Sitzung 7: Zufall

Reenactment Historischer Computer
(Sommersemester 2022)

Nils Reiter
nils.reiter@uni-koeln.de

17. Mai 2022

INSTITUT FÜR
DIGITAL HUMANITIES
UNIVERSITÄT ZU KÖLN

CRETA
CENTER FOR REFLECTED TEXT ANALYTICS

Today

1. Real randomness
2. Pseudo randomness in computers
3. Randomness by Lutz on a Z22

Motivation: Why do Computers Need Random Numbers?

- Cryptography: Generate numbers that cannot be predicted
- Statistics: Select $n\%$ of the items
  - Without pre-determining which ones!
- Video games: Make enemy behaviour less predictable
- ...

Sources for Real Randomness

- Some physical phenomena show some random behaviour
  - Atmospheric noise, based on lightning strikes in the atmosphere (ca. 40 lightning flashes per second)
    - We cannot predict when and where exactly a lightning strikes
  - Radioactive decay
    - Iodine-123 halves its mass in 13 hours, but we don't know which atom decays when
  - Mouse movement in a computer
    - Less random, because some areas are used more frequently than others
  - Throwing dice
    - Mostly random, but assumptions apply

Real Randomness in a Computer

`https://www.random.org/`

- Randomness via atmospheric noise
- Distributed radios pick up noise
- Web page provides access and statistics

Real Randomness in a Computer

`https://www.random.org/`

- Randomness via atmospheric noise
- Distributed radios pick up noise
- Web page provides access and statistics
- Could someone affect the numbers by broadcasting a radio signal?
  - Theoretically, yes. Practically, no.

Real Randomness in a Computer

`https://www.random.org/`

- Randomness via atmospheric noise
- Distributed radios pick up noise
- Web page provides access and statistics
- Could someone affect the numbers by broadcasting a radio signal?
  - Theoretically, yes. Practically, no.
- Why is there a limit to the amount of randomness I can use per day?
  - Generating true random numbers takes time
  - Each radio: ca. $12\,000\ \frac{\text{bit}}{\text{s}}$
  - If all bits are used, one has to wait

True

www.amazon.de/TrueRNG-V3-Hardware-Random-Gen

amazon.de | Computers & Accessories | Hello, Nils Account & Lists | Returns & Orders | Discover Prime | 0 Shopping-Basket

Deliver to Nils Dortmund 44141 | Cyber Monday Sale | AmazonBasics | Prime Video | Nils's Amazon | Cyber Monday Ends in 12 : 16 : 26 HOURS MINUTES SECONDS

Computers | Deals | PCs & Laptops | Tablets | PCs | PC Gaming | Computer Accessories | Computer Components | Monitors | Printers | Best Sellers

Computer & Accessories › Accessories › Cables & Accessories › Cables › USB Cables

Truerng V3 – USB Hardware Zufallszahlengenerator

Brand: ubld.it

4.4 out of 5 stars 25 ratings

Our Price: €97.44 & **FREE Delivery**
Prices for items sold by Amazon include temporarily reduced VAT. Depending on your delivery address, VAT may vary at Checkout. For other items, please see details. Information on the reduced VAT in Germany.

New (4) from €97.44 + FREE Shipping

- High performance Speed: > 400 kilobit/second
- Improved Internal Whitening
- Native support for Windows and Linux

€97.44
& **FREE Delivery**

Arrives: **Dec 18 - 31** Details

Fastest delivery: **Dec 3 - 5** Details

In stock.

Quantity: 1

Add to Basket

Buy Now

Secure transaction

Dispatched from and sold by Boss Bross. For Returns, please check the seller link.

