Randomness as a Resource

Brian Hayes

A reprint from

American Scientist

the magazine of Sigma Xi, the Scientific Research Society

Volume 89, Number 4
July–August, 2001
pages 300–304

Randomness as a Resource

Brian Hayes

Randomness is not something we usually look upon as a vital natural resource, to be carefully conserved lest our grandchildren run short of it. On the contrary, as a close relative of chaos, randomness seems to be all too abundant and everpresent. Everyone has a closet or a file drawer that offers an inexhaustible supply of disorder. Entropy—another cousin of randomness—even has a law of nature saying it can only increase. And, anyway, even if we were somehow to use up all the world's randomness, who would lament the loss? Fretting about a dearth of randomness seems like worrying that humanity might use up its last reserves of ignorance.

Nevertheless, there is a case to be made for the proposition that high-quality randomness is a valuable commodity. Many events and processes in the modern world depend on a steady supply of the stuff. Furthermore, we don't know how to manufacture randomness; we can only mine it from those regions of the universe that have the richest deposits, or else farm it from seeds gathered in the natural world. So, even if we have not yet reached the point of clear-cutting the last proud acre of old-growth randomness, maybe it's not too early to consider the question of long-term supply.

The Randomness Industry

To appreciate the value of randomness, just imagine a world without it. What would replace the referee's coin flip at the start of a football game? How would a political poll-taker select an unbiased sample of the electorate? Then of course there's the Las Vegas problem. Slot machines devour even more randomness than they do silver dollars. Inside each machine an electronic device spews out random numbers 24 hours a day, whether or not anyone is playing.

There's also a Monte Carlo problem. I speak not of the Mediterranean principality but of the simulation technique named for that place. The Monte Carlo method got its start in the 1940s at Los Alamos, where physicists were struggling to predict the fate of neutrons moving through uranium and other materials. The Monte Carlo approach to this problem is to trace thousands of simulated neutron paths. Whenever a neutron strikes a nucleus, a random number determines the outcome of the event—reflection, absorption or fission. Today the Monte Carlo method is a major industry not only in physics but also in economics and some areas of the life sciences, not to mention hundreds of rotisserie baseball leagues.

Many computer networks would be deadlocked without access to randomness. When two nodes on a network try to speak at once, politeness is not enough to break the impasse. Each computer might be programmed to wait a certain interval and then try again, but if all computers followed the same rule, they'd keep knocking heads repeatedly until the lights went out. The Ethernet protocol solves this problem by deliberately not giving a fixed rule. Instead, each machine picks a random number between 1 and n, then waits n units of time before retransmitting; the probability of a second collision is reduced to $1/n$.

Computer science has a whole technology of "randomized algorithms." On first acquaintance the very idea of a randomized algorithm may seem slightly peculiar: An algorithm is supposed to be a deterministic procedure—one that allows no scope for arbitrary choice or caprice—so how can it be randomized? The contradiction is resolved by making the randomness a resource external to the algorithm itself. Where an ordinary algorithm is a black box receiving a stream of bits as input and producing another stream of bits as output, a randomized algorithm has a second input stream made up of random bits.

Sometimes the advantage of a randomized algorithm is clearest when you take an adversarial view of the world. Randomness is what you need to foil an adversary who wants to guess your intentions or predict your behavior. Suppose you are writing a program to search a list of items for some specified target. Given any predetermined search strategy—left to right, right to left, middle outward—an adversary can arrange the list so that the target item is always in the last place you look. But a randomized version of the procedure

Brian Hayes is Senior Writer for American Scientist. *Address: 211 Dacian Avenue, Durham, NC 27701. Internet: bhayes@amsci.org*

