HOW TO AVOID YOURSELF

Brian Hayes

Every Sunday morning you go for a walk in the city, heading nowhere in particular, with just one rule to your rambling: You never retrace your steps or cross your own path. If you have already walked along a certain block or passed through an intersection, you refuse to set foot there again.

This recipe for tracing a loopless path through a grid of city streets leads into some surprisingly dark back alleys of mathematics—not to mention byways of physics, chemistry, computer science and biology. Avoiding yourself, it turns out, is a hard problem. The exact analysis of self-avoiding walks has stumped mathematicians for half a century; even counting the walks is a challenge.

My own initiation into the trials of self-avoidance came when I began experimenting with a simple model of the folding of protein molecules, a story I told in an earlier "Computing Science" column (see Hayes 1998). Protein folding is close to the historical roots of the self-avoiding walk, which was first conceived as a tool for understanding the geometry of long-chain polymer molecules. A polymer writhing and wriggling in solution forms a random tangle—random, that is, except that no two atoms can occupy the same position at the same time. This "excluded volume effect" in the polymer is modeled by the walk's insistence on avoiding itself.

Self-avoiding walks have also found applications elsewhere in the sciences, such as the physics of magnetic materials and the study of phase transitions. And they are of interest as purely mathematical objects. Many of the obvious questions about them have resisted rigorous analysis, and so the best answers known so far come from computer-intensive methods.

All of the walks I shall describe here take place on a two-dimensional square lattice, which is a grid of city streets reduced to its mathematical essence. The lattice consists of all points on the plane that have integer x and y coordinates. Walks begin at the origin, the point with coordinates $x=0$ and $y=0$. A single step always moves from the current lattice site to one of the four nearest-neighbor sites. By convention the length of a walk, n, is defined as the number of steps, and so the number of lattice sites visited is $n+1$.

I Wonder as I Wander

In trying to understand the self-avoiding walk, a good place to begin is with a walk that doesn't bother to avoid itself but lurches over the landscape entirely at random. At each step of such a walk you choose one of the four neighboring lattice sites with equal probability, and move there. If you repeat this process a few hundred times, and draw a line behind you as you go, the result is a scribble with a random but nonetheless distinctive and recognizable geometry.

How many different random paths can be traced on a square lattice? They are easy to count. From any fixed point of origin there are just four walks consisting of a single step, namely those going one unit north, east, south or west. On the second step each of these walks can be continued in any of four directions, and so there are 16 two-step walks. For every further step the number of walks is again multiplied by 4, so that the number of n-step walks is 4^n.

An interesting question to ask about random walks is whether the walker ever returns to the starting point. Eighty years ago George Pólya showed that the answer depends on the dimensionality of the lattice. In one or two dimensions a random walker is certain to come back home if the walk continues long enough. But three or more dimensions offer enough room to get lost in, and a return cannot be guaranteed no matter how long the walk goes on.

Pólya's result immediately tells us something about self-avoiding walks on a two-dimensional lattice. If a random walk's probability of returning home is 1, the probability of *not* revisiting the origin must be 0. And since the origin is one of the places that self-avoiding walks avoid, an arbitrarily long self-avoiding walk must be highly improbable—so rare and exceptional that you have almost no chance of finding one. This scarcity is one reason self-avoiding walks are so hard to study. And yet, paradoxically, another reason is that they're so numerous it's a challenge to count them all.

Brian Hayes is a former editor of American Scientist. *Address: 211 Dacian Avenue, Durham, NC 27701. Internet: bhayes@amsci.org.*

