PEMBANGKIT BILANGAN ACAK

Mata Kuliah Pemodelan & Simulasi

Pertemuan Ke- 7

Riani L.

Jurusan Teknik Informatika

Universitas Komputer Indonesia

Pembangkit Bilangan Acak (Random Number Generator)

- CARA MEMPEROLEH :
  - ZAMAN DAHULU, dgn cara :
    - Melempar dadu
    - Mengocok kartu
  - ZAMAN MODERN (>1940), dgn cara :

    membentuk bilangan acak secara numerik/ aritmatik(menggunakan komputer) , disebut "Pseudo Random Number" (bilangan pseudo acak).

- PEMBANGKIT BILANGAN ACAK, HARUS :
  - Berdistribusi uniform(0,1) dan tidak berkorelasi antar bilangan.
  - Membangkitkan cepat, storage tidak besar
  - Dapat di "reproduce"
  - Periode besar, karena mungkin bil.acak dibangkitkan berulang

Bilangan Acak ?

- Bilangan acak adalah bilangan yang tidak dapat diprediksi kemunculannya
- Tidak ada komputasi yang benar-benar menghasilkan deret bilangan acak secara sempurna
- Bilangan acak yang dibangkitkan oleh komputer adalah bilangan acak semu (Pseudo Random Number), karena menggunakan rumus-rumus matematika
- Banyak algoritma atau metode yang dapat digunakan untuk membangkitkan bilangan acak
- Bilangan acak dapat dibangkitkan dengan pola tertentu yang dinamakan dengan distribusi mengikuti fungsi distribusi yang ditentukan

Sifat-Sifat Pembangkit PRN

- Independent : tiap variablenya harus bebas dari ketentuan, seperti :
  - Z_{i-1} = merupakan hasil akhir
  - Z_0 = merupakan angka pertama yang bebas tertentu
  - a = merupakan angka konstan yang dapat bebas dengan ketentuan tersendiri
  - c = merupakan angka bebas tetapi tidak ada hubungan tertentu dengan m
- Uniform : suatu distribusi yang umum (distribusi probabilitas) dan sama untuk semua besaran yang dikeluarkan/diambil. Hal ini berarti bahwa diusahakan probabilitasnya sama untuk setiap penarikan random number tersebut.
- Dense : Density Probabilitas Distribution harus mengikuti syarat probabilitas (antara 0 dan 1). Hal ini berarti dalam penarikan angka-angka yang dibutuhkan dari Random Number Generator cukup banyak dan dibuat sedemikian rupa sehingga $0 \leq$ R.N. ≤ 1
- Efficient : artinya dapat cukup sederhana dan dalam menggunakan cara ini harus terlebih dahulu memilih angka-angka untuk variable-variabelnya yang cocok. Hal ini berarti dalam penarikan random number tersebut harus dapat menentukan angka-angka untuk variabelnya yang sesuai sehingga dapat berjalan terus-menerus.

Penentuan Random Number

a. Tabel Random Number; table ini sudah banyak ditemukan mulai dari enam digit sampai dengan belas digit.

b. Electronic Random Number; number ini banyak juga dipergunakan dalam percobaan penelitian.

c. Conguential Pseudo Random Number Generator, yang terdiri dari tiga bagian :
   a. Linear Congruential Generator (LCG)
   b. Multiplicative Random Number Generator
   c. Mixed Congruential Random Number Generator

Linear Congruential Generator (LCG)

- Metode ini digunakan untuk membangkitkan bilangan acak dengan distribusi uniform
- Pseudo RNG, berbentuk :

$$Z_i = (aZ_{i-1} + c) \bmod m$$

Dimana :

Z_i = bilangan acak ke-*i* dari deretnya

Z_{i-1} = bilangan acak sebelumnya

a = faktor pengali

c = *increment*

m = modulus

Kunci pembangkit adalah Z_0 yang disebut **umpan** (*seed*).

