Generatorji psevdo naključnih števil

Andrej Flajs
uporabna matematika
andrej.flajs@hermes.si

6. avgust 2001

0.1 UVOD

Ko slišimo oz. preberemo besedo naključno, večina ljudi pomisli na igralnice in prvi **generatorji psevdo naključnih števil** (GPNŠ) so nastali prav v igralniške namene. Vendar so GPNŠ uporabni tudi drugje:

- simulacije naravnih pojavov,
- numerična analiza (npr. računanje določenih integralov po metodi Monte Carlo),
- kriptografija,
- testiranje računalniških programov,
- računalniške igre.

Sedaj vemo, kje se uporabljajo GPNŠ, ne vemo pa, kaj sploh so. To so generatorji, ki nam generirajo zaporedje števil $X_1, X_2, ..., X_n, ...$, za katera velja, da iz $X_1, X_2, ..., X_{n-1}$ ne moremo, oziroma zelo težko določimo število X_n. **Psevdo** pa rečemo, ker zaporedje dobimo po neki formuli, ki pa jo zelo težko uganemo iz zaporedja samega.

Z razvojem računalništva, se je začelo pojavljati veliko GPNŠ, vendar prevladujejo izpeljanke iz "Linear Congruental" metode.
Za vsak GPNŠ ne moremo reči, da vrne zadovoljive rezultate. Najbolj pogosti slabosti sta:

- cikličnost (zaporedje se začne ponavljati),
- konvergenca zaporedja.

Vse nove generatorje moramo zato dobro stestirati. Obstaja veliko različnih testov za katerimi pa je veliko matematične teorije in si bomo zato ogledali samo uporabo.

0.2 GENERATORJI PSEVDO NAKLJUČNIH ŠTEVIL

Ko se je začelo računalništvo razvijati, se je pojavilo tudi veliko GPNŠ, saj so računalniki omogočili hitro računanje velikih kvadratov, ki jih potrebuje *"Midle Squere" metoda*, do množenja in seštevanja v končnih kolobarjih, ki ga potrebuje *"Linear Congruental"* metoda.

"Middle Squere" metoda
To metodo je John von Neumann prvič predstavil leta 1946. Vzamemo npr. 10 mestno število ga kvadriramo in za naslednji člen zaporedja vzamemo srednjih 10 števk. Kakor je metoda preprosta, je tudi slaba, saj za večino začetnih vrednosti zaporedje konvergira k neki vrednosti.
Primer: Izračun naslednjega člena zaporedja

$$X_n = 8527419631$$

$$X_n^2 = 72716885563164176161$$

$$X_{n+1} = 8855631641$$

"Super Random" metoda
Ta metoda je sestavljena iz 13 korakov, ki pa se uporabijo na vsakem koraku samo nekateri. Kateri koraki se v določenem trenutku uporabijo, določi neki drugi GPNŠ. Ta metoda je tipični primer, da GPNŠ, ki uporabljajo druge GPNŠ niso priporočljivi za uporabo.

"Linear Congruential" metoda
To metodo si bomo ogledali malo bolj podrobno, ker je ena največkrat uporabljenih. Obstaja pa tudi veliko modifikacij, ki so boljše in tudi slabše. Člen zaporedja X_{n+1} izračunamo po formuli:

$$X_{n+1} = (a \cdot X_n + c) \, mod \, m$$

pri čemer moramo v naprej izbrati **čarobne vrednosti**:

- X_0 začetna vrednost,

- a multiplikator,
- c zamik,
- m modul.

Čarobne vrednosti moramo zelo pazljivo izbrati, ker je od njih zelo odvisna kvaliteta GPNŠ.

