiv a \left[ a^{n} Z_{0} + c \frac{a^{n} - 1}{a - 1} \right] + c \mod m$$

$$\Leftrightarrow \qquad aZ_{n} + c \equiv \left[ a^{n+1} Z_{0} + c \frac{a^{n+1} - a}{a - 1} + c \right] \mod m$$

$$\Leftrightarrow \qquad aZ_{n} + c \equiv \left[ a^{n+1} Z_{0} + c \frac{a^{n+1} - 1}{a - 1} \right] \mod m \qquad (2)$$

Dado que la relación de congruencia es, en particular, una relación de equivalencia se tiene la transitividad y por las ecuaciones (??) y (??) concluimos la demostración al obtener:

$$Z_{n+1} \equiv \left[ a^{n+1}Z_0 + c\frac{a^{n+1}-1}{a-1} \right] \bmod m$$

2. ¿Qué se puede decir de el periodo de  $Z_i \equiv aZ_{i-1} \mod m$  con a=630,360,016 y  $m=2^{31}-1$ 

Dado que es un GLC multiplicativo no cumple el teorema del periodo completo (c=0 por lo que no es primo relativo con m) de forma que el periodo máximo que podría alcanzar es m-1

- 3. Sin calcular ninguna  $Z_i$ , determinar cuál de los siguientes GLC's mixtos tienen periodo completo.
  - (a)  $Z_i \equiv [13Z_i + 13] \mod 16$
  - (b)  $Z_i \equiv [12Z_i + 13] \mod 16$
  - (c)  $Z_i \equiv [13Z_i + 12] \mod 16$
  - (d)  $Z_i \equiv [Z_i + 12] \mod 16$
  - (e)  $Z_i \equiv [aZ_i + c] \mod m$  con a = 2814749767109, c = 59482661568307 y  $m = 2^{48}$

# Solución:

Para resolver dicho problema se realizó una función en python que permite saber si un GLC tiene periodo completo o no, lo hace tras verificar que cumpla las 3 hipótesis del teorema del periodo completo, es decir, verifica:

- a) Que c y m son primos relativos
- b) Que si q es un número primo que divide a m, entonces q también divide a -1 (a  $\equiv 1$ ) mod q, para cada factor primo de m.)
- c) Finalmente, que si 4 divide a m, entonces 4 divide a -1. (a  $\equiv 1$  m'od 4, si 4 divide a m).

El programa está escrito en python 3 y el código fuente se muestra a continuación:

Tras ejecutar el programa en los ejercicios proporcionados se obtuvo que los generadores dadas por las expresiones a), d) y e) tienen periodo completo mientras que los dados por b) y c) no, a continuación se muestran los resultados

```
1 a,c,m = 13,13,16
2 complete_period(a,c,m)
```

True

```
1 a,c,m = 12,13,16
2 complete_period(a,c,m)
```

Falla condición 2:
2 es primo y divide a m=16 pero no a (a-1)=11
Falla condición 3:
4 divide a m=16 pero no a (a-1)=11

False

```
1 a,c,m = 13,12,16
2 complete_period(a,c,m)
```

Los números no son primos relativos, su MCD(12,16) = 4

#### False

```
1 a,c,m = 1,12,13
2 complete_period(a,c,m)
```

True

```
1 a,c,m = 2814749767109, 59482661568307,2**48
2 complete_period(a,c,m)
```

True

4. Mostrar que el promedio de las  $U_i's$  tomadas de un ciclo completo de un GLC de periodo completo es  $\frac{1}{2}-\frac{1}{m}$ 

### Demostración:

Afirmación: Dado un generador de ciclo completo, si  $Z_i \equiv \left[a^i Z_0 + c \frac{a^i - 1}{a - 1}\right] \mod m$  entonces  $\{Z_i \mid 0 \leq i < m, \} = \{0, 1, ..., m - 1\}$ . Para probar dicha afirmación basta notar que por un lado  $\{Z_i \mid 0 \leq i < m, \} \subset \{0, 1, ..., m - 1\}$  por la definición de los  $Z_i's$ . Por otro lado  $\{0, 1, ..., m - 1\} \subset \{Z_i \mid 0 \leq i < m, \}$  pues en caso contrario el generador no sería completo.

Con dicha afirmación, tenemos:

$$\frac{1}{m} \sum_{i=1}^{m} U_i = \frac{1}{m} \sum_{i=1}^{m} \frac{Z_i}{m}$$

$$= \frac{1}{m^2} \sum_{i=1}^{m} Z_i$$

$$= \frac{1}{m^2} \sum_{i \in \mathbb{N}, i < m} i$$

$$= \frac{1}{m^2} \frac{(m-1)(m)}{2}$$

$$= \frac{(m-1)}{2m}$$

$$= \frac{m}{2} - \frac{1}{2m}$$

5. Generar 10,000 números con U (0, 1) de Excel. Hacer un breve estudio para probar la calidad de los generadores; aplicar las pruebas de uniformidad e independencia a cada conjunto de datos. Resumir resultados en NO MAS de 2 cuartillas, incluyendo gráficas. De acuerdo a tus resultados, ¿cómo calificarías al generador de Excel?