Contoh 1 LCG :

Membangkitkan bilangan acak sebanyak 8 kali dengan a = 2, c = 7, m = 10, dan Z_0= 2

$Z_1 = (2.2+7) \bmod 10 = 1$

$Z_2 = (2.1+7) \bmod 10 = 9$

$Z_3 = (2.9+7) \bmod 10 = 5$

$Z_4 = (2.5+7) \bmod 10 = 7$

$Z_5 = (2.7+7) \bmod 10 = 1$

$Z_6 = (2.1+7) \bmod 10 = 9$

$Z_7 = (2.9+7) \bmod 10 = 5$

$Z_8 = (2.5+7) \bmod 10 = 7$

Bilangan acak yang dibangkitkan adalah :

1 9 5 7 1 9 5 7

→ Terjadi pengulangan bilangan secara periodik (4)

Contoh 2 LCG :

Membangkitkan bilangan acak sebanyak 8 kali dengan a = 4, c = 7, m = 15, dan Z_0= 3

$Z_1 = (4.3+7) \bmod 15 = 4$

$Z_2 = (4.4+7) \bmod 15 = 8$

$Z_3 = (4.8+7) \bmod 15 = 5$

$Z_4 = (4.5+7) \bmod 15 = 12$

$Z_5 = (4.12+7) \bmod 15 = 10$

$Z_6 = (4.10+7) \bmod 15 = 2$

$Z_7 = (4.2+7) \bmod 15 = 0$

$Z_8 = (4.0+7) \bmod 15 = 7$

Bilangan acak yang dibangkitkan adalah :

4 8 5 12 10 2 0 7

→ Tidak terjadi pengulangan bilangan secara periodik

- Terjadi pengulangan pada periode tertentu atau setelah sekian kali pembangkitan, hal ini adalah salah satu sifat pembangkitan dari metode ini dan PRNG pada umumnya
- LCG mempunyai periode tidak lebih besar dari m, dan pada kebanyakan kasus periodenya kurang dari itu
- *LCG* mempunyai periode penuh ($m - 1$) jika memenuhi syarat berikut:
  1. c relatif prima terhadap m.
  2. $a - 1$ dapat dibagi dengan semua faktor prima dari m
  3. $a - 1$ adalah kelipatan 4 jika m adalah kelipatan 4
  4. $m >$ maks(a, c, Z_0)
  5. $a > 0$, $c > 0$
- Penentuan konstanta LCG (a, c, dan m) sangat menentukan baik tidaknya bilangan acak yang diperoleh dalam arti memperoleh bilangan acak yang seakan-akan tidak terjadi pengulangan.

Contoh 3 LCG :

a = 21, c = 3, m = 16 digunakan untuk menghasilkan angka acak PRN

$Z_i = (21.Z_{i-1} + 3) \bmod 16$

Z_0 = 13 (pilih angka antara 0 dan 15 (diperoleh dari m-1) sebagai seed value/starting value)

$Z_1 = (21.\ Z_0 + 3) \bmod 16$

$= (21.13+3) \bmod 16$

$= 276 \bmod (16)$

= 4 (random number)

Random variate :

$U_i = Z_i/16$

$= 4/16$

= 0,2500

TABLE 3.1 Example LCG $Z_i = (21Z_{i-1} + 3) \bmod(16)$, with $Z_0 = 13$

i	$21Z_{i-1} + 3$	Z_i	$U_i = Z_i/16$
0		13	
1	276	4	0.2500
2	87	7	0.4375
3	150	6	0.3750
4	129	1	0.0625
5	24	8	0.5000
6	171	11	0.6875
7	234	10	0.6250
8	213	5	0.3125
9	108	12	0.7500
10	255	15	0.9375
11	318	14	0.8750
12	297	9	0.5625
13	192	0	0.0000
14	3	3	0.1875
15	66	2	0.1250
16	45	13	0.8125
17	276	4	0.2500
18	87	7	0.4375
19	150	6	0.3750
20	129	1	0.0625

Panduan untuk memilih *a, c* dan *m* untuk mewujudkan **panjang siklus maksimum *(maximum cycle length)*** :

- $m = 2^b$, dimana b ditentukan berdasarkan jumlah *bit* per kata dalam komputer yang digunakan. Sebagian komputer menggunakan 32 bit per kata, sehingga angka 31 merupakan pilihan yang baik untuk *b*
- *c* dan *m* sedemikian sehingga faktor persekutuan terbesar *(greatest common factor)* adalah 1 (integer positif satu-satunya yang membagi *m* dan *c* adalah 1)
- $a = 1 + 4k$, dimana k adalah bilangan integer
- Panjang siklus maksimum yang dapat dicapai sebuah LCG adalah m
- LCG dapat mencapai panjang siklus penuh lebih dari 2.1 milyar (2^{31}) bilangan acak