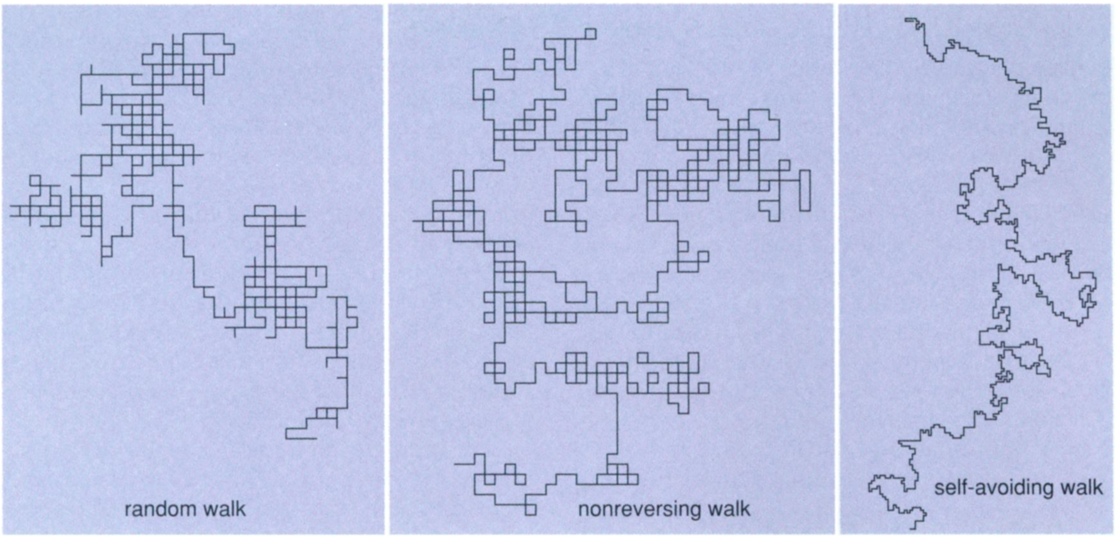


Figure 1. Random and nonreversing walks on the square lattice look very different from a self-avoiding walk, which is necessarily nonbranching. Each walk consists of 1,000 steps; the self-avoiding walk is shown at reduced scale.

Don't Look Back

An intermediate stage between a purely random walk and a self-avoiding walk is the nonreversing walk. As a recipe for an urban perambulation, it allows you to go left, right or forward at each intersection, but not to make a U-turn and go back the way you just came. Thus a nonreversing walker on a square lattice has four choices for the first step, but only three choices for each step thereafter. The number of n-step walks is $4 \times 3^{n-1}$, which for large n converges to 3^n.

Typical examples of random walks, nonreversing walks and self-avoiding walks can be distinguished at a glance. The random walk usually consists of dense regions, where most of the lattice points have been visited at least once, connected by tendrils through more sparsely settled territory. A trace of the walk looks something like a map of towns and cities along a river. The nonreversing walk is similar but suggests a more open landscape—perhaps suburban sprawl rather than the city center. And the trace of a self-avoiding walk looks not like cities along a river but like the river itself, or like a coastline.

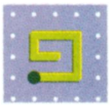
These differences in appearance are reflected in quantitative measures of the walks' geometry. One important measure is the square of the distance between the end points of the walk. For n-step random walks the mean-squared displacement is n; for nonreversing walks it is $2n$. Self-avoiding walks are qualitatively different. The mean-squared displacement grows as a nonlinear function of n, which appears to be $n^{3/2}$.

There is another important difference between random walks and self-avoiding walks. Every random walk can go on forever; you can always take one more step. But a self-avoiding walk can stumble into a blind alley, getting trapped at a lattice site where none of the neighbors are unvisited. In other words, sometimes you can't avoid yourself no matter how hard you try. On any given step the probability of getting boxed in is small—a little less than 1 percent—but if you extend a walk indefinitely, it is certain to wander into a dead end eventually. This is another way of saying that self-avoiding walks are rare and special; they have to beat the odds to survive.

Counting Your Steps

Just how many distinct n-step self-avoiding walks can you take on a square lattice? There is no simple exact formula, analogous to the expression 4^n that enumerates random walks, but upper and lower bounds can be stated. The number of self-avoiding walks has to be less than 3^n because that is the number of nonreversing walks, which include the self-avoiding walks as a subset. Similarly, it's easy to construct subsets of the self-avoiding walks whose numbers grow as 2^n; an example is the family of walks that move only north or east at each step. Thus the number of n-step self-avoiding walks should lie between 2^n and 3^n. Tighter bounds than these have been established, but still the only known way to get an exact tally is to actually trace out all the n-step walks and count them.