6. Probar que la parte fraccional de la suma de uniformes en [0,1]:  $U_1+U_2+\ldots+U_k$  es también uniforme en el intervalo [0,1].

7. Un generador de Fibonacci obtiene  $X_{n+1}$  a partir de  $X_n$  y  $X_{n-1}$  de la siguiente forma:

$$X_{i+1} = (X_i + X_{i-1}) \mod m$$

Con  $X_0$  y  $X_1$  dados. Para m=5 solo dos ciclos son posibles, encontrarlos y al periodo.

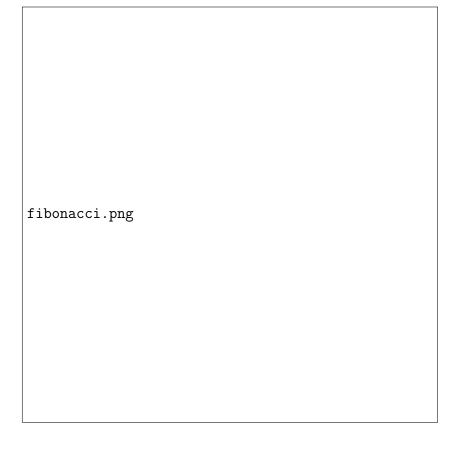
# Solucion:

Para la solución a dicho problema se implementaron las siguientes funciones en python3

La función "fiborecibe como parámetros  $X_0$  y  $X_1$ , las raíces y m, el módulo. Y genera el los números producidos por la iteración hasta caer en un ciclo, como ejemplos:

Haciendo uso de dicha función, la siguiente función obtiene todos los posibles ciclos de el generador de fibonacci para un n dado, para n=5 tenemos los siguientes resultados:

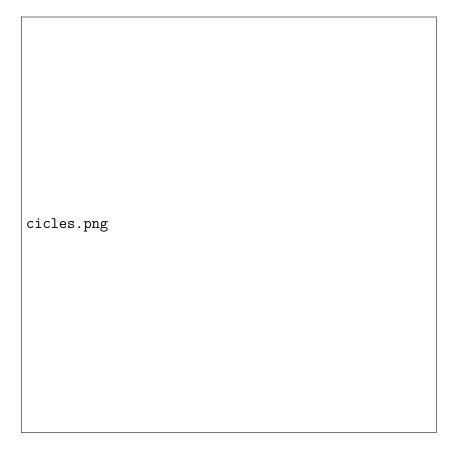
Notamos que, además del ciclo trivial, hay 2 ciclos distintos.



8. Genera 10,000 números con una semilla de  $Z_0 = 1$  usando el generador  $Z_n = 75 Z_{n-1} \mod (2^{31} - 1)$  Clasifica los números en 10 celdas de igual tamaño y prueben por uniformidad usando la prueba  $\xi^2$  con un nivel de confianza del 90 %. Aplicar también la prueba de rachas.

## Solución

```
\label{eq:continuous} \begin{tabular}{l} \hline \textbf{GLC} = \textbf{Function}(\textbf{z}0,\textbf{a},\textbf{c},\textbf{m},\textbf{k}) \\ \textbf{df} = \textbf{data.frame}(\textbf{Ui} = \textbf{z}0/\textbf{m}) \ \textbf{z} = (\textbf{z}0^*\textbf{a} + \textbf{c}) \text{for (i in 2:k) df} = \textbf{rbind}(\textbf{df}, \\ \textbf{data.frame}(\textbf{Ui} = \textbf{z}/\textbf{m})) \ \textbf{z} = (\textbf{z}^*\textbf{a} + \textbf{c}) \ \textbf{return (df)} \\ \textbf{df} = \textbf{GLC}(1,7^5,0,2^31-1,10000) head(df) \\ \textbf{h} = \textbf{hist}(\textbf{df}Ui,breaks = 10,right = FALSE,plot = FALSE) breaks_cdf < -punif(hbreaks) \ \textbf{null.probs} \ \textbf{j-breaks}_cdf[-1] -breaks_cdf[-length(breaks_cdf)]a < -punif(hbreaks) \ \textbf{mull.probs} \ \textbf{j-breaks}_cdf[-1] -breaks_cdf[-length(breaks_cdf)]a < -punif(hbreaks_cdf) \ \textbf{j-breaks}_cdf[-1] -breaks_cdf[-1] -b
```



 $-chisq.test(h\text{counts},\, \mathbf{p}=\text{null.probs},\, \text{rescale.p}=\mathbf{T},\, \text{simulate.p.value}=\mathbf{T})$  a  $_{\odot}$