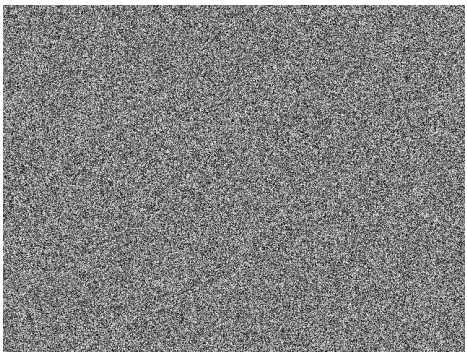
TrueRNG

- `https://ubld.it/products/truerng-hardware-random-number-generator/`
- Source of randomness: Avalanche effect in a semiconductor junction
  - "Avalanche noise is caused by the behavior of electrons in a reverse biased diode"

    Lampert et al. (2016, 2)
- Read speed: $400\ \frac{\text{kbit}}{\text{s}}$

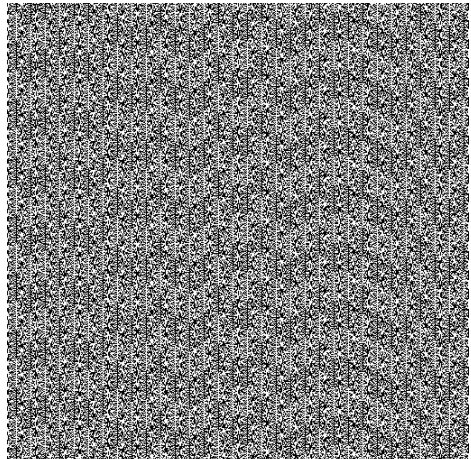
TrueRNG

- ▶
- ▶
- ▶



true randomness from random.org

g-hard
t in a s
avior o



16, 2)

php's rand() function

Pseudo Random

- Computers don't have random numbers (because they are deterministic machines)

Pseudo Random

- Computers don't have random numbers (because they are deterministic machines)
- Computers: Pseudo-random numbers
  - Start with a seed value x_0
  - Function to generate x_{n+1} from x_n
    - This is a research area on its own

Pseudo Random

- Computers don't have random numbers (because they are deterministic machines)
- Computers: Pseudo-random numbers
  - Start with a seed value x_0
  - Function to generate x_{n+1} from x_n
    - This is a research area on its own
- Possible seed value sources
  - Mouse movement (not really random ...)
  - Memory allocation
  - Fixed number (0, 23, 42, ...)

Pseudo Random

- Computers don't have random numbers (because they are deterministic machines)
- Computers: Pseudo-random numbers
  - Start with a seed value x_0
  - Function to generate x_{n+1} from x_n
    - This is a research area on its own
- Possible seed value sources
  - Mouse movement (not really random ...)
  - Memory allocation
  - Fixed number (0, 23, 42, ...)
- Why is it useful to set the seed to a fixed value?
  - Predictability of lab conditions: Reproducibility of results
  - Not used when models are applied productively

Pseudo Random

- Computers don't have random numbers (because they are deterministic machines)
- Computers: Pseudo-random numbers
  - Start with a seed value x_0
  - Function to generate x_{n+1} from x_n
    - This is a research area on its own
- Possible seed value sources
  - Mouse movement (not really random ...)
  - Memory allocation
  - Fixed number (0, 23, 42, ...)
- Why is it useful to set the seed to a fixed value?
  - Predictability of lab conditions: Reproducibility of results
  - Not used when models are applied productively

```
1 import random
2
3 random.seed(5)
4 for i in range(10):
5  print(random.randint(0,1000000))
```

Pseudo Random

- Computers don't have random numbers (because they are deterministic machines)
- Computers: Pseudo-random numbers
  - Start with a seed value x_0
  - Function to generate x_{n+1} from x_n
    - This is a research area on its own
- Possible seed value sources
  - Mouse movement (not really random ...)
  - Memory allocation
  - Fixed number (0, 23, 42, ...)
- Why is it useful to set the seed to a fixed value?
  - Predictability of lab conditions: Reproducibility of results
  - Not used when models are applied productively

```
1 import random
2
3 random.seed(5)
4 for i in range(10):
5  print(random.randint(0,1000000))
```

```
653159
267853
777820
375951
833820
723985
988230
882388
775839
683704
```

