Metody numeryczne
Wykład nr 11
Generatory liczb pseudolosowych

Aneta Wróblewska

UMCS, Lublin

May 28, 2024

Losowość, liczby losowe i pseudolosowe oraz ich wykorzystanie

Losowość

- W naturze, technice, ekonomii i życiu społecznym często spotykamy zjawiska, które wydają się być losowe.
- Trudność w przewidywaniu ich przyszłych zachowań lub określeniu przyczyn wynika z braku informacji, błędów w obserwacji, czy ograniczeń technicznych w dostępie do danych.
- Przyczyny losowości mogą być związane z właściwościami fizycznymi zjawisk lub ich złożonością, uniemożliwiającą modelowanie deterministyczne.

Losowość w matematyce i kryptografii

- Rozkład liczb pierwszych wśród liczb naturalnych.Chociaż można określić średnią częstość ich występowania, dokładne rozmieszczenie liczb pierwszych jest nieznane i wydaje się losowe. Pojawienie się liczby pierwszej w ciągu liczb naturalnych jest trudne do przewidzenia bez szczegółowej analizy.
- Znajdują zastosowanie w statystycznych badaniach reprezentatywnych, kontroli jakości, badaniach rynkowych oraz w naukach eksperymentalnych do planowania eksperymentów.
- Liczby losowe są niezbędne w metodach Monte Carlo, używanych do obliczeń prawdopodobieństw i optymalizacji.
- W kryptografii, liczby losowe pełnią kluczową rolę jako klucze w szyfrach, zwiększając bezpieczeństwo przesyłanych informacji.

Losowość w symulacjach i systemach komunikacyjnych

- Symulacje komputerowe używają liczb losowych do imitacji realnych procesów opisanych równaniami, uwzględniając losowe czynniki wpływające na te zjawiska.
- Badania te często stanowią jedyną metodę ilościowej analizy złożonych procesów, które nie mogłyby być inaczej zbadane.
- Liczby losowe tworzą złudzenie realizmu w grach komputerowych, symulatorach treningowych oraz w grach strategicznych.
- Rozwój technologii komunikacyjnych, jak telefonia komórkowa i sieci komputerowe, intensyfikuje potrzebę skutecznych generatorów liczb losowych.

Ciąg losowy

- **Ciąg** liczbowy nazywamy **losowym**, jeśli nie istnieje krótszy algorytm, który by go opisał niż sam ciąg.
- Z ciągu takiego nie można wywnioskować żadnych reguł umożliwiających jego odtworzenie bez znajomości wszystkich jego elementów.
- Nie jest też możliwe przewidzenie żadnego elementu ciągu na podstawie pozostałych.

Ciągi pseudolosowe

- Mimo, że istnieją reguły opisujące ciąg, nasza niewiedza może uniemożliwić ich identyfikację.
- Taki **ciąg** nazywamy **pseudolosowym** i w wielu przypadkach może być traktowany jak ciąg losowy.
- W sytuacjach praktycznych pseudolosowe ciągi liczb mogą efektywnie symulować losowe zachowania.
- W przeciwieństwie do prawdziwie losowych, liczby pseudolosowe generowane są przez algorytmy deterministyczne.
- Ważne jest, aby generowane ciągi liczb były nieprzewidywalne i miały właściwości statystyczne podobne do ciągów losowych.

Przykłady generatorów ciągów liczb pseudolosowych

- **LCG (Linear Congruential Generator)**: Jedna z najstarszych i najprostszych metod generowania liczb pseudolosowych.
- **Mersenne Twister**: Znany z bardzo długiego okresu i wysokiej jakości generowanych ciągów.
- **Cryptographically Secure Pseudorandom Number Generators (CSPRNG)**: Generatory o wysokiej nieprzewidywalności, stosowane w kryptografii.

Historyczne metody otrzymywania liczb losowych

Wprowadzenie do historycznych metod liczb losowych

- Od dawna istnieje zapotrzebowanie na liczby losowe w badaniach statystycznych.
- Jednymi z pierwszych źródeł liczb losowych były tablice liczb losowych.
- Przykłady historycznych zbiorów liczb losowych obejmują różne techniki ich generacji.

