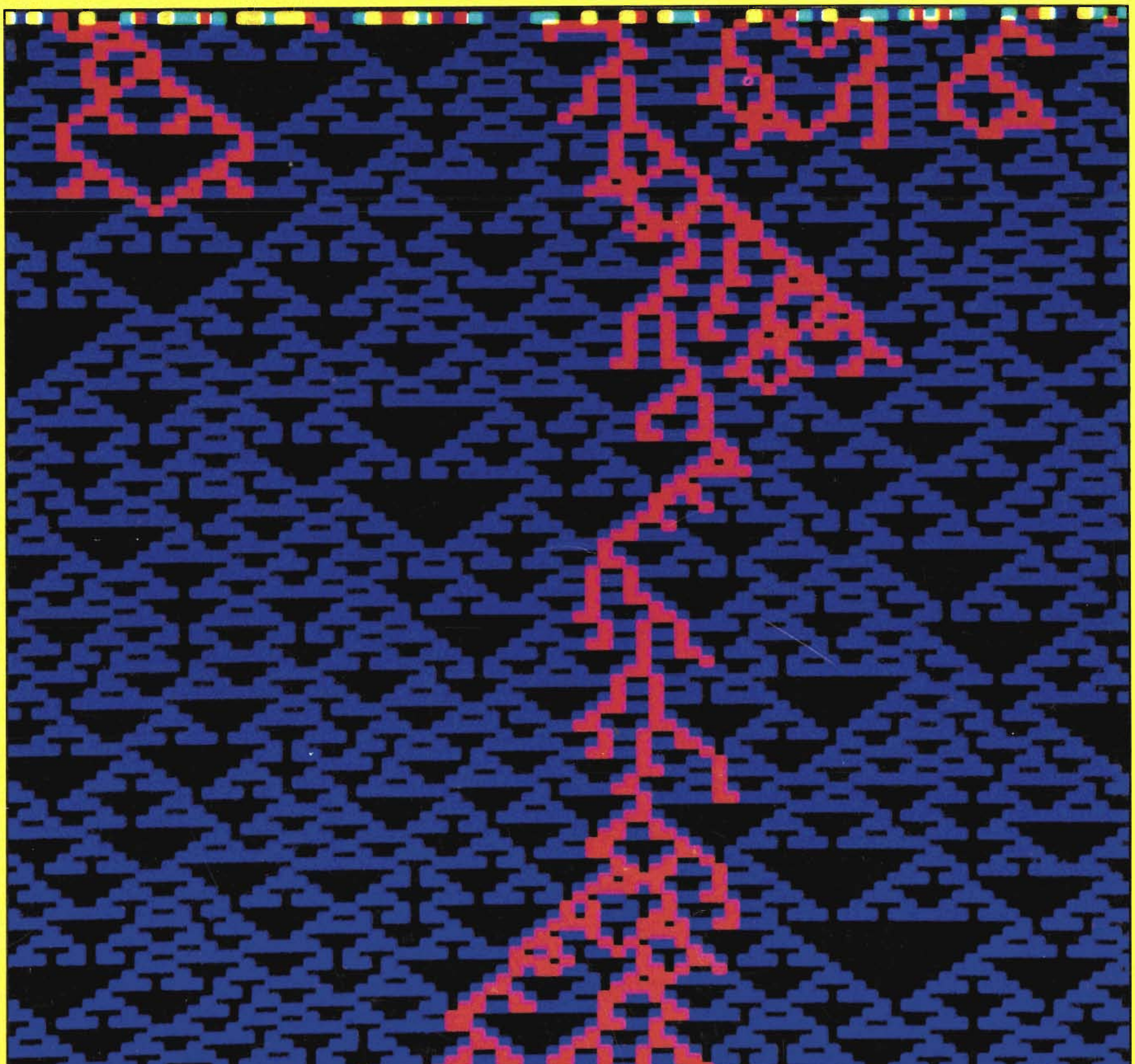
Number 9 Fall 1983

Los Alamos Science

LOS ALAMOS NATIONAL LABORATORY



EDITOR'S NOTE

This issue is evidence of our growing ability to understand complex systems and the essential role that computers play in this increasing capability.

The title of the cover article, "Cellular Automata," refers to discrete mathematical systems that can provide a useful model for many complex natural systems—in particular, systems with many degrees of freedom that evolve in a more or less discrete fashion. The idea of such a construct grew out of von Neumann's interest in developing a theory of information encompassing both natural automata (living systems) and manmade automata (computer and communications networks). His work in automata theory in the late 1940s followed naturally from his pioneering work on the logical design of the new electronic automaton, the digital computer.