If you were asked to enumerate all the self-avoiding walks of, say, 15 steps, how would you answer? One useful rejoinder would be: Show me all the 14-step walks and I'll construct the 15-step ones. Adding the final step to each walk is straightforward: Just try each of the four possible directions, accepting a move if the neighboring site is vacant, and otherwise rejecting it. Then the question becomes how do you generate all the 14-step walks, and of course the answer is first to produce the 13-step walks. This regress continues back to the 0-step walk, which is just the single point at the origin.



trapped!

The procedure is uncomplicated, but the counting itself is arduous. There are 284 walks of five steps, and 44,100 of 10 steps. By $n=15$ the number of walks has reached 6,416,596, and at $n=20$ it is 897,697,164. The rate of growth is so steep and relentless that merely fine-tuning a program to improve its efficiency yields only a paltry reward. Counting the walks of $n+1$ steps takes longer than counting all the walks from 1 through n steps.

Most of the records for counting self-avoiding walks belong to A. J. Guttmann and his colleagues at the University of Melbourne. As early as 1972 Guttmann was part of a group (led by M. F. Sykes of the University of London) that counted all walks of up to 24 steps. In 1987 Guttmann raised the bar to 27 steps, then later to 29. Others then reached 30 and 34 steps, and Guttmann's group went on to 39. Then in 1996, in an extraordinary feat of computing, A. R. Conway and Guttmann enumerated all the self-avoiding walks through $n=51$. There are 14,059,415,980,606,050,644,844 walks of 51 steps. Performing this computation required an algorithm more sophisticated than the one sketched here, as well as an Intel Paragon supercomputer that dedicated 1,024 processors and 10 gigabytes of memory to the task.

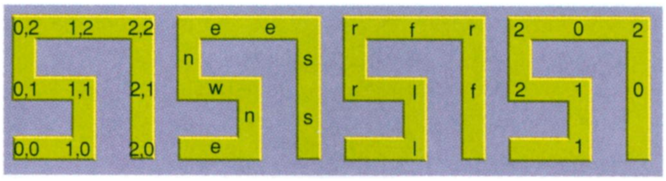


Figure 2. Lattice walks can be represented as a list of *x,y* coordinates, as compass directions or as *left, right* and *forward* commands. The last representation can also be encoded in a ternary (base 3) number.

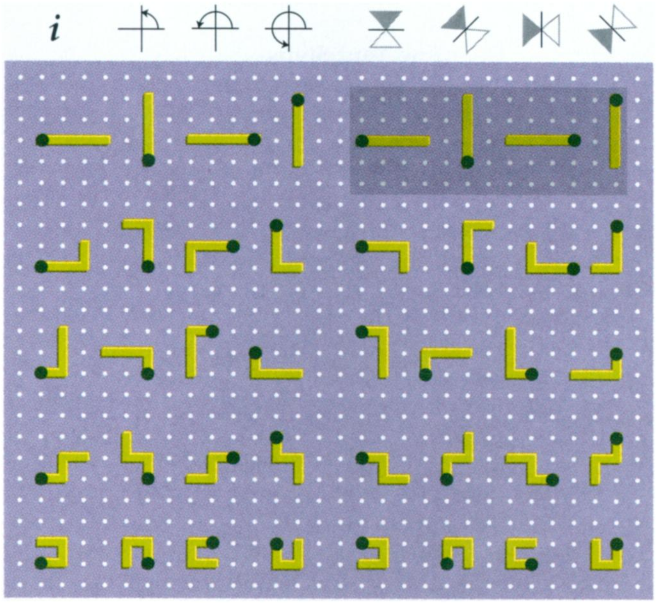


Figure 3. Symmetries reduce the number of unique walks by a factor of nearly 8. Each of the five three-step paths in the leftmost column is acted on by three rotations and four reflections, yielding 40 walks altogether; but for straight walks *(first row)* the effect of reflection is the same as that of rotation, reducing the total to 36 walks.

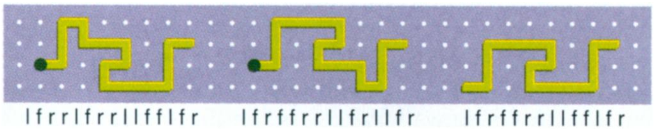


Figure 4. Unrooted walks have an additional symmetry. The two walks at left would be indistinguishable if the dots marking the starting points were erased. The walk at right is a palindrome, with a list of directions that reads the same in either direction.

