

an Article from | **SCIENTIFIC
AMERICAN**

SEPTEMBER, 1984 VOL. 251, NO. 3

Computer Software in Science and Mathematics

Computation offers a new means of describing and investigating scientific and mathematical systems. Simulation by computer may be the only way to predict how certain complicated systems evolve

by Stephen Wolfram

Scientific laws give algorithms, or procedures, for determining how systems behave. The computer program is a medium in which the algorithms can be expressed and applied. Physical objects and mathematical structures can be represented as numbers and symbols in a computer, and a program can be written to manipulate them according to the algorithms. When the computer program is executed, it causes the numbers and symbols to be modified in the way specified by the scientific laws. It thereby allows the consequences of the laws to be deduced.

Executing a computer program is much like performing an experiment. Unlike the physical objects in a conventional experiment, however, the objects in a computer experiment are not bound by the laws of nature. Instead they follow the laws embodied in the computer program, which can be of any consistent form. Computation thus extends the realm of experimental science: it allows experiments to be performed in a hypothetical universe. Computation also extends theoretical science. Scientific laws have conventionally been constructed in terms of a particular set of mathematical functions and constructs, and they have often been developed as much for their mathematical simplicity as for their capacity to model the salient features of a phenomenon. A scientific law specified by an algorithm, however, can have any consistent form. The study of many complex systems, which have resisted analysis by traditional mathematical methods, is consequently being made possible through computer experiments and computer models. Computation is emerging as a major new approach to science, supplementing the long-standing methodologies of theory and experiment.

There are many scientific calculations, of course, that can be done by conventional mathematical means, without the aid of the computer. For example,

given the equations that describe the motion of electrons in an arbitrary magnetic field, it is possible to derive a simple mathematical formula that gives the trajectory of an electron in a uniform magnetic field (one whose strength is the same at all positions). For more complicated magnetic fields, however, there is no such simple mathematical formula. The equations of motion still yield an algorithm from which the trajectory of an electron can be determined. In principle the trajectory could be worked out by hand, but in practice only a computer can go through the large number of steps necessary to obtain accurate results.

A computer program that embodies the laws of motion for an electron in a magnetic field can be used to perform computer experiments. Such experiments are more flexible than conventional laboratory experiments. For example, a laboratory experiment could readily be devised to study the trajectory of an electron moving under the influence of the magnetic field in a television tube. No laboratory experiment, however, could reproduce the conditions encountered by an electron moving in the magnetic field surrounding a neutron star. The computer program can be applied in both cases.

The magnetic field under investiga-

tion is specified by a set of numbers stored in a computer. The computer program applies an algorithm that simulates the motion of the electron by changing the numbers representing its position at successive times. Computers are now fast enough for the simulations to be carried out quickly, and so it is practical to explore a large number of cases. The investigator can interact directly with the computer, modifying various aspects of a phenomenon as new results are obtained. The usual cycle of the scientific method, in which hypotheses are formulated and then tested, can be followed much faster with the aid of the computer.

Computer experiments are not limited to processes that occur in nature. For example, a computer program can describe the motion of magnetic monopoles in magnetic fields, even though magnetic monopoles have not been detected in physical experiments. Moreover, the program can be modified to embody various alternative laws for the motion of magnetic monopoles. Once again, when the program is executed, the consequences of the hypothetical laws can be determined. The computer thus enables the investigator to experiment with a range of hypothetical natural laws.

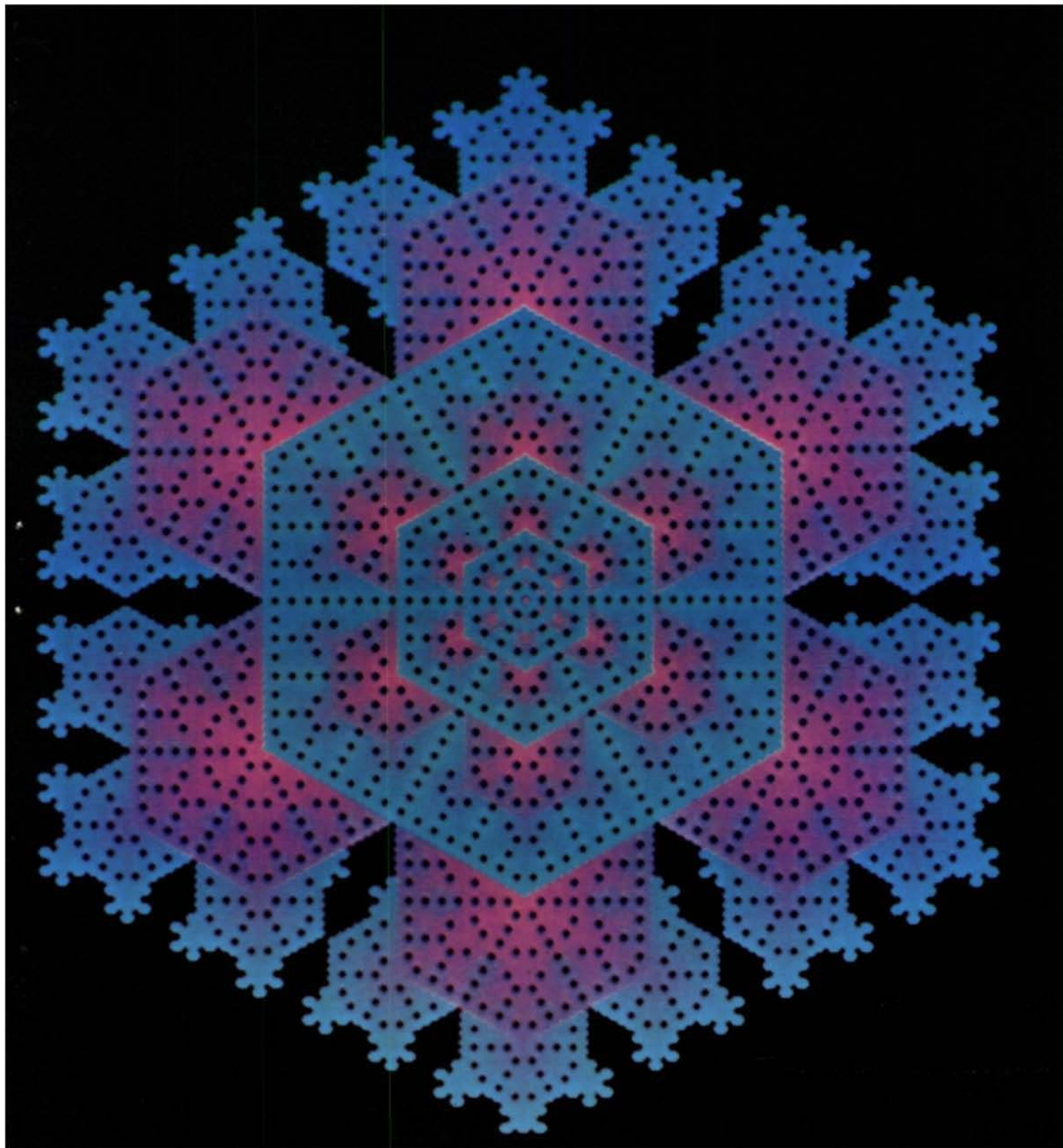
COMPUTER SIMULATION has made it practical to consider many new kinds of models for natural phenomena. Here the stages in the formation of a snowflake are generated by a computer program that embodies a model called a cellular automaton. According to the model, the plane is divided into a lattice of small, regular hexagonal cells. Each cell is assigned the value 0, which corresponds to water vapor (black), or the value 1, which corresponds to ice (color). Beginning with a single red cell in the center of the illustration, the simulated snowflake grows in a series of steps. At each step the subsequent value of any cell on the boundary of the snowflake depends on the total value of the six cells that surround it. If the total value is an odd number, the cell becomes ice and takes on the value 1; otherwise the cell remains vapor and keeps the value 0. The successive layers of ice formed in this way are shown as a sequence of colors, ranging from red to blue every time the number of layers doubles. The calculation required for each cell is simple, but for the pattern shown more than 10,000 calculations were needed. The only practical way to generate the pattern is by computer simulation. The illustration was made with the aid of a program written by Norman H. Packard of the Institute for Advanced Study.

The computer can also be used to study the properties of abstract mathematical systems. Mathematical experiments carried out by computer can often suggest conjectures that are subsequently established by conventional mathematical proof. Consider a mathematical system that can be introduced to model the path of a beam of elec-

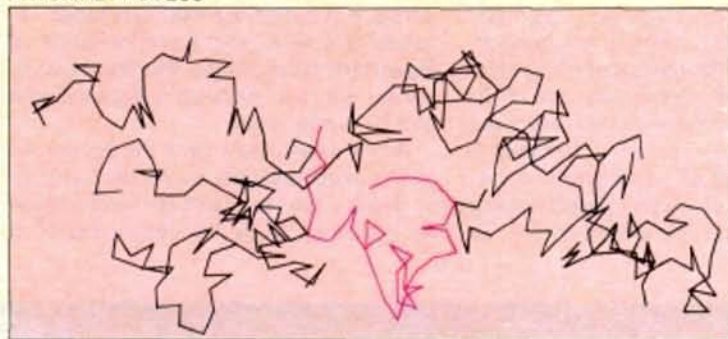
trons traveling through the magnetic fields in a circular particle accelerator. The transverse displacement of an electron as it passes a point on one of its revolutions around the accelerator ring is given by some fraction x between 0 and 1. The value of the fraction corresponding to the electron's displacement on the next revolution is then $ax(1 - x)$,

where a is a number that can range between 0 and 4. The formula gives an algorithm from which the sequence of values for the electron's displacement can be worked out.