Wczesne tablice liczb losowych

- W 1927 roku L.H. Tippett opublikował pierwszą tablicę losowych cyfr, składającą się z 41600 cyfr pochodzących z danych spisu powszechnego w Wielkiej Brytanii.
- W 1939 roku R.A. Fisher i F. Yates wydali tablicę 15000 losowych cyfr, które zaczerpnęli z cyfr od 15 do 19 z tablic logarytmicznych.
- Kendall, Babington i Smith w tym samym roku zaprezentowali tablicę 100000 cyfr losowych uzyskanych za pomocą "elektrycznej ruletki".

Tablice liczb losowych w Polsce i RAND Corporation

- W 1951 roku w Polsce GUS opracował własną tablicę liczb losowych, korzystając z pasków drukujących maszyn liczbowych.
- W 1955 roku RAND Corporation stworzyła tablicę miliona cyfr losowych, używając impulsów binarnych, co umożliwiało łatwe zastosowanie w obliczeniach komputerowych.
- Wady tablic liczb losowych obejmowały ograniczoną długość i konieczność rozwoju algorytmów do generowania nowych ciągów losowych.

Generowanie ciągów liczb losowych na podstawie tablicy cyfr losowych

1. Wybór losowej pięciocyfrowej liczby z tablicy.
2. Modyfikacja pierwszej cyfry liczby modulo 2. Tak zmieniona liczba pięciocyfrowa wskazuje numer wiersza w tablicy.
3. Zredykowana dwucyfrowego końcówka liczby modulo 50 wskazuje numer kolumny.
4. Proces rozpoczyna się od wybranej pozycji w tablicy, co tworzy losowy ciąg.

Współczesne metody otrzymywania liczb losowych

Generatory fizyczne liczb losowych

- Współczesne metody generacji liczb losowych dzielą się na algorytmiczne i fizyczne.
- Algorytmy matematyczne pozwalają na wielokrotne otrzymanie tego samego ciągu pseudolosowego.
- Generatory fizyczne opierają się na mierzalnych parametrach procesów fizycznych, które zachodzą w sposób losowy.

Generatory fizyczne

Przykłady generatorów fizycznych

- Mechaniczne urządzenia losujące, takie jak moneta, kostka do gry, czy ruletka.
- Licznik Geigera, mierzący promieniowanie jądrowe, które zachodzi losowo.
- Elektroniczne liczniki impulsów, np. z dysków komputerowych, monitorów czy kart dźwiękowych.
- Urządzenia generujące losowe bity z klawiatury lub arytmometru w intensywnie używanych komputerach.
- Specjalnie skonstruowane elektroniczne urządzenia generujące liczby losowe, np. z diod szumowych, często dostępne jako karty komputerowe.

Ważność testowania generatorów fizycznych

- Każdy wygenerowany ciąg liczb losowych musi być testowany przed użyciem, aby zapewnić, że zachowuje on właściwości losowości.
- W przypadku awarii urządzenia, wygenerowane ciągi mogą stracić swoje losowe właściwości.
- Dla celów kryptograficznych, ciągi liczb losowych powinny być zapisywane w dwóch kopiah na zewnętrznych nośnikach danych.

Generatory algorytmiczne

Podstawą algorytmicznego generowania liczb losowych jest uzyskanie ciągu liczb całkowitych z przedziału [1;M] w sposób równomierny.

Z losowych liczb całkowitych $X_i, i = 1, 2, \ldots$ o równomiernym rozkładzie w [1;M] uzyskuje się liczby $R_i, i = 1, 2, \ldots$ o ciągłym rozkładzie równomiernym na [0; 1] przez przekształcenie $R_i = \frac{X_i}{M}$.

Generatory kongruencyjne

Najbardziej znanym sposobem generowania liczb pseudolosowych jest metoda opracowana przez Lehmera w 1951 zwana **liniowym generatorem kongruentnym** (ang. LCG - Linear Congruential Generator). Można wyróżnić dwa podstawowe wzory do obliczania liczb pseudolosowych z wykorzystaniem generatora LCG.

Addytywny LCG:

$$X_{n+1} = a \cdot X_n + c \mod M$$

Multiplikatywny LCG:

$$X_{n+1} = a \cdot X_n \mod M$$

gdzie:

- X_n - n-ta liczba psudolosowa
- a - mnożnik
- c - parametr, dla generatora multiplikatywnego $c = 0$
- M - ilość generowanych liczb

Niech N będzie okresem generatora. Ważne informacje dotyczące długości generatora liniowego zawarte są w następujących twierdzeniach:

Twierdzenie 1

Maksymalny okres generatora liniowego wynosi $N = 2^{m-2}$, gdy $M = 2^m$ dla $m \geqslant 3$ i $a \equiv 3 \mod 8$ lub $a \equiv 5 \mod 8$.

