Curso:
Métodos de Monte Carlo
Unidad 4, Sesión 9: Números aleatorios (parte 2)

Departamento de Investigación Operativa
Instituto de Computación, Facultad de Ingeniería
Universidad de la República, Montevideo, Uruguay

dictado semestre 1 - 2016

Contenido:

1. Generadores de números seudo-aleatorios.

2. Propiedades deseables.

3. Principales familias de generadores de números seudo-aleatorios.

Generadores de números seudo-aleatorios

Como vimos en la sesión pasada, los generadores de números seudo-aleatorios son la forma más comúnmente usada para obtener secuencias de valores que puedan utilizarse en lugar de la secuencia de muestras de variables aleatorias uniformes e independientes, necesarias para todo método de Monte Carlo.

En general, todos los generadores de números seudo-aleatorios se implementan a través de una función $f : [0, Q-1]^t -> [0, Q-1]$, que recibe en entrada t valores entre 0 y $Q-1$, y genera uno adicional en este mismo rango. La función se aplica para generar una secuencia $\{Z_n, n \geq 0\}$ de números, donde el número $Z_n = f(Z_{n-t}, Z_{n-t+1}, \ldots, Z_{n-1})$ se calcula en función de los t anteriores en la secuencia, y donde $Z_0, Z_1, \ldots, Z_{t-1}$ son valores de inicialización llamados $semillas$, y que deben elegirse con tanto cuidado como la propia función f.

Como en general estamos interesados en valores $U(0, 1)$, se suele dividir los valores Z_n entre Q, para obtener números (racionales) entre 0 y 1.

Esto involucra un error de aproximación, que de todas formas existe ya que en los lenguajes de programación usuales empleamos siempre una representación finita de los números reales y por lo tanto existe una pérdida de precisión.

Dado que un generador de números seudo-aleatorio es una función determinística y sobre un espacio discreto de valores, es inevitable que exista un ciclo en la misma, es decir que exista un n_0 y un n_1 tal que $(Z_{n_0-t}, Z_{n_0-t+1}, \ldots, Z_{n_0-1}) = (Z_{n_1-t}, Z_{n_1-t+1}, \ldots, Z_{n_1-1})$; esto implica a su vez que $Z_{n_0} = Z_{n_1}$, y que de allí en más todos los valores se repitan. Se llama período al largo de la secuencia sin repeticiones, y en todos los casos el mismo será menor o igual a Q^t (que es la cantidad de secuencias de largo t distintas posibles).

Propiedades deseables

Supongamos que el objetivo del empleo de generadores de números seudo-aleatorios es el de simular computacionalmente una secuencia de variables aleatorias independientes con distribución uniforme $(0, 1)$ (ésta es la utilidad que le damos en este curso, otras aplicaciones como las criptográficas pueden tener requerimientos diferentes).

Las principales propiedades que nos interesan son las siguientes:

- Período largo: dado que como acabamos de ver todo generador de números seudo-aleatorio genera una secuencia cíclica, es importante asegurarse que el largo de dicha secuencia sea lo mayor posible, y en particular que exceda los requerimientos de la aplicación que los emplea (para ejemplificar: si vamos a emplear un método Monte Carlo con N replicaciones, y cada replicación emplea M números aleatorios distintos, es necesario que el período del generador exceda NM; de no ser así, los valores comenzarían a repetirse y se perdería la validez estadística del experimento).

- Eficiencia computacional: es conveniente que el generador sea rápido (requiera el menor número de instrucciones posible para generar un valor aleatorio), y emplee poca memoria.

- Repetibilidad: el usar un generador de números seudo-aleatorio permite repetir exactamente la misma secuencia de números, esto es muy importante tanto desde un punto de vista conceptual para permitir la duplicación de un experimento (que otra persona pueda obtener el mismo resultado que el reportado en un informe científico), pero también desde una perspectiva metodológica para emplear técnicas de reducción de varianza, y ya a nivel técnico (programación) para depurar programas que si no tendrían un comportamiento distinto en cada ejecución.

- Portabilidad: es deseable que el generador funcione de la misma manera en distintos sistemas operativos y lenguajes, sin depender tampoco del hardware empleado, ya que esto permite la repetibilidad y garantiza que las conclusiones sobre propiedades estadísticas obtenidas con pruebas en

una plataforma son aplicables a todas.

- En muchos casos es necesario contar con varias secuencias independientes distintas; en muchos generadores de números seudo-aleatorios, esto se puede implementar mediante la posibilidad de calcular la secuencia a partir de un valor n arbitrario, ya que si el período es muy largo, es posible calcular entonces las subsecuencias que comienzan en distintas ubicaciones del mismo y emplearlas como generadores virtuales independientes.

- Uniformidad e independencia: claramente el conjunto de propiedades anteriores, si bien importante, no alcanza para definir un buen generador de números seudo-aleatorios (por ejemplo, el generador $f(x) = (x + 1) \text{mod } Q$ podría cumplir con todos los requisitos, pero resultaría completamente inútil en la práctica). Aunque sabemos que los números generados por un proceso determinístico no pueden cumplir en sentido estricto con estas propiedades, deseamos un generador tal que las secuencias generadas ´´parezcan´´ independientes y uniformemente

distribuidas, es decir, que tengan un comportamiento frente a diversos tests estadísticos similar al que tendría una secuencia de variables aleatorias con estas características.

Familias de generadores

Los generadores de números aleatorios más utilizados y empleados actualmente entran en alguna de las siguientes categorías:

- Basados en recurrencias lineales
  - de un paso,
  - de múltiples pasos,
  - de múltiples pasos, y módulo 2,
- Basados en combinaciones de recurrencias lineales.
- Basados en recurrencias no lineales.

