TECNOLÓGICO NACIONAL DE MÉXIC

Ingeniería en Sistemas Computacional

Simulaci

Unidad II: Números Pseudoaleatori

Material de clase desarrollado para la asignatura de **Simulación** para Ingeniería en Sistemas Computacio

Unidad	Temas	Subtemas
2	Números pseudoaleatorios	2.1 Métodos de generación de números Pseudoaleatorio 2.2 Pruebas estadísticas. 2.2.1 De uniformidad. (chi cuadrada, kolmogorov-Smimov). 2.2.2 De aleatoriedad. (corridas arriba y debajo de la media y longitud de corridas). 2.2.3 De independencia. (Autocorrelación, prueba de huecos, prueba del póquer, prueba de Yule). 2.3 Método de Monte Carlo 2.3.1 Caracteristicas. 2.3.2 Aplicaciones. 2.3.3 Solución de problemas.

Competencias:

- Conocer la diferencia entre un número aleatorio y un pseudoaleatorio.
- Identificar y aplicar los métodos de generación de números pseudoaleatorios.
- Aplicar e interpretar las pruebas estadísticas a los números pseudoaleatorios.
- Seleccionar el generador de números pseudoaleatorios a utilizar en la unidad siguiente.
- Aplicar el método de Montecarlo a la solución de un problema matemático.

SIMULACIÓN

GENERACIÓN DE NÚMEROS PSEUDOALEATORIOS

- Se llama números pseudoaleatorios a una sucesión determinística números en el intervalo [0,1] que tiene las mismas propiedades estadísti que una sucesión de números aleatorios. Los números pseudoaleatorios s necesarios cuando se pone en práctica un modelo de simulación, para obte observaciones aleatorias a partir de distribuciones de probabilidad. números aleatorios generados en un inicio por una computadora casi siem son números aleatorios enteros

SIMULACIÓN

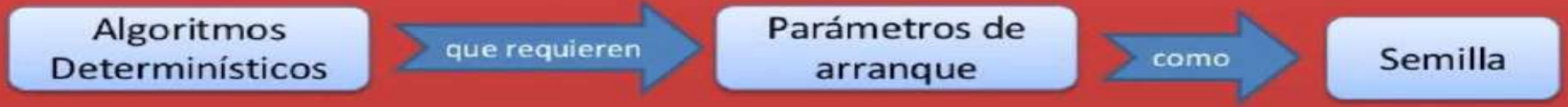
- La aplicación de los números aleatorios se remonta a los tiempos de la primera revolución industrial, cuando los procesos manuales tuvieron que reemplazarse por procesos mecanizados como consecuencia de la explosión demográfica que se estaba presentando en los países desarrollados con la disminución de las tasas de mortalidad y el aumento de las tasas de natalidad y que, para satisfacer las necesidades de la población cada vez más creciente hubo necesidad de incrementar la producción de toda clase de bienes y servicios.

SIMULACIÓN

- El procedimiento usado por una computadora para generar números aleatorios se llama: GENERADOR DE NÚMEROS ALEATORIOS
- La referencia a secuencias de números aleatorios significa que: e algoritmo produce muchos números aleatorios en serie.
- Es posible identificar diferentes métodos usados a través de la historia para generar números aleatorios que pudieran ser utilizados en los procesos de simulación de las actividades industriales. Dichos métodos pudiéramos clasificarlos en:
    1. Manuales
    2. Tablas de Biblioteca
    3. Generadores Analógicos
    4. Generadores Digitales
    5. Métodos de Recurrencia o Congruenciales

¿Cómo se generan los números pseudo aleatorios entre 0 y 1?