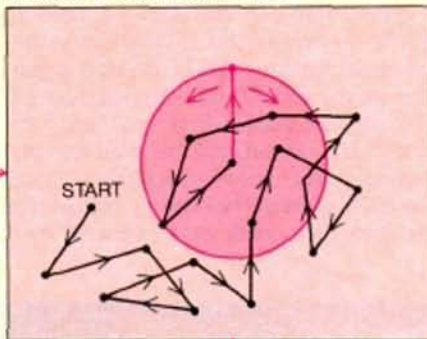
A few trials show how the properties of the sequence depend on the value of a . If a is equal to 2 and the initial value of x is equal to .8, the next value of x ,



PHYSICAL PROCESS



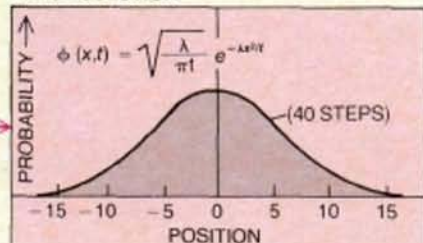
ALGORITHMIC DESCRIPTION



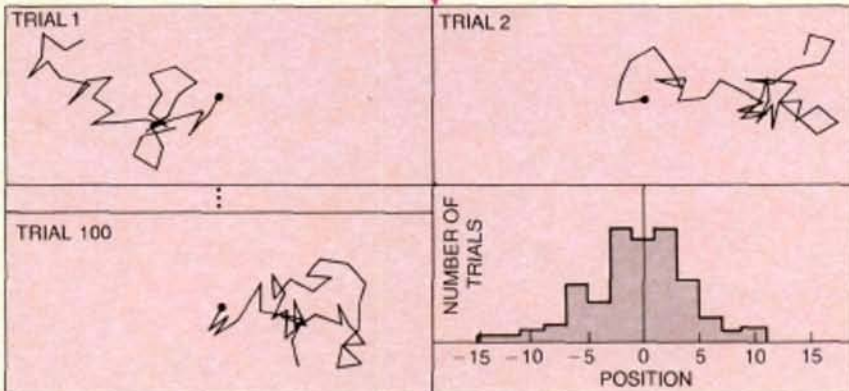
DIFFERENTIAL EQUATION

$$\frac{\partial \phi}{\partial t} = \lambda \left(\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} \right)$$

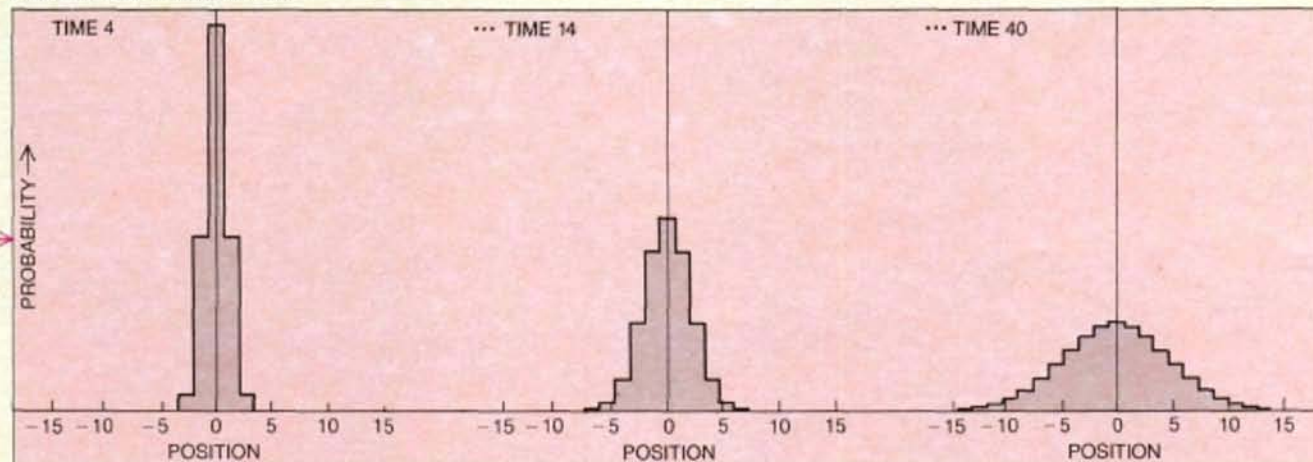
EXACT SOLUTION



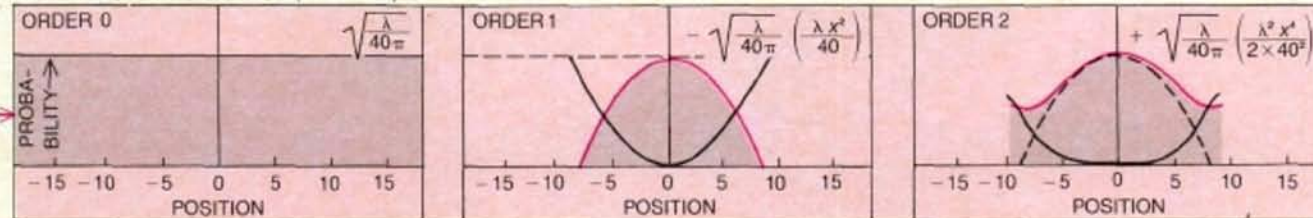
COMPUTER EXPERIMENT (40 STEPS)



NUMERICAL APPROXIMATION



ALGEBRAIC APPROXIMATION (40 STEPS)



MATHEMATICAL AND COMPUTATIONAL METHODS are applied in various ways in the study of random walks. A random walk is a model for such physical processes as the Brownian motion of a small particle suspended in a liquid. The particle undergoes random deflections as it is bombarded by the molecules in the liquid; its path can thus be described as a sequence of steps, each taken in a random direction. The most direct way to deduce the consequences of the model is by a computer experiment. Many random walks are simulated on a computer and their average properties are measured. The diagram shows a histogram in which the height of each bin records the number of simulated random walks that were found to have reached a particular range of positions after a certain time. As more trials are included, the shape of the histogram approaches that of the exact distribution of positions. For an ordinary random walk it is possible to derive the exact distribution directly: A differential equation can be

constructed for the distribution, and the equation is simple enough for an exact solution to be given. For most differential equations, however, no such exact solution can be obtained, and approximations must be made. In numerical approximations the smooth variation of quantities in the differential equation is approximated by a large number of small increments. The results shown in the diagram were obtained by a computer program in which the spatial and temporal increments were small fractions of the lengths and times for individual steps in the random walk. Algebraic approximations to the differential equation are found as a series of algebraic terms. The diagram shows the first three terms in such a series. The contribution of each term is shown as a solid black line or curve. The line or curve is added by superposition to the broken black line or curve that represents the previous order in the approximation. The result of the superposition is the current order in the approximation (solid colored curves).

which is given by $ax(1-x)$, is equal to .32. If the formula is applied again, the value of x obtained is .4352. After several iterations the sequence of values for x converges to .5. Indeed, when a is small and x is any fraction between 0 and 1, the sequence quickly settles down to give the same value of x for each revolution of the electron.

As a increases, however, a phenomenon called period doubling can be observed. When a reaches 3, the sequence begins to alternate between two values of x . As a continues to increase, first four, then eight and finally, when it reaches about 3.57, an entire range of values for x appear. This behavior could not readily be guessed from the construction of the mathematical system, but it is immediately suggested by the computer experiment. The detailed properties of the system can then be established by a conventional proof.

The mathematical processes that can be described by a computer program are not limited to the operations and functions of conventional mathematics. For example, there is no conventional mathematical notation for the function that reverses the order of the digits in a number. Nevertheless, it is possible to define and apply the function in a computer program. The computer makes it practical to introduce scientific and mathematical laws that are intrinsically algorithmic in nature. Consider the chain of events set up when an electron accelerated to a high energy is fired into a block of lead. There is a certain probability that the electron emits a photon of a particular energy. If a photon is emitted, there is a certain probability that it gives rise to a second electron and a positron (the antiparticle of the electron). Each member of the pair can in turn emit more photons, so that a cascade of particles is eventually generated. There is no simple mathematical formula that can describe even the elements of the process. Nevertheless, an algorithm for the process can be incorporated into a computer program, and the outcome of the process can be deduced by executing the program. The algorithm serves as the basic law that describes the process.

The mathematical basis of most conventional models of natural phenomena is the differential equation. Such an equation gives relations between certain quantities and their rates of change. For example, a chemical reaction proceeds at a rate proportional to the concentrations of the reacting chemicals, and that relation can be expressed by a differential equation. A solution to the equation would give the concentration of each reactant as a function of time. In some simple cases it is possible to find a complete solution to the equation in terms of standard mathematical functions. In most cases, however, no

such exact solution can be obtained, and one must resort to approximation.