Cellular automata are in many ways directly analogous to parallel-processing digital computers and are therefore most easily investigated on digital computers. For many years research in this area was limited to a few isolated individuals, but recently scientists from a wide range of disciplines have become interested in this subject, in part because of increased access to computer facilities. Last March they came together at a Los Alamos conference to discuss both the intrinsic properties of cellular automata behavior and their possible use as models for natural systems. DNA sequences and their evolution, lattice-spin systems in physics, microtubules in biology, and complex chemical systems are among the applications already under investigation. One of the speakers and author of the cover article, Stephen Wolfram, has been credited by many as making this infant field into a science. His comprehensive study of one-dimensional automata has elucidated statistical features of self-organizing systems and has led to his discovery of four universal classes of behavior. This discovery may be a key to defining complexity as a quantitative concept.

In contemplating a general theory of complicated automata back in 1948, von Neumann asked this question: Is it possible to build automata that can create things more complicated than themselves in analogy with the processes of evolution? Genes contain, in some sense, a description of the whole organism, its replication, and its evolutionary change. What logical design underlies their operation?

Since then we have come to understand that DNA operates much like a programmed computer and that the genetic code is essentially the "microcode" that describes how the computer is run. But the basic flow chart that governs the operation and evolutionary development of a whole DNA molecule remains a mystery. As Walter Goad describes in this issue, one hope for greater understanding is comparative analysis of the many DNA sequences that are rapidly becoming known. Such analysis has been facilitated by the formation of GenBank, a national computer-based library of DNA sequences at Los Alamos, where scientists are not only collecting and annotating the library entries but also examining similarities among them with one of today's supercomputers, the Cray-1.

Our understanding of the natural automaton we call the gene is advancing through laboratory studies as well. In this issue scientists from the Life Sciences Division explain how they unraveled, step by step, the action of a particular gene cluster known as the metallothionein locus. This gene cluster, which defends against certain types of heavy-metal poisoning, contains an on/off switch that may be attached to other genes (such as the oncogenes implicated in cancer) to control their expression. The authors' elaborate and careful studies on the locus itself are now providing a sound basis for using it in such an exciting way.

The issue closes with a report on the recent conference sponsored by the National Security Agency and Los Alamos National Laboratory and entitled "Frontiers of Supercomputing." Among the Laboratory organizers of the conference were Bill Buzbee, Nick Metropolis, and David Sharp, who describe the quantum jump in computing that would be made possible by massively parallel computer architectures. This jump would have far-reaching impact on our national security and economy as well as on scientific research. The ability to tackle complex problems more realistically is the promise of the next generation of supercomputers and the key to progress in numerous activities of the modern world. ■

Necia Grant Cooper

On the cover.
A space-time pattern of a one-dimensional cellular automaton generated on a computer by Stephen Wolfram. The automaton evolves from a random initial configuration (colored squares across the top) to exhibit class 3 behavior (pattern of blue squares) and class 4 behavior (pattern of red squares). For explanation of these universal behavioral classes see Wolfram's article, "Cellular Automata."