Guttmann's long series of enumerations yields an estimate of the asymptotic growth rate in the number of walks—the rate to which the series apparently converges in the limit of large n. Based on the known data, increasing n by 1 multiplies the number of self-avoiding walks by about 2.638; in other words, the number of n-step walks is proportional to $n^{2.638}$.

Even though self-avoiding walks are so numerous we can't count any but the shortest of them, they still remain rarities among all possible lattice walks. At $n=20$ fewer than one walk in 1,200 is self-avoiding. At $n=50$, the ratio is one out of 240 million.

Coming and Going

If you look at a complete set of n-step self-avoiding walks, the first thing you're likely to notice is a lot of repetition. Many of the walks have the same basic shape; they differ only by a rotation or reflection.

On the square lattice any path can be rotated into four orientations and reflected across four axes (vertical, horizontal and two diagonals). Should the results of these transformations be considered eight distinct walks or eight variations on a single walk? The answer depends on what you want to do with the walks, but if you're merely counting them, it's clearly foolish to count by ones when you can count by eights. Programs for walk counting generate just one of the eight configurations, and then multiply to get the total number.

To eliminate the fourfold rotational symmetry of the lattice, you can generate only those walks that start with a step in some particular direction, say east. The four mirror symmetries disappear if you consider only walks whose first turn after the initial step is in one specified direction, say north. In this way the number of walks to be counted is reduced to approximately one-eighth the total number. Why *approximately*? Because of one small complication: A straight walk makes no turns away from its initial direction and is therefore left unchanged by mirror reflection. Hence there are only four distinguishable straight walks instead of eight, and the total number of walks is reduced by 4.

In the symmetries described so far, walks are considered to be *rooted*, that is, one end of each walk is distinguished as the starting end, as if it were planted in the ground. In many contexts, however, the direction in which a walk is traversed has no significance. For example, if you were folding a polymer along the path of a self-avoiding walk, you could start at either end of the walk. Viewing self-avoiding walks as unrooted paths makes many pairs of walks indistinguishable, so that the number of distinct *n*-step walks is reduced by a further factor of approximately 2.

I have been unable to find any published literature on enumerating unrooted self-avoiding walks, although it seems unlikely I am the first to consider the problem. The search for a practical algorithm proved an interesting challenge.

For unrooted walks, two paths through the lattice are identical if traversing one of them forward yields the same path as following the other one backward. The symmetry is easiest to understand when you represent a walk not as a list of lattice sites or as absolute directions but as instructions telling you what turns to take at every intersection along the path. Suppose the route from your home to your office is abbreviated [*l f r f f*], where *l* stands for *left*, *r* for *right* and *f* for *forward*. (The notation is borrowed from the turtle graphics system of the Logo programming language.) On your way home, if you want to retrace your steps, you would obey the turtle-graphics commands [*f f l f r*]. The two lists of instructions are related by a transformation I shall call retroreflection: The sequence of letters is reversed, and all the *lefts* and *rights* are interchanged.

In writing a program to enumerate the unrooted self-avoiding walks, the aim is to select one member of each retroreflected pair and discard the other. (It doesn't matter which one is kept.) The direct solution would be to maintain an archive of all the walks seen so far. Then as you generate each new walk, you reverse the list of turtle-graphics commands, flip the *lefts* and *rights*, and compare the result with the archive of stored walks, keeping the new walk only if the retroreflected version hasn't been seen already. This strategy would work, but it would be hideously slow and a memory hog.

There is a better way. The key idea is to transform the list of turtle-graphics commands into a number, specifically a ternary (base 3) number, with the digit 0 representing *forward*, 1 representing *left* and 2 representing *right*. Then every *n*-step self-avoiding walk has a unique encoding as an (*n*–1)-digit ternary number; equally important, every (*n*–1)-digit ternary number specifies an *n*-step nonreversing (though not necessarily self-avoiding) walk. For example, counting in ternary from 0000, 0001, 0002, 0010 up to 2222 generates every five-step nonreversing walk. Discarding the walks that fail a test for self-intersections leaves just the self-avoiding five-step walks. (In practice, it's better to write the walk numbers in "balanced ternary" notation, where the digits are –1, 0 and +1; then interchanging *left* and *right* is just multiplying by –1. Balanced ternary is the only numbering system I know that has its own Web page.)