1. Modul
   Modul mora biti čim večji, saj je perioda zaporedja (t.j. kdaj se členi zaporedja začnejo ponavljati) vedno manjša od modula. Ker pa vrednosti računamo z računalnikom, mora biti izbran tudi tako, da se z njim hitro računa. Zato je ponavadi oblike $m = 2^n \pm 1$, kjer ima m malo deliteljev, ali pa $m = največje\ manjše\ praštevilo\ od\ 2^n \pm 1$. n je pri tem dolžina besede $(8, 16, 32, 64, \ldots)$

$2^e - 1$	e	$2^e + 1$
$3 \cdot 5 \cdot 17 \cdot$	8	257
$3 \cdot 5 \cdot 17 \cdot 257$	16	65537
$3 \cdot 5 \cdot 17 \cdot 257 \cdot 65537$	32	$614 \cdot 6700417$
$3 \cdot 5 \cdot 17 \cdot 257 \cdot 641 \cdot 65537 \cdot 6700417$	64	$274177 \cdot 67280421310721$

2. Multiplikator
   Tudi tukaj nam je glavno merilo dolžina periode, vendar moramo zelo paziti tudi na to, da je zaporedje sploh naključno. Multiplikator si lahko izberemo tako, da bomo imeli periodo dolžine m, se pravi najdaljšo možno, zaporedje pa ne bo naključno.(npr.: Izberemo si lahko multiplikator=zamik=1, vendar je potem tako zaporedje $X_{n+1} = (X_n + 1)\, mod\, m$ ni naključno.)
   Če lahko dosežemo dolžino periode m, potem si lahko izberemo začetni člen zaporedja X_0 poljubno, ker pridejo na vrsto vsa števila med 0 in $m - 1$ v vsaki periodi.

Izpeljanke "Linear Congruental" metode

Razlika od "Linear Congruental"metode je v rekurzuvni formuli.

1. $X_{n+1} = ((a \cdot X_n)\, mod\, (m+1) + c)\, mod\, m$
2. $X_{n+1} = (d \cdot X_n^2 + a \cdot X_n + c)\, mod\, m$
3. $X_{n+1} = (X_n + X_{n-1})\, mod\, m$

Čeprav zgledajo formule bolj komplicirane, pa se vse izpeljanke ne izkažejo vedno bolje od osnovne "Linear Congruental"metode.

0.3 Statistični testi

V prejšnjem poglavju smo rekli, da kakšen GPNŠ ni dober, oziroma da so nekateri boljši od drugih. Sedaj si bomo pa ogledali nekaj testov, ki nam to tudi pokažejo. Večina testov temelji na dve osnovnih:

1. χ^2-test,
2. Kolmogorov-Smirnov test,

zato si bomo ta dva malo bolj natančno ogledali, ostale pa samo našteli.

χ^2-test

χ^2-test si je najlažje ogledati na primeru. Metali bomo dve kocki, kot rezultat pa gledamo vsoto cifer. Najprej pa še oznaki:

- Y_s kolikokrat smo vrgli vsoto s (*število zadetkov*)
- p_s verjetnost, da bomo vrgli vsoto s

Če kocki vržemo $n = 144$ krat, lahko dobimo naslednji rezultat:

vsota s	=	2	3	4	5	6	7	8	9	10	11	12
Y_s	=	2	4	10	12	22	29	21	15	14	9	6
$n \cdot p_s$	=	4	8	12	16	20	24	20	16	12	8	4

Na podlagi takega poizkusa izračunamo vrednost:

$$V = \frac{(Y_2 - n \cdot p_2)^2}{n \cdot p_2} + \frac{(Y_3 - n \cdot p_3)^2}{n \cdot p_3} + \cdots + \frac{(Y_{12} - n \cdot p_{12})^2}{n \cdot p_{12}}$$

in ji rečemo χ^2-statistika opazovanih vrednosti $Y_2, \cdots, Y_{12}$. Za naš primer je $V = 7\frac{7}{48}$. Iz poizkusa tudi vidimo, da si lahko poljubno izberemo število metov za 10 različnih vsot, število metov 11-te vsote pa je določeno s številom vseh metov. Pravimo, da je *stopnja svobode* 10.
Stopnja svobode ν pove, za koliko rezultatov si lahko poljubno izberemo število zadetkov in so potem vsi določeni (oz. maksimalno število neodvisnih slučajnih spremenljivk). *Stopnja svobode* je edini parameter za izračun χ^2-porazdelitve. Nam je ni potrebno računati, ker že obstajajo tabele, ki jih

najdemo v različnih knjigah[1] in na internetu.

