Random number generators

Comp Sci 1570 Introduction to C++
1 Definitions

2 Random number generation
   Methods
   Hardware
   Software
   Hybrid
   Example: LCG
   C++
   rand()
   srand()
Random number generator

- The generation of a sequence of numbers or symbols that cannot be reasonably predicted better than by a random chance, usually through a random-number generator (RNG).
- RNGs have applications in gambling, statistical sampling, computer simulation, cryptography, completely randomized design, and other areas where producing an unpredictable result is desirable.
- Two principal methods, computational pseudorandom number generation (PRNG) and a hardware random number generator (true random number generation, TRNG).
1 Definitions

2 Random number generation
   Methods
   Hardware
   Software
   Hybrid
   Example: LCG
   C++
   rand()  srand()
1 Definitions

2 Random number generation
   Methods
      Hardware
      Software
      Hybrid
      Example: LCG
      C++
      rand()
      srand()
1 Definitions

2 Random number generation
   Methods
   Hardware
   Software
   Hybrid
   Example: LCG
   C++
   rand()
   srand()
• Measures some physical phenomenon expected to be random and then compensate for possible biases in the measurement process.

• E.g., atmospheric noise, thermal noise, and other external electromagnetic and quantum phenomena like cosmic background radiation or radioactive decay as measured over short timescales represent sources of natural entropy.

• In security applications, hardware generators are generally preferred over pseudo-random algorithms, where feasible.

• Speed at which entropy can be harvested from natural sources is dependent on the underlying physical phenomena being measured.

• Thus, sources of naturally occurring "true" entropy are said to be blocking – they are rate-limited until enough entropy is harvested to meet the demand.

• On most Linux distributions, the pseudo device file /dev/random will block until sufficient entropy is harvested from the environment.
• Computational algorithms can produce long sequences of apparently random results, which are in fact completely determined by a shorter initial value, known as a seed value or key.

• As a result, the entire seemingly random sequence can be reproduced if the seed value is known.

• Called a pseudorandom number generator (PNRG)

• Does not rely on sources of naturally occurring entropy, though it may be periodically seeded by natural sources.

• This generator type is non-blocking, so they are not rate-limited by an external event, making large bulk reads a possibility.

• While a pseudorandom number generator based solely on deterministic logic can never be regarded as a "true" random number source in the purest sense of the word, in practice they are generally sufficient even for demanding security-critical applications.
- Some systems take a hybrid approach, providing randomness harvested from natural sources when available, and falling back to periodically re-seeded software-based cryptographically secure pseudorandom number generators (CSPRNGs).

- The fallback occurs when the desired read rate of randomness exceeds the ability of the natural harvesting approach to keep up with the demand.

- This approach avoids the rate-limited blocking behavior of random number generators based on slower and purely environmental methods.
1 Definitions

2 Random number generation
   Methods
   Hardware
   Software
   Hybrid
   Example: LCG
   C++
   rand()
   srand()
Definitions

Random number generation

Methods
Hardware
Software
Hybrid

Example: LCG

C++

rand()
srand()

Linear congruential generator

- An example of a PNRG is a linear congruential generator (LCG), which is an algorithm that yields a sequence of pseudo-randomized numbers calculated with a discontinuous piecewise linear equation.
- One of the oldest and best-known pseudorandom number generator algorithms.
- Generator is defined by a recurrence relation:

\[ X_{n+1} = (aX_n + c) \mod m \]

where \( X \) is the sequence of pseudorandom values, and
- \( m, 0 < m \) the "modulus"
- \( a, 0 < a < m \) the "multiplier"
- \( c, 0 \leq c < m \) the "increment"
- \( X_0, 0 \leq X_0 < m \) the "seed" or "start value"

are integer constants that specify the generator.

If \( c = 0 \), the generator is often called a multiplicative congruential generator (MCG), or Lehmer RNG.
If \( c \neq 0 \), the method is called a mixed congruential generator.
Work out an example

$$X_{n+1} = (aX_n + c) \mod m$$

where $X$ is the sequence of pseudorandom values, and $m, 0 < m$ the ”modulus”
$a, 0 < a < m$ the ”multiplier”
$c, 0 \leq c < m$ the ”increment”
$X_0, 0 \leq X_0 < m$ the ”seed” or ”start value”
are integer constants that specify the generator.
1 Definitions

2 Random number generation
   Methods
   Hardware
   Software
   Hybrid
   Example: LCG
   C++
   rand()
   srand()
C++ function: `rand()`

```c
int rand (void);
```

- Returns a pseudo-random integral number in the range between 0 and `RAND_MAX`, the value of which is library-dependent, but is guaranteed to be at least 32767 on any standard library implementation.
- Algorithm returns a sequence of apparently non-related numbers each time it is called.
- Uses a seed to generate the series, which should be initialized to some distinctive value using function `srand`. 
void srand (unsigned int seed);

- PRNG is initialized using the argument passed as seed.
- For every different seed value used in a call to srand, the pseudo-random number generator can be expected to generate a different succession of results in the subsequent calls to rand.
- Two different initializations with the same seed will generate the same succession of results in subsequent calls to rand.
- If seed is set to 1, the generator is reinitialized to its initial value and produces the same values as before any call to rand or srand.
- In order to generate random-like numbers, srand is usually initialized to some distinctive runtime value, like the value returned by function time.
- This is distinctive enough for most trivial randomization needs.