The reason for treating the walks as numbers is that it imposes a total ordering on them. Given any two distinct finite numbers, there is always a larger and a smaller. Likewise, given any two distinct walks related by the retroreflective symmetry, one walk's ternary encoding must be less than the other's. This immediately suggests an efficient algorithm for splitting the symmetrical pairs: As you generate each walk, compare the ternary representation with its retroreflection. If the original number is greater, discard the walk; otherwise keep it. In this scheme there is no need to maintain and search an archive of walks; every decision is made with a single numerical comparison.

n	unrooted walks	total self-avoiding walks
1	1	4
2	2	12
3	4	36
4	9	100
5	22	284
6	56	780
7	147	2,172
8	388	5,916
9	1,047	16,268
10	2,806	44,100
11	7,600	120,292
12	20,437	324,932
13	55,313	881,500
14	148,752	2,374,444
15	401,629	6,416,596
16	1,078,746	17,245,332
17	2,905,751	46,466,676
18	7,793,632	124,658,732
19	20,949,045	335,116,620
20	56,112,530	897,697,164
21	150,561,752	2,408,806,028
22	402,802,376	6,444,560,484
23	1,079,193,821	17,266,613,812
24	2,884,195,424	46,146,397,316

Figure 5. Exact enumerations show the unrooted walks are not quite one-sixteenth the rooted ones.

Figure 5 gives the number of unrooted self-avoiding walks of up to 24 steps. As would be expected, the numbers are approximately one-sixteenth the number of all self-avoiding walks of the

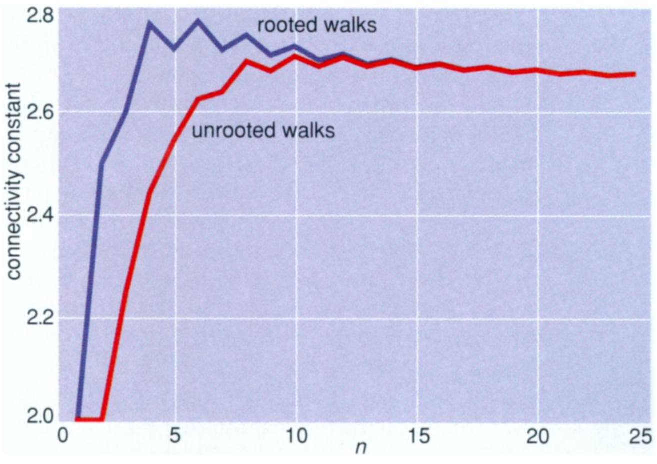


Figure 6. Connectivity constant measures the average number of sites available to a walk and determines the rate of growth in the number of walks. The limiting value for large *n* is believed to be about 2.638.

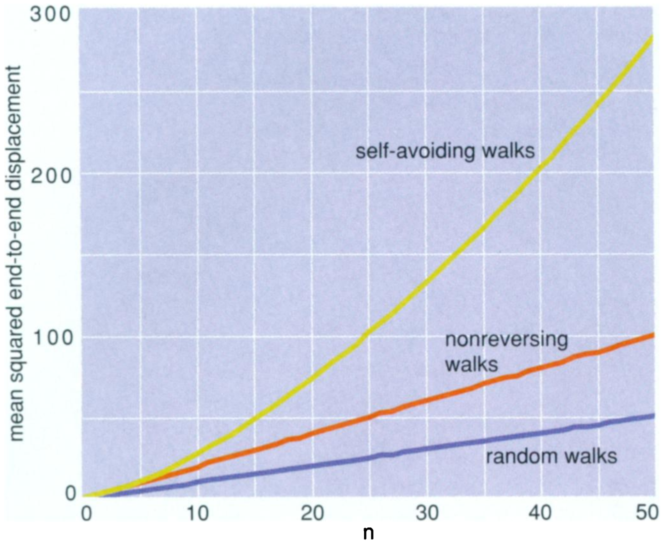


Figure 7. Mean-squared end-to-end distance grows as a linear function of *n* for random and nonreversing walks, but for self-avoiding walks seems to be equal to the 3/2 power of *n*.

