# „Matematyczna Ruletka” – Czyli jak się robi liczby (pseudo)losowe.

WIESŁAW PŁACZEK

Instytut Informatyki Uniwersytetu Jagiellońskiego

**Plan:**

- **Wstęp.**
- **Co to są liczby losowe i skąd się biorą?**
- **Liczby pseudolosowe i ich generatory.**
- **Postawowe rodzaje generatorów matematycznych.**
- **Jakość generatorów liczb losowych.**

## Metody Monte Carlo – krótka historia:

- **Rok 1768:** G. Buffon (francuski matematyk) – eksperymentalne wyznaczenie wartości liczby $\pi$ przez rzucanie igły na poliniowaną kartkę papieru.
- **Lata 1930-te:** E. Fermi – obliczenia dyfuzji neutronów w oparciu o liczby losowe
  $\rightarrow$ Fermiac – mechaniczne urządzenie do obliczeń typu Monte Carlo.
- **Lata 1940-te:** J. von Neumann, S. Ulam, N. Metropolis – podstawy matematyczne metod Monte Carlo (MC) i rachunki MC dużej skali w ramach projektu Manhattan (prace nad bombą jądrową).
  $\rightarrow$ termin: metody Monte Carlo – kryptonim dla tajnych badań.
- **Lata 1950-te:** Szybki rozwój metod MC, ale ograniczone zastosowania – brak wydajnych maszyn cyfrowych.
- **Pojawienie się szybkich komputerów:** Wzrost zainteresowania metodami MC – zastosowania w wielu dziedzinach: nauki przyrodnicze, techniczne, ekonomia, socjologia, itd. (rozwiązywanie skomplikowanych problemów numerycznych).

# Co to są liczby losowe i skąd się biorą?

,,there is no such thing as a random number – there are only methods to produce random numbers"

John von Neumann

**Metody Monte Carlo – algorytmy korzystające z liczb losowych!**

**Liczba losowa** – konkretna wartość przyjmowana przez zmienną losową.

$\rightarrow$ Sekwencja liczb prawdziwie losowych – nieprzewidywalna i niereprodukowalna!

▶ **Źródła liczb prawdziwie losowych – generatory fizyczne:**

- ,,mechaniczne" – np. rzut monetą, losowanie z urny, ruletka itd.
- oparte o losowe procesy fizyczne – np. rozpad radioaktywny, promienie kosmiczne, szumy w urządzeniach elektronicznych (głównie tzw. szum biały) itd.

● **Wady generatorów fizycznych:**

- zbyt wolne dla typowych potrzeb obliczeniowych (szczególnie ,,mechaniczne);
- problemy ze stabilnością – szczególnie generatory oparte o procesy fizyczne, np. niewielka zmiana warunków fizycznych źródła lub otoczenia może spowodować istotne zmiany własności probabilistycznych otrzymywanych liczb losowych $\rightarrow$ potrzebne dodatkowe urządzenia testujące i korygujące.

▷ Dawniej: **tablice liczb losowych** – mało praktyczne! $\rightarrow$ Dziś wracają (?)

(1995: Marsaglia, CD-ROM 650MB liczb losowych: szumy elektroniczne $\oplus$ muzyka rap)

# Liczby pseudolosowe i ich generatory

,,Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

John von Neumann

**Liczby pseudolosowe** – liczby otrzymywane wg. ścisłej formuły matematycznej, ale mające ,,wygląd" liczb losowych, tzn. ich własności statystyczne są bardzo zbliżone do własności liczb prawdziwie losowych (ktoś, kto nie zna formuły, nie jest w stanie odróżnić ich od liczb prawdziwie losowych).

▶ **Źródła liczb pseudolosowych – generatory matematyczne (programowe):**

∗ dobre własności statystyczne generowanych liczb,

∗ wygodne w użyciu (proste, szybkie, . . . ).

→ Praktycznie całkowicie wyparły generatory fizyczne!

Dlatego powszechnie liczby pseudolosowe określa się mianem liczb losowych, a odpowiednie numeryczne algorytmy ich otrzymywania – generatorami liczb losowych.

● **Pierwszy generator matematyczny: generator kwadratowy von Neumanna:**

→ Formuła: $$X_n = \lfloor X_{n-1}^2 \cdot 10^{-m} \rfloor - \lfloor X_{n-1}^2 \cdot 10^{-3m} \rfloor \cdot 10^{2m}$$

gdzie: $X_i$, $m$ – liczby całkowite nieujemne, $X_0$ – stała początkowa,

$\lfloor \cdot \rfloor$ – oznacza część całkowitą liczby.

⇒ Generuje $2m$-cyfrowe sekwencje liczb – niestety krótkie serie, zależne od $X_0$!