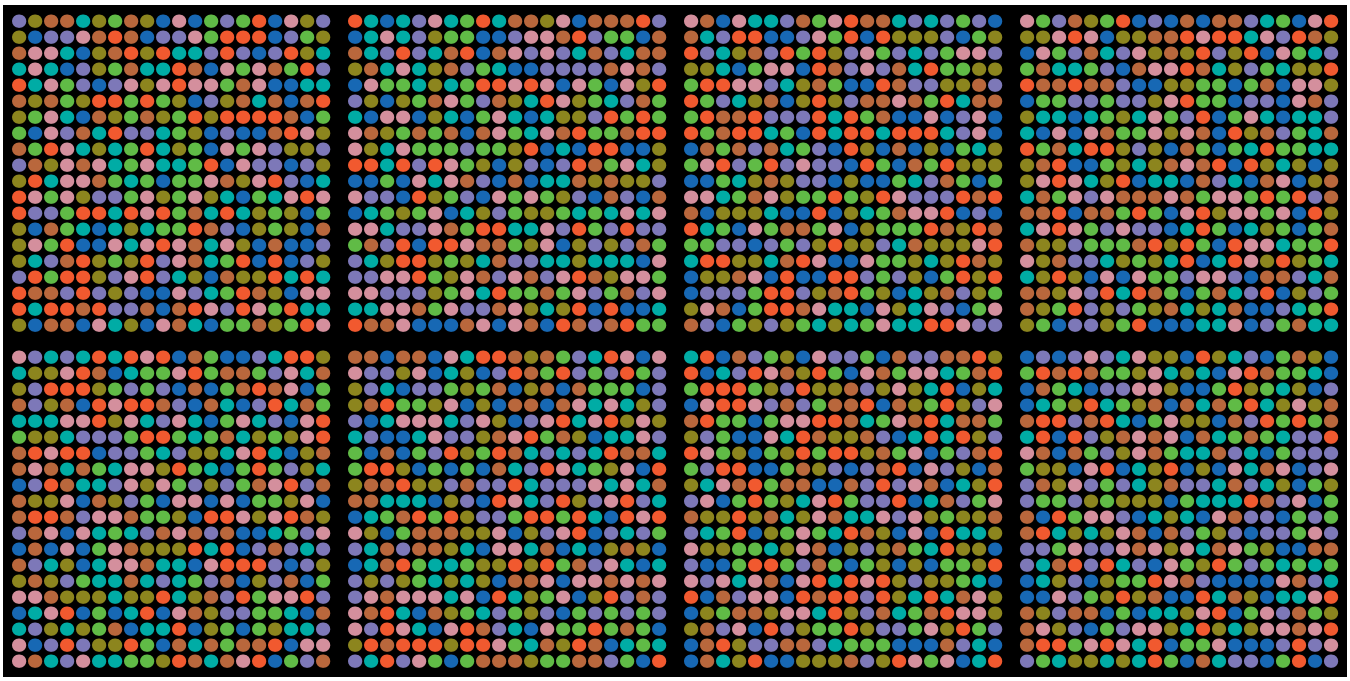

Figure 1. Eight specimens of randomness come from very different sources but yield similar patterns. The 400 colored dots in each panel represent 1,200 random bits taken three at a time. The sources of randomness are, from left to right and top to bottom: a pseudo-random generator, the 1955 Rand Corporation table, the HG202 random-bit generator (faust.irb.hr/~stipy/), the Pennsylvania Daily Number lottery (www.palottery.com), 1,200 coin flips, rapid pounding on a keyboard, an electrocardiogram of atrial fibrillation (www.physionet.org), and six Lava Lites (lavarand.sgi.com).

can't be outguessed so easily; the adversary can't know where to hide the target because the program doesn't decide where to search until it begins reading random bits. In spite of the adversary's best efforts, you can expect to find the target after sifting through half the list.

Still another field that can't do without randomness is cryptography, where calculated disorder is the secret to secrecy. The strongest of all cipher systems require a random key as long as the message that's being sent. The late Claude E. Shannon proved that such a cipher is absolutely secure. That is, if the key is truly random, and if it is used only once, an eavesdropper who intercepts an encrypted message can learn nothing about the original text, no matter how much time and effort and computational horsepower are brought to bear on the task. Shannon also showed that no cipher with a key shorter than the message can offer the same degree of security. But a long key is a considerable inconvenience—hard to generate, hard to distribute.

Much of the emphasis in recent cryptological research has been on ways to get by with less randomness, but a recent proposal takes a step in the other direction. The idea is to drown an adversary in a deluge of random bits. The first version of the scheme was put forward in 1992 by Ueli M. Maurer of the Swiss Federal Institute of Technology; more recent refinements (not yet published) have come from Michael O. Rabin of Harvard University and his student Yan Zong Ding.

