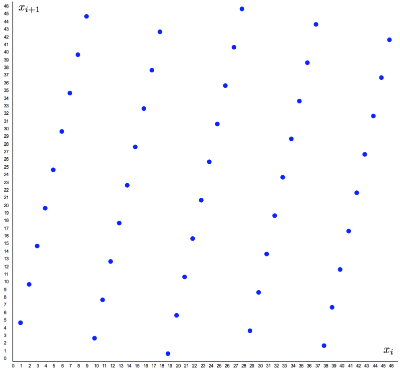
Linear Congruential Generators



Linear Congruential Generators

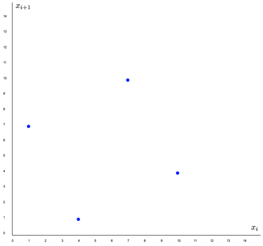
- A linear congruential generator produces a sequence of integers x_i for $i = 1, 2, \ldots$ starting with the given initial (seed) value x_0 as

$$x_{i+1} = a \cdot x_i + b \pmod{n}$$

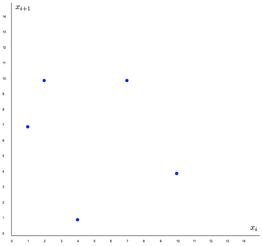
  where the multiplication and addition operation is performed modulo n, and therefore, $x_i \in \mathcal{Z}_n$
- This is a deterministic algorithm; the same x_i value will always produce the same x_{i+1} value, and the same seed x_0 will produce the same sequence $x_1, x_2, \ldots$
- There are only finitely many $x_i \in \mathcal{Z}_n$, and the sequence will repeat
- The period of the sequence is w such that $x_{i+w} = x_i$ for any $i \geq 0$

Linear Congruential Generators

- For $(a, b, n, x_0) = (3, 4, 15, 1)$, i.e., $a = 3$, $b = 4$, 15, and $x_0 = 1$, we obtain the sequence: $1, 7, 10, 4, 1, 7, 10, 4 \ldots$
  The period is $w = 4$
- For $(a, b, n, x_0) = (3, 4, 15, 2)$, we obtain the sequence: $2, 10, 4, 1, 7, 10, 4, 1, 7, \ldots$
  The period is $w = 4$



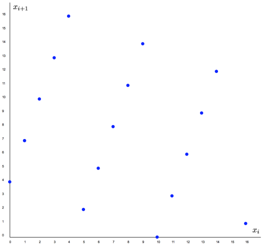
$(3, 4, 15, 1)$



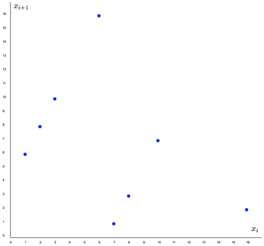
$(3, 4, 15, 2)$

Linear Congruential Generators

- For $(a, b, n, x_0) = (3, 4, 17, 1)$, we obtain the sequence:
  $1, 7, 8, 11, 3, 13, 9, 14, 12, 6, 5, 2, 10, 0, 4, 16, 1, 7, 8 \ldots$
  The period is $w = 16$
- For $(a, b, n) = (2, 4, 17)$, and $x_0 = 2$, we obtain the sequence:
  $1, 6, 16, 2, 8, 3, 10, 7, 1, 6, 16, 2, \ldots$
  The period is $w = 8$



$(3, 4, 17, 1)$



$(2, 4, 17, 2)$

Period of LCGs

Theorem

Given a LCG with parameters (a, b, p) such that p is prime, the period w is equal to the order of the element a in the multiplicative group $\mathcal{Z}_p^*$ for all x_0 seed values, except the period is 1 if $x_0 = -(a-1)^{-1} \cdot b \bmod p$.

- The group order is $p-1$
- The period w is a divisor of $p-1$
- The order of a primitive element is $p-1$
- The maximum period $w = p-1$ occurs when a is a primitive
- The theorem states that, if the starting point is

$$x_0 = -(a-1)^{-1} \cdot b \bmod p$$

  then the period is $w = 1$
- This is called "bad seed" since it causes minimum period $w = 1$

Bad Seed

- Assume $x_0 = -(a-1)^{-1} \cdot b \pmod{p}$
- Write this as $x_0 = ub \pmod{p}$ where $u = -(a-1)^{-1} \pmod{p}$
- This implies $u \cdot (a-1) = -1$, and thus, $au = u - 1$
- By starting with $x_0 = ub$, we obtain

$$\begin{aligned} x_0 &= ub \pmod{p} \\ x_1 &= ax_0 + b \pmod{p} \\ &= aub + b \pmod{p} \\ &= (au+1)b \pmod{p} \\ &= (u-1+1)b \pmod{p} \\ &= ub \end{aligned}$$

- Therefore, all subsequent x_is will be equal to ub
- The period $w = 1$

Period of LCGs

- For $(a, b, n) = (3, 4, 17)$, the order of the group is equal 16
- The element $a = 3$ is primitive since

$$\{3^1, 3^2, 3^3, \ldots, 3^{16}\} = \{3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1\}$$

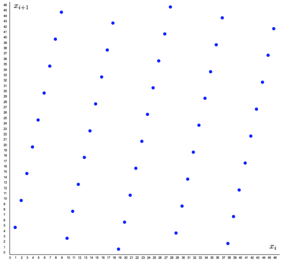
- The bad seed value is

$$\begin{aligned} x_0 &= -(a-1)^{-1} \cdot b \pmod{17} \\ &= -(3-1)^{-1} \cdot 4 \pmod{17} \\ &= -2^{-1} \cdot 4 \pmod{17} \\ &= 15 \pmod{17} \end{aligned}$$