Los Alamos Science

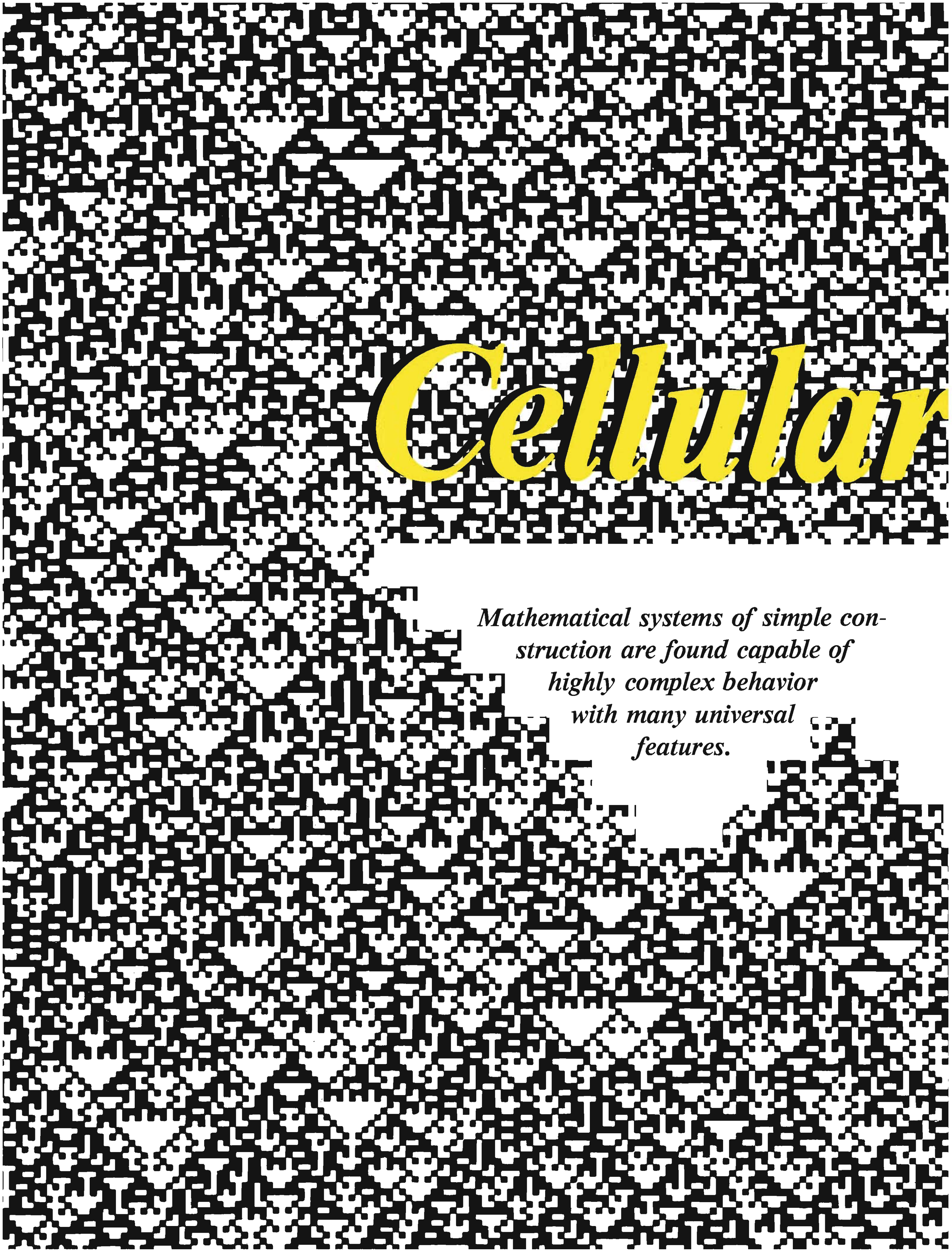
Fall 1983 Number 9

CONTENTS

Editor
Necia Grant Cooper
Art Director
Gloria Sharp
Associate Editor
Nancy Shera
Science Writer
Roger Eckhardt
Feature Writer
Judith M. Lathrop
Editorial Coordinator
Elizabeth P. White
Production Coordinator
Judy Gibes
Copy Chief
Elizabeth P. White
Illustrators
Anita Flores, Karen Hindman
Janice Taylor (type)
Photography
John Flower, Fred Rick
Bill Jack Rodgers, LeRoy N. Sanchez
Richard Robinson (assistant)
Black and White Photo Laboratory Work
Tom Baros, Louise Carson
Gary Desharnais, Chris Lindberg
Dan Morse
Color Photo Laboratory Work
Peter Anders
Phototypesetting
Kris Mathieson, Bessie Vigil
Chris West
Pasteup
Judy Gibes
Printing
Robert C. Crook, Jim E. Lovato
Circulation
Elizabeth P. White