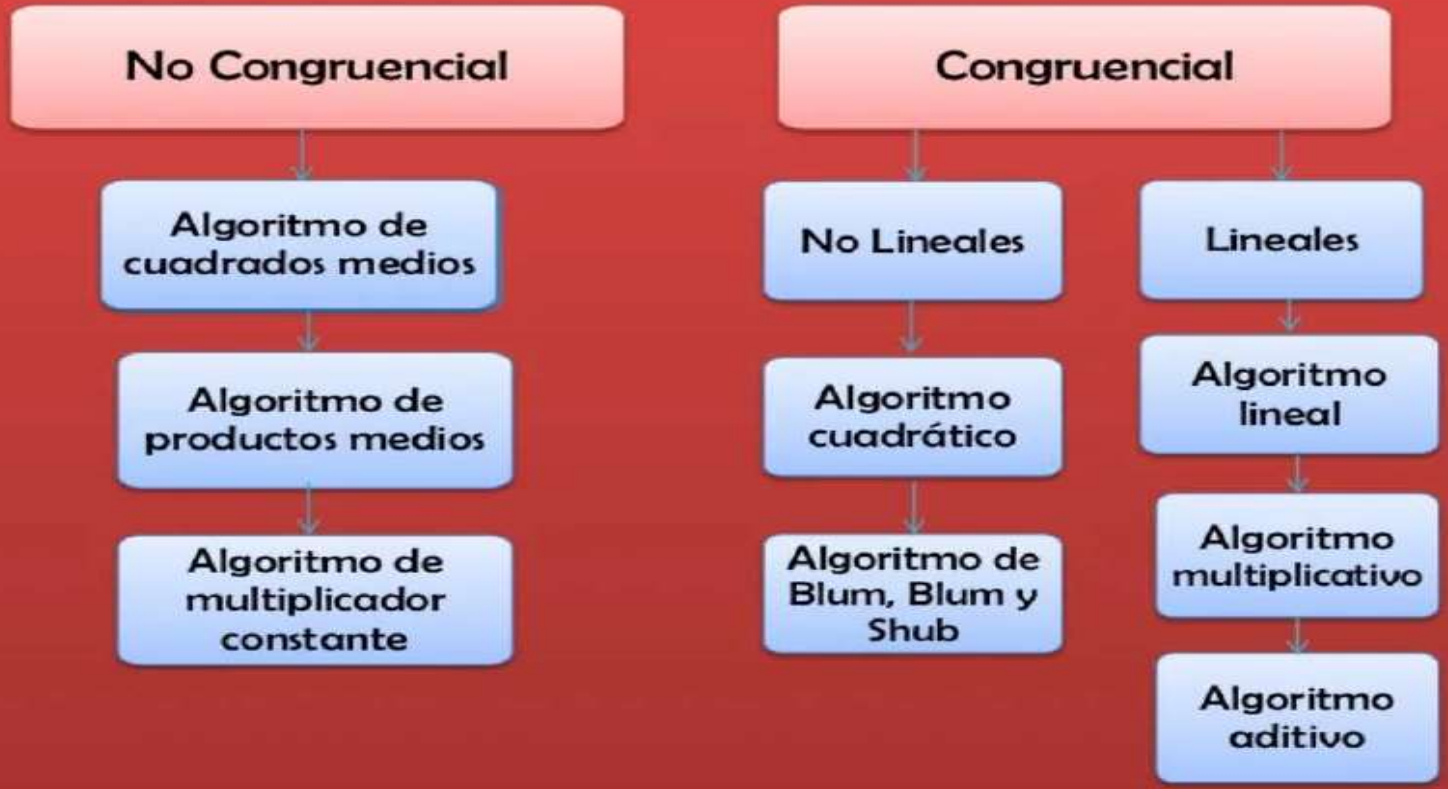
Los números pseudo aleatorios se generan mediante:



Los algoritmos determinísticos se dividen en:

- Congruenciales
- No Congruenciales

¿Cómo se generan los números pseudo aleatorios entre 0 y 1?



Método de los cuadrados medios

Este algoritmo no congruencial fue propuesto en la década de los cuarenta del siglo XX por Von Neumann y Metropolis.[1] Requiere un número entero detonador (llamado *semilla*) con D dígitos, el cual es elevado al cuadrado para seleccionar del resultado los D dígitos del centro; el primer número r_1 se determina simplemente anteponiendo el "0." a esos dígitos. Para obtener el segundo r_1 se sigue el mismo procedimiento, sólo que ahora se elevan al cuadrado los D dígitos del centro que se seleccionaron para obtener el primer r_1. Este método se repite hasta obtener n números r_i. A continuación se presentan con más detalle los pasos para generar números con el algoritmo de cuadrados medios.

