Random Number Generators

Uniform random numbers can be generated in many different ways. Since our interest in random numbers is for use within simulations, we need to be able to generate them on a computer. This is done using mathematical functions called **random number generators.**

Most random number generators use some form of a congruential relationship. Examples of such generators include the linear congruential generator, the multiplicative generator, and the mixed generator. The linear congruential generator is by far the most widely used. In fact, most built-in random number functions on computer systems use this generator. With this method, we produce a sequence of integers $x_1, x_2, x_3, \ldots$ between 0 and $m - 1$ according to the following recursive relation:

$$x_{i+1} = (ax_i + c) \text{ modulo } m \qquad (i = 0, 1, 2, \ldots)$$

The initial value of x_0 is called the seed, a is the constant multiplier, c is the increment, and m is the modulus. These four variables are called the parameters of the generator. Using this relation, the value of x_{i+1} equals the remainder from the division of $ax_i + c$ by m. The random number between 0 and 1 is then generated using the equation

$$R_i = \frac{x_i}{m} \qquad (i = 1, 2, 3, \ldots)$$

For example, if $x_0 = 35$, $a = 13$, $c = 65$, and $m = 100$, the algorithm works as follows:

Iteration 0 Set $x_0 = 35$, $a = 13$, $c = 65$, and $m = 100$.

Iteration 1 Compute

$$\begin{aligned} x_1 &= (ax_0 + c) \text{ modulo } m \\ &= [13(35) + 65] \text{ modulo } 100 \\ &= 20 \end{aligned}$$

Deliver

$$\begin{aligned} R_1 &= \frac{x_1}{m} \\ &= \frac{20}{100} \\ &= 0.20 \end{aligned}$$

Iteration 2 Compute

$$\begin{aligned} x_2 &= (ax_1 + c) \text{ modulo } m \\ &= [13(20) + 65] \text{ modulo } 100 \\ &= 25 \end{aligned}$$

Deliver

$$\begin{aligned} R_2 &= \frac{x_2}{m} \\ &= \frac{25}{100} \\ &= 0.25 \end{aligned}$$

Iteration 3 Compute

$$x_3 = (ax_2 + c) \text{ modulo } m$$

$$\vdots$$

and so on.

Each random number generated using this method will be a decimal number between 0 and 1. Note that although it is possible to generate a 0, a random number cannot equal 1. Random numbers generated using congruential methods are called **pseudorandom numbers.** They are not true random numbers in the technical sense, because they are completely determined once the recurrence relation is defined and the parameters of the generator are specified. However, by carefully selecting the values of a, c, m, and x_0, the pseudorandom numbers can be made to meet all the statistical properties of true random numbers. In addition to the statistical properties, random number generators must have several other important characteristics if they are to be used efficiently within computer simulations. (1) The routine must be fast; (2) the routine should not require a lot of core storage; (3) the random numbers should be replicable; and (4) the routine should have a sufficiently long cycle—that is, we should be able to generate a long sequence without repetition of the random numbers.

There is one important point worth mentioning at this stage: Most programming languages have built-in library functions that provide random (or pseudorandom) numbers directly. Therefore, most users need only know the library function for a particular system. In some systems, a user may have to specify a value for the seed, x_0, but it is unlikely that a user would have to develop or design a random number generator. However, for more information on random numbers and random number generators, the interested reader may consult Banks and Carson (1984), Knuth (1998), or Law and Kelton (1991).