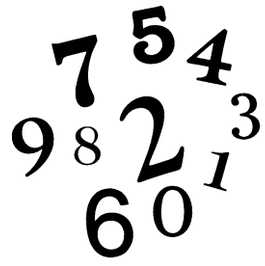
Numbers at Random

Number theory supplies a superior random-number generator



By IVARS PETERSON

Random numbers are a precious commodity. They're widely used but difficult to conjure up.

Scientists, engineers and statisticians use random numbers — ideally, patternless strings of digits — for tackling a wide range of problems, from modeling molecular behavior to analyzing the statistical significance of data. Consider, for instance, the problem of determining how much and what wavelengths of light escape a star's atmosphere. To find an answer, astrophysicists can simulate the process, using a computer to track a large number of photons as they interact with atoms. For each of the myriad photon-atom interactions possible in a typical simulation, the computer furnishes a random number to specify the angle and energy at which the photon emerges from its atomic assignation.

Such computer simulations consume vast quantities of random numbers. As scientists turn to increasingly speedy and powerful computers, they run the risk of using up all the random numbers that a computer's standard set of instructions for generating random numbers can typically produce without starting to repeat the list. At the same time, because no deterministic, computer-based process can generate a string of truly random numbers, they must watch for subtle deviations from randomness among the generated digits.

"We've got a new situation in which conventional random-number generators just won't do," says George Marsaglia, a computer scientist and statistician at Florida State University in Tallahassee. "Even on a [personal computer], you can run something for a few days and exhaust the generator."

To better meet researchers' needs for lengthy strings of random numbers, Marsaglia and Florida State colleague Arif Zaman have developed a new class of computer-based random-number generators that produce "astonishingly long" sequences of random numbers. These random-number generators feature longer "periods" than conventional methods, which means they generate a longer list of random numbers before the list starts repeating itself.

Called "add-with-carry" and "subtract-with-borrow," the novel methods show great promise, Marsaglia says. Moreover, because these methods are closely tied to a branch of mathematics known as number theory, Marsaglia and Zaman can prove mathematically that their random-number generators have long periods.

The traditional procedures for generating a list of random numbers — flipping a coin, rolling dice, shuffling cards, shaking an urn filled with numbered balls — are too cumbersome for everyday use by researchers. Scientists prefer computers for generating vast stocks of random numbers quickly and efficiently. Indeed, nearly all types of computers and many computer programs have a built-in set of instructions for generating random numbers.

A computer normally processes numbers as sequences of ones and zeros, or bits. A random-number generator scrambles the bits of a given number to produce a new number in such a way that the result appears to be independent of previously generated numbers and represents a random selection from the set of all available numbers.

The most commonly used bit-scrambling method involves multiplication. A starting number of, say, 10 digits is multiplied by a given, specific number, or constant, and then the last 10 digits of the product are taken as the new random number. That number, multiplied by the constant, generates the next random number. This procedure forms the basis for "congruential" random-number generators.

Such methods usually produce numbers that pass simple tests of randomness and mimic well the expected behavior of true random sequences. For example, on the average, a high number is followed by a lower one as often as a low number follows a higher one.

"With a proper choice of multiplier. . ., such a generator produces a sequence of numbers that are difficult to distinguish from truly random numbers," Marsaglia says. "A good congruential generator could be used to run the casinos in Las Vegas and Atlantic City and all the state lotteries with no one the wiser."

But the constraints imposed by handling numbers of only a certain length limit these generators to sequences of a few billion or so random numbers. That's not enough for many applications.

"Modern computer speeds and exotic architectures make possible massive Monte Carlo simulations for which standard generators may not be suitable," Marsaglia says. Monte Carlo simulations use random numbers to replicate a system's behavior, whether the motion of a molecule or fluctuations in the stock market.

Furthermore, if too many random numbers come from a relatively small pool, the selected numbers as a group may no longer pass some of the more stringent tests of randomness. "People have run into such problems already," says Andrew M. Odlyzko, a mathematician at AT&T Bell Laboratories in Murray Hill, N.J.