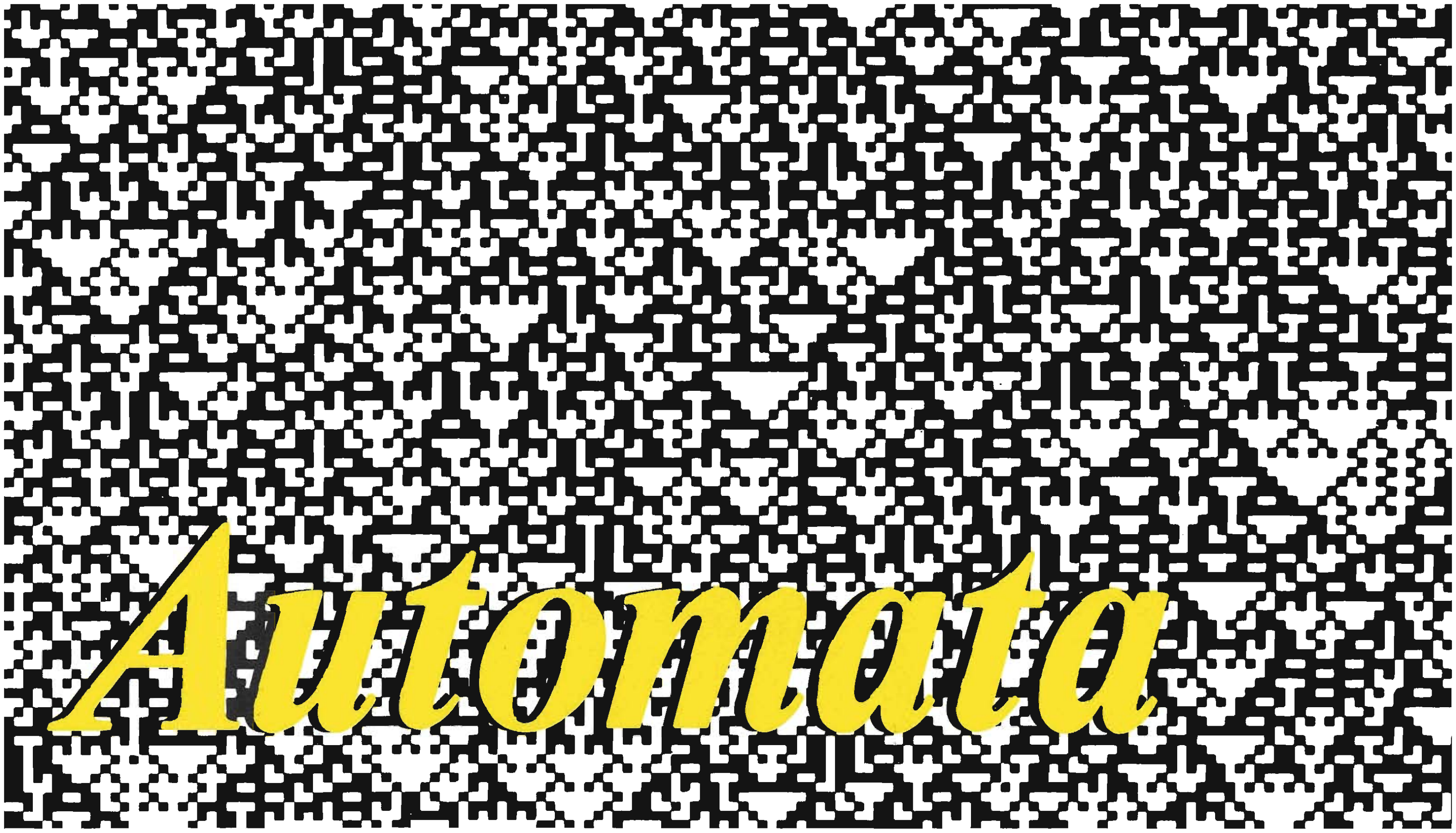
Los Alamos Science is published by Los Alamos National Laboratory, an Equal Opportunity Employer operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.

Address mail to
Los Alamos Science
Los Alamos National Laboratory
Mail Stop M708
Los Alamos, New Mexico 87545



Cellular

Mathematical systems of simple construction are found capable of highly complex behavior with many universal features.