The commonest approximations are numerical. Suppose one term of a differential equation gives the instantaneous rate of change of a quantity with time. The term can be approximated by the total change in the quantity over some small interval and then substituted into the differential equation. The resulting equation is in effect an algorithm that determines the approximate value of the quantity at the end of an interval, given its value at the beginning of the interval. By applying the algorithm repeatedly for successive intervals, the approximate variation of the quantity with time can be found. Smaller intervals yield more accurate results. The calculation required for each interval is quite simple, but in most cases it must be repeated many times to achieve an acceptable level of accuracy. Such an approach is practical only with a computer.

The numerical methods embodied in computer programs have been employed to find approximate solutions to differential equations in a wide variety of disciplines. In some cases the solutions have a simple form. In many cases, however, the solutions show complicated, almost random behavior, even though the differential equations from which they arise are quite simple. For such cases experimental mathematics must be used.

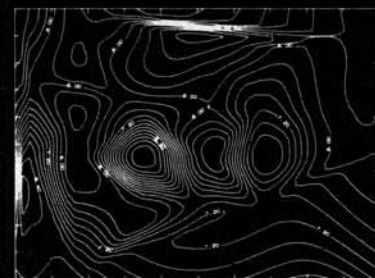
In practical applications one often finds not only that differential equations are complicated but also that there are many of them. For example, the theoretical models of nuclear explosions employed in the design of weapons and the study of supernovas involve hundreds of differential equations that describe the interactions of many isotopes. In practice such models are always used in the form of computer programs: only a computer can follow the interrelations among so many quantities.

The results of some numerical calculations, such as the abundance of helium in the universe, can be stated as single numbers. In most cases, however, one is concerned with the variation of certain quantities as the parameters of a calculation are changed. When the number of parameters is only one or two, the results can be displayed as a graph. When there are more than two parameters, however, the results often can be stated succinctly only as a mathematical formula. Exact formulas usually cannot be found, but it is often possible to derive approximate formulas. Such formulas are particularly convenient because, unlike graphs or tables of numbers, they can be inserted directly into other calculations.

A common form for an approximate formula is a series of terms. Each term includes a variable raised to some power

Contouring

Mainframe Power on the IBM Personal Computer with the In-Situ CONTUR Software



CONTUR is useful for a variety of applications, including geology, biology, chemistry, physics, mathematics, civil engineering, hydrology, and mining. The program supports Hewlett-Packard and Houston Instruments plotters.

For more information about CONTUR and other In-Situ software please call us.



In-Situ Inc.

P.O. Box 1
Laramie, WY 82070 USA

Telephone: 307 742-8213
TWX: 910-949-4944

AUDIO-FORUM® offers the best in self-instructional foreign language courses using audio cassettes — featuring those used to train U.S. State Dept. personnel in Spanish, French, German, Portuguese, Japanese, Greek, Hebrew, Arabic, Chinese, Italian, and more.

Learn a foreign language on your own! Free Catalog

Call (203) 453-9794, or fill out and send this ad to —
Audio-Forum

Room 604, On-the-Green
Guilford, CT 06437

Name _____

Address _____

City _____

State/Zip _____

I am particularly interested in (check choice):

- ☐ Spanish ☐ French ☐ German ☐ Polish
☐ Greek ☐ Russian ☐ Vietnamese
☐ Bulgarian ☐ Turkish ☐ Hausa
☐ Other _____

er; the power is larger in each successive term. When the value of the variable is small, the terms in the series become progressively smaller; thus for small values of x the sum of the first few terms in an infinite series such as $1 - x + x^2 - x^3 + \dots$ gives an accurate approximation to the sum of the entire series, which is $1/(1+x)$. The first few terms in a series are usually easy to evaluate, but the complexity of the terms increases rapidly thereafter. In order to evaluate terms that include large powers of x the computer becomes essential.

In principle computer programs can operate with any well-defined mathematical construct. In practice, however, the kinds of construct that can be used in a particular program are largely determined by the computer language in which the program is written. Numerical methods require only a limited set of mathematical constructs, and the programs that embody such methods can be written in general-purpose computer languages such as C, FORTRAN or BASIC. The derivation and manipulation of formulas require operations on higher-level mathematical constructs such as algebraic expressions, for which new computer languages are needed. Among the languages of this kind now in use is the SMP language that I have developed.

SMP is a language for manipulating symbols. It operates not only with numbers but also with symbolic expressions that can represent mathematical formulae.

For example, in SMP the algebraic expression $2x - 3y + 5x - y$ would be simplified to the form $7x - 4y$. This transformation is a general one, valid for any possible numerical values of x and y . The standard operations of algebra and mathematical analysis are among the fundamental instructions in SMP [see illustration on page 196].

The SMP language also includes operations that allow higher-level mathematical constructs to be defined and manipulated, much as they are in ordinary mathematical work. Real numbers (which include all rational and irrational values) as well as complex numbers (which have both a real and an imaginary part) are fundamental in SMP. The mathematical constructs known as quaternions, which are generalizations of the complex numbers, are not fundamental. They can nonetheless be defined in SMP, and rules can be specified for their addition and multiplication. In this way the mathematical knowledge of SMP can be extended.

Some of the advantages of a language such as SMP can be compared to the advantages of using a calculator instead of a table of logarithms. By now the widespread availability of electronic calculators and computers has made such tables obsolete: it is far more convenient to call on an algorithm in a computer to obtain a logarithm than it is to look up the result in a table. Similarly, with a language such as SMP it has become possible

to make the entire range of mathematical knowledge available in algorithmic form. For example, the calculation of integrals, conventionally done with the aid of a book of tables, can increasingly be left to a computer. The computer not only carries out the final calculations quickly and without error but also automates the process of finding the relevant formulas and methods.

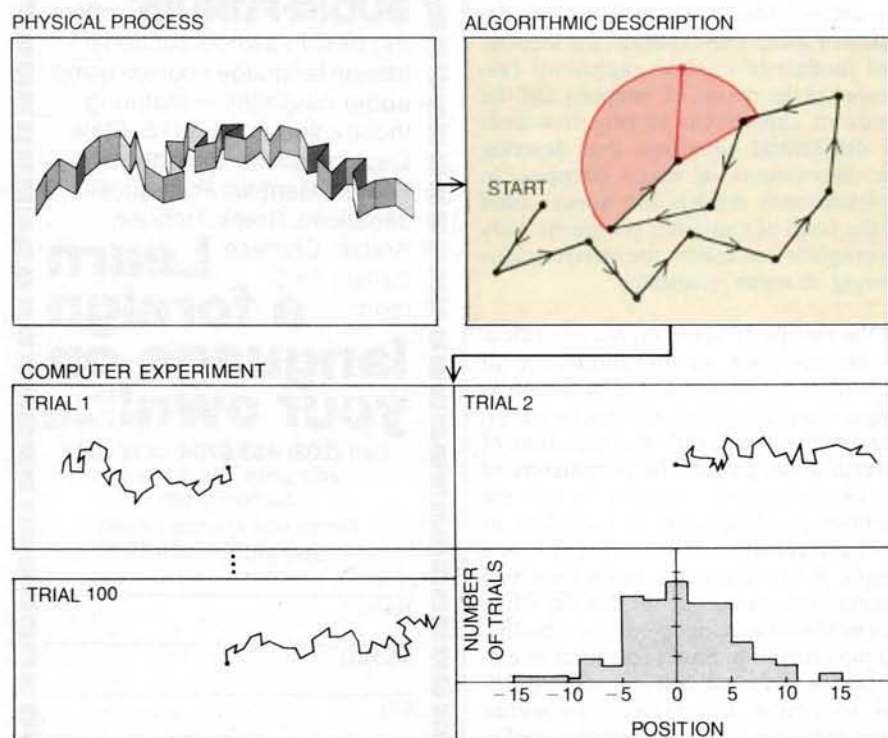
In SMP an expanding collection of definitions is being assembled in order to provide for a wide variety of mathematical calculations. One can now find in SMP the definition of variance in statistics, and one can immediately apply the definition to calculate the variance in a particular case. Such definitions enable programs written in the SMP language to call on increasingly sophisticated mathematical knowledge.

Differential equations give adequate models for the overall properties of physical processes such as chemical reactions. They describe, for example, the changes in the total concentration of molecules; they do not, however, account for the motions of individual molecules. These motions can be modeled as random walks: the path of each molecule is like the path that might be taken by a person in a milling crowd. In the simplest version of the model the molecule is assumed to travel in a straight line until it collides with another molecule; it then recoils in a random direction. All the straight-line steps are assumed to be of equal length. It turns out that if a large number of molecules are following random walks, the average change in the concentration of molecules with time can in fact be described by a differential equation called the diffusion equation.

There are many physical processes, however, for which no such average description seems possible. In such cases differential equations are not available and one must resort to direct simulation. The motions of many individual molecules or components must be followed explicitly; the overall behavior of the system is estimated by finding the average properties of the results. The only feasible way to carry out such simulations is by computer experiment: essentially no analysis of the systems for which analysis is necessary could be made without the computer.