1. Seleccionar una semilla (X_0) con D dígitos ($D > 3$).
2. Sea Y_0 = resultado de elevar X_0 al cuadrado; sea X_1 = los D dígitos del centro, y sea $r_1 = 0.$ D dígitos del centro.
3. Sea Y_i = resultado de elevar X_i al cuadrado; sea X_{i+1} = los D dígitos del centro, y sea $r_i = 0.$ D dígitos del centro para toda $i = 1, 2, 3, \ldots n$.
4. Repetir el paso 3 hasta obtener los n números r_i deseados.

Nota: Si no es posible obtener los D dígitos del centro del número Y_i, agregue ceros a la izquierda del número Y_i.

SIMULACIÓN

Ejemplo:

- Suponga que se desean generar números aleatorios por el método de los cuadrados medios con 1111 como semilla y e=8.

i	número al cuadrado (n_{i-1}^2)	n_i	r_i
1	01234321	2343	0.2343
2	05489649	4896	0.4896
3	23970816	9708	0.9708
4	94245264	2452	0.2452
5	06012304	0123	0.0123
6	00015129	0151	0.0151

Ejemplo 2.1

Generar los primeros 5 números r_i a partir de una semilla $X_0 = 5735$, de donde se puede observar que $D = 4$ dígitos.

Solución:

$$Y_0 = (5735)^2 = 32890225 \qquad X_1 = 8902 \qquad r_1 = 0.8902$$
$$Y_1 = (8902)^2 = 79245604 \qquad X_2 = 2456 \qquad r_2 = 0.2456$$
$$Y_2 = (2456)^2 = 06031936 \qquad X_3 = 0319 \qquad r_3 = 0.0319$$
$$Y_3 = (0319)^2 = 101761 \qquad X_4 = 0176 \qquad r_4 = 0.0176$$
$$Y_4 = (0176)^2 = 030976 \qquad X_5 = 3097 \qquad r_5 = 0.3097$$

El algoritmo de cuadrados medios generalmente es incapaz de generar una secuencia de r_i con periodo de vida n grande. Además, en ocasiones sólo es capaz de generar un número, por ejemplo, si $X_0 = 1000$, entonces $X_1 = 0000$; $r_i = 0.0000$ y se dice que el algoritmo se degenera con la semilla de $X_0 = 1000$.

Algoritmo de los Productos Medios

- La mecánica de generación de números pseudoaleatorios de este algoritmo no congruencial es similar a la del algoritmo de cuadrados medios. La diferencia entre ambos radica en que el algoritmo de productos medios requiere dos semillas, ambas con *D* dígitos; además en lugar de elevarlas al cuadrado, las semillas se multiplican y del producto se seleccionan los *D* dígitos del centro, los cuales formarán el primer número pseudoaleatorio $r_i = 0.D$ dígitos.

SIMULACIÓN

- Después se elimina una semilla, y la otra se multiplica por el primer número de *D* dígitos, para luego seleccionar del producto los *D* dígitos que conformarán un segundo número r(.. Entonces se elimina la segunda semilla y se multiplican el primer número de *D* dígitos por el segundo número de *D* dígitos; del producto se obtiene el tercer número *rj* Siempre se irá eliminando el número más antiguo, y el procedimiento se repetirá hasta generar los *n* números pseudo aleatorios.

SIMULACIÓN

A continuación se presentan con más detalle los pasos del método para generar números con el algoritmo de producto medios.

1. Seleccionar una semilla (X_0) con D dígitos ($D > 3$)
2. Seleccionar una semilla (X_1) con D dígitos ($D > 3$)
3. Sea $Y_0 = X_0 * X_1$; sea X_2 = los D dígitos del centro, y sea $r_i = 0.D$ dígitos del centro.
4. Sea $Y_i = X_i * X_{i+1}$; sea X_{i+2} = los D dígitos del centro, y sea $r_{i+1} = 0.D$ dígitos del centro para toda $i = 1, 2, 3, \ldots n$.
5. Repetir el paso 4 hasta obtener los n números r_i deseados.