The heart of the plan is to set up a public beacon—perhaps a satellite—continually broadcasting random bits at a rate so high that no one could store more than a small fraction of them. Parties who want to communicate in privacy share a relatively short key that they both use to select a sequence of random bits from the public broadcast; the selected bits serve as an enciphering key for their messages. An eavesdropper cannot decrypt an intercepted message without a record of the random broadcasts, and cannot keep such a record because it would be too voluminous.

How much randomness would the beacon have to broadcast? Rabin and Ding mention a rate of 50 gigabits per second, which would fill up some 800,000 CD-ROMs per day.

Supply-Side Issues

Whatever the purpose of randomness, and however light or heavy the demand, it seems like producing the stuff ought to be a cinch. At the very least it should be easier to make random bits than non-random ones, in the same way that it's easier to make a mess than it is to tidy up. If computers can perform long and intricate calculations where a single error could spoil the entire result, then surely they should be able to churn out some patternless digital junk. But they can't. There is no computer program for randomness.

Of course most computer programming languages will cheerfully offer to generate random numbers for you. In Lisp the expression *(random 100)* produces an integer in the range between 0 and 99, with each of the 100 possible values having equal probability. But these are *pseudo*-random numbers: They "look" random, but under

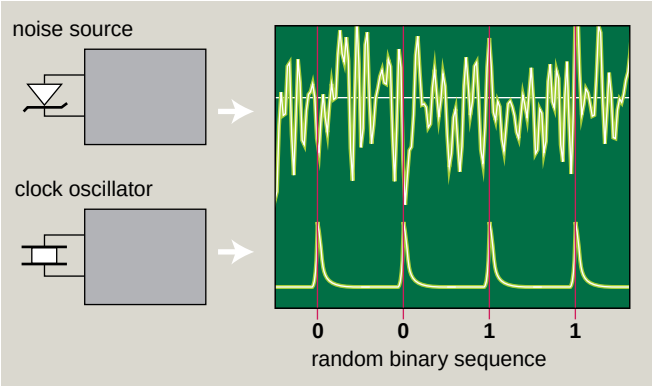

Figure 2. Thermal noise in electronic circuits, which is usually a nuisance to be suppressed, becomes a resource to be exploited in random number generators. In one scheme the noise signal is measured at regular intervals defined by a sequence of clock pulses; if the voltage at the instant of a pulse is positive, a 1 is emitted, and otherwise a 0.

the surface there is nothing unpredictable about them. Each number in the series depends on those that went before. You may not immediately perceive the rule in a series like 58, 23, 0, 79, 48..., but it's just as deterministic as 1, 2, 3, 4....

The only source of true randomness in a sequence of pseudo-random numbers is a "seed" value that gets the series started. If you supply identical seeds, you get identical sequences; different seeds produce different numbers. The crucial role of the seed was made clear in the 1980s by Manuel Blum, now of Carnegie Mellon University. He pointed out that a pseudo-random generator does not actually generate any randomness; it stretches or dilutes whatever randomness is in the seed, spreading it out over a longer series of numbers like a drop of pigment mixed into a gallon of paint.

For most purposes, pseudo-random numbers serve perfectly well—often better than true random numbers. Almost all Monte Carlo work is based on them. Even for some cryptographic applications—where standards are higher and unpredictability is everything—Blum and others have invented pseudo-random generators that meet most needs. Nevertheless, true randomness is still in demand, if only to supply seeds for pseudo-random generators. And if true randomness cannot be created in any mathematical operation, then it will have to come from some physical process.

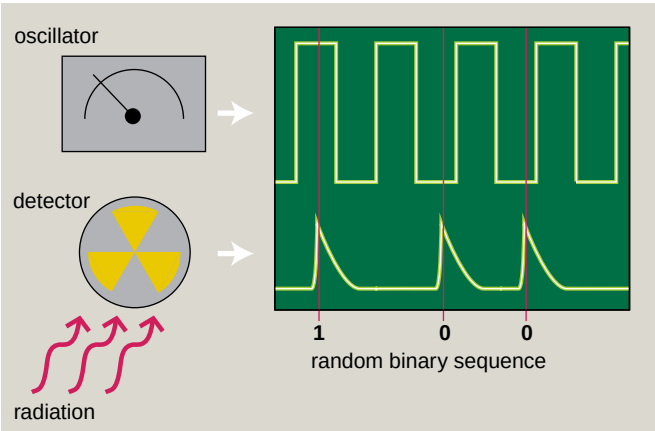

Figure 3. Radioactive decay offers another source of randomness. When a decay event is detected, the digit emitted depends on the polarity of a square-wave signal at that instant.