The self-avoiding random walk is an example of a process that can apparently be studied only by direct simulation. It can be described by a simple algorithm that is similar to the ordinary random walk. It differs in that the successive steps in the self-avoiding random walk must not cross the path taken by any previous steps. The folding of long molecules such as DNA can be modeled as a self-avoiding random walk.

The introduction of the single con-



COMPUTATIONAL METHODS alone are used in the study of self-avoiding random walks. Self-avoiding random walks, which arise as models for physical processes such as the folding of polymer molecules, differ from ordinary random walks in that each step must avoid all previous steps. The complication makes it impossible to construct a simple differential equation that describes the average properties of the walk. Conventional mathematical approaches are thus ineffective. Properties of the self-avoiding random walk are found by direct simulation.

straint makes the self-avoiding random walk much more complicated than the ordinary random walk. Indeed, there is no simple average description, analogous to the diffusion equation, that is known for the self-avoiding random walk. In order to investigate its prop-

erties it seems one has no choice but to carry out a direct computer experiment. The procedure is to generate a large number of sample random walks, choosing a random direction at each step. The properties of all the walks are then averaged. Such a procedure is an

example of the Monte Carlo method, so called because its application depends on the element of chance.

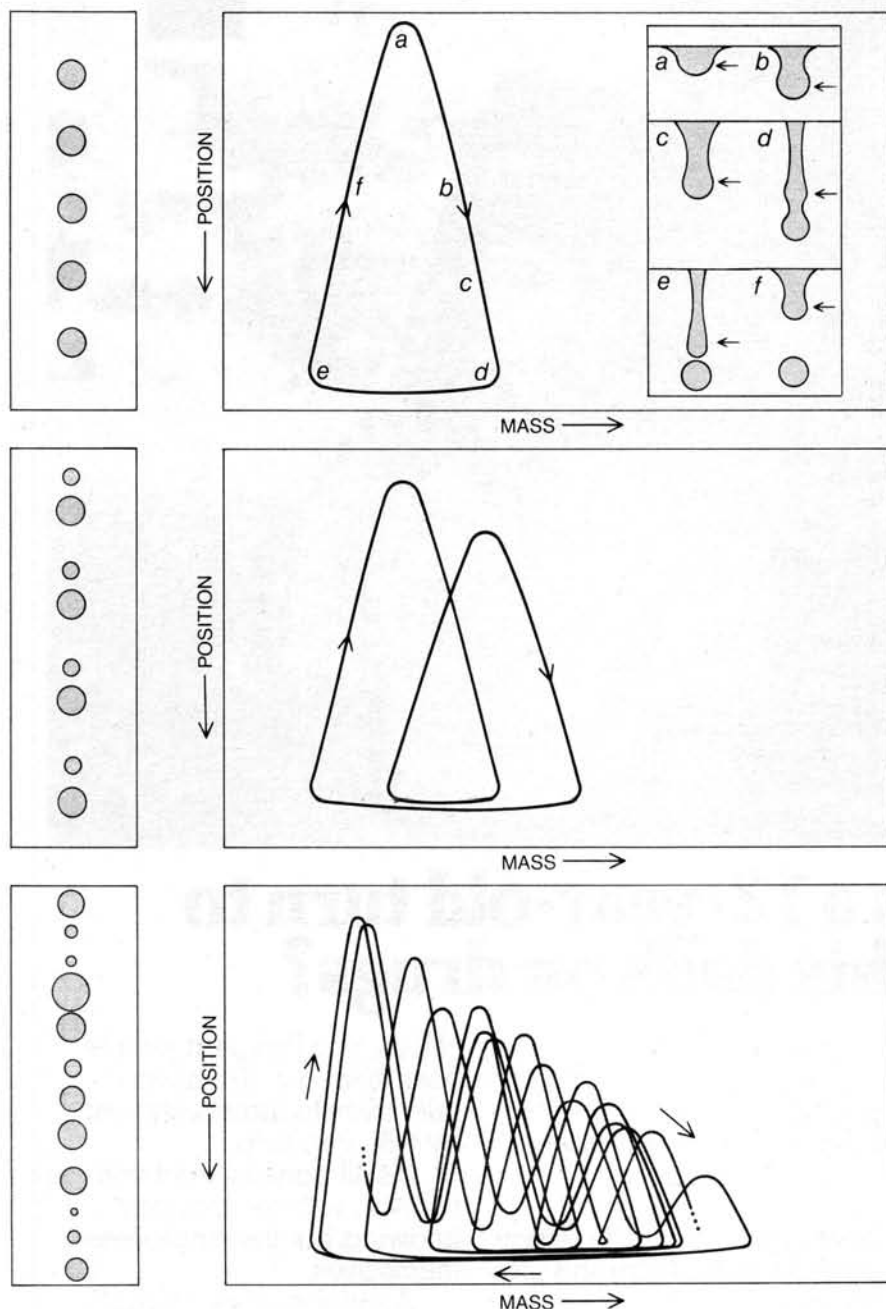
Several examples have been given of systems whose construction is quite simple but whose behavior is extremely complicated. The study of such systems is leading to a new field called complex-systems theory, in which the computational method plays a central role. The archetypal example is fluid turbulence, which develops, for example, when water flows rapidly around an obstruction. The set of differential equations satisfied by the fluid can easily be stated. Nevertheless, the patterns of fluid flow to which the equations give rise have largely defied mathematical analysis or description. In practice the patterns are found either through observation of the actual physical system or, as far as possible, through computer experiment.

It is suspected there is a set of mathematical mechanisms common to many systems that give rise to complicated behavior. The mechanisms can best be studied in systems whose construction is as simple as possible. Such studies have recently been done for a class of mathematical systems known as cellular automata. A cellular automaton is made up of many identical components; each component evolves according to a simple set of rules. Taken together, however, the components generate behavior of essentially arbitrary complexity.

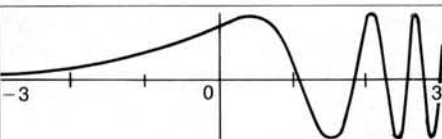
The components of a cellular automaton are mathematical "cells," arranged in one dimension at a sequence of equally spaced points along a line or in two dimensions on a regular grid of squares or hexagons. Each cell carries a value chosen from a small set of possibilities, often just 0 and 1. The values of all the cells in the cellular automaton are simultaneously updated at each "tick" of a clock according to a definite rule. The rule specifies the new value of a cell, given its previous value and the previous values of its nearest neighbors or some other nearby set of cells.

Consider a one-dimensional cellular automaton in which each cell can have the value 0 or 1. Even in such a simple case the overall behavior of the cellular automaton can be quite complex; the most effective way to investigate the behavior is by computer experiment. Most of the properties of cellular automata have in fact been conjectured on the basis of patterns generated in computer experiments. In some cases they have later been established by conventional mathematical arguments.

Cellular automata can serve as explicit models for a wide variety of physical processes. Suppose ice is represented on a two-dimensional hexagonal grid by cells with the value 1 and water vapor is represented by cells with the value 0. A cellular-automaton rule can then be



CHAOTIC BEHAVIOR is seen in many natural systems. A familiar example is the dripping faucet, described by a mathematical model formulated in terms of a differential equation by Robert Shaw of the Institute for Advanced Study. When the rate at which water flows through the faucet is very low, drops of equal size are formed at regular intervals (*left*). The model implies that if the position of the top of each drop that forms (*arrows*) is plotted against the mass of the drop, a simple closed curve called a limit cycle is obtained (*right*). The evolution of the system is represented by a point that traces the curve with time. If the flow is increased, the behavior of the system suddenly becomes more complicated. A phenomenon known as period doubling occurs, and pairs of drops, often of different sizes, are formed in each cycle. If the flow is further increased, there is a sequence of additional period doublings. Finally, just before the water flowing from the faucet becomes continuous an irregular stream of drops is produced. The drops have an entire range of sizes, and the intervals between the formation of consecutive drops appear to be random. The behavior of the system is then described by an irregular curve called a strange or chaotic attractor. The form of the curve is implied by the differential equation, but in practice it can be found only by numerical-approximation techniques.