- **Typowy schemat konstrukcji generatora:**

  **1)** Ustalić stałe początkowe: $X_0, X_1, \ldots, X_{k-1}$.

  **2)** Jeżeli wygenerowano już $(n-1)$ liczb, to liczbę $X_n$ obliczyć wg. wzoru:

$$X_n = f(X_{n-1}, X_{n-2}, \ldots, X_{n-k}),\ n \geq k.$$

▷ Najczęściej generowane są liczby całkowite lub bity $\Rightarrow$ liczby rzeczywiste o rozkładzie równomiernym na przedziale $[0,1)$, ozn. $U(0,1)$.

- **Okres generatora:**

  Ciągi liczb z generatora matematycznego – ciągi okresowe.

  Niech $P, \nu$ – liczby naturalne, a $X_0, X_1, \ldots$ – ciąg liczb losowych,

$$P \text{ – } \textbf{okres generatora} \text{ (ciągu)} \Leftrightarrow \exists_{\nu,P} : X_i = X_{i+jP}\ (j = 1, 2, \ldots)\ \forall_{i \geq \nu}.$$

  Okres na ogół da się wyznaczyć teoretycznie (choć czasem może to być trudne!).

  ▷ Wymagania co do okresu generatora:

  Jeżeli $N$ – ilość liczb z generatora używanych w obliczeniach, to powinno być:

$$N \ll P.$$

  $\rightarrow$ W praktyce wymaga się: $N \lesssim \sqrt{P}$.

  Popularny ostatnio: Mersenne Twister (Matsumoto & Nishimura, 1998): $P \approx 10^{6000}$

# Postawowe rodzaje generatorów matematycznych

1. **Generatory liniowe:**

   Ogólna postać: $X_n = (a_1 X_{n-1} + a_2 X_{n-2} + \ldots + a_k X_{n-k} + c) \bmod m,$

   gdzie: $a_1, \ldots, a_k, c, m$ – parametry generatora (ustalone liczby całkowite $\geq 0$),

   $a \bmod b$ oznacza resztę z dzielenia liczby $a$ przez liczbę $b$.

   Okres: $P \leq m^k - 1$ (maksymalny tylko dla odpowiednio dobranych parametrów)

   ▷ Popularne implementacje (np. Pascal, język C/C++):

   $$k = 1: \quad X_n = (a X_{n-1} + c) \bmod m$$

   ▶ Podstawowa wada: Wielowymiarowe rozkłady tworzą regularne hiperpłaszczyzny – tzw. ,,efekt Marsaglii".

2. **Generatory SR** (ang. shift-register):

   Dla bitów: $b_n = (a_1 b_{n-1} + \ldots + a_k b_{n-k}) \bmod 2,$

   gdzie: $a_1, \ldots, a_k \in \{0, 1\}$ – stałe binarne.

   Łatwe w implementacji, bo: $(a + b) \bmod 2 = a \operatorname{xor} b$

   $\Rightarrow$ Liczby $U \in [0, 1)$ wg. schematu Tauswortha: $U_i = \sum_{j=1}^{L} 2^{-J} b_{is+j}$, $s \leq L$.

   Okres: $P \leq 2^k - 1$

   ▶ Wady: Nie spełniają nowszych testów statystycznych!

   ▷ Generator Tezuki (1995): Kombinacja 3 generatorów SR, $P \approx 10^{26}$, stat. OK.

▷ Mersenne Twister (Matsumoto & Nishimura): ulepszony SR, $P = 2^{19937} - 1$.

3. **Uogólnione generatory Fibonacciego**:

   Ogólna postać: $X_n = (X_{n-r} \circ X_{n-s}) \bmod m, \quad n \geq r,\ r > s \geq 1,$

   gdzie operator $\circ \in \{+, -, \times, \mathrm{xor}\}$.

   Okres: $P \leq (2^r - 1)\frac{m}{2}$

   Własności statystyczne: najlepsze dla $\times$, najgorsze dla xor.

   ▷ Popularny generator RANMAR (Marsaglia, Zaman, Tsang):

   Kombinacja 2 generatorów, $P \approx 10^{43}$, b. dobre własności statystyczne.

4. **Generatory SWB** (ang. subtract-with-borrow) – Marsaglia & Zaman (1991):

   Schemat: $X_n = (X_{n-r} \ominus X_{n-s}) \bmod m, \quad n \geq r,\ r > s \geq 1,$

   gdzie: $x \ominus y \bmod m = \begin{cases} x - y - c + m & \text{oraz } c = 1 \text{ gdy } x - y - c < 0, \\ x - y - c & \text{oraz } c = 0 \text{ gdy } x - y - c \geq 0, \end{cases}$

   na początku: $c = 0$.