Extracting randomness from the material world also sounds like an easy enough job. Unpredictable events are all around us: the stock market tomorrow, the weather next week, the orbital position of Pluto in 50 million years. Yet finding events that are *totally* patternless turns out to be quite difficult. The stories of the pioneering seekers after randomness are chronicles of travail and disappointment.

Consider the experience of the British biometrician W. F. R. Weldon and his wife, the former Florence Tebb. Evidently they spent many an evening rolling dice together—not for money or sport but for science, collecting data for a classroom demonstration of the laws of probability. But in 1900 Karl Pearson analyzed 26,306 of the Weldons' throws and found deviations from those laws; there was an excess of fives and sixes.

In 1901 Lord Kelvin tried to carry out what we would now call a Monte Carlo experiment, but he ran into trouble generating random numbers. In a footnote he wrote: "I had tried numbered billets (small squares of paper) drawn from a bowl, but found this very unsatisfactory. The best mixing we could make in the bowl seemed to be quite insufficient to secure equal chances for all the billets."

In 1925 L. H. C. Tippett had the same problem. Trying to make a random selection from a thousand cards in a bag, "it was concluded that the mixing between each draw had not been sufficient, and there was a tendency for neighbouring draws to be alike." Tippett devised a more elaborate randomizing procedure, and two years later he published a table of 41,600 random digits. But in 1938 G. Udny Yule submitted Tippett's numbers to statistical scrutiny and reported evidence of "patchiness."

Ronald A. Fisher and Frank Yates compiled another table of 15,000 random digits, using two decks of playing cards to select numbers from a large table of logarithms. When they were done, they discovered an excess of sixes, and so they replaced 50 of them with other digits "selected at random." (Two of their statistical colleagues, Maurice G. Kendall and Bernard Babington Smith, comment mildly: "A procedure of this kind may cause others, as it did us, some misgiving.")

The ultimate random-number table arrived with a thump in 1955, when the Rand Corporation published a 600-page tome titled *A Million Random Digits with 100,000 Normal Deviates*. The Rand randomizers used "an electronic roulette wheel" that selected one digit per second. Despite the care taken in the construction of this device, "Production from the original machine showed statistically significant biases, and the

engineers had to make several modifications and refinements of the circuits." Even after this tune-up, the results of the month-long run were still unsatisfactory; Rand had to remix and shuffle the numbers before the tables passed statistical tests.

Today there is little interest in publishing tables of numbers, but machines for generating randomness are still being built. Many of them find their source of disorder in the thermal fluctuations of electrons wandering through a resistor or a semiconductor junction. This noisy signal is the hiss or whoosh you hear when you turn up an amplifier's volume control. Traced by an oscilloscope, it certainly looks random and unpredictable, but converting it into a stream of random bits or numbers is not straightforward.

The obvious scheme for digitizing noise is to measure the signal at certain instants and emit a 1 if the voltage is positive or a 0 if it is negative. But it's hard to build a measuring circuit with a precise and consistent threshold between positive and negative voltage. As components age, the threshold drifts, causing a bias in the balance between 1s and 0s. There are circuits and computational tricks to correct this problem, but the need for such fixes suggests just how messy it can be getting a physical device to conform to a mathematical ideal—even when the ideal is that of pure messiness.

Another popular source of randomness is the radioactive decay of atomic nuclei, a quantum phenomenon that seems to be near the ultimate in unpredictability. A simple random-number generator based on this effect might work as follows. A Geiger-Müller tube detects a decay event, while in the background a free-running oscillator generates a high-frequency square-wave signal—a train of positive and negative pulses. At the instant of a nuclear decay, the square wave is sampled, and a binary 1 or 0 is output according to the polarity of the pulse at that moment. Again there are engineering pitfalls. For example, the circuitry's "dead time" after each event may block detection of closely spaced decays. And if the positive and negative pulses in the square wave differ in length even slightly, the output will be biased.

