

# Pseudozufallszahlengeneratoren

---

**Gerrit Blöss**

Fachbereich Informatik der Universität Hamburg

Vogt-Kölln-Straße 30

Hamburg, Germany

[5bloess@informatik.uni-hamburg.de](mailto:5bloess@informatik.uni-hamburg.de)

# Pseudozufallszahlengeneratoren

---

- Anwendungsbereiche & Anforderungen
- Kongruenzgeneratoren
- Test-Verfahren
- Sicherheitslücken
- Ausblick
- Fazit
- Referenzen

# Überblick

---

- **Anwendungsbereiche & Anforderungen**
- Kongruenzgeneratoren
- Test-Verfahren
- Sicherheitslücken
- Ausblick
- Fazit
- Referenzen

# Anwendungsbereiche & Anforderungen

---

- Kryptographische Anwendungen
- Monte-Carlo-Algorithmen & Optimierungsverfahren
- Spiele

# Überblick

---

- Allgemeines
- **Kongruenzgeneratoren**
  - Lineare Kongruenzgeneratoren
  - Multiplikative Kongruenzgeneratoren
  - Additive Kongruenzgeneratoren
  - Fibonacci-Generator
- Test-Verfahren
- Sicherheitslücken
- Ausblick
- Fazit
- Referenzen

# Lineare Kongruenzgeneratoren

---

$$x_{i+1} \equiv a \cdot x_i + c \pmod{M}$$

- Kongruenz bezüglich  $M$  „verschleiert“ Spuren der Erzeugung
- $x_{i+1}$ : erzeugte Zufallszahl
- $x_i$ : „Saat“
- $a$  und  $c$  für möglichst gute Ergebnisse gewählt
- $M$  meist Registergröße des Zielsystems

# Beispiel eines linearen Kongruenzgenerators

---

$$x_{i+1} \equiv 24298 \cdot x_i + 99991 \pmod{199017}$$

- Verwendet von Texas Instruments für TI-59
- Die ersten 20 Zahlen mit  $x_0 = 1$

124289; 190155; 107509; 56531; 74895; 87253; 45284;  
45630; 94024; 179000; 124473; 83596; 148097; 134520;  
11743; 41027; 97884; 33256; 145259; 36678

# Multiplikative Kongruenzgeneratoren

---

$$x_{i+1} \equiv a \cdot x_i + c \pmod{M}$$

- Mit  $c = 0$  (Wegfall der additiven Komponente)
- Erhöht Geschwindigkeit der Erzeugung minimal
- Kann Qualität der erzeugten Zahlen verringern

# Additive Kongruenzgeneratoren

---

$$x_{i+1} \equiv a \cdot x_i + c \pmod{M}$$

- Mit  $a = 1$  (Wegfall der multiplikativen Komponente)
- Erzeugte Zahlen von geringer Qualität, meist unbrauchbar

# Fibonacci-Algorithmus

---

$$x_{i+1} = x_i + x_{i-d} \bmod M$$

- Für tatsächliche Fibonacci-Zahlen:  $d = 1$
- Meist unzureichende Qualität der Zahlen

# Überblick

---

- Allgemeines
- Kongruenzgeneratoren
- **Test-Verfahren**
  - Typen & Bewertungskriterien
  - Chi-Quadrat-Test
  - Runs-Test
- Sicherheitslücken
- Ausblick
- Fazit
- Referenzen

# Test-Verfahren: Typen & Bewertungskriterien

---

- Theoretische und empirische Verfahren
- Kriterien:  
**Zufälligkeit, Verteilung, Unabhängigkeit**

# Chi-Quadrat-Test

---

- Zahlenmenge in Kategorien aufteilen
- Prüfen, wie nahe Anzahl von Werten in jeder Kategorie am Erwartungswert ist

# Chi-Quadrat-Test: Kategorien

---

$$\frac{s-1}{k} < x_i \leq \frac{s}{k}$$

- **s**: Kategorie
- **k**: Anzahl der Kategorien
- **$x_i$** : Zuzuteilende Zahl

# Chi-Quadrat-Test: Berechnung der Gleichverteilung

---

$$\chi^2 = \sum_{s=1}^k \frac{(y_s - n \cdot p_s)^2}{n \cdot p_s}$$

- **$n$** : Anzahl der erzeugten Zahlen
- **$p_s$** : Wahrscheinlichkeit, dass  $x_j \in s$
- **$n \cdot p_s$** : Erwartete Anzahl von Zahlen in der Kategorie  $s$
- **$y_s$** : Tatsächliche Anzahl von Zahlen in  $s$
- Je näher  $\chi^2$  an 0, desto bessere Gleichverteilung
- Kann nicht als einziger Test auf Qualität hindeuten!