Nota: Si no es posible obtener los D dígitos del centro del número Y_i, agregue ceros a la izquierda del número Y_i.

SIMULACIÓN

Generar los primeros 5 números r_i a partir de las semillas $X_0 = 5015$ y $X_1 = 5734$; observe que ambas semillas tienen $D = 4$ dígitos.

Solución:

$Y_0 = (5015)(5734) = 28756010$ $\quad X_2 = 7560$ $\quad r_1 = 0.7560$

$Y_1 = (5734)(7560) = 43349040$ $\quad X_3 = 3490$ $\quad r_2 = 0.3490$

$Y_2 = (7560)(3490) = 26384400$ $\quad X_4 = 3844$ $\quad r_3 = 0.3844$

$Y_3 = (3490)(3844) = 13415560$ $\quad X_5 = 4155$ $\quad r_4 = 0.4155$

$Y_4 = (3844)(4155) = 15971820$ $\quad X_6 = 9718$ $\quad r_5 = 0.9718$

Algoritmo de multiplicador constante

Este algoritmo no congruencial es similar al algoritmo de productos medios. Los siguientes son los pasos necesarios para generar números pseudoaleatorios con el algoritmo de multiplicador constante.

1. Seleccionar una semilla (X_0) con D dígitos ($D > 3$).
2. Seleccionar una constante (a) con D dígitos ($D > 3$).
3. Sea $Y_0 = a^*X_0$; sea X_1 = los D dígitos del centro, y sea $r_i = 0.D$ dígitos del centro.
4. Sea $Y_i = a^*X_i$; sea X_{i+1} = los D dígitos del centro, y sea $r_{i+1} = 0.D$ dígitos del centro para toda $i = 1, 2, 3, \ldots n$.
5. Repetir el paso 4 hasta obtener los n números r_i deseados.

Nota: Si no es posible obtener los D dígitos del centro del número Y_i, agregue ceros a la izquierda del número Y_i.

SIMULACIÓN

Generar los primeros 5 números r_i a partir de la semilla $X_0 = 9803$ y con la constante $a = 6965$. Observe que tanto la semilla como la constante tienen $D = 4$ dígitos.

Solución:

$$Y_0 = (6965)(9803) = 68277895 \qquad X_1 = 2778 \qquad r_1 = 0.2778$$
$$Y_1 = (6965)(2778) = 19348770 \qquad X_2 = 3487 \qquad r_2 = 0.3487$$
$$Y_2 = (6965)(3487) = 24286955 \qquad X_3 = 2869 \qquad r_3 = 0.2869$$
$$Y_3 = (6965)(2869) = 19982585 \qquad X_4 = 9825 \qquad r_4 = 0.9825$$
$$Y_4 = (6965)(9825) = 68431125 \qquad X_5 = 4311 \qquad r_5 = 0.4311$$

Algoritmo lineal

Este algoritmo congruencial fue propuesto por D. H. Lehmer[5] en 1951. Según Law y Kelton,[3] no ha sido el más usado. El algoritmo congruencial lineal genera una secuencia de números enteros por medio de la siguiente ecuación recursiva:

$$X_i + 1 = (ax_i + c)\text{mod}(m) \qquad i = 0, 1, 2, 3, \ldots, n$$

donde X_0 es la semilla, a es la constante multiplicativa, c es una constante aditiva, y m es el módulo. $X_0 > 0$, $a > 0$, $c > 0$ y $m > 0$ deben ser números enteros. La operación "mod (m)" significa multiplicar X_i por a, sumar c, y dividir el resultado entre m para obtener el residuo X_{i+1}. Es importante señalar que la ecuación recursiva del algoritmo congruencial lineal genera una secuencia de números enteros $S = \{0, 1, 2, 3, \ldots, m - 1\}$, y que para obtener números pseudoaleatorios en el intervalo (0,1) se requiere la siguiente ecuación:

$$r_i = \frac{X_i}{m-1} \qquad i = 0, 1, 2, 3, \ldots, n$$

SIMULACIÓN

Generar 4 números entre 0 y 1 con los siguientes parámetros: $X_0 = 37$, $a = 19$, $c = 33$ y $m = 100$.