Twierdzenie 2

Dla $M = p$, gdzie p jest liczbą pierwszą, generator liniowy posiada maksymalny okres równy p. Ten okres jest osiągany, gdy a jest pierwiastkiem pierwotnym liczby p.

Generatory kongruencyjne

- Liczby generowane przez addytywny LCG (Linear Congruential Generator) mogą przyjmować wartości z przedziału od 0 do $M - 1$.
- Multiplikatywny LCG generuje wartości z przedziału od 1 do $M - 1$.
- Po M wykonaniach, wartości zaczynają się powtarzać, co wynika z własności działania modulo.

Konfiguracja generatora liczb losowych

- Konieczne jest podanie wartości oznaczającej ziarno (ang. *seed*), które jest początkową wartością X_0.
- Aby generator działał poprawnie, wartości a, c i M nie mogą być przypadkowe.

Zasady doboru parametrów

- Wartość zmiennej M jest o jeden większa od największej wartości losowej, jaką algorytm będzie mógł wygenerować. Często wybiera się tutaj potęgi 10 albo 2,
- Parametr c musi być względnie pierwszy z M.
- Wartość wyrażenia $a - 1$ jest wielokrotnością wszystkich dzielników zmiennej M. Wartość a nie powinna być zbyt duża, ale też nie za mała - dobrym wyborem jest tutaj liczba o jeden rząd mniejsza (o jedną cyfrę krótsza) od M,

D.E.Knuth pokazał, iż niektóre wartości mogą prowadzić do wygenerowania bardzo krótkich cykli, co oczywiście jest wysoce niepożądane. Na przykład dla $b = 19$, $M = 381$ oraz $X_1 = 0$ otrzymamy sekwencję: $0, 1, 20, 0, 1, 20, \cdots$

- Główną wadą generatorów liniowych jest ich przewidywalność - punkty o współrzędnych (X_n, X_{n+1}) układają się na linii prostej.

Popularne generatory liczb losowych

Nazwa	**M**	**a**	**c**
Numerical Recipes	2^{32}	1664525	1013904223
Borland C/C++	2^{32}	22695477	1
GNU Compiler Collection	2^{32}	69069	5
ANSI C	2^{32}	1103515245	12345
Borland Delphi, Virtual Pascal	2^{32}	134775813	1
Microsoft Visual/Quick C/C++	2^{32}	214013	2531011
ANSIC	2^{31}	1103515245	12345
MINSTD	$2^{31} - 1$	16807	0

Table: Wybrane generatory liczb losowych

Przykład generatora addytywnego LCG

- Parametry generatora: $a = 4,\ c = 2,\ M = 9,\ X_0 = 0.$
- Wzór:

$$X_n = (4 \cdot X_{n-1} + 2) \mod 9$$

Rozwiązanie - kolejne wartości X_n

- $X_0 = (4 \cdot 0 + 2) \mod 9 = 2$
- $X_1 = (4 \cdot 2 + 2) \mod 9 = 1$
- $X_2 = (4 \cdot 1 + 2) \mod 9 = 6$
- $X_3 = (4 \cdot 6 + 2) \mod 9 = 8$
- $X_4 = (4 \cdot 8 + 2) \mod 9 = 7$
- $X_5 = (4 \cdot 7 + 2) \mod 9 = 3$
- $X_6 = (4 \cdot 3 + 2) \mod 9 = 5$
- $X_7 = (4 \cdot 5 + 2) \mod 9 = 4$
- $X_8 = (4 \cdot 4 + 2) \mod 9 = 0$
- $X_9 = (4 \cdot 0 + 2) \mod 9 = 2$
- $X_{10} = (4 \cdot 2 + 2) \mod 9 = 1$
- $X_{11} = (4 \cdot 1 + 2) \mod 9 = 6$
- $X_{12} = (4 \cdot 6 + 2) \mod 9 = 8$
- $X_{13} = (4 \cdot 8 + 2) \mod 9 = 7$
- $X_{14} = (4 \cdot 7 + 2) \mod 9 = 3$
- $X_{15} = (4 \cdot 3 + 2) \mod 9 = 5$
- $X_{16} = (4 \cdot 5 + 2) \mod 9 = 4$
- $X_{17} = (4 \cdot 4 + 2) \mod 9 = 0$

Generator Lehmera

Generator Lehmera, nazywany czasami generatorem liczb losowych Parka-Millera (od Stephena K. Parka i Keitha W. Millera), to rodzaj Liniowego Generatora Kongruencyjnego (LCG), który operuje w multiplikatywnej grupie modulo M. Jego działanie określa wzór:

$$X_{k+1} = a \cdot X_k \mod M,$$

gdzie:

- M – liczba pierwsza lub potęga liczby pierwszej,
- a – element mający wysoki rząd modulo M (np. pierwiastek pierwotny),
- X_0 – ziarno, liczba względnie pierwsza z M.