same length, and the ratio approaches 1:16 more closely as *n* increases; and yet the ratio is not exact. The reason is another little complication, analogous to the problem of straight walks that are invariant under reflection. In this case a class of walks pass through retroreflection unchanged. They are the palindromic walks. Consider the path [*f l f r f*]; if you reverse the sequence of instructions and interchange *lefts* and *rights*, you wind up with the same sequence of commands again. The path is its own retroreflection. To get a correct census of the unrooted walks it's crucial that such paths be counted once and only once.

Random Thoughts on Self-Avoidance

Unless a new algorithm comes along, exhaustive enumerations of self-avoiding walks seem unlikely to advance much beyond the current limit of $n=51$. Knowledge of longer walks has come mainly from random sampling. This process too is computationally intensive.

For most purposes, a random sample is meant to be selected with uniform probability from the set of all *n*-step self-avoiding walks. Unfortunately, the most obvious algorithm does not yield walks with this distribution. It's easy enough to build an *n*-step walk one step at a time by choosing directions at random; the question is what to do if the walk collides with itself before reaching *n* steps. The temptation is simply to back up one step and try another direction, but that practice leads to a biased sample of walks. To ensure a fair sample you have to abandon a failed walk entirely and start over.

My own experiments with random sampling have relied on the ternary-number representation. I choose an *n*-digit balanced-ternary number at random, then check the corresponding walk for self-intersections. If the walk fails the test, I generate a new random number and try again.

Algorithms like this one readily produce large samples of 60- or 70-step walks, or smaller numbers of 100-step walks. As the walks get longer, however, the proportion of candidates that pass the self-avoidance test declines sharply. At $n=100$ you are proposing more than 50,000 walks for every one that turns out to be self-avoiding. At $n=200$ the acceptable walks would be rarer than one in a billion.

Other algorithms extend the range of exploration into the thousands of steps. Thirty years ago Zeev Alexandrowicz of the Weizmann Institute of Science suggested a method called dimerization, which exploits a divide-and-conquer strategy familiar in many other areas of computer science. Dimerization works because it's much easier to create two 50-step walks than a single 100-step walk. You build the two shorter walks, and string them together end-to-end. Of course the two half-walks may collide, in which case you have to start over, but failure turns out to be much less likely than in the step-by-step technique. The procedure can be invoked recursively to build the 50-step walks from 25-step components, and so on. What's particularly sweet about this algorithm is that it lends itself to a very simple and transparent implementation; I found it easier to get right than the less-efficient step-by-step methods.

Another technique, called the pivot algorithm, also goes back 30 years; it was first described by Moti Lal of the Unilever Research Laboratory and more recently has been refined and extended by Neal Madras of York University and Alan D. Sokal of New York University. The pivot algorithm is quite different from all the others described here. It does not actually generate a self-avoiding walk but instead takes one walk and transforms it into another. The idea is to randomly choose a pivot point somewhere along the walk, and then rotate or reflect or reverse the segment on one side of the pivot. If the result is a self-avoiding path, the move is accepted; otherwise you choose a new pivot and try again. Successive walks in the sequence are highly correlated, but repeating the transformation many times wipes out all memory of former configurations.

Rigorous Self-Avoidance

Computational studies of self-avoiding walks have produced a rich harvest of empirical results. Theorems have been harder to come by. For example, studies of the mean-squared end-to-end displacement, based on both complete enumerations and on random samples, strongly support the hypothesis mentioned above that the displacement grows as $n^{3/2}$. Indeed, everyone "knows" that this result is correct and exact. But so far no one has proved it; no one has even proved that the exponent must be greater than 1 or less than 2.

Rigorous results on the counting of self-avoiding walks are also scarce. The empirical evidence suggests that for large *n* the number of walks grows as $n^{2.638}$, but this growth law has not been explained from first principles. Until recently, it wasn't even certain that the number of walks invariably increases as n gets larger; because walks can become trapped, it seemed possible that there might be some range of values where there are fewer $(n+1)$-step walks than n-step walks. In 1990, however, George L. O'Brien proved that the series increases monotonically.