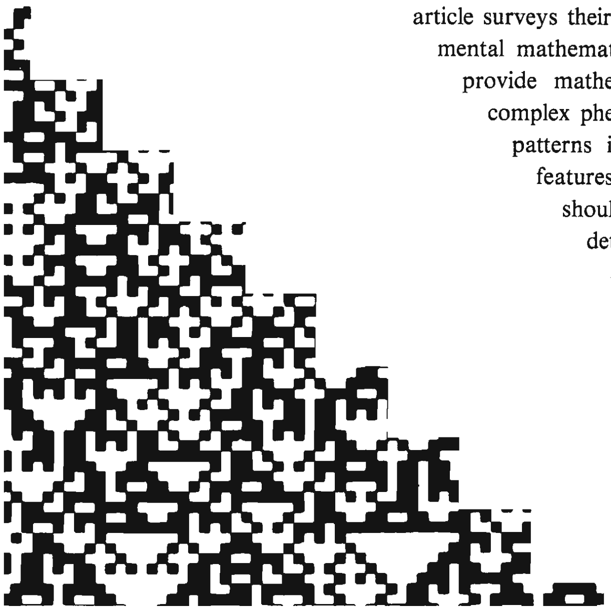
Automata

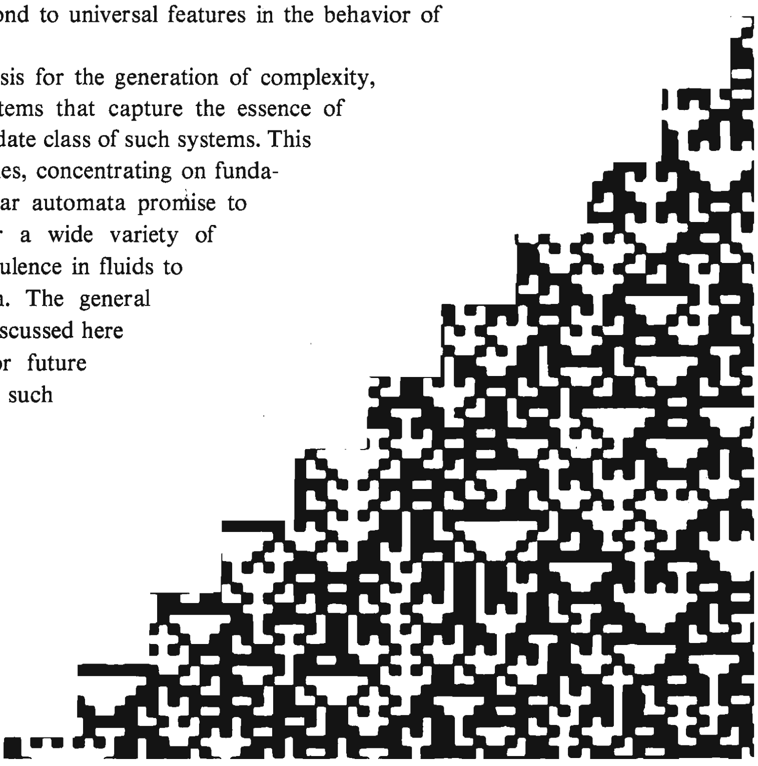
by Stephen Wolfram

It appears that the basic laws of physics relevant to everyday phenomena are now known. Yet there are many everyday natural systems whose complex structure and behavior have so far defied even qualitative analysis. For example, the laws that govern the freezing of water and the conduction of heat have long been known, but analyzing their consequences for the intricate patterns of snowflake growth has not yet been possible. While many complex systems may be broken down into identical components, each obeying simple laws, the huge number of components that make up the whole system act together to yield very complex behavior.

In some cases this complex behavior may be simulated numerically with just a few components. But in most cases the simulation requires too many components, and this direct approach fails. One must instead attempt to distill the mathematical essence of the process by which complex behavior is generated. The hope in such an approach is to identify fundamental mathematical mechanisms that are common to many different natural systems. Such commonality would correspond to universal features in the behavior of very different complex natural systems.

To discover and analyze the mathematical basis for the generation of complexity, one must identify simple mathematical systems that capture the essence of the process. Cellular automata are a candidate class of such systems. This article surveys their nature and properties, concentrating on fundamental mathematical features. Cellular automata promise to provide mathematical models for a wide variety of complex phenomema, from turbulence in fluids to patterns in biological growth. The general features of their behavior discussed here should form a basis for future detailed studies of such specific systems.





The Nature of Cellular Automata and a Simple Example

Cellular automata are simple mathematical idealizations of natural systems. They consist of a lattice of discrete identical sites, each site taking on a finite set of, say, integer values. The values of the sites evolve in discrete time steps according to deterministic rules that specify the value of each site in terms of the values of neighboring sites. Cellular automata may thus be considered as discrete idealizations of the partial differential equations often used to describe natural systems. Their discrete nature also allows an important analogy with digital computers: cellular automata may be viewed as parallel-processing computers of simple construction.

As a first example of a cellular automaton, consider a line of sites, each with value 0 or 1 (Fig. 1). Take the value of a site at position i on time step t to be $a_i^{(t)}$. One very simple rule for the time evolution of these site values is

$$a_i^{(t+1)} = a_{i-1}^{(t)} + a_{i+1}^{(t)} \mod 2 \; , \qquad (1)$$

where mod 2 indicates that the 0 or 1 remainder after division by 2 is taken. According to this rule, the value of a particular site is given by the sum modulo 2 (or, equivalently, the Boolean algebra "exclusive or") of the values of its left- and right-hand nearest neighbor sites on the previous time step. The rule is implemented simultaneously at each site.* Even with this very simple rule quite complicated behavior is nevertheless found.

Fractal Patterns Grown from Cellular Automata. First of all, consider evolution ac-

In the very simplest computer implementation a separate array of updated site values must be maintained and copied back to the original site value array when the updating process is complete.