Solución:

$$X_1 = (19*37 + 33) \bmod 100 = 36 \qquad r_1 = 36/99 = 0.3636$$
$$X_2 = (19*36 + 33) \bmod 100 = 17 \qquad r_2 = 17/99 = 0.1717$$
$$X_3 = (19*17 + 33) \bmod 100 = 56 \qquad r_3 = 56/99 = 0.5656$$
$$X_4 = (19*56 + 33) \bmod 100 = 97 \qquad r_4 = 97/99 = 0.9797$$

En el ejemplo anterior se dieron de manera arbitraria cada uno de los parámetros requeridos: X_0, a, c, m. Sin embargo, para que el algoritmo sea capaz de lograr el máximo periodo de vida N, es preciso que dichos parámetros cumplan ciertas condiciones. Banks, Carson, Nelson y Nicol[1] sugieren lo siguiente:

$m = 2^g$

$a = 1 + 4k$

k debe ser entero

c relativamente primo a m

g debe ser entero

Bajo estas condiciones se obtiene un periodo de vida máximo: $N = m = 2^g$. Veamos un ejemplo más, tomando en cuenta lo anterior.

Generar suficientes números entre 0 y 1 con los parámetros $X_0 = 6$, $k = 3$, $g = 3$ y $c = 7$, hasta encontrar el periodo de vida máximo (N).

Como podemos ver, si se cumplen las condiciones que Banks, Carson, Nelson y Nicol sugieren, se logrará el periodo máximo $N = m = 8$. A continuación se presenta el desarrollo de la generación de los números r_r

$$a = 1 + 4\,(3) = 13 \text{ y } m = 2^3 = 8$$

$X_0 = 6$

$X_1 = (13*6 + 7) \bmod 8 = 5$	$r_1 = 5/7 = 0.714$
$X_2 = (13*5 + 7) \bmod 8 = 0$	$r_2 = 0/7 = 0.000$
$X_3 = (13*0 + 7) \bmod 8 = 7$	$r_3 = 7/7 = 1.000$
$X_4 = (13*7 + 7) \bmod 8 = 2$	$r_4 = 2/7 = 0.285$
$X_5 = (13*2 + 7) \bmod 8 = 1$	$r_5 = 1/7 = 0.142$
$X_6 = (13*1 + 7) \bmod 8 = 4$	$r_6 = 4/7 = 0.571$
$X_7 = (13*4 + 7) \bmod 8 = 3$	$r_7 = 3/7 = 0.428$
$X_8 = (13*3 + 7) \bmod 8 = 6$	$r_8 = 6/7 = 0.857$

Es importante mencionar que el número generado en $X_8 = 6$ es exactamente igual a la semilla X_0, y si continuáramos generando más números, éstos se repetirían. Además, sabemos que el algoritmo congruencial lineal genera una secuencia de números enteros $S = \{0, 1, 2, 3, \ldots, m - 1\}$. Observe que en este caso se genera la secuencia $S = \{0, 1, 2, 3, 4, 5, 6, 7\}$.

Consideremos de nuevo el ejemplo anterior, pero tratemos de infringir de manera arbitraria alguna de las condiciones. Supongamos que $a = 12$; se sabe que a no es el resultado de $1 + 4k$, donde k es un entero. Veamos el comportamiento del algoritmo congruencial lineal ante tal cambio.

Solución:

$$a = 1 + 4\,(3) = 13 \text{ y } m = 2^3 = 8$$

$X_0 = 6$

$X_1 = (12*6 + 7) \bmod 8 = 7 \qquad r_1 = 7/7 = 1.000$

$X_2 = (12*7 + 7) \bmod 8 = 3 \qquad r_2 = 3/7 = 0.428$

$X_3 = (12*3 + 7) \bmod 8 = 3 \qquad r_3 = 3/7 = 0.428$

El periodo de vida en este caso es $N = 2$, de manera que, como puede ver, el periodo de vida máximo no se logra. Como conclusión tenemos que si no se cumple alguna de las condiciones, el periodo de vida máximo $N = m$ no se garantiza, por lo que el periodo de vida será menor que m.