Membuat Fungsi Pembangkit Bilangan Acak dengan LCG

```
function x=LCM(xs)
% Membangkitkan bilangan acak dengan LCM
a=23; c=15; m=257;
x=mod(a*xs+c,m);
```

Fungsi ini menghasilkan satu bilangan x, dengan memasukkan x sebelumnya (xs), sedangkan a, c dan m merupakan konstanta yang harus didefinisikan

Memanggil Bilangan Acak dengan Fungsi LCG

Membangkitkan 4 bilangan acak dengan x(0) = 10 adalah sebagai berikut:

```
>> LCM(10)
ans =
   245
>> LCM(245)
ans =
   253
>> LCM(253)
ans =
   180
>> LCM(180)
ans =
    43
```

Membangkitkan 20 bilangan acak dengan x(0) = 150 adalah sebagai berikut:

```
>> xs=150;
>> for i=1:20
x(i)=LCM(xs);
xs=x(i);
end
>> x
x =
  Columns 1 through 6
   124    40   164   189   250   111
  Columns 7 through 12
   255   226    73   152   170    70
  Columns 13 through 18
    83   125    63   179    20   218
  Columns 19 through 20
   146    32
```

Multiplicative Random Number Generator

$$Z_i = (a.Z_{i-1}) \bmod m$$

Dimana :

- Bilangan pseudo dimulai dgn nilai awal Z_0 yang disebut benih.
- a & m : bilangan bulat positif tertentu
- $A.Z_{i-1}$ dibagi dgn m dan sisanya diambil sebagai nilai Z_n

- Agar Zn berprilaku acak yang dapat dipertanggungjawabkan :
  - Modulo m dipilih sebesar mungkin untuk memperbesar periode
  - a dipilih agar korelasi antar Z_n minimum
  - Benih Z_o: bilangan Bulat positif ganjil, $Z_o<m$
  - Bilangan acak : $U_i = Z_n/m$

Untuk pemilihan nilai-nilai yang terbaik dijabarkan sebagai berikut :

a. Pemilihan nilai : m (modulo) merupakan suatu angka integer yang cukup besar dan merupakan satu kata dari yang dipakai pada computer. Contoh :
   - Dalam computer IBM 360/370 sistem sebuah kata adalah 32 bits panjangnya, berarti angka integer yang terbesar dalam satu kata computer (computer words) adalah : $2^{32-1} -1 = 2^{31} – 1 = 2147488647$
   - Maka nilai m hasrus lebih satu integer, atau : $m = 2^{32-1} +1 = 2147.483.648$
   - Untuk mesin computer system 1130/1800 IBM yang dikenal dengan 16 BITS Words maka untuk memilih m adalah : $m = 2^{16-1} = 32.768$
   - Sedangkan untuk memilih microcomputer dengan 8 BITS akan digunakan :
   - $m = 2^{8-1} = 128$
   - Dengan nilai m ini akan merupakan pembagi dari nilai (a x Z1) yang mengikuti operasi modulo

- Hal ini akan menjadikan mesin computer hanya dapat tertinggi dengan integer m-1 dan apabila produk-produknya lebih besar dari nilai-nilai ini akan mengakibatkan overflow/hang.

b. Pemilihan konstanta multiplier : a harus tepat.
   - Pemilihan nilai a harus bilangan prima terhadap m. a juga harus bilangan ganjil (odd number). Pemilihan yang terbaik adalah dengan rumus yang lebih mendekat pada ketepatan.
   - Untuk system IBM 1130/1800 dengan : 16 Bits akan diperoleh
   - Dan untuk mikrokomputer dengan 8 Bits, maka akan diperoleh :

c. Pemilihan untuk Z_0, yang dikenal dengan : SEED = Z_0 mengharuskan relative belakangan prima terhadap m. Hal ini dapat diperhatikan dengan mudah apabila dicari untuk m adalah angka berpangkat 2 (dua) → angka exporer dari angka 2. Dengan demikian untuk Z_0 adalah setiap angka-angka yang ganjil (odd number) seperti : $I_{SEED} = Z_0 = 12357$

   Dapat diambil sembarang asalkan bilangan ganjil dan biasanya cukup besar.

d. Bilangan c yang dipilih harus bukan merupakan kelipatan dari m dan juga harus bilangan ganjil.