Rudimentary Example of an Add-With-Carry Random-Number Generator

Starting values (seeds): 0, 1; Carry bit: 0

Previous value		Current value		Carry bit		New value	
0	+	1	+	0	=	1	
1	+	1	+	0	=	2	
1	+	2	+	0	=	3	
2	+	3	+	0	=	5	
3	+	5	+	0	=	8	
5	+	8	+	0	=	13	(save 3, change carry bit to 1)
8	+	3	+	1	=	12	(save 2, carry bit remains 1)
3	+	2	+	1	=	6	(because answer is less than 10, carry bit reverts to 0)
2	+	6	+	0	=	8	

Marsaglia and Zaman's new class of random-number generators hinges on the use of what are called Fibonacci sequences. The classic example of a Fibonacci sequence consists of the numbers 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, and so on. Except for the first two, each number in this sequence equals the sum of the previous two numbers. For example, 89 = 34 + 55. The next number in the sequence would be 55 + 89, or 144.

Taking only the last digit of each num-

ber in the sequence produces the "random" numbers 0, 1, 1, 2, 3, 5, 8, 3, 1, 4, 5, 9, 4,. . . . When applied to seeds, or starting numbers, about 1,800 bits long, such a procedure forms the basis of "lagged-Fibonacci" random-number generators.

To extend this generator's period, Marsaglia and Zaman make a simple but crucial modification. They introduce a "carry bit," which may have an initial value of either 0 or 1. Such an "add-with-carry" generator then has some of the features associated with "long addition," an operation familiar to students who learned their arithmetic in the days before calculators.

For example, suppose the two initial values are 0 and 1, and the initial carry bit is 0. Each new digit in the sequence equals the sum of the previous two digits plus the carry bit. If the result is larger than 10, only the last digit is saved, and the carry bit used in computing the next digit in the sequence becomes 1 (see table). This procedure generates the "random" sequence 0, 1, 1, 2, 3, 5, 8, 3, 2, 6, 8,. . . .

By selecting appropriate starting values and slightly altering the arithmetic, computer programmers can create a whole class of "add-with-carry" generators with periods considerably longer than those of generators now in use. A similar approach underlies the "subtract-with-borrow" generators.

"In a way, you can view this type of generator as a randomness amplifier," Marsaglia says. "You start with random seeds thousands of bits long and generate these long strings of random numbers."

To prove that the new methods have long periods, Marsaglia and Zaman delve into aspects of number theory concerning prime numbers and the mathematical characteristics of fractions expressed as decimals. "This elementary material has been known for hundreds of years," Marsaglia notes, "but it is seldom mentioned in modern books."

These number-theoretic proofs, which involve factoring large numbers, show that "subtract-with-borrow" generators have periods of 10^{250} or more. The simplest congruential random-number generators, which are commonly installed in personal computers, have periods of only a few billion (10^9) numbers.

"We get tremendously long periods with very simple arithmetic," Marsaglia says.

But a long period by itself isn't enough. As sophisticated tests of randomness demonstrate, no string of numbers generated by a simple computer process can be truly random. In fact, nearly every scheme now in use for generating random numbers by computer has some flaw. Often, the flaws are difficult to detect and analysts require sensitive techniques to discern the subtle numerical patterns that may lie hidden in vast arrays of digits.

"Many people use random-number generators blindly," Odlyzko says. "But among people who are aware of what happens, there is a lot of concern about the quality of random-number generators. This is quite an active area of research, and there are concerns that many simulations may not be valid because people have been using [random-number generators] incorrectly."

"For any generator, there are situations for which it gives bad results," Marsaglia says. "What we want is enough experience so that we can provide guidelines about when not to use a particular random-number generator."

The new Marsaglia-Zaman random-number generators, described earlier this year in ANNALS OF APPLIED PROBABILITY (Vol.1, No.3), have already attracted the attention of scientists interested in high-quality, long-period sources of random numbers. These generators have passed a variety of the most stringent tests of randomness available. They are also remarkably efficient, and they don't take up inordinate amounts of a computer's memory.

"We've written a program for the subtract-with-borrow generator for personal computers, and we've had hundreds of requests for copies," Marsaglia says. "Physicists seem to be among the biggest users." □