	$p = 99\%$	$p = 95\%$	$p = 75\%$	$p = 50\%$	$p = 25\%$	$p = 5\%$	$p = 1\%$
$\nu = 10$	2.558	3.940	6.737	9.342	12.55	18.31	23.21

Iz tabele preberemo, kakšna je verjetnost, da bo vrednost V večja od neke, ki jo preberemo v vrstici s *stopnjo svobode*. Najbolje je, da pri poizkušanju dobimo vrednosti V čim bližje tisti pri 50%.
Sedaj si oglejmo test za 4 različne GPNŠ po metodi "Linear Congruential":

- A: $X_0 = 0$, $a = 3141592653$, $c = 2718281829$, $m = 2^{35}$
- B: $X_0 = 0$, $a = 2^7 + 1$, $c = 1$, $m = 2^{35}$
- C: $X_0 = 47594118$, $a = 23$, $c = 0$, $m = 10^8 + 1$
- D: $X_0 = 314159265$, $a = 2^{18} + 1$, $c = 1$, $m = 2^{35}$

A			B			C			D		
		○								●	●
			○						⋆	⋆	●
⋆						○	○		●	●	●
○				●	○				●	●	⋆
									●	●	●

znak	Verjetnost za V	sprejemljivost
●	$0 - 1\%$ in $99 - 100\%$	slabo
⋆	$1 - 5\%$ in $95 - 99\%$	sumljivo
○	$5 - 10\%$ in $90 - 95\%$	skoraj sumljivo

Vse ostale vrednosti so sprejemljive.
Več ima tabela obarvanih polj, slabši je generator. V tem testu je generator D še posebej slab. Vidimo, da je res pomembno kako si izberemo čarobne vrednosti.

[1] [1] Chapter3 - Random Numbers, str. 39,1965

Kalmogorov-Smirnov test

χ^2 test je bil dober, če smo kot rezultat GPNŠ dobivali končne naravne vrednosti, da smo lahko izračunali V. Kolmogorov-Smirnov test pa je narejen za GPNŠ, ki vračajo vrednosti med 0 in 1.
Najprej definiramo *porazdelitveno funkcijo*:

$$F(x) = verjetnost,\ da\ je\ X \leq x$$

pri čemer je X število, ki ga vrne GPNŠ. Porazdelitvene funkcije za različne primere že najdemo določene v nekaterih statističnih knjigah in na internetu. Na podlagi poizkusov določimo *poizkusno porazdelitveno funkcijo*:

$$F_n = \frac{število\ X_1, \ldots, X_n \leq x}{n}$$

pri čemer je n število metov v poizkusu.
Poizkusna porazdelitvena funkcija se mora čim bolje ujemati z porazdelitveno funkcijo. Zato pri Kolmogorov-Smirnov testu gledamo funkciji:

$$K_n^+ = \sqrt{n} \max_{-\infty<x<+\infty} (F_n(x) - F(x))$$

$$K_n^- = \sqrt{n} \max_{-\infty<x<+\infty} (F(x) - F_n(x))$$

V knjigah[2] najdemo tabele zvrednostmi porazdelitev za K_n^+ in K_n^-

	$p = 99\%$	$p = 95\%$	$p = 75\%$	$p = 50\%$	$p = 25\%$	$p = 5\%$	$p = 1\%$
$n = 1$	0.01000	0.05000	0.2500	0.5000	0.7500	0.9500	0.9900
$n = 2$	0.01400	0.06749	0.2929	0.5176	0.7071	1.0980	1.2728
$n = 3$	0.01699	0.07919	0.3112	0.5147	0.7539	1.1017	1.3589
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Podobno kot pr χ^2 testu gledamo, koliko časa sta K_n^+ in K_n^- večja od vrednosti v tabeli. Sedaj ni več parameter stopnja svobode, ampak število metov v vsakem poizkusu. Naredimo lahko enake tabele kot prej:

[2] [1] Chapter3 - Random Numbers,str. 44,1965

Rang vrednosti K_n^+:

A			B			C			D		
								●	⋆	⋆	
							○	○	⋆	●	●
	○			⋆			⋆		⋆	●	●
				●			⋆				○
				○					●		

Rang vrednosti K_n^-:

A			B			C			D		
						○				○	
					○	⋆					
						⋆					
			⋆						●	●	⋆
			○		○			⋆	●	●	●

znak	Verjetnost za K_n^+ in K_n^-	sprejemljivost
●	$0-1\%$ in $99-100\%$	slabo
⋆	$1-5\%$ in $95-99\%$	sumljivo
○	$5-10\%$ in $90-95\%$	skoraj sumljivo

Ta test[3] je narejen za štiri GPNŠ po metodi "Linear Congruental"iz dela o χ^2-testu.

[3][1] Chapter3 - Random Numbers,str. 48,1965

Poizkusni testi

Ogledali si bomo kako naredimo nekaj testov, ki nekaj povedo o naključnosti členov zaporedja. Uporabljali bomo naslednji naključni zaporedji:

$$\langle U_n \rangle = U_0, U_1, U_2, \ldots$$

je zaporedje naključnih števil z vrednostmi med 0 in 1.

$$\langle Y_n \rangle = Y_0, Y_1, Y_2, \ldots$$

zaporedje dobimo iz $\langle U_n \rangle$ po pravilu $Y_n = \lfloor dU_n \rfloor$. d si moramo seveda primerno izbrati. (Na binarnem računalniku npr. 64 ali 128.) Vrednsti zaporedja $\langle Y_n \rangle$ ležijo med 0 in d.

- Frekvenčni test
  S tem testom preverimo, če so členi naključnega zaporedja enakomerno porazdeljeni med najmanjšo in najvišjo možno vrednostjo. Ta test lahko naredimo na dva načina. Lahko naredimo Kolmogorov-Smirnov test na zaporedju $\langle U_n \rangle$ in pri tem vzamemo za porazdelitveno funkcijo $F(x) = x \; za \; 0 \leq x \leq 1$ ali pa χ^2 test na zaporedju $\langle Y_n \rangle$. Za vsako število r, $0 \leq r < d$ štejemo kolikokrat je $Y_j = r$ za $0 \leq j < n$. Možnih rezultatov je d, verjetnost za vsako vrednost pa je $\frac{1}{d}$.

- Serijski test
  Pri serijskem testu štejemo kolikokrat so pari $(Y_{2j}, Y_{2j+1}) = (q, r)$ $0 \leq j < n$; $0 \leq q, r < d$. Naredimo χ^2-test pri čemer je vseh možnih parov (q, r) d^2, za vsakega pa vzamemo verjetnost $\frac{1}{d^2}$. Namesto parov lahko vzamemo tudi trojke, četverke, itd..

- Gap test
  S tem testom gledamo razlike med dvema členoma zaporedja. Naj bosta α, β dve realni števili $0 \leq \alpha < \beta \leq 1$. Opazujemo dolžino podzaporedja $U_j, \ldots, U_{j+r}$ pri katerem leži U_{j+k} med α in β ostali U-ji pa ne. Rečemo, da ima gap dolžino r. Test naredimo na zaporedju $\langle U_n \rangle$. Izberemo si primeren t in štejemo podzaporedja z gapi dolžine $0, 1, 2, \ldots, t-1$ in podzaporedja z dolžino gapa $\geq t$. To naredimo za n podzaporedij. Število podzaporedij, ki pripadajo posamezni dolžini gapa, zapišemo v zaporedje $C_0, \ldots, C_{t-1}, C_t$. Sedaj na $\langle C_n \rangle$ naredimo

χ^2-test z $t+1$ možnimi rezultati in verjetnostmi $p_0 = p$, $p_1 = p(1-p)$,...,$p_{t-1} = p(1-p)^{t-1}$,$p_t = (1-p)^t$, pri čemer je $p = \beta - \alpha$. To je verjetnost, da $\alpha \leq U_j \leq \beta$. Gap test pogosto naredimo za $\alpha = 0$, $\beta = 1$.