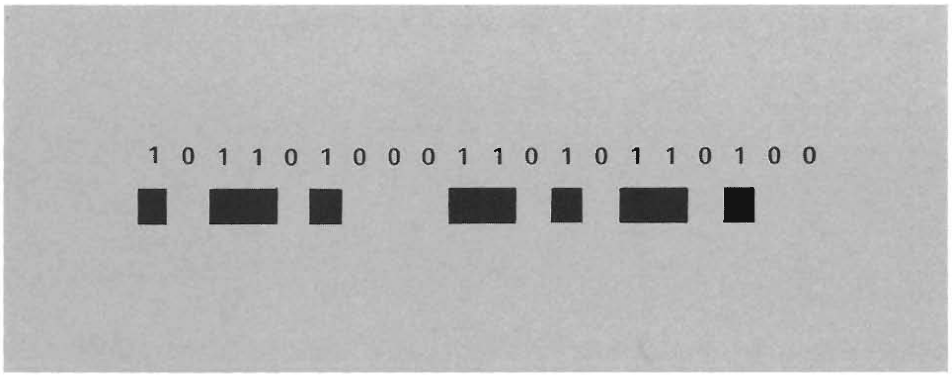


Fig. 1. A typical configuration in the simple cellular automaton described by Eq. 1, consisting of a sequence of sites with values 0 or 1. Sites with value 1 are represented by squares; those with value 0 are blank.

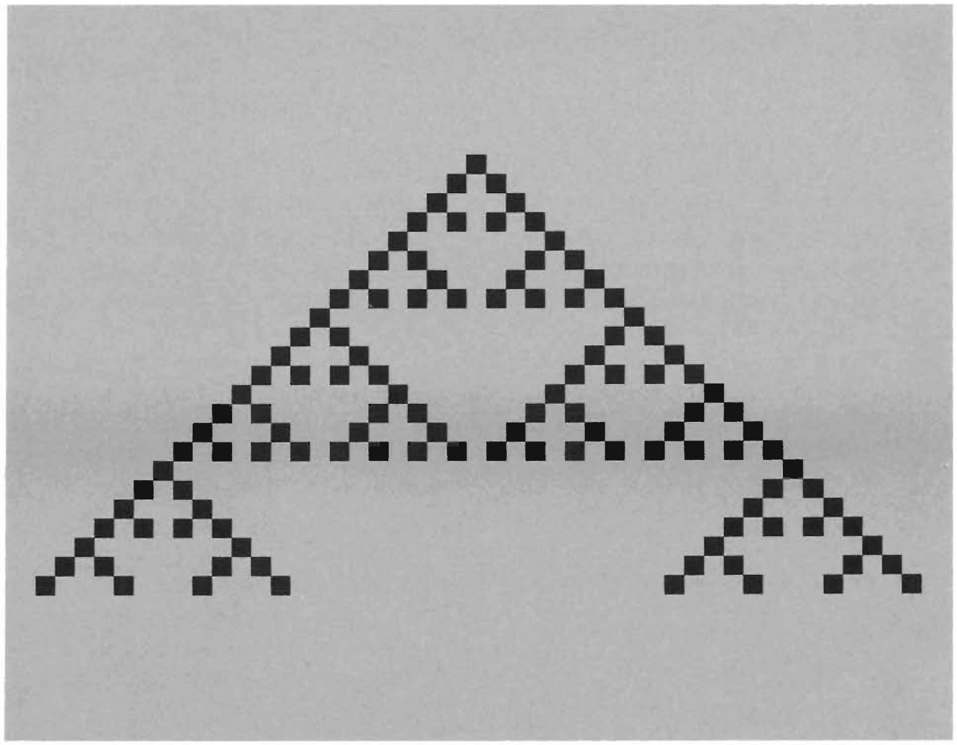


Fig. 2. A few time steps in the evolution of the simple cellular automaton defined by Eq. 1, starting from a "seed" containing a single nonzero site. Successive lines are obtained by successive applications of Eq. 1 at each site. According to this rule, the value of each site is the sum modulo 2 of the values of its two nearest neighbors on the previous time step. The pattern obtained with this simple seed is Pascal's triangle of binomial coefficients, reduced modulo 2.