Contoh :

Misal komputer berkapasitas 12 bit word

- $W = 12$

  $m = 2^{w-1} = 2^{11} = 2048$

  $a = 67 \Rightarrow a \approx 2^{6}$ & $a \equiv 3 \pmod 8$

  misal : $Zo = 129$

- $Z_1 = (67)(129) \bmod 2048 = 451$

  $Z_2 = (67)(451) \bmod 2048 = 1545$

  $Z_3 = (67)(1545) \bmod 2048 = 1115$

  $Z_4 = (67)(1115) \bmod 2048 = 977$

Contoh :

- $U_1 = 451/2048 = 0{,}22015$

  $U_2 = 1545/2048 = 0{,}754395$

  $U_3 = 1115/2048 = 0{,}544434$

  $U_4 = 977/2048 = 0{,}477051$

- Periode : $m/4 = 2048/4 = 512$

  $U_1 = U_{513}$

  $U_2 = U_{514}$

Mixed Congruential Random Number Generator

- Pseudo Random Number ini dapat dirumuskan dengan :

$$Z_n = a^n Z_0 + \frac{a^n - 1}{a - 1} . C (mod. m)$$

- Rumus Pseudo Random Number generator ini adalah dengan syarat utama n harus sejumlah bilangan integer (bulat) dan lebih besar dari nol, rumus ini dikenal juga dengan nama ‘Linier Congruential RNG’

- Namun apabila nilai C = 0 maka akan diperoleh rumus yang dikenal ‘Multiplicative Congruen RNG’. Rumus multiplivative ini cukup baik untuk masa-masa yang akan dating karena sedikit sekali storage memori yang dibutuhkan.

- beberapa kondisi syarat-syaratnya sebagai berikut :
  - C = adalah bilangan relative prima terhadap n
  - a = 1 (mod.q) untuk setiap factor prima q dari m
  - a = 1 (mod 4) apabila 4 adalah suatu factor dari m
- Kondisi 1 berarti bahwa pembagi umum yang terbesar dari c dan m adalah satu. Dan kondisi ini mudah dicapai.
- Kondisi 2 berarti :

$$a - q\left(\frac{a}{q}\right) = 1$$

  Apabila $k = \left(\frac{a}{q}\right)$ akan dapat diperoleh untuk a, yaitu a= 1 +qk

  Dimana q adalah faktor prima dari m
- Kondisi 3 : berarti a = 1 + 4k

  Apabila : = adalah integer. Artinya m bilangan bulat dapat dibagi 4

Penerapannya

Simulasi kejadian "acak" *("random" events)* dalam sebuah restoran *drive-through*

- Waktu tiba mobil di jendela restoran *drive-through*
- Waktu yang diperlukan pengemudi untuk memesan
- Jumlah *hamburger,* minuman, dan kentang yang diorder
- Waktu yang diperlukan oleh restoran untuk menyiapkan pesanan