INPUT	OUTPUT	COMMENT
6+17	23	Evaluate a numerical expression.
6/7+8/9	110/63	Evaluate a numerical expression with exact fractions.
2x-3x+1	1-x	Simplify an algebraic expression.
Ex[(x-1)(x+1)]	-1+x ²	Expand algebraic expressions made up of products of terms. The notation x ^y stands for x raised to the power of y. A space between two non-numerical expressions stands for multiplication.
Ex[(x-a) ² (x+2a) ⁵]	8a ⁶ x ⁶ +21a ⁵ x ⁵ +10a ⁴ x ⁴ -40a ⁴ x ³ -48a ⁵ x ² +16a ⁶ x+32a ⁷ +x ⁷	
Fac[x ² -1]	(-1+x)(1+x)	Factor algebraic expressions.
Fac[x ⁶ -6x ⁴ +4x ³ +9x ² -12x+4]	(-1+x) ⁴ (2+x) ²	
Sol[x ² -3x+1=0,x]	$\{x \rightarrow \frac{3-5^{1/2}}{2}, x \rightarrow \frac{3+5^{1/2}}{2}\}$	Solve an equation for the variable x.
Sol[{x+3a y=4,y-15x=6b},{x,y}]	$\{x \rightarrow \frac{4}{1+45a} - \frac{18ab}{1+45a}, y \rightarrow \frac{60}{1+45a} + \frac{6b}{1+45a}\}$	Solve a pair of simultaneous equations for the variables x and y.
Ps[(1+x ³)E ^x ,x,0,6]	$1+x+\frac{x^2}{2}+\frac{7x^3}{6}+\frac{25x^4}{24}+\frac{61x^5}{120}+\frac{121x^6}{720}$	Find a power-series approximation for the expression e ^x (1+x) ³ for x close to 0, keeping terms up to order x ⁶ .
t:x-2a t ² -2t+1	1+4a-2x+(-2a+x) ²	Assign the value x-2a to the symbol t; simplify the expression t ² -2t+1 for this value.
f[2]:6x+1 f[3]:4-x a f[2]+b f[3]+c f[1]	a(1+6x)+b(4-x)+c f[1]	Assign the value 6x+1 to f[2] and the value 4-x to f[3]; evaluate an expression involving f[1], f[2] and f[3], where f[1] is not yet specified.
f	{[2]:1+6x,[3]:4-x}	Print the object f, which is a list whose elements are indexed by the numbers shown in the brackets.
f[1]:7 f	{7,1+6x,4-x}	Assign the value 7 to f[1]; print the object f, which is now given as a vector, or ordered list of elements.
f ² -8	{41,-8+(1+6x) ² ,-8+(4-x) ² }	Find the square and then subtract 8 from each element of the vector f; the result is a new vector.
f[p]:5x f[p ²]:6x f	{[p ²]:6x,[p]:5x,[1]:7,[2]:1+6x,[3]:4-x}	Assign values for elements of f that have non-numerical indexes; print the object f.
f[\$x]:\$x ² f	{[p ²]:6x,[p]:5x,[1]:7,[2]:1+6x,[3]:4-x,\$x[\$x]:\$x ² }	Assign a value for f[\$x], where \$x is any expression; the general definition is placed at the end of the list f and is used only when none of the preceding special cases apply. Print the object f.
f[p]+f[2]+f[a]	1+11x+a ²	Evaluate the expression f[p]+f[2]+f[a]; the general definition for f[\$x] is applied in order to evaluate f[a].
g[\$x_:=Natp[\$x]]:\$x g[\$x-1] g[1]:1 g	{[1]:1,\$x_:=Natp[\$x]]:\$x g[\$x-1]}	Define the factorial function g[x] for natural numbers x, where g[N] is equal to 1×2×...×N. The definition is given by a recursive formula in which g[x] is specified in terms of g[\$x-1]. The expression \$x_:=Natp[\$x] indicates that \$x must be a natural, or positive whole, number.
g[5]	120	Evaluate g[5], the factorial of 5.
Abs[3] Abs[-3] Abs[-x]	3 3 Abs[x]	Find the absolute values of -3, 3 and -x.
Abs[\$x \$x]:Abs[\$x] Abs[\$\$x]		Define the absolute value of the product of two arbitrary expressions \$x and \$\$x to be the product of their absolute values.
Abs[\$x^(n_:=Natp[n])]:Abs[\$x]^n		Define the absolute value of the arbitrary expression \$x raised to the natural-number power n to be the absolute value of \$x to the power n.
Abs[a b^2 c]	Abs[a] Abs[b] ² Abs[c]	Find the absolute value of the product a×b ² ×c according to the standard rules of algebra and the definitions given for the absolute-value function.
Graph[Sin[E^x],x,-3,3]		Plot a graph of the function sin(e ^x) for values of x from -3 to 3.

MATHEMATICAL CALCULATIONS are carried out by a computer in this example of a dialogue in the SMP computer language developed by the author. The computer manipulates algebraic formulas and other symbolic constructs as well as numbers. The commands in

the language include all the operations of standard mathematics. The last few panels show how new operations can be defined. Properties of the absolute-value function are defined and then are applied by the computer to simplify any expression that includes the function.

used to simulate the successive stages in the freezing of a snowflake. The rule states that once a cell is frozen it does not thaw. Cells exposed at the edge of the growing pattern freeze unless they have so many ice neighbors that they cannot dissipate enough heat to freeze. Snowflakes grown in a computer experiment from a single frozen cell according to this rule show intricate treelike patterns, which bear a close resemblance to real snowflakes. A set of differential equations can also describe the growth of snowflakes, but the much simpler model given by the cellular automaton seems to preserve the essence of the process by which complex patterns are created. Similar models appear to work for biological systems: intricate patterns of growth and pigmentation may be accounted for by the simple algorithms that generate cellular automata.

Simulation by computer is the only method now used for investigating many of the systems discussed so far. It is natural to ask whether simulation, as a matter of principle, is the most efficient possible procedure or whether there is a mathematical formula that could lead more directly to the results. In order to address the question the correspondence between physical and computational processes must be studied more closely.

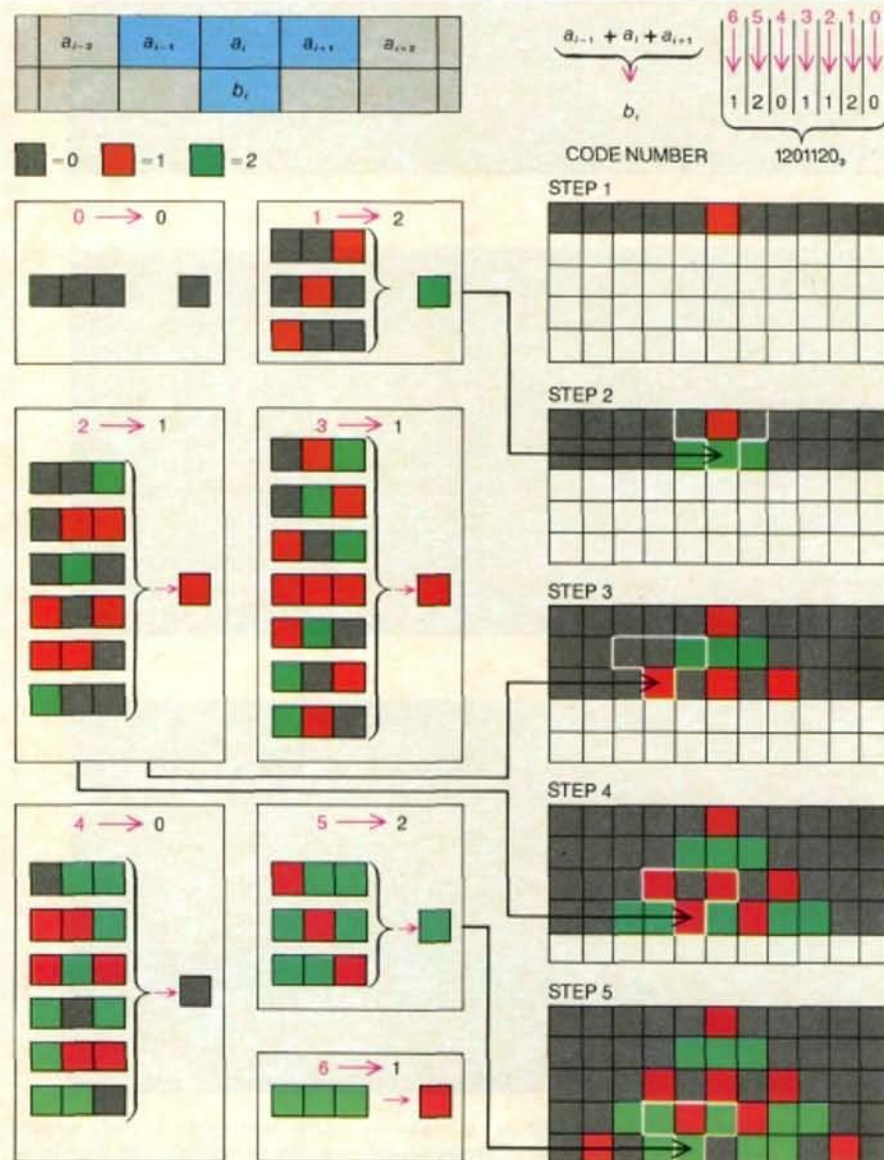
It is presumably true that any physical process can be described by an algorithm, and so any physical process can be represented as a computational process. One must determine how complicated the latter process is. In cellular automata the correspondence between physical and computational processes is particularly clear. A cellular automaton can be regarded as a model of a physical system, but it can also be regarded as a computational system closely analogous to an ordinary digital computer. The sequence of initial cell values in a cellular automaton can be understood as abstract data or information, much like the sequence of binary digits in the memory of a digital computer. During the evolution of a cellular automaton the information is processed: the values of the cells are modified according to definite rules. Similarly, the digits stored in the memory of the digital computer are modified by rules built into the central processing unit of the computer.

The evolution of a cellular automaton from some initial configuration may thus be viewed as a computation that processes the information carried by the configuration. For cellular automata exhibiting simple behavior the computation is a simple one. For example, it may serve only to pick out sequences of three consecutive cells whose initial values are equal to 1. On the other hand, the evolution of cellular automata that show complicated behavior may correspond to a complicated computation.