Hardware random-number generators are available as off-the-shelf components you can plug into a port of your computer. Most of them rely on thermal electronic noise. If your computer has one of the latest Intel Pentium processors, you don't need to plug in a peripheral: The random-number generator is built into the CPU chip. There are also several Web sites that serve up free samples of randomness. George Marsaglia of Florida State University has some 4.8 billion carefully tested random bits available to the public. And there are less-conventional sources of randomness, most famously "lavarand," at Silicon Graphics, where random bits are extracted from images of the erupting blobs inside six Lava Lite lamps. (Lately the lamps have gone out, although samples remain available at lavarand.sgi.com.)

The Empyrean and the Empirical

As a practical matter, reserves of randomness certainly appear adequate to meet current needs. Consumers of randomness need not fear rolling blackouts this summer. But what of the future? The great beacon of randomness proposed by Rabin and Ding would require technology that remains to be demonstrated. They envision broadcasting 50 billion random bits per second, but randomness generators today typically run at speeds closer to 50 kilobits per second.

The prospect of scaling up by a factor of a million demands attention to quality as well as quantity. For most commodities, quantity and quality have an inverse relation. A laboratory buying milligrams of a reagent may demand 99.9 percent purity, whereas a factory using carloads can tolerate a lower standard. In the case of randomness, the trade-off is turned upside down. If you need just a few random numbers, any source will do; it's hard to spot biases in a handful of bits. But a Monte Carlo experiment burning up billions of random numbers is exquisitely sensitive to the faintest trends and patterns. The more randomness you consume, the better it has to be.

Why is it hard to make randomness? The fact that maintaining perfect *order* is difficult surprises no one; but it comes as something of a revelation that perfect *disorder* is also beyond our reach. As a matter of fact, perfect disorder is the more troubling concept—it is hard not only to attain but also to define or even to imagine.

The prevailing definition of randomness was formulated in the 1960s by Gregory J. Chaitin of IBM and by the Russian mathematician A. N. Kolmogorov. The definition says that a sequence of bits is random if the shortest computer program for generating the sequence is at least as long as the sequence itself. The binary string 101010101010 is not random because there is an easy rule for creating it, whereas 111010001011 is unlikely to have a generating program much shorter than "print 111010001011." It turns out that almost all strings of bits are random by this criterion—they have no concise description—and

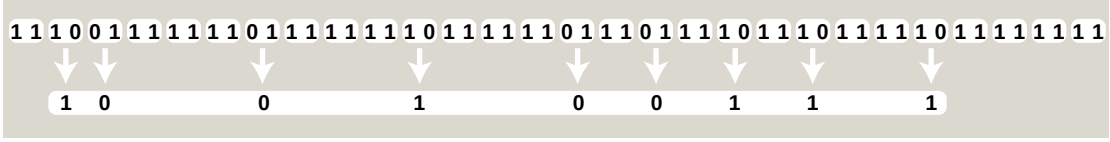

Figure 4. Biased stream of random bits is rebalanced through a trick invented by John von Neumann. The bits are taken two at a time. All 00 and 11 pairs are discarded, then each 01 is replaced by 0 and each 10 by 1. The bias is thereby eliminated, but at least three-fourths of the bits are thrown away. Repeating the procedure can correct some subtler flaws.

yet no one has ever exhibited a single string that is certified to be random. The reason is simple: The first string certified to have no concise description would thereby acquire a concise description—namely that it's the first such string.

The Chaitin-Kolmogorov definition is not the only aspect of randomness verging on the paradoxical or the ironic. Here is another example: True random numbers, captured in the wild, are clearly superior to those bred in captivity by pseudo-random generators—or at least that's what the theory of randomness implies. But Marsaglia has run the output of various hardware and software generators through a series of statistical tests. The best of the pseudo-random generators earned excellent grades, but three hardware devices flunked. In other words, the fakes look more convincingly random than the real thing.

To me the strangest aspect of randomness is its role as a link between the world of mathematical abstraction and the universe of ponderable matter and energy. The fact that randomness requires a physical rather than a mathematical source is noted by almost everyone who writes on the subject, and yet the oddity of this situation is not much remarked.