Generator Lehmera

W 1988 roku, Park i Miller zasugerowali wartości $M = 2^{31} - 1$ (liczba Mersenne'a M_{31}) i $a = 7^5 = 16807$ (pierwiastek pierwotny modulo M_{31}), obecnie znane jako MINSTD. Mimo krytyki, te parametry pozostały w użyciu do dzisiaj, włącznie z zastosowaniami w CarbonLib i minstd_rand0 z C++11. Park, Miller i Stockmeyer zasugerowali później użycie mnożnika $a = 48271$ zamiast 16807.

Generator Lehmera

W Sinclair ZX-81 i jego następcach użyto wartości $M = 2^{16} + 1$ (liczba Fermata F_4) i $a = 75$. CRAY użył generatora Lehmera z $M = 2^{48} - 1$ i $a = 44485709377909$. GNU Scientific Library zawiera kilka generatorów Lehmera, w tym MINSTD i RANF.

Generator Lehmera, choć jest specjalnym przypadkiem LCG (z $c = 0$), posiada unikalne ograniczenia i właściwości. W szczególności, ziarno X_0 musi być względnie pierwsze do M. Maksymalny okres generatora Lehmera, kiedy M jest liczbą pierwszą i a jest pierwiastkiem pierwotnym, równa się $M - 1$.

Uogólniony generator liniowy

- Uogólnienie generatora liniowego polega na uwzględnieniu kilku poprzednich wartości przy obliczaniu bieżącej wartości:

$$X_n = a_1 X_{n-1} + \ldots + a_k X_{n-k} + b \mod M$$

- Przy odpowiednim doborze stałych $a_1, \ldots, a_k, b < M$, generator ten może osiągnąć maksymalny okres równy M.
- Uogólniony generator liniowy, mimo większej złożoności, nadal nie nadaje się do zastosowań kryptograficznych.

Generator Fibonacciego jest jednym z odmian uogólnionego generatora liniowego liczb losowych, który opiera się na ciągu Fibonacciego:

$$X_M = X_{M-1} + X_{M-2} \mod M \quad (M \geqslant 2)$$

Ten rodzaj generatora charakteryzuje się lepszymi parametrami jakościowymi niż inne generatory liniowe, jednak jego główną wadą są duże korelacje między wyrazami ciągu, co oznacza, że ciągi te spełniają warunek rozkładu, ale nie warunek niezależności.

Lagged Fibonacci Generator

Uogólnienie generatora Fibonacciego

W celu zmniejszenia korelacji, można uogólnić generator Fibonacciego do postaci zwaną "lagged Fibonacci generator":

$$X_M = X_{M-p} + X_{M-q} \mod M \quad (M \geqslant p, p > q \geqslant 1)$$

gdzie p i q są opóźnieniami generatora. Generator taki można modyfikować zastępując działanie dodawania innymi operacjami, takimi jak odejmowanie, mnożenie, XOR:

$$X_M = X_{M-q} \diamond X_{M-p} \mod M \quad (M \geqslant p, p > q \geqslant 1),$$

gdzie $\diamond$ reprezentuje wybrany operator.

Przykład zastosowania generatora

$$m = 17, \quad p = 3, \quad q = 1, \quad X_0 = 7, \quad X_1 = 16, \quad X_2 = 5$$

$$X_M = X_{M-q} + X_{M-p} \mod m$$

$$\begin{aligned} X_3 &= X_0 + X_2 \mod 17 = 7 + 5 \mod 17 = 12, \\ X_4 &= X_1 + X_3 \mod 17 = 16 + 12 \mod 17 = 11, \\ X_5 &= X_2 + X_4 \mod 17 = 5 + 11 \mod 17 = 16. \end{aligned}$$

Kolejne wartości ciągu: 7, 16, 5, 12, 11, 16, 11, 5, 4, 15, 3, 7,...