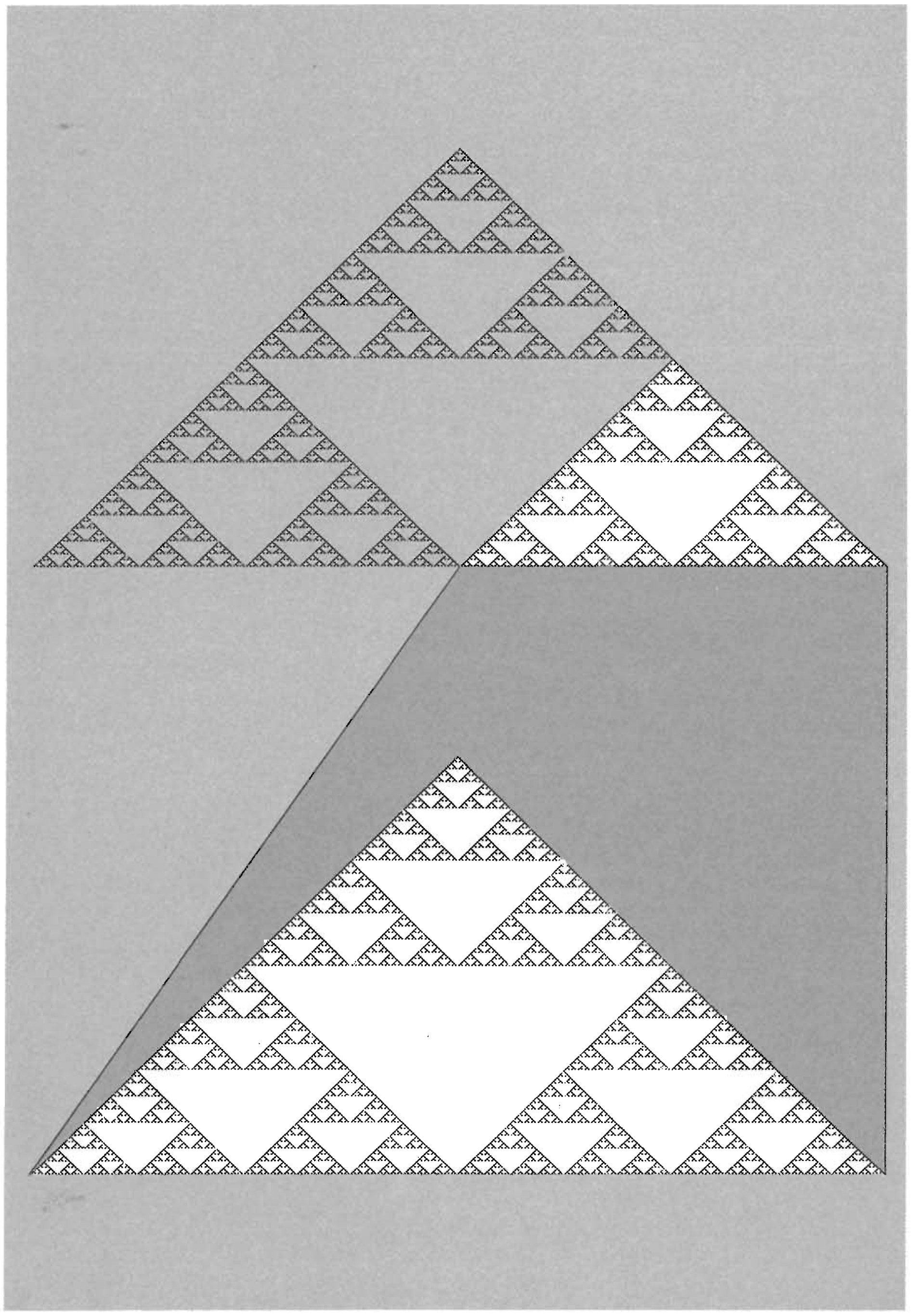


Fig. 3. Many time steps in the evolution of the cellular automaton of Fig. 2, generated by applying the rule of Eq. 1 to about a quarter of a million site values. The pattern obtained is "self similar": a part of the pattern, when magnified, is indistinguishable from the whole. The pattern has a fractal dimension of $log_2 3 \simeq 1.59$.

cording to Eq. 1 from a "seed" consisting of a single site with value 1, all other sites having value 0. The pattern generated by evolution for a few time steps already exhibits some structure (Fig. 2). Figure 3 shows the pattern generated after 500 time steps. Generation of this pattern required application of Eq. 1 to a quarter of a million site values. The pattern of Figs. 2 and 3 is an intricate one but exhibits some striking regularities. One of these is "self-similarity." As illustrated in Fig. 3, portions of the pattern, when magnified, are indistinguishable from the whole. (Differences on small scales between the original pattern and the magnified portion disappear when one considers the limiting pattern obtained after an infinite number of time steps.) The pattern is therefore invariant under rescaling of lengths. Such a self-similar pattern is often called a fractal and may be characterized by a fractal dimension. The fractal dimension of the pattern in Fig. 3, for example, is $\log_2 3 = \log 3/\log 2 \simeq 1.59$. Many natural systems, including snowflakes, appear to exhibit fractal patterns. (See Benoit B. Mandelbrot, *The Fractal Geometry of Nature,* W. H. Freeman and Company, 1982.) It is very possible that in many cases these fractal patterns are generated through evolution of cellular automata or analogous processes.