- Poker test
  Za poker test iz naključnega zaporedja naredimo peterke (skupine po pet stevil), ki jih porazdelimo v skupine:

vse različne	*abcde*	full house	*aaabb*
en par	*aabcd*	štiri enake	*aaaab*
dva para	*aabbc*	pet enakih	*aaaaa*
tri enake	*aaabc*		

  Sedaj naredimo χ^2 test na podlagi števila peterk v vsaki skupini.

- Permutacijski test
  Iz zaporedja $\langle U_n \rangle$ naredimo n skupin po t elementov.

  $$(U_{jt}, U_{jt+1}, \ldots, U_{jt+t-1}) \;\; 0 \leq j < n$$

  Elemente v vsaki skupini lahko uredimo na $t!$ načinov. Preštejemo koliko skupin pripada kakšni od ureditev in naredimo χ^2 test s številom možnih rezultatov $k = t!$ in verjetnostjo vsakega $\frac{1}{t!}$.

- Testi podzaporedij
  Včasih kakšen algoritem uporablja več naključnih števil naenkrat. Recimo X, Y, Z in jih jemlje iz nekega naključnega zaporedja v trojicah. V tem primeru moramo vedeti, da je podzaporedje, ki vsebuje vsak tretji element tudi naključno. Zato je dobro, da vse teste, ki smo jih že opisali, naredimo tudi na nekaterih podzaporedjih. Ponavadi je opaziti, da so podzaporedja manj naključna kot cela zaporedja.

0.4 Teoretični testi

Čeprav lahko vsak GPNŠ testiramo s testi, ki smo jih spoznali, je bolje, če imamo o vsakem tudi kakšne teoretične rezultate, ki nam povedo nekaj o naključnem zaporedju, ki ga dobimo z njim. Teoretični rezultati nam več povedo o samem delovanju GPNŠ, res pa je, da moramo za vsakega razviti svojo teorijo, ki velja zanj. V knjigi[4] si lahko ogledate nekaj teorije o *linear congruential metodi*.

0.5 Spektralni test

Ta test sta razvil R.R. Coveyou in R.D. MacPerson leta 1965. Pomemben je predvsem zato, ker je označil za slabe vse GPNŠ po *linear congruential metodi*, ki so znani do sedaj in zanje iz drugih testov vemo, da so slabi. Zato je to eno najmočnejših testov. Združuje lastnosti tako praktičnih kot tudi teoretičnih testov. Podobnost s teoretičnim testom je v tem, da testiramo zaporedje na celotni *periodi*, s praktičnimi pa ga veže računalniški program, ki nam vrne rezultate. V knjigi[5] si lahko ogledate nekaj teorije in sam algoritem za izvedbo spektralnega testa.

0.6 Literatura

[1] $D.E.Knuth: \ The\ Art\ of\ Computer\ Programming\ vol.2$
[2] $http://valley.interact.nl/av/com/orion/rng/home.html\#what$
[3] $http://www.npac.syr.edu/projects/random/index.html$
[4] $http://www.fourmilab.ch/random/$
[5] $http://random.mat.sbg.ac.at/test/$
[6] $http://webnz.com/robert/true_rng.html$
[7] $http://www.merrymeet.com/jon/usingrandom.html$

[4] [1] Chapter3 - Random Numbers,str. 69-79,1965
[5] [1] Chapter3 - Random Numbers,str. 82-97,1965