Zaawansowane uogólnienia

Generator Fibonacciego można rozbudować o więcej poprzednich punktów:

$$X_M = X_{M-p_1} \diamond X_{M-p_2} \diamond \ldots \diamond X_{M-p_k} \mod M \quad (M \geqslant p_k > \ldots > p_1 \geqslant 1)$$

Przykłady znanych generatorów tego typu to generator Marsagli (Marsa-LFIB4) i generator Ziffa (Ziff98), które wykorzystują różne operacje i mają różne parametry p_i.

Kwadratowy generator kongruencyjny

Aby uniknąć liniowej zależności kolejnych wartości ciągu, zastosowano zależność kwadratową:

$$X_{n+1} = aX_n^2 + bX_n + c \mod M$$

Dla odpowiednich wartości $a, b, c, X_0 < M$, maksymalny okres tego generatora może być równy M.

Generator wykorzystujący wielomiany permutacyjne

Generator wykorzystuje wielomiany permutacyjne postaci:

$$g(X) = \sum_{k=0}^{n} a_k X^k \quad a_1, \ldots, a_n \in \{0, 1, \ldots, M-1\}$$

Maksymalny okres generatora jest równy M. W przeciwieństwie do generatora liniowego, nie jest przewidywalny.

Generator inwersyjny

Generator inwersyjny, używany do generowania liczb losowych o równomiernym rozkładzie, wykorzystuje wzór:

$$X_{n+1} = \begin{cases} (aX_n^{-1} + b) \mod p & \text{dla } X_n \neq 0; \\ b & \text{dla } X_n = 0; \end{cases}$$

gdzie p jest liczbą pierwszą. Maksymalny okres generatora, przy odpowiednich wartościach a i b, może wynosić $p - 1$.

Metody testowania generatorów

Aby ocenić, czy przyjąć, czy odrzucić generator lub wygenerowany ciąg bitów stosujemy w praktyce zestawy odpowiednich testów statystycznych. Testy te powinny potwierdzać:

- równomierny rozkład ciągu bitów,
- losowość rozkładu,
- niezależność kolejnych bitów.

W literaturze dostępnych jest wiele testów do oceny generatorów liczb losowych. Można wyróżnić testy ogólne, dotyczące rozkładów liczb całkowitych, oraz specyficzne, skoncentrowane na ciągach binarnych. Zestawy testów powinny być kompletne, ale także wystarczająco ograniczone, aby były wykonalne w praktyce podczas użytkowania generatora.

Testy według normy FIPS-140-3

Praktyczne zastosowanie testów

Weryfikacja generatorów często korzysta ze standardów amerykańskich, np. normy FIPS-140-3, która określa cztery podstawowe testy dla ciągów długości 20 000 bitów:

1. Test monobitowy - sprawdza, czy liczba jedyneczek jest w określonych granicach.
2. Test pokerowy - bada równomierność rozkładu segmentów czterobitowych.
3. Test serii - analizuje ciągłość serii takich samych bitów.
4. Test długich serii - sprawdza, czy występują zbyt długie serie jednego bitu.

Norma FIPS-140-3 stawia wysokie wymagania dla akceptacji ciągów bitów, wymagając, aby poziom istotności testów (prawdopodobieństwo odrzucenia ciągu bitów mogącego pochodzić z prawidowego źródła) wynosił $0,0001$.

Test monobitowy

Test monobitowy jest podstawowym testem statystycznym używanym do sprawdzania generatorów liczb losowych. Jego głównym celem jest ocena, czy liczba jedynkowych bitów ('1') w wygenerowanym ciągu bitów leży w granicach, które są statystycznie prawdopodobne dla równomiernie losowego ciągu. Test wykorzystuje statystykę chi-kwadrat do oceny, czy zaobserwowane odchylenie liczby jedynek od oczekiwanej średniej jest statystycznie istotne.

Test monobitowy

Procedura testu monobitowego składa się z kilku kroków:

1. Obliczenie liczby jedynek w ciągu 20000 bitów:

$$X = \text{liczba jedynek w ciągu 20000 bitów}$$

2. Ocena, czy wygenerowany ciąg można uznać za akceptowalny, na podstawie kryterium:

$$9725 < X < 10275$$

Jeżeli liczba jedynkowych bitów spełnia ten warunek, ciąg jest uznawany za zgodny z oczekiwaniami dla równomiernego rozkładu.