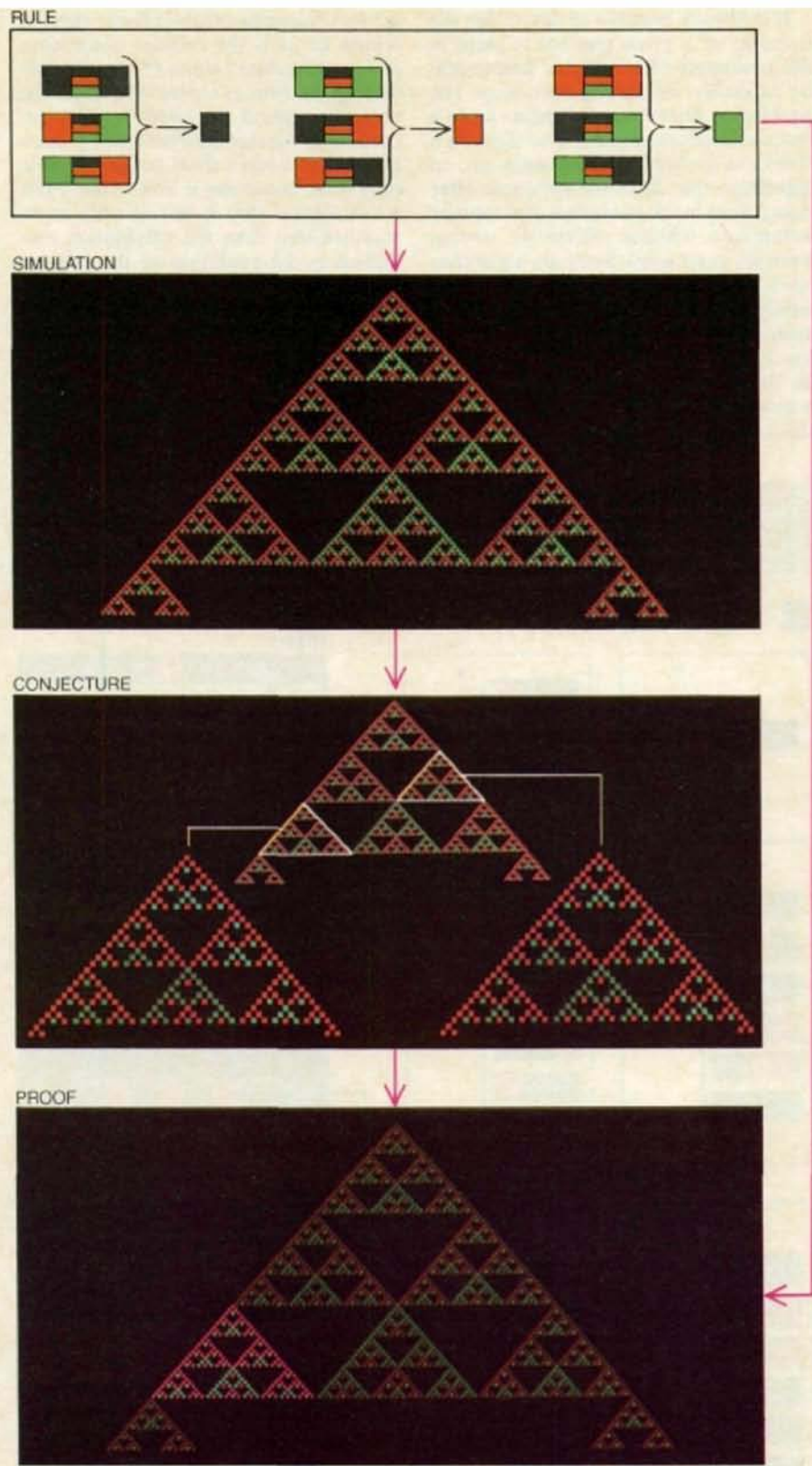
It is always possible to determine the outcome of a given number of steps in the evolution of a cellular automaton by explicitly simulating each step. The problem is whether or not there can be a more efficient procedure. Can there be a short cut to step-by-step simulation, an algorithm that finds the outcome after many steps in the evolution of a cellular automaton without effectively tracing through each step? Such an algorithm could be executed by a computer, and it would predict the evolution of a cellular automaton without explicitly simulating it. The basis of its operation would be that the computer could carry out a more sophisticated computation than the cellular automaton could and so

achieve the same result in fewer steps. It would be as if the cellular automaton were to calculate 7 times 18 by explicitly finding the sum of seven 18's, while the computer found the same product according to the standard method for multiplication. Such a short cut is available only if the computer is able to carry out a calculation that is intrinsically more sophisticated than the calculation embodied in the evolution of the cellular automaton.

One can define a certain class of problems called computable problems that can be solved in a finite time by following definite algorithms. A simple computer such as an adding machine can solve only a small subset of these.



CELLULAR AUTOMATA are simple models that appear to capture the essential features of a wide variety of natural systems. A one-dimensional cellular automaton is made up of a line of cells, shown in the diagram as colored squares. Each cell can take on a number of possible values, represented by different colors. The cellular automaton evolves in a series of steps, shown as a sequence of rows of squares progressing down the page. At each step the values of all the cells are updated according to a fixed rule. In the case illustrated the rule specifies the new value of a cell in terms of the sum of its previous value and the previous values of its immediate neighbors. Such rules are conveniently specified by code numbers defined as shown in the diagram; the subscript 3 is given because each cell can take on one of three possible values.



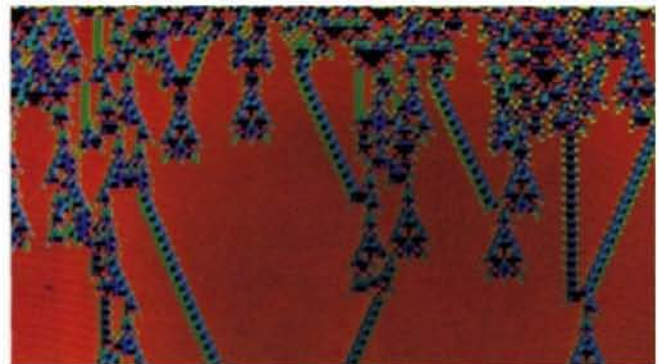
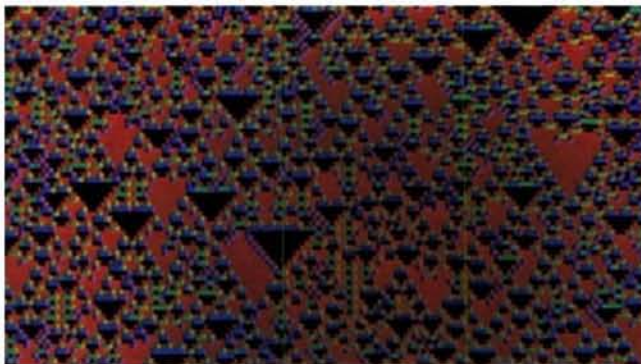
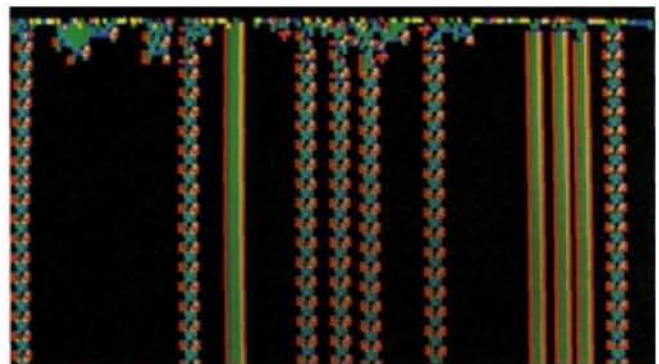
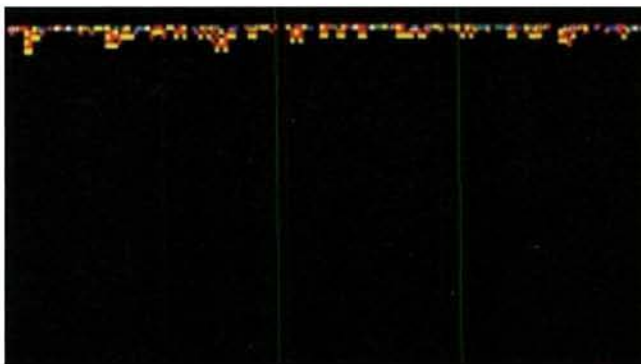
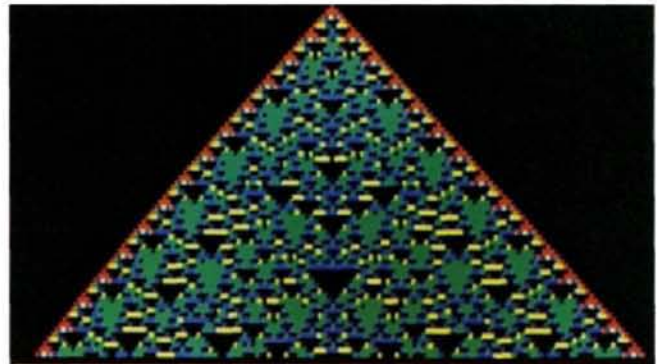
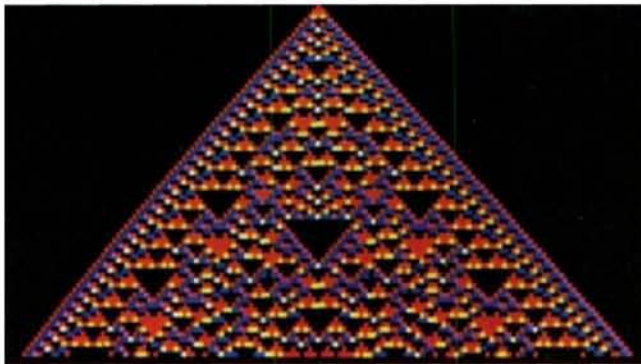
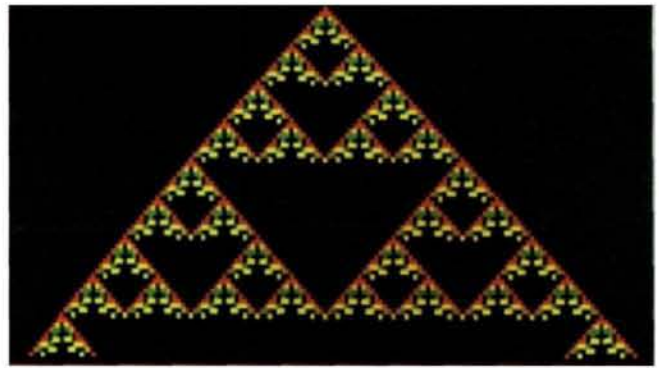
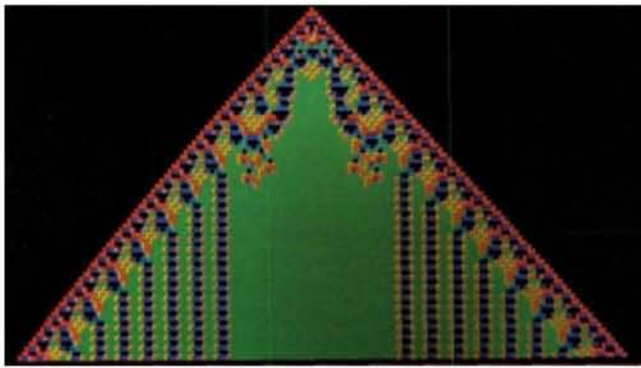
EXPERIMENTAL MATHEMATICS is an exploratory technique made possible largely through the use of computers. Any set of mathematical rules can be applied repeatedly by a computer and their consequences explored in an experimental fashion. For example, in order to study a pattern generated by the cellular automaton defined by the rule shown, one begins by explicitly simulating on a computer many steps in the evolution of the cellular automaton. Inspection of the pattern obtained then leads to the conjecture that it is fractal, or self-similar, in the sense that parts of it, when enlarged, have the same overall form as the whole. The conjecture, once made, is comparatively easy to prove by conventional mathematical techniques. The proof can be based on the fact that the initial conditions for growth from certain cells in the pattern are the same as the conditions for growth from the very first cell. There are an increasing number of mathematical results that were discovered in computer experiments. Some of them have subsequently been reproduced by conventional mathematical arguments.