Lutz' Pseudo Random Numbers Generator

```
1700 B5
1701 T1712
1702 B1713
1703 LLA0
1704 LLA0
1705 A1713
1706 RA0
1707 RA0
1708 RA0
1709 RA0
1710 U1713
1711 CI15
1712 0
1713 12345678'

1720 F1700
1721 U14
```

E 1720E

Lutz' Pseudo Random Numbers Generator

```
1700 B5
1701 T1712
1702 B1713
1703 LLA0
1704 LLA0
1705 A1713
1706 RA0
1707 RA0
1708 RA0
1709 RA0
1710 U1713
1711 CI15
1712 0  'E1721
1713 12345678'

1720 F1700
1721 U14
```

	a	b	c	5
0		E1700		E1721
1		B5	E1701	
2	E1721	E1701		
3	E1721	T1712	E1702	
4	0	E1702		
5	0	B1713	E1703	
6	12345678	E1702		
7	"	LLA0	E1704	
8	49382712	E1704		
9	"	LLA0	E1705	
10	197530848	E1705		
11	"	A1713	E1706	
12	205 876 576	E1706		
13		...		
14	13117782	E1710		
15	"	U1713	E1711	

Lutz' Pseudo Random Numbers Generator

```
1700 B5
1701 T1712
1702 B1713
1703 LLA0
1704 LLA0
1705 A1713
1706 RA0
1707 RA0
1708 RA0
1709 RA0
1710 U1713
1711 CI15
1712 0
1713 12345678
1720 F1700
1721 U14
```

	a	b	c	5
15		E1700		E1721
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				

```
B5
T1712
B1713
LLA0
LLA0
A1713
RA0
RA0
RA0
RA0
U1713
CI15
0
12345678'

F1700
U14
```

	a	b	c	5	1712
1		E1700		E1721	0
2		B5	E1701	E1721	0
3	E1721	E1701		E1721	0
4	E1721	T1712	E1702	E1721	0
5	0	E1702		E1721	E1721
6	0	B1713	E1703	E1721	E1721
7	12345678	E1703		E1721	E1721
8	12345678	LLA0	E1704	E1721	E1721
9	49382712	E1704		E1721	E1721
10	49382712	LLA0	E1705	E1721	E1721
11	197530848	E1705		E1721	E1721
12	197530848	A1713	E1706	E1721	E1721
13	209876526	E1706		E1721	E1721
14	209876526	RA0	E1707	E1721	E1721
15	104938263	E1707		E1721	E1721
16	104938263	RA0	E1708	E1721	E1721
17	52469131	E1708		E1721	E1721
18	52469131	RA0	E1709	E1721	E1721
19	26234565	E1709		E1721	E1721
20	26234565	RA0	E1710	E1721	E1721
21	13117282	E1710		E1721	E1721
22	13117282	U1713	E1711	E1721	E1721
23	13117282	E1711		E1721	E1721
24	13117282	CI15	E1712	E1721	E1721
25	2	E1712		E1721	E1721
26	2	E1721	E1713	E1721	E1721

Randomness in Lutz' Program

$$
\begin{aligned}
x_0 &= 12345678 \\
x_{n+1} &= (x_n \ll 4) + x_n \gg 4 \\
&= (16x_n + x_n) \gg 4 \\
r_n &= x_{n+1} \wedge 16 \; 15
\end{aligned}
$$

```
1  class LutzRandom:
2    def __init__(self):
3      self.x = 12345678
4
5    def nextnumber(self):
6      x1 = ((self.x << 4) + self.x) >> 4
7      self.x = x1 & (2**38-1)
8      return (x1 & 15)
```

- Simple algorithm to generate pseudo random numbers
- Internal value increases continuously – 38bit overflow quickly
  - First overflow after 200 numbers

Randomness in Lutz' Program

Evaluation