# Runs-Test

---

- Betrachten aufeinander folgender gleicher Ereignisse
- Meistens:
  - Run-up-Test:  
gleich bleibende / aufsteigende Zahlenfolgen
  - Run-down-Test:  
abfallende Zahlenfolgen

# Runs-Test: Berechnung

---

$$p = \frac{r}{(r + 1)!}$$

- ***r***: Länge des Runs
- ***p***: Wahrscheinlichkeit, dass der Run auftritt

# Runs-Test mit Chi-Quadrat-Test

---

- Chi-Quadrat-Test mit beim Run-Test gewonnenen Zahlen
- Bessere Aussagekraft über Zufälligkeit der erzeugten Zahlen
- Allgemein: **Testkombinationen führen zu brauchbareren Ergebnissen**

# Überblick

---

- Allgemeines
- Kongruenzgeneratoren
- Test-Verfahren
- **Sicherheitslücken**
  - Mögliche Sicherheitslücken
  - Das geknackte Pokerspiel
- Ausblick
- Fazit
- Referenzen

# Sicherheitslücken

---

- Sicherheitslücken beim Generator:
  - Wiederholung der Zahlen nach geringer Periode
  - Vorhersagbarkeit von Zahlen
- Sicherheitslücke bei der Verwendung:
  - Unsichere/vorhersagbare Saat

# Das geknackte Pokerspiel

---

- Pokervariante mit öffentlichen Karten
- Möglichkeiten, Blatt mit 52 Karten anzuordnen:  **$2^{226}$**
- Saat nur 32-Bit-Zahl  
→ Reduktion auf  **$2^{32}$**  Möglichkeiten
- Saat: verstrichene Millisekunden seit 0 Uhr  
→ nur ca.  **$2^{27}$**  Möglichkeiten
- Synchronisation mit Server-Uhr  
→ Prüfung von **200.000** Saat-Werten ausreichend
- Durch Kenntnis einiger Karten per Brute Force Erraten des gesamten Blatts möglich

# Überblick

---

- Allgemeines
- Kongruenzgeneratoren
- Test-Verfahren
- Sicherheitslücken
- **Ausblick**
- Fazit
- Referenzen

# Ausblick: Mersenne Twister

---

- 1998 veröffentlichter Generator
- Wiederholung erst nach  $2^{19937}-1$  Werten
- Gute Gleichverteilung
- Hohe Geschwindigkeit

# Überblick

---

- Allgemeines
- Kongruenzgeneratoren
- Test-Verfahren
- Sicherheitslücken
- Ausblick
- **Fazit**
- Referenzen

# Fazit

---

- Kongruenzgeneratoren:  
ausreichende Sicherheit für viele Anwendungen
- Falsche Anwendung von Generatoren führt zu  
Unsicherheiten

# Referenzen

---

- Sohst, F. *Verschlüsselung mit dem RSA-Verfahren*. Universität Hamburg, Deutschland, 2005.
- Heppner, C. *Tabu-Search – Übersicht / Einführung in eine moderne Meta-Heuristik*. Universität Hamburg, Deutschland, 2005.
- Gess, J. L. *A Survey of Random Number Generation Theory*. Naval Air Development Center, Warminster, Pennsylvania, USA, 1971.
- Götz, S., Reichel, H.-C., Müller, R. et. al. *Mathematik-Lehrbuch 6*. öbvhpt Verlagsgesellschaft, Wien, 2005.
- Knuth, D. E. *The Art of Computer Programming, 3rd edition, volume 2: Seminumerical Algorithms*. Addison-Wesley, Reading, Massachusetts, USA, 1997. <http://itmanagement.earthweb.com/entdev/article.php/616221>, am: 22.01.2006
- Arkin, B., Hill, F., Schmid, M. et. al. *How we Learned to Cheat in Online Poker: A Study in Software Security*. Software Security Group, 1999.
- Matsumoto, M. und Nishimura, T. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. In: *ACM Transactions on Modeling and Computer Simulations, Vol. 8, 3–30*, 199