problems. There exist universal, or general-purpose, computers, however, that can solve any computable problem. A real digital computer is essentially such a universal machine. The instructions that can be executed by the central processing unit of the computer are rich enough to serve as the elements of a computer program that can embody any algorithm. A number of systems in addition to the digital computer have been shown to be capable of universal computation. Several cellular automata are among them: for example, universal computation has been proved for a simple two-dimensional cellular automaton with a 0 or a 1 in each cell. It is strongly suspected that several one-dimensional cellular automata are also universal computers. The simplest candidates have three possible values at each cell and rules of evolution that take account only of the nearest-neighbor cells.

Cellular automata that are capable of universal computation can mimic the behavior of any possible computer; since any physical process can be represented as a computational process, they can mimic the action of any possible physical system as well. If there were an algorithm that could work out the behavior of these cellular automata faster than the automata themselves evolve, the algorithm would allow any computation to be speeded up. Because this conclusion would lead to a logical contradiction, it follows there can be no general short cut that predicts the evolution of an arbitrary cellular automaton. The calculation corresponding to the evolution is irreducible: its outcome can be found effectively only by simulating the evolution explicitly. Thus direct simulation is indeed the most efficient method for determining the behavior of some cellular automata. There is no way to predict their evolution; one must simply watch it happen.

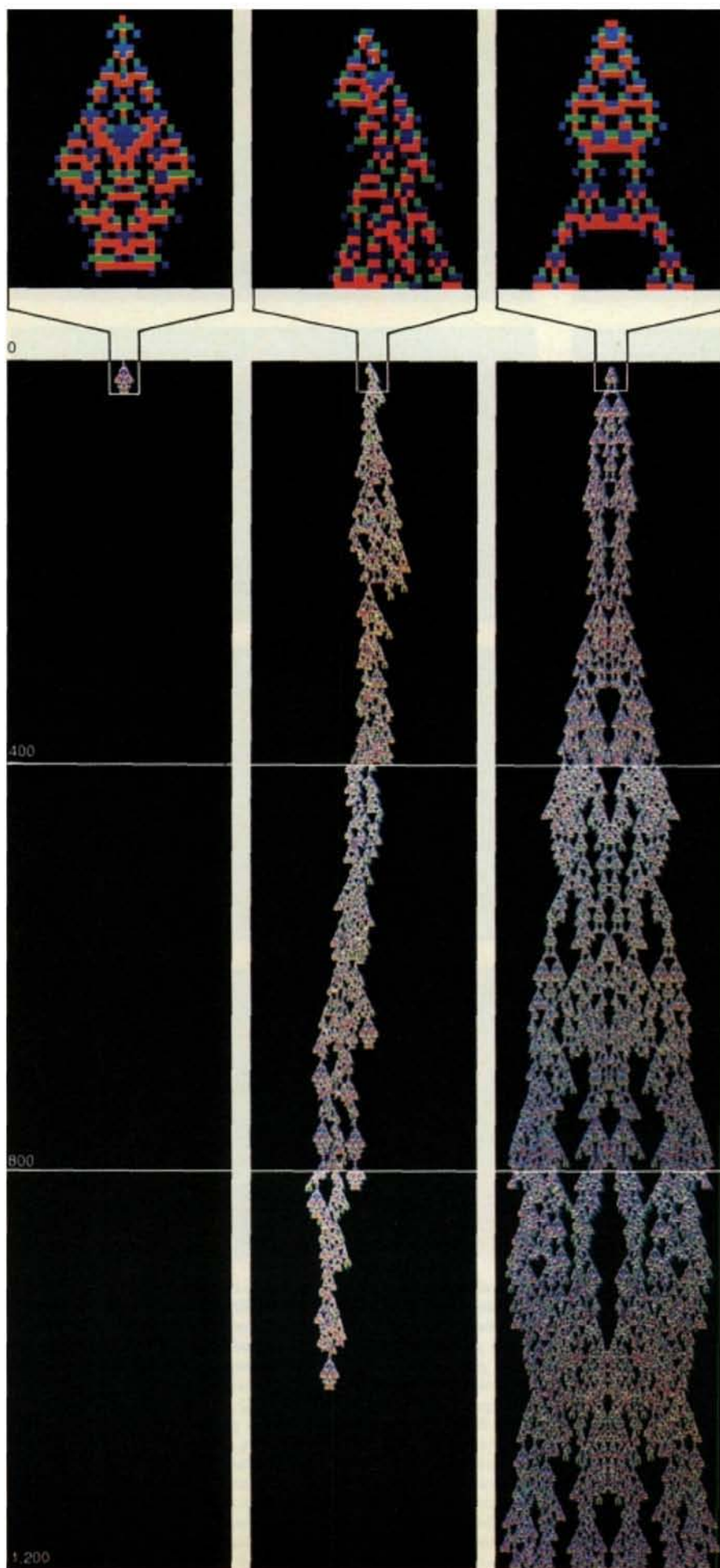
It is not yet known how widespread the phenomenon of computational irreducibility is among cellular automata or among physical systems in general. Nevertheless, it is clear that the elements of a system need not be very complicated for the overall evolution of the system to be computationally irreducible. It may be that computational irreducibility is almost always present when the behavior of a system appears complicated or chaotic. General mathematical formulas that describe the overall behavior of such systems are not known, and it is possible no such formulas can ever be found. In that case, explicit simulation in a computer experiment is the only available method of investigation.