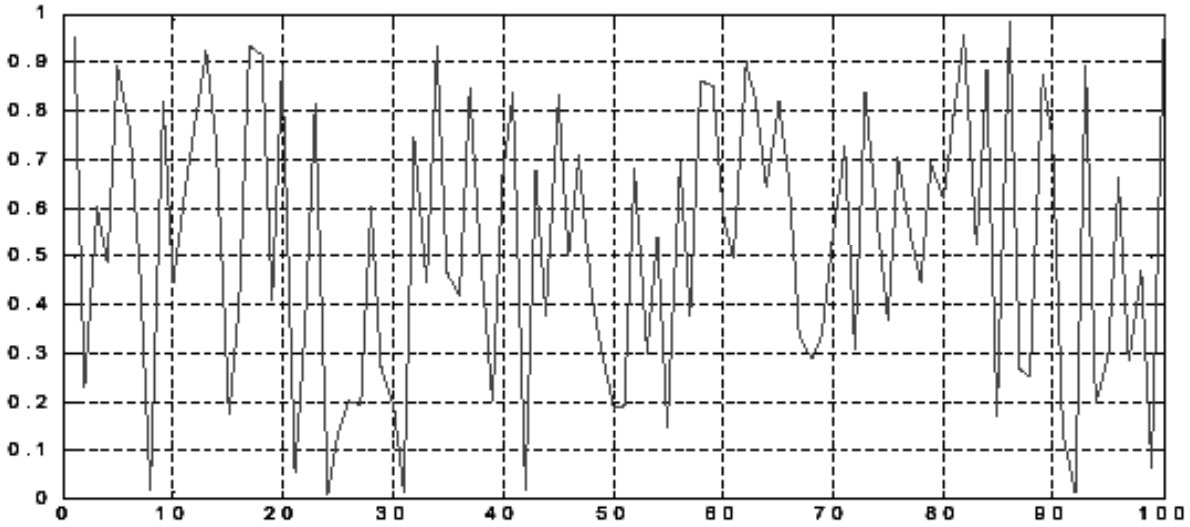
- Panjang rentetan bilangan acak dapat dibagi-bagi dalam segmen yang lebih kecil, yang disebut aliran/***stream.***
- Contoh: *stream 1*: pola kedatangan mobil ke jendela restoran *drive-through; stream 2:* waktu yang diperlukan oleh pengemudi untuk memesan

Bagaimana Penerapannya

- Memutuskan berapa banyak bilangan acak yang ditempatkan dalam masing-masing *stream*
- Bagilah urutan pembangkit dari bilangan acak dalam beberapa *stream*
- Bangkitkan keseluruhan urutan bilangan acak *(cycle length)*
- Catat nilai Z_i yang menandai permulaan masing-masing *stream*
- Masing-masing *stream* memiliki nilai awal sendiri atau yang disebut sebagai *seed value*

Distribusi Bilangan Acak & Grafiknya

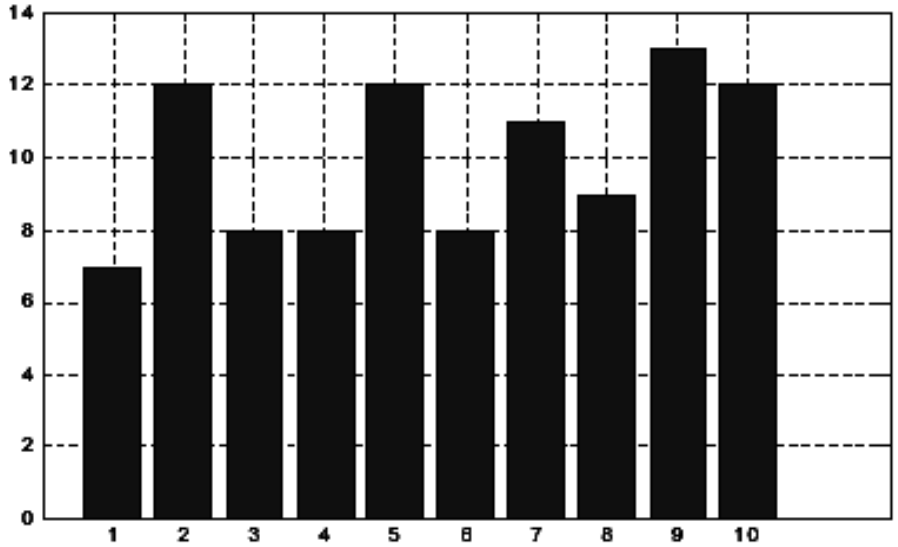
- Bilangan acak dapat dibangkitkan dengan pola tertentu yang mengikuti fungsi distibusi yang ditentukan
- Untuk mengetahui distribusi suatu bilangan acak digunakan histogram atau PDF



Perintah dalam Matlab

```
x=rand(1,100);
plot(x), grid
```

Grafik di atas tidak dapat menggambarkan apa-apa selain nilai maksimum & minimum



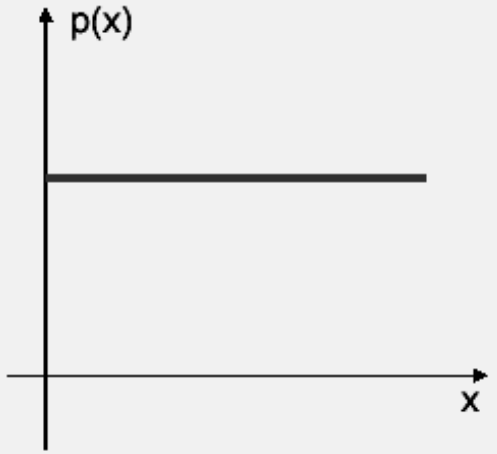
Perintah dalam Matlab

```
x=rand(1,100);
h=hist(x,10);
bar(h), grid
```