Algoritmo congruencial multiplicativo

El algoritmo congruencial multiplicativo surge del algoritmo congruencial lineal cuando $c = 0$. Entonces la ecuación recursiva es:

$$X_{i+1} = (aX_i) \bmod (m) \qquad i = 0, 1, 2, 3, \ldots, n$$

En comparación con el algoritmo congruencial lineal, la ventaja del algoritmo multiplicativo es que implica una operación menos a realizar. Los parámetros de arranque de este algoritmo son X_0, a y m, los cuales deben ser números enteros y mayores que cero. Para transformar los números X_i en el intervalo (0, 1) se usa la ecuación $r_i = x_i/(m-1)$. De acuerdo con Banks, Carson, Nelson y Nicol,[1] las condiciones que deben cumplir los parámetros para que el algoritmo congruencial multiplicativo alcance su máximo periodo N, son:

$m = 2^g$
$a = 3 + 8k$ o $a = 5 + 8k$
$k = 0, 1, 2, 3, \ldots$
X_0 = debe ser un número impar
g debe ser entero

A partir de estas condiciones se logra un periodo de vida máximo $N = k/4 = 2^{g-2}$

SIMULACIÓN

Generar suficientes números entre 0 y 1 con los siguientes parámetros: $X_0 = 17$, $k = 2$ y $g = 5$, hasta encontrar el periodo o ciclo de vida.

Solución:

$$a = 5 + 8\,(2) = 21 \qquad \text{y} \qquad m = 32$$

$X_0 = 17$

$X_1 = (21*17) \bmod 32 = 5 \qquad r_1 = 5/31 = 0.612$

$X_2 = (21*5) \bmod 32 = 9 \qquad r_2 = 9/31 = 0.2903$

$X_3 = (21*9) \bmod 32 = 29 \qquad r_3 = 29/31 = 1.9354$

$X_4 = (21*29) \bmod 32 = 1 \qquad r_4 = 1/31 = 0.3225$

$X_5 = (21*1) \bmod 32 = 21 \qquad r_5 = 21/31 = 0.6774$

$X_6 = (21*21) \bmod 32 = 25 \qquad r_6 = 25/31 = 0.8064$

$X_7 = (21*25) \bmod 32 = 13 \qquad r_7 = 13/31 = 0.4193$

$X_8 = (21*13) \bmod 32 = 17 \qquad r_8 = 17/31 = 0.5483$

Si la semilla X_0 se repite, volverán a generarse los mismos números. Por lo tanto, el periodo de vida es $n = 8$, el cual corresponde a $N = m/4 = 32/4 = 8$.

Ahora bien, si quebrantamos la condición de que la semilla sea un número impar, digamos con $X_0 = 12$, tenemos:

Solución:

$X_0 = 12$

$X_1 = (21*12) \bmod 32 = 28 \qquad r_1 = 28/31 = 0.9032$

$X_2 = (21*28) \bmod 32 = 12 \qquad r_2 = 12/31 = 0.3870$

En vista de que la semilla X_0 se repite, volverán a generarse los mismos números. Por lo tanto, el periodo de vida es $N = 2$.

Algoritmo congruencial aditivo

Este algoritmo requiere una secuencia previa de n números enteros $X_1, X_2, X_3, X_4, \ldots, X_n$ para generar una nueva secuencia de números enteros que empieza en $X_{n+1}, X_{n+2}, X_{n+3}, X_{n+4}, \ldots$
Su ecuación recursiva es:

$$X_i = (x_{i+1} + X_{i-n}) \bmod (m) \qquad i = n+1, n+2, n+3, \ldots N$$

Los números r_i pueden ser generados mediante la ecuación

$$r_i = x_i / m - 1$$

SIMULACIÓN

Generar 7 números pseudoaleatorios entre cero y uno a partir de la siguiente secuencia de números enteros: 65, 89, 98, 03, 69; $m = 100$.