Test pokerowy

Test pokerowy to metoda statystyczna używana do sprawdzania generatorów liczb losowych, zwłaszcza w kontekście ich zdolności do produkcji ciągów bitów, które nie wykazują widocznych wzorców i są równomiernie rozłożone. Test skupia się na analizie segmentów czterobitowych w ciągu bitów.
Jest to krytyczne dla zastosowań, które wymagają wysokiej jakości losowości, takich jak algorytmy kryptograficzne.

Test pokerowy

Procedura testu pokerowego obejmuje następujące kroki:

1. Dzielenie ciągu 20000 bitów na 5000 segmentów czterobitowych.
2. Liczenie częstości pojawienia się każdego z 16 możliwych czterobitowych segmentów (od 0000 do 1111).
3. Obliczanie statystyki chi-kwadrat z wyników częstości:

$$X = \frac{16}{5000}\left(\sum_{i=0}^{15}[f(i)]^2\right) - 5000$$

   gdzie $f(i)$ to liczba wystąpień i-tego czterobitowego segmentu.
4. Ocena zgodności rozkładu segmentów z oczekiwanym rozkładem równomiernym, gdzie zakres akceptowalnych wartości statystyki X wynosi:

$$2.16 < X < 46.17$$

Test serii

Test serii ocenia, czy liczba serii różnej długości w ciągu bitów jest zgodna z oczekiwaniami dla ciągu bitów o równomiernym rozkładzie losowości. Serią nazywamy ciąg kolejnych bitów o tej samej wartości.
Test serii pozwala zidentyfikować ciągi bitów, które mogą nie wykazywać oczekiwanej losowości ze względu na nieodpowiednią liczbę lub rozkład długości serii. Jest to kluczowe dla zastosowań kryptograficznych i innych, gdzie jakość losowości bezpośrednio wpływa na bezpieczeństwo i efektywność systemów.

Test serii

Test serii skupia się na analizie długości serii bitów w wygenerowanym ciągu:

1. Podział ciągu 20000 bitów na serie jednorodnych bitów (1 lub 0).
2. Liczenie liczby serii o różnej długości. Specyficzne kategorie długości to 1, 2, 3, 4, 5 oraz 6 i więcej.
3. Porównanie wyników z teoretycznymi wartościami oczekiwanymi dla idealnie losowego ciągu bitów. Akceptowalne zakresy dla liczby serii różnych długości są następujące:
   - Długość serii 1: 2343 do 2657
   - Długość serii 2: 1135 do 1365
   - Długość serii 3: 542 do 708
   - Długość serii 4: 251 do 373
   - Długość serii 5: 111 do 201
   - Długość serii 6 i więcej: 111 do 201

Różne rodzaje testów statystycznych

W literaturze można znaleźć różne grupy testów, które badają rozmaite cechy badanej populacji oraz warunki eksperymentu losowego, w tym:

- Testy losowości, które określają, czy ciąg wyników może być traktowany jako ciąg zmiennych losowych.
- Testy zgodności, które określają, czy obserwacje mają ten sam rozkład prawdopodobieństwa.
- Testy normalności, sprawdzające, czy dane pochodzą z rozkładu normalnego.
- Testy dotyczące parametrów rozkładu, potwierdzające wiarygodność estymowanych parametrów.
- Testy niezależności, które określają, czy zmienne są niezależne.

Podsumowanie - kryteria dobrego generatora

- **Okres**: Idealny generator miałby okres równy lub zbliżony do maksymalnej możliwej liczby unikatowych stanów, co oznacza, że ciąg wartości nie powtarza się przez bardzo długi czas.
- **Równomierność**: Równomierność odnosi się do równego prawdopodobieństwa każdej z możliwych wartości w zakresie generatora. Równomierność generatora można ocenić za pomocą testów statystycznych, takich jak test chi-kwadrat, które pozwalają określić, czy rozkład generowanych liczb jest bliski idealnemu rozkładowi równomiernemu.
- **Nieprzewidywalność**: Nieprzewidywalność oznacza, że nie jest możliwe efektywne przewidzenie kolejnych wartości w ciągu na podstawie jakiejkolwiek skończonej liczby wcześniejszych wartości. Nieprzewidywalność można oceniać przez analizę korelacji i innych wzorców w wygenerowanych ciągach. Testy takie jak test następnego bitu są przykładami metod oceny tego kryterium.