Mathematics and theoretical computer science inhabit a realm of idealized and immaterial objects: points and lines, sets, numbers, algorithms, Turing machines. For the most part, this world is self-contained; anything you need in it, you can make in it. If a calculation calls for the millionth prime number or the cube root of 2, you can set the computational machinery in motion without ever leaving the precincts of mathland. The one exception is randomness. When a calculation asks for a random number, no mathematical apparatus can supply it. There is no alternative but to reach outside the mathematical empyrean into the grubby world of noisy circuits and decaying nuclei. What a strange maneuver! If some purely mathematical statement—say the formula for solving a quadratic equation—depended on the mass of the earth or the diameter of the hydrogen atom, we would find this disturbing or absurd. Importing randomness into mathematics crosses the same boundary.

Of course there is another point of view: If we choose to look upon mathematics as a science limited to deterministic operations, it's hardly a surprise that absence-of-determinism can't be found there. Perhaps what is really extraordinary is not that randomness lies outside mathematics but that it exists anywhere at all.

Or does it? The savants of the 18th century didn't think so. In their clockwork universe the chain of cause and effect was never broken. Events that appeared to be random were merely too complicated to submit to a full analysis. If we failed to predict the exact motion of an object—a roving comet, a spinning coin—the fault lay not in the unruliness of the movement but in our ignorance of the laws of physics or the initial conditions.

The issue is seen differently today. Quantum mechanics has cast a deep shadow over causality, at least in microscopic domains. And "deterministic chaos" has added its own penumbra, obscuring the details of events that might be predicted in principle, but only if we could gather an unbounded amount of information about them. To a modern sensibility, randomness reflects not just the limits of human knowledge but some inherent property of the world we live in. Nevertheless, it seems fair to say that most of what goes on in our neighborhood of the universe is mainly deterministic. Coins spinning in the air and dice tumbling on a felt table are not conspicuously quantum-mechanical or chaotic systems. We choose to describe their behavior through the laws of probability only as a matter of convenience; there's no question the laws of angular momentum are at work behind the scenes. If there is any genuine randomness to be found in such events, it is the merest sliver of quantum uncertainty. Perhaps this helps to explain why digging for randomness in the flinty soil of physics is such hard work.

Bibliography

Aumann, Yonatan, and Michael O. Rabin. 1999. Information theoretically secure communication in the limited storage space model. In *CRYPTO '99: 19th Annual International Cryptology Conference*, Santa Barbara, Calif., August 15–19, 1999, pp. 65–79. Berlin: Springer-Verlag.

Ding, Yan Zong, and Michael O. Rabin. 2001. Provably secure and non-malleable encryption. Abstract.

Fisher, R. A., and F. Yates. 1938. *Statistical Tables for Biological, Agricultural and Medical Research*. London: Oliver & Boyd.

Ford, Joseph. 1983. How random is a coin toss? *Physics Today* (April 1983) pp. 40–47.

Intel Platform Security Division. 1999. The Intel random number generator. ftp://download.intel.com/design/security/rng/techbrief.pdf

Lord Kelvin. 1901. Nineteenth century clouds over the dynamical theory of heat and light. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, Series 6, 2:1–40.

Marsaglia, George. 1995. *The Marsaglia Random Number CDROM, Including the DIEHARD Battery of Tests of Randomness*. Tallahassee, Fla.: Department of Statistics, Florida State University.

Maurer, Ueli M. 1992. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology* 5(1):53–66.

Pearson, Karl. 1900. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, Series 5, 50:157–175.

The Rand Corporation. 1955. *A Million Random Digits with 100,000 Normal Deviates*. Glencoe, Ill.: Free Press.

Shannon, C. E. 1949. Communication theory of secrecy systems. *Bell System Technical Journal* 28:656–715.

Tippett, L. H. C. 1927. Random sampling numbers. *Tracts for Computers,* No. 15. London: Cambridge University Press.

Vincent, C. H. 1970. The generation of truly random binary numbers. *Journal of Physics E* 3(8):594–598.

von Neumann, John. 1951. Various techniques used in connection with random digits. In *Collected Works*, Vol. 5, pp. 768–770. New York: Pergamon Press.