Sean $X_1 = 65, X_2 = 89, X_3 = 98, X_4 = 03, X_5 = 69$. Para generar $r_1, r_2, r_3, r_4, r_5, r_6$ y r_7 antes es necesario generar $X_6, X_7, X_8, X_9, X_{10}, X_{11}, X_{12}$.

Solución:

$$X_6 = (X_5 + X_1) \bmod 100 = (69 + 65) \bmod 100 = 34 \qquad r_1 = 34/99 = 0.3434$$

$$X_7 = (X_6 + X_2) \bmod 100 = (34 + 89) \bmod 100 = 23 \qquad r_2 = 23/99 = 0.2323$$

$$X_8 = (X_7 + X_3) \bmod 100 = (23 + 98) \bmod 100 = 21 \qquad r_3 = 21/99 = 0.2121$$

$$X_9 = (X_8 + X_4) \bmod 100 = (21 + 03) \bmod 100 = 24 \qquad r_4 = 24/99 = 0.2424$$

$$X_{10} = (X_9 + X_5) \bmod 100 = (24 + 69) \bmod 100 = 93 \qquad r_5 = 93/99 = 0.9393$$

$$X_{11} = (X_{10} + X_6) \bmod 100 = (93 + 34) \bmod 100 = 27 \qquad r_6 = 27/99 = 0.2727$$

$$X_{12} = (X_{11} + X_7) \bmod 100 = (27 + 23) \bmod 100 = 50 \qquad r_7 = 50/99 = 0.5050$$

Algoritmos congruenciales no lineales

Algoritmo congruencial cuadrático

Este algoritmo tiene la siguiente ecuación recursiva:

$$X_{i+1} = (aX_i^2 + bX_i + c) \bmod (m) \qquad i = 0,1,2,3,...N$$

En este caso, los números r_i pueden ser generados con la ecuación $r_i = x_i/(m-1)$. De acuerdo con L'Ecuyer,[4] las condiciones que deben cumplir los parámetros m, a, b y c para alcanzar un periodo máximo de $N = m$ son:

$m = 2^g$

a debe ser un número par

c debe ser un número impar

g debe ser entero

$(b-1) \bmod 4 = 1$

De esta manera se logra un periodo de vida máximo $N = m$.

SIMULACIÓN

Generar, a partir del algoritmo congruencial cuadrático, suficientes números enteros hasta alcanzar el periodo de vida, para esto considere los parámetros $X_0 = 13$, $m = 8$, $a = 26$, $b = 27$ y $c = 27$. Como todas las condiciones estipuladas para los parámetros se satisfacen, es de esperarse que el periodo de vida del generador sea $N = m = 8$, tal como podrá comprobar al revisar los cálculos correspondientes, que se presentan a continuación.

Solución:

$$X_1 = (26*13^2 + 27*13 + 27) \bmod (8) = 4$$
$$X_2 = (26*4^2 + 27*4 + 27) \bmod (8) = 7$$
$$X_3 = (26*7^2 + 27*7 + 27) \bmod (8) = 2$$
$$X_4 = (26*2^2 + 27*2 + 27) \bmod (8) = 1$$
$$X_5 = (26*1^2 + 27*1 + 27) \bmod (8) = 0$$
$$X_6 = (26*0^2 + 27*0 + 27) \bmod (8) = 3$$
$$X_7 = (26*3^2 + 27*3 + 27) \bmod (8) = 6$$
$$X_8 = (26*6^2 + 27*6 + 27) \bmod (8) = 5$$
$$X_9 = (26*5^2 + 27*5 + 27) \bmod (8) = 4$$

Por otro lado, el algoritmo cuadrático genera una secuencia de números enteros $S = \{0, 1, 2, 3, ..., m-1\}$, al igual que el algoritmo congruencial lineal.

2.2.7.2 Algoritmo de Blum, Blum y Shub[2]

Si en el algoritmo congruencial cuadrático $a = 1$, $b = 0$ y $c = 0$, entonces se construye una nueva ecuación recursiva:

$$X_{i+1} = (X_i^2) \bmod (m) \qquad i = 0,1,2,3,...n$$

La ecuación anterior fue propuesta por Blum, Blum y Shub[2] como un nuevo método para generar números que no tienen un comportamiento predecible.