Much of physical science has traditionally focused on the study of computationally reducible phenomena, for which simple overall descriptions can be given. In real physical systems, however,



2213310 _s	4200410 _s	<div style="display: flex; flex-direction: column; align-items: center;"> <div style="width: 10px; height: 10px; background-color: black; margin-bottom: 5px;"></div>=0 <div style="width: 10px; height: 10px; background-color: red; margin-bottom: 5px;"></div>=1 <div style="width: 10px; height: 10px; background-color: green; margin-bottom: 5px;"></div>=2 <div style="width: 10px; height: 10px; background-color: blue; margin-bottom: 5px;"></div>=3 <div style="width: 10px; height: 10px; background-color: yellow; margin-bottom: 5px;"></div>=4 </div>
331240 _s	2024310 _s	
1100400 _s	2231000 _s	
131210 _s	3211310 _s	

COMPLEX BEHAVIOR can develop even in systems with simple components. The eight cellular automata shown in the photographs are made up of lines of cells that take on one of five possible values. The value of each cell is determined by a simple rule based on the values of its neighbors on the previous line. Each pattern is generated by the rule whose code number is given in the key (see illustration on page 197). The patterns in the upper four photographs are grown from a single colored cell. Even in this case the patterns generated can be complex, and they sometimes appear quite random. The complex patterns formed in such physical processes as the flow of a turbulent fluid may well arise from the same mechanism. Complex patterns generated by cellular automata can also serve as a source of effectively random numbers, and they can be applied to encrypt messages by converting a text into an apparently random form. The patterns in the lower four photographs begin with disordered states. Even though the values of the cells in these initial states are chosen at random, the evolution of the cellular automata gives rise to structures of four basic classes. In the two classes shown in the third row of photographs the long-term behavior of the cellular automata is comparatively simple; in the two classes shown in the bottom row it can be highly complex. The behavior of many natural systems may well conform to this classification.



computational reducibility may well be the exception rather than the rule. Fluid turbulence is probably one of many examples of computational irreducibility. In biological systems computational irreducibility may be even more widespread: it may turn out that the form of a biological organism can be determined from its genetic code essentially only by following each step in its development. When computational irreducibility is present, one must adopt a methodology that depends heavily on computation.

One of the consequences of computational irreducibility is that there are questions that can be asked about the ultimate behavior of a system but that cannot be answered in full generality by any finite mathematical or computational process. Such questions must therefore be considered undecidable. An example of such a question is whether a particular pattern ever dies out in the evolution of a cellular automaton. It is straightforward to answer the question for some definite number of steps, say 1,000: one need only simulate 1,000 steps in the evolution of the cellular automaton. In order to determine the answer for any number of steps, however, one must simulate the evolution of the cellular automaton for a potentially infinite number of steps. If the cellular automaton is computationally irreducible, there is no effective alternative to such direct simulation.

The upshot is that no calculation of any fixed length can be guaranteed to determine whether a pattern will ultimately die out. It may be possible to tell the fate of a particular pattern after tracing only a few steps in its evolution, but there is no general way to tell in advance how many steps will be re-

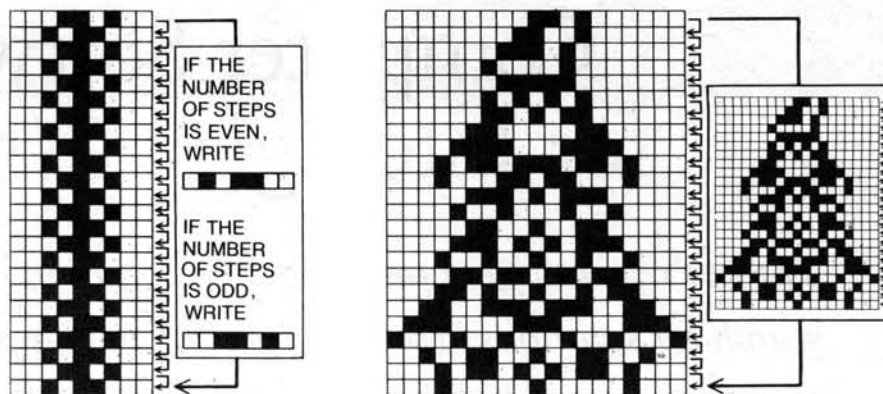
UNDECIDABLE PROBLEMS can arise in the mathematical analysis of models of physical systems. For example, consider the problem of determining whether a pattern generated by the evolution of a cellular automaton will ever die out, so that all the cells become black. The patterns generated by the cellular automaton shown above are so complicated that the only possible general approach to the solution of the problem is to explicitly simulate the evolution of the cellular automaton. The pattern obtained from the initial state shown at the left is found to die out after just 16 steps. The initial state in the center yields a pattern that takes 1,016 steps to die out. The initial state at the right gives rise to a pattern whose fate remains unclear even after a simulation carried out over many thousands of steps. In general no finite simulation of a fixed number of steps can be guaranteed to determine the ultimate behavior of the cellular automaton. Hence the problem of whether or not a particular pattern ultimately dies out, or halts, is said to be formally undecidable. The cellular automaton shown here follows a rule specified by the code number 3311100320₄.

quired. The ultimate form of a pattern is the result of an infinite number of steps, corresponding to an infinite computation; unless the evolution of the pattern is computationally reducible, its consequences cannot be reproduced by any finite computational or mathematical process.

The possibility of undecidable questions in mathematical models for physical systems can be viewed as a manifestation of Gödel's theorem on undecidability in mathematics, which was proved by Kurt Gödel in 1931. The theorem states that in all but the simplest mathematical systems there may be propositions that cannot be proved or disproved by any finite mathematical or logical process. The proof of a given proposition may call for an indefinitely large number of logical steps. Even propositions that can be stated succinctly can require an arbitrarily long proof. In practice there are many simple mathematical theorems for which the only known proofs are very long. In addition the cases that must be examined to prove or refute conjectures are often quite complicated. In number theory, for example, there are many cases in which the smallest number having some special property is extremely large; the number can often be found only by testing each whole number in turn. Such phenomena are making the computer an essential tool in many mathematical investigations.

Computational irreducibility implies many fundamental limitations on the scope of theories for physical systems. It may be possible to model a system at many levels, from simulating the motions of individual molecules to solving differential equations for overall properties. Computational irreducibility implies there is a highest level at which abstract models can be made; above that level results can be found only by explicit simulation.

When the level of description becomes computationally irreducible, undecidable questions also begin to appear. Such questions must be avoided in the formulation of a theory, much as the simultaneous measurement of the position and velocity of an electron—impossible according to the uncertainty principle—is avoided in quantum mechanics. Even if such questions are eliminated, there is still the practical difficulty of answering questions that in principle can be answered. The degree of difficulty depends strongly on the nature of the objects involved in the simulation. If the only way to predict the weather were to simulate the motions of every molecule in the atmosphere, no practical calculations could be carried out. Nevertheless, the relevant features of the weather can probably be studied by considering the interactions of large volumes of the



COMPUTATIONAL IRREDUCIBILITY is a phenomenon that seems to arise in many physical and mathematical systems. The behavior of any system can be found by explicit simulation of the steps in its evolution. When the system is simple enough, however, it is always possible to find a short cut to the procedure: once the initial state of the system is given, its state at any subsequent step can be found directly from a mathematical formula. For the system shown schematically at the left, the formula merely requires that one find the remainder when the number of steps in the evolution is divided by 2. Such a system is said to be computationally reducible. For a system such as the one shown schematically at the right, however, the behavior is so complicated that in general no short-cut description of the evolution can be given. Such a system is computationally irreducible, and its evolution can effectively be determined only by the explicit simulation of each step. It seems likely that many physical and mathematical systems for which no simple description is now known are in fact computationally irreducible. Experiment, either physical or computational, is effectively the only way to study such systems.

atmosphere, and so useful simulations should be possible.

The efficiency with which a computationally irreducible system can be simulated depends on the computational sophistication of each step in its evolution. The steps in the evolution of the system can be simulated by instructions in a computer program. The fewer the instructions needed to reproduce each step, the more efficient the simulation. Higher-level descriptions of physical systems typically call for more sophisticated steps, much as single instructions in higher-level computer languages correspond to many instructions in lower-level ones. One time step in the numerical approximation of a differential equation that describes a jet of gas requires a computation more sophisticated than the one needed to follow a collision between two molecules in the gas. On the other hand, each step in the higher-level description given by a differential equation accounts for an immense number of steps in the lower-level description of molecular collisions. The resulting gain in efficiency more than makes up for the fact that the individual steps are more sophisticated.

In general the efficiency of a simulation increases with higher levels of description, until the operations needed for the higher-level description are matched with the operations carried out directly by the computer doing the simulation. It is most efficient for the computer to be as close an analogue to the system being simulated as possible.

There is one major difference between most existing computers and physical systems or models of them: computers process information serially, whereas

physical systems process information in parallel. In a physical system modeled by a cellular automaton the values of all the cells are updated together at each time step. In a standard computer program, however, the simulation of the cellular automaton is carried out by a loop that updates the value of each cell in turn. In such a case it is straightforward to write a computer program that performs a fundamentally parallel process with a serial algorithm. There is a well-established framework in which algorithms for the serial processing of information can be described. Many physical systems, however, seem to require descriptions that are essentially parallel in nature. A general framework for parallel processing does not yet exist, but when it is developed, more effective high-level descriptions of physical phenomena should become possible.

The introduction of the computer in science is comparatively recent. Already, however, computation is establishing a new approach to many problems. It is making possible the study of phenomena far more complex than the ones that could previously be considered, and it is changing the direction and emphasis of many fields of science. Perhaps most significant, it is introducing a new way of thinking in science. Scientific laws are now being viewed as algorithms. Many of them are studied in computer experiments. Physical systems are viewed as computational systems, processing information much the way computers do. New aspects of natural phenomena have been made accessible to investigation. A new paradigm has been born.