Generadores basados en recurrencias lineales

Las familias de generadores más estudiadas son probablemente las basadas en la recurrencia lineal

$$Z_i = \left(\sum_{s=1}^{p} A_s Z_{i-s} \right) \text{ mod } Q,$$

donde $A_1, \ldots, A_p$, son enteros no negativos tales que $A_p > 0$ y Q es un entero.

Cuando $p = 1$, tenemos generadores basados en recurrencias de un paso, que durante muchos años fueron el método de elección (y aún hoy predominan en las bibliotecas estandarizadas de lenguajes como C y Java).

Dentro de estos métodos, tenemos los métodos multiplicativos congruenciales,

$$Z_i = AZ_{i-1} \text{ mod } Q,$$

y los métodos mixtos congruenciales (o lineales congruenciales),

$$Z_i = AZ_{i-1} + C \text{ mod } Q,$$

introducidos por D.H. Lehmer en 1949.

Estas funciones son de las más simples imaginables, lo que ha permitido el estudio teórico y empírico de su comportamiento, con resultados que permiten conocer con mucha precisión en qué condiciones estos generadores tienen buenos y malos resultados, y los límites teóricos de su precisión. Quizá sorprendentemente, con una buena elección de los valores de A, C y Q, es posible tener generadores de ciclo Q o $Q-1$, y con un comportamiento muy bueno frente a tests de independencia y uniformidad.

En particular, un generador mixto congruencial tiene período Q sí y solo sí

1. C y Q son primos entre sí;

2. $A-1$ es múltiplo de p, para todo primo p que divide a Q;

3. $A - 1$ es múltiplo de 4, si Q es múltiplo de 4.

Y si Q es primo, un generador multiplicativo congruencial tiene período $Q - 1$ sí y solo sí A es una raíz primitiva de Q, es decir que

1. $A^{Q-1} - 1 \bmod Q = 0$;

2. para todo entero $I < Q - 1$, $(A^I - 1)/Q$ no es entero.

Estas condiciones, necesarias y suficientes para garantizar los máximos períodos alcanzables, no son suficientes para garantizar el buen comportamiento de los generadores del punto de vista de la distribución de los valores en las secuencias generadas; es más, existen numerosos ejemplos de implementaciones realizadas con valores inadecuados de estos parámetros, que han resultado en generadores con muy malas características que pueden invalidar los resultados de experimentos realizados en base a las secuencias por ellos generadas.

Existen también apreciaciones respecto a la eficiencia computacional y facilidad de implementación teniendo en cuenta la precisión (tamaño de palabra) de las computadoras existentes, que hace que estos métodos resulten especialmente rápidos para ciertas combinaciones de valores.

La sección "LCG: Linear congruential generators" del reporte "A collection of selected pseudorandom number generators with linear structures", por Karl Entacher (1997), referencia `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.53.3686&rep=rep1&type=pdf` (lectura opcional - último acceso 2016-04-11) contiene información sobre generadores de esta familia que han sido propuestos y utilizados en diversos contextos.

Empleando múltiples pasos en la recursión lineal (es decir, cuando $p > 1$) es posible lograr períodos mayores y también mejores comportamientos en cuanto a la distribución de la secuencia generada. Estos generadores reciben muchas veces el nombre de generadores recursivos múltiples (Multiple Recursive Generators, MRG). Si bien su teoría también ha sido desarrollada de manera detallada, los resultados obtenidos muestran que la

elección de buenos parámetros de un generador puede ser un problema intratable en tiempos razonables.

Sin embargo, cuando se considera el caso especial en que $Q = 2$ (es decir, cuando lo que generamos son bits aleatorios), es posible salvar estos problemas y desarrollar generadores de excelente calidad. Los generadores de la forma $B_i = (\sum_{s=1}^{p} A_s B_{i-s}) \mod 2$ donde A_i y B_i son números binarios se conocen por el nombre de Feedback Shift Register Generators (este nombre surge porque sería posible implementar un generador de este tipo directamente en hardware a través de un circuito de diseño estándar, conocido por feedback shift register). Con una correcta elección de los A_i, es posible obtener períodos de largo $2^p - 1$ (el máximo alcanzable con una recursión de esta forma).

Dado que estos generadores proveen secuencias de bits, si necesitamos obtener números en un rango $(0, 2^w)$ alcanza con tomar w bits consecutivos del generador. Formalmente, cada número aleatorio Z_i tendrá la representación binaria $B_{wi} B_{wi-1} \ldots B_{w(i-1)+1}$.

Una generalización del concepto resulta en los llamados Generalized

Feedback Shift Register Generators (GFSRG), que utilizan la misma formulación para generar los bits aleatorios, pero con reglas más generales para formar los números aleatorios como secuencias de los bits generados.

Lectura adicional: artículo "A New Class of Linear Feedback Shift Register Generators", Pierre L'Ecuyer and Francois Panneton, Proceedings of the 2000 Winter Simulation Conference, Dec. 2000, 690–696, accesible en `http://www.informs-sim.org/wsc00papers/091.PDF` (último acceso 2016-04-11).

Preguntas para auto-estudio

- ¿Que es el período de un generador de números seudo-aleatorios? ¿Cuál es el máximo período posible para un generador que recibe t valores entre 0 y $Q-1$ en entrada?

- ¿Cuáles son las propiedades deseables en un generador de números seudo-aleatorios?

- ¿Cuáles son los generadores basados en recurrencias lineales?

- Los métodos multiplicativos congruenciales, a qué familia de generadores pertenecen?