Even if the asymptotic growth law is correct, however, it is only an approximation—perhaps good enough for chemists and physicists but not wholly satisfying to mathematicians. Ideally, one would like a formula for calculating the exact number of walks for any value of n, without all the laborious counting. Is that too much to ask? Most likely it is. Conway and Guttmann have given compelling arguments (though not quite a proof) that no simple analytic function predicts the exact number of self-avoiding walks.

Perhaps the absence of such a function tells us something important about the nature of self-avoiding walks. The number of walks is perfectly definite and knowable; there is nothing random or uncertain about the number of ways to arrange a nonintersecting path on a lattice. So why can't we calculate it? I don't know the answer, but I would point out that there are many objects in mathematics that exhibit the same curious mixture of determinism and unpredictability. The prime example is the prime numbers. Again there is nothing uncertain or statistical about what makes a number prime, but if there is any pattern in the distribution of the primes, it remains totally inscrutable. As with the self-avoiding walks, there are good approximations for the number of primes, but no one has found (or expects to find) an exact formula that will reliably point to every prime. This stubborn resistance to total analysis is part of what makes the primes interesting. Perhaps self-avoiding walks belong in the same category of perpetually tantalizing mathematical structures.

Bibliography

Alexandrowicz, Z. 1969. Monte Carlo of chains with excluded volume: a way to evade sample attrition. *Journal of Chemical Physics*. 51:561–565.

Allwright, James. Balanced Ternary Web Pages. http://castle.ecs.soton.ac.uk:8080/cgi-bin2/ternary

Conway, A. R., I. G. Enting and A. J. Guttmann. 1993. Algebraic techniques for enumerating self-avoiding walks on the square lattice. *Journal of Physics A: Mathematical and General Physics* 26:1519–1534.

Conway, A. R., and A. J. Guttmann. 1996. Square lattice self-avoiding walks and corrections to scaling. *Physical Review Letters* 77:5284–5287.

Domb, C., and M. E. Fisher. 1958. On random walks with restricted reversals. *Proceedings of the Cambridge Philosophical Society* 54:48–59.

Guttmann, A. J., T. Prellberg and A. L. Owczarek. 1993. On the symmetry classes of planar self-avoiding walks. *Journal of Physics A: Mathematical and General Physics* 26:6615–6623.

Guttmann, A. J., and Jian Wang. 1991. The extension of self-avoiding random walk series in two dimensions. *Journal of Physics A: Mathematical and General Physics* 24:3107–3109.

Hayes, Brian. 1998. Computing Science: Prototeins. *American Scientist* 86:216–221.

Hughes, Barry D. 1995. *Random Walks and Random Environments. Vol. 1: Random Walks.* Oxford: Clarendon Press.

Lal, Moti. 1969. 'Monte Carlo' computer simulation of chain molecules, I. *Molecular Physics* 17:57–64.

Madras, Neal, and Gordon Slade. 1993. *The Self-Avoiding Walk.* Boston: Birkhäuser.

Madras, Neal, and Alan D. Sokal. 1988. The pivot algorithm: A highly efficient Monte Carlo method for the self-avoiding walk. *Journal of Statistical Physics* 50:109–186.

O'Brien, George L. 1990. Monotonicity of the number of self-avoiding walks. *Journal of Statistical Physics* 59:969–979.

Slade, Gordon. 1994. Self-avoiding walks. *The Mathematical Intelligencer* 16(1):29–35.

Slade, Gordon. 1996. Random walks. *American Scientist* 84:146–153.

Sykes, M. F. 1961. Some counting theorems in the theory of the Ising model and the excluded volume problem. *Journal of Mathematical Physics* 2:52–62.

Wang, Jian. 1989. A new algorithm to enumerate the self-avoiding random walk. *Journal of Physics A: Mathematical and General Physics* 22:L969–L971.

The text of this "Computing Science" column and links to Internet resources for further exploration are available on the *American Scientist* Web site:

http://www.amsci.org/amsci/issues/comsci98/compsci1998-07.html