   ▶ Wady: Nie spełniają nowszych testów statystycznych!

   ▷ Generator RCARRY (Marsaglia & Zaman, 1991):

   $P \approx 10^{171}$, prosty i szybki, ale nie spełna niektórych nowszych testów.

5. **Generatory MWC** (ang. multiply-with-carry) – Marsaglia:

   Schemat: $X_n = (a_1 X_{n-1} + a_2 X_{n-2} + \ldots + a_r X_{n-r} + c) \bmod m,$

   gdzie: $c = \lfloor (a_1 X_{n-1} + a_2 X_{n-2} + \ldots + a_r X_{n-r})/m \rfloor$ – tzw. wartość przeniesienia do następnego kroku.

   ▶ Zalety: Proste, szybkie, łatwe w implementacji, mają długie okresy, bardzo dobre własności statystyczne.

   ▷ Kilka przykładów generatorów podanych przez Marsaglię.

6. **Generatory nieliniowe** (od połowy lat 1980):

   ▷ Eichenauer & Lehn: $X_n = (a X_{n-1}^{-1} + b) \bmod m$

   gdzie: $c^{-1}$ = liczba całkowita: $c \cdot c^{-1} \bmod m = 1$, $m$ – liczba pierwsza.

   ▷ Eichenauer-Hermann: $X_n = (a(n + n_0) + b)^{-1} \bmod m$

   → Kolejna wartość $X_n$ może być uzyskiwana niezależnie od poprzednich.

   ▷ L. Blum, M. Blum, Shub: $X_n = X_{n-1}^2 \bmod m$; $m$ – iloczyn liczb pierwszych

   → Zastosowania w kryptologii.

   ▶ Zalety: Dobre własności statystyczne (przechodzą pomyślnie wszystkie testy).

   ▶ Wady: Są dość wolne.

- **Kombinacje generatorów – na ogół dają lepsze wyniki, ale nie zawsze!**

# Jakość generatorów liczb losowych

,,Random number generators should not be chosen at random."

Donald Knuth

Jak sprawdzić czy dany generator jest dobry?

Generator jest dobry, to znaczy, że produkuje sekwencje liczb o własnościach liczb prawdziwie losowych. ← Jak to sprawdzić?

- **Tradycyjne podejście:**

Formułowanie własności dla liczb losowych z rozkładu U(0,1) i testowanie, czy sekwencje liczb z generatora posiadają te własności.

→ Ale takich własności można sformułować nieskończenie wiele ⇒ nieskończenie wiele testów!

▷ W praktyce można jedynie udowodnić, że generator jest zły (nie spełnia któregoś z testów), ale nie da się udowodnić, że jest dobry (to że generator przeszedł pomyślnie $n$ testów nie gwarantuje, że przejdzie pomyślnie $(n+1)$-szy, którym akurat może okazać się nasz rozwiązywany problem!).

Testowanie generatorów – selekcja negatywna. Pozytywne wyniki określonej liczby testów zwiększają jedynie nasze zaufanie do generatora, ale nie gwarantują jego niezawodności.

# Jakość generatorów liczb losowych

▷ Sformułowano wiele rozmaitych wyrafinowanych testów, patrz np. D. Knuth, ,,Sztuka programowania", tom 2, WNT Warszawa, 2002.

→ Np. Bateria testów G. Marsaglii ,,DIEHARD" (http://stat.fsu.edu/~geo/diehard.html)

– przyczyniła się do wyeliminowania wielu generatorów, m.in. fizycznych.

▶ Nie bardzo wiadomo dlaczego formuły rekurencyjne produkują liczby, które wyglądają jak losowe? → Faktycznie, jest to zaskakujące!

● **1993: M. Lüscher – wreszcie teoria generowania liczb losowych! (?)**

Martin Lüscher – fizyk, specjalista od kwantowej teorii pola na siatkach

▷ Artykuł: hep-lat/9309020, Comput. Phys. Commun. **79** (1994) 100:

Operacyjna definicja losowości w sensie wymaganym dla obliczeń Monte Carlo, oparta o chaotyczne zachowanie w klasycznych układach dynamicznych (teorie Kołmogorowa i Arnolda) – eksponenty Lapunowa i entropia Kołmogorowa do badania chaotyczności liczb z generatora.

▶ **Generator RANLUX:** oparty o generator SWB RCARRY Marsaglii i Zamana, z dodanym algorytmem odrzucania pewnych sekwencji liczb – w celu zapewnienia dostatecznej ,,chaotyczności" generowanych liczb losowych.

Okres: $P \approx 10^{171}$. → **Dotąd nie znaleziono żadnego defektu!**