Grafik histogram menunjukkan seringnya kemunculan suatu nilai, dalam hal ini dapat menggambarkan distribusi dari bilangan acak yang dibangkitkan

Bilangan Acak Berdistribusi Uniform

- Bilangan acak yang dibangkitkan menggunakan fungsi rand atau metode LCM adalah bilangan acak yang berdistribusi uniform.
- Pada distribusi uniform, kemungkinan munculnya setiap bilangan adalah sama.
- PDF yang ditampilkan seperti gambar disebelah kanan



Histogram & PDF Bilangan Acak Berdistribusi Uniform

- Bangkitkan 1000 bilangan acak bulat 0 s/d 9 dengan fungsi:

```
x=floor(10*rand(1,1000))
```

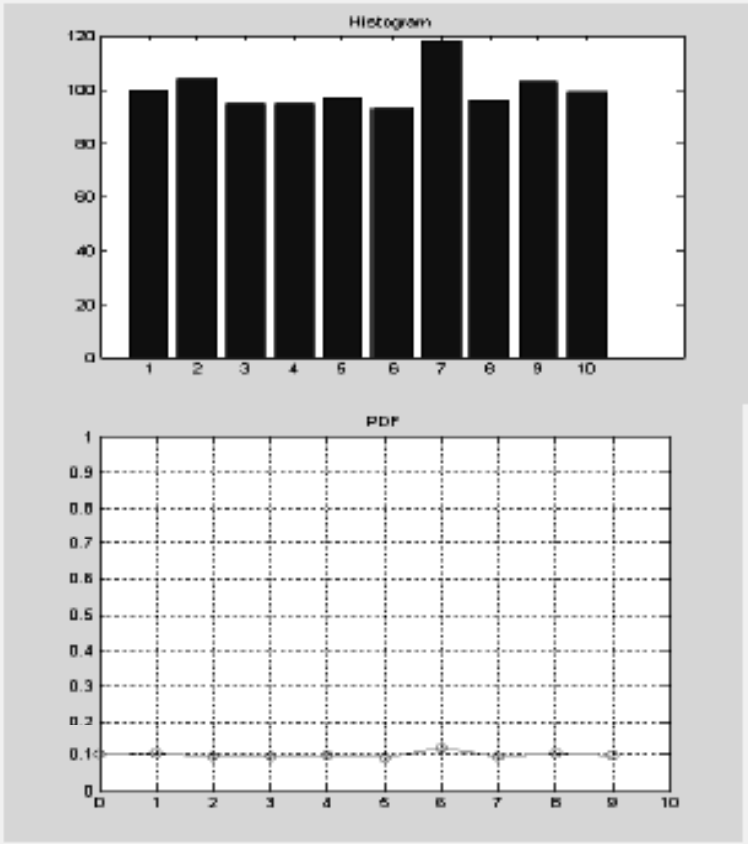
- Tentukan histogram dengan cara:

```
h=hist(x,10);
Figure(1), bar(h), title(histogram)
```

- Tentukan PDF dengan cara:

```
t=0:9;
P=h/sum(h);
Figure(2), plot(t,p), grid, title('PDF');
```

```
x=floor(10*rand(1,1000));
h=hist(x);
figure(1), bar(h), title('Histogram')
t=0:9;
p=h/sum(h);
figure(2), plot(t,p,'o-'), title('PDF')
grid
axis([0 10 0 1])
```



Bagaimana cara membangkitkan random variate ?

Bagaimana cara membangkitkan pengamatan/observasi *(random variates)* dari distribusi selain distribusi uniform (0,1) ?



Transformasikan observasi yang dihasilkan dari pembangkit bilangan acak ke distribusi yang diinginkan



Nilai yang ditransformasikan → *variate* dari distribusi yang dimaksud