Pseudo Random Number Generators (PRNG)

Desired properties in general:

1. Must have a long enough period
2. Uniform distribution of numbers for large quantities of generated numbers
3. Independence of successive values
4. High quality of the dimensional distribution of the output sequence

Usually, PRNGs use an internal state that is "somehow" is scrambled each time a new number is requested.

Parallel PRNGs

In addition to the above mentioned properties, parallel PRNGs require the following qualities:

1. Easy generation (initialization) of parallel states
2. Independence of the parallel states
3. Each parallel state should provide an independent sequence
4. Each independent state should provide a high quality sequence of PRNs
5. The state must be efficiently updated in parallel with simple operations

Mersenne Twister

The **Mersenne Twister** is a pseudorandom number generator (PRNG). It is by far the most widely used general-purpose PRNG. Its name derives from the fact that its period length is chosen to be a Mersenne prime.

Matsumoto, M.; Nishimura, T. (1998). "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator". *ACM Transactions on Modeling and Computer Simulation*. **8** (1): 3–30. CiteSeerX 10.1.1.215.1141

Advantages

- Permissively-licensed and patent-free for all variants except CryptMT.
- Passes numerous tests for statistical randomness, including the Diehard tests and most, but not all of the TestU01 tests.[39]
- A very long period of 2^19937 − 1. Note that while a long period is not a guarantee of quality in a random number generator, short periods, such as the 2^{32} common in many older software packages, can be problematic.
- *k*-distributed to 32-bit accuracy for every $1 \leq k \leq 623$ (for a definition of *k*-distributed, see below)
- Implementations generally create random numbers faster than true random methods. A study found that the Mersenne Twister creates 64-bit floating point random numbers approximately twenty times faster than the hardware-implemented, processor-based RDRAND instruction set.

Disadvantages

- Relatively large state buffer, of 2.5 KiB, unless the TinyMT variant is used.
- Mediocre throughput by modern standards, unless the SFMT variant is used.
- Exhibits two clear failures (linear complexity) in both Crush and BigCrush in the TestU01 suite. The test, like Mersenne Twister, is based on F_2 algebra.

[39] There are a number of other generators that pass all the tests (and numerous generators that fail badly).

- Multiple instances that differ only in seed value (but not other parameters) are not generally appropriate for Monte-Carlo simulations that require independent random number generators, though there exists a method for choosing multiple sets of parameter values.
- Poor diffusion: can take a long time to start generating output that passes randomness tests, if the initial state is highly non-random—particularly if the initial state has many zeros. A consequence of this is that two instances of the generator, started with initial states that are almost the same, will usually output nearly the same sequence for many iterations, before eventually diverging. The 2002 update to the MT algorithm has improved initialization, so that beginning with such a state is very unlikely.[45] The GPU version (MTGP) is said to be even better.[46]
- Contains subsequences with more 0's than 1's. This adds to the poor diffusion property to make recovery from many-zero states difficult.
- Is not cryptographically secure, unless the CryptMT variant is used. The reason is that observing a sufficient number of iterations (624 in the case of MT19937, since this is the size of the state vector from which future iterations are produced) allows one to predict all future iterations.

What is TinyMT

TinyMT is a new small-sized variant of Mersenne Twister (MT) introduced by Mutsuo Saito and Makoto Matsumoto in 2011. There are two types of TinyMT, tinymt32 and tinymt64. tinymt32 outputs 32-bit unsigned integers and single precision floating point numbers. On the other hand, tinymt64 outputs 64-bit unsigned integers and double precision floating point numbers.

The purpose of TinyMT is not to replace Mersenne Twister. TinyMT has far shorter period than Mersenne Twister. The merit of TinyMT is in its small size of the internal state of 127 bits, far smaller than 19937 of Mersenne Twister. The purpose of TinyMT may be used in a situation where large state generators such as Mersenne Twister

are difficult to use. According to statistical test (BigCrush in TestU01 and AdaptiveCrush), the quality of the outputs of TinyMT seems pretty good, taking the small size of the internal state into consideration.

TinyMT has following features:

- The periods of generated sequences are 2^127-1.
- The size of internal state space is 127 bits.
- The state transition function is **F**2-linear.
- The output function is not **F**2-linear.
- TinyMTDC generates distinct parameter sets for TinyMT.
- TinyMTDC can generate a large number of parameter sets (over 2^32 x 2^16)
- Parameter generation of TinyMTDC is fast.

http://www.math.sci.hiroshima-u.ac.jp/m-mat/MT/TINYMT/index.html