- When $x_0 = 15$, we obtain the sequence: $15, 15, 15, \ldots$
- When $x_0 = 15$, the period is $w = 1$

The LCG for $(a, b, n, x_0) = (5, 0, 47, 1)$

- If we select $b = 0$, the bad seed value for any n will always be $x_0 = -(a-1)^{-1} \cdot b = 0 \pmod{n}$ for any a or n
- Therefore, it would be easy to detect and avoid the bad seed 0
- For example, for $(a, b, n, x_0) = (5, 0, 47, 1)$, we obtain the maximal period since 5 is a primitive root mod 47, and the bad seed is automatically avoided for a nonzero x_0



$(5, 0, 47, 1)$

A Practical LCG

- Since our processors have fixed data length, it is a good idea to select a prime as large as the word size, since we will perform mod p arithmetic
- It turns out that $2^{31} - 1 = 2,147,483,647$ is a prime number; furthermore, a suitable primitive element in $\mathcal{Z}_p$ for $p = 2^{31} - 1$ is found as $a = 7^5 = 16,807$
- The primitive root a is chosen to be near the square root of p, therefore, we have a good, practical, general-purpose LCG, given as

$$
\begin{aligned}
x_{i+1} &= a \cdot x_i \pmod{p} \\
p &= 2^{31} - 1 = 2,147,483,647 \\
a &= 7^5 = 16,807
\end{aligned}
$$

Since a is a primitive element, the period of LCG is $w = 2^{31} - 2$

Cryptographic Strength of LCGs

- Does the LCG satisfy requirements R1 and R2?
- Analysis and experiments show that LCGs with large p (such as the previous practical LCG) are (almost) acceptable as statistically random, but there are some deficiencies
- Unfortunately, the LCGs do not satisfy R2 since they are is highly predictable: Assuming a and p are known, given a single element x_i, any future element of the sequence can be computed as $x_{i+k} = a^k x_i \bmod n$
- Similarly, given x_i, any past element of the sequence can be computed as $x_{i-k} = a^{-k} x_i = (a^{-1})^k \bmod n$
- Inversion: the seed x_0 can be computed if any element x_i of the sequence is known, by working back from i down to 0

Cryptographic Strength of LCGs

- In general, we need to assume that a and p are fixed parameters of the RNG and therefore they are not changeable, i.e., they are not part of the key (x_0, the seed) — they can be discovered by reverse engineering
- If we can bundle a and p with the seed x_0, then we can claim more security — it would be much harder to discover the key (a, p, and x_0) given a limited number of elements x_i from the sequence $x_1, x_2, \ldots$
- Note that $x_{i+1} = a \cdot x_i \bmod p$ implies $x_{i+1} = a \cdot x_i + N \cdot p$ for some integer N; however, N is different for every pair (x_{i+1}, x_i), we have

$$x_{i+1} = a \cdot x_i + N_i \cdot p$$

  and therefore, if we have k pairs of the known elements (x_j, x_k) then we will also have $k + 2$ unknowns, i.e., a, p, and N_i for $i = 1, 2, \ldots, k$

Cryptographic Strength of LCGs

- Still, equations of the form $x_{i+1} = a \cdot x_i + N_i \cdot p$ can be solved using lattice reduction techniques, and therefore, we do not have strong assumptions of cryptographic strength
- There is also practical constraint in a LCG with all three parameters (a, p, x_0) are considered as the key
- We know that p has to be a prime and a has to be a primitive element of the group, that means a key generation algorithm has needs to incorporate these properties and generate such keys
- On the other hand, in a LCG with fixed parameters (a, p) we need not worry about key with special properties — the only key, the seed x_0, is just a random integer: any integer would be fine; also, since $b = 0$, the only "bad seed" is 0, and easy to avoid

GLIBC `random()`

- The GNU C library's `random()` function is a LCG with three steps
- The first step is based on the prime modulus $p = 2^{31} - 1$ and the primitive element $a = 16,807$
- Given the seed value s, the first step computes 33 elements $x_1, x_2, \ldots, x_{33}$:

$$
\begin{aligned}
x_0 &= s \\
x_i &= a \cdot x_{i-1} \pmod{p} \quad \text{for} \quad i = 1, 2, \ldots, 30 \\
x_{31} &= x_0 \\
x_{32} &= x_1 \\
x_{33} &= x_2
\end{aligned}
$$

GLIBC `random()`

- The second step is based on the addition operation mod $q = 2^{32}$
- In the second step, new x_i values are computed for $i = 34, 35, \ldots, 343$

$$x_i = x_{i-3} + x_{i-31} \pmod{q} \quad \text{for} \quad i = 34, 35, \ldots, 343$$

- In the final step, the output values are generated using the previous mod q addition operation and the logical right shift operation $(\cdot)_{rs}$ as follows

$$\begin{aligned} x_i &= x_{i-3} + x_{i-31} \pmod{q} \quad \text{for} \quad i \geq 344 \\ r_j &= (x_{j+344})_{rs} \quad \text{for} \quad i \geq 0 \end{aligned}$$

- Inversion: Two consecutive different moduli and the right shift make the inversion more difficult, however, since there are 2^{32} different seed values, exhaustive search is possible