Self-Organization in Cellular Automata. Figure 4 shows evolution according to Eq. 1 from a "disordered" initial state. The values of sites in this initial state are randomly chosen: each site takes on the value 0 or 1 with equal probability, independently of the values of other sites. Even though the initial state has no structure, evolution of the cellular automaton does manifest some structure in the form of many triangular "clearings." The spontaneous appearance of these clearings is a simple example of "self-organization."

The pattern of Fig. 4 is strongly reminiscent of the pattern of pigmentation found on the shells of certain mollusks (Fig. 5). It is quite possible that the growth of these pigmentation patterns follows cellular automaton rules.

In systems that follow conventional thermodynamics, the second law of thermodynamics implies a progressive degradation of any initial structure and a universal tendency to evolve with time to states of maximum entropy and maximum disorder. While many natural systems do tend toward disorder, a large class of systems, biological ones being prime examples, show a reverse trend: they spontaneously generate structure with time, even when starting from disordered or structureless initial states. The cellular automaton in Fig. 4 is a simple example of such a self-organizing system. The mathematical basis of this behavior is revealed by considering global properties of the cellular automaton. Instead of following evolution from a particular initial state, as in Fig. 4, one follows the overall evolution of an ensemble of many different initial states.

It is convenient when investigating global properties to consider finite cellular automata that contain a finite number N of sites whose values are subject to periodic boundary conditions. Such a finite cellular automaton may be represented as sites arranged, for example, around a circle. If each site has two possible values, as it does for the rule of Eq. 1, there are a total of 2^N possible states, or configurations, for the complete finite cellular automaton. The global evolution of the cellular automaton may then be represented by a finite state transition graph plotted in the "state space" of the cellular automaton. Each of the 2^N possible states of the complete cellular automaton (such as the state 110101101010 for a cellular automaton with twelve sites) is represented by a node, or point, in the graph, and a directed line connects each node to the node generated by a single application of the cellular automaton rule. The trajectory traced out in state space by the directed lines connecting one particular node to its successors thus corresponds to the time evolution of the cellular

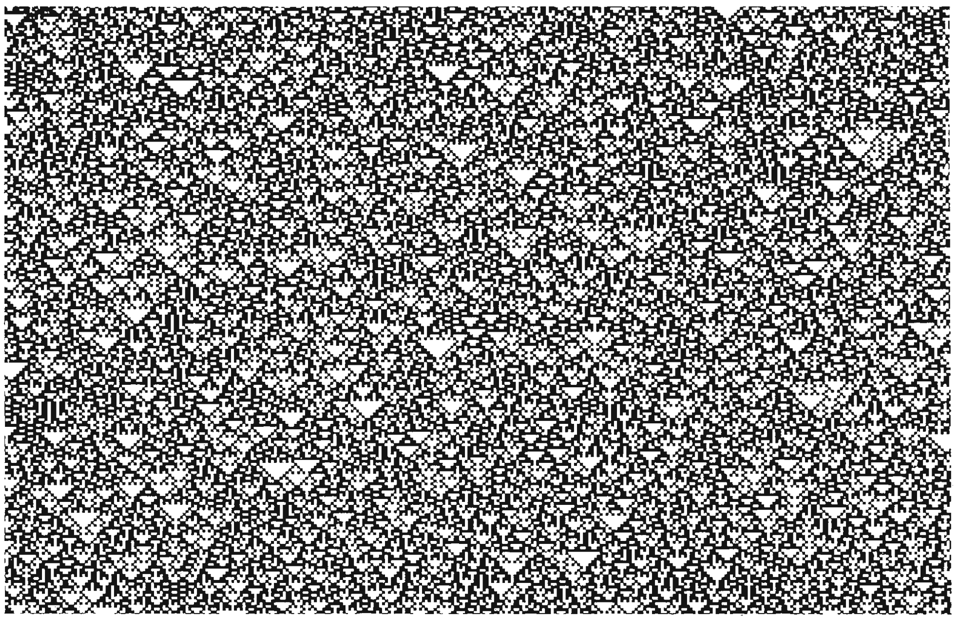


Fig. 4. Evolution of the simple cellular automaton defined by Eq. 1, from a disordered initial state in which each site is taken to have value 0 or 1 with equal, independent probabilities. Evolution of the cellular automaton even from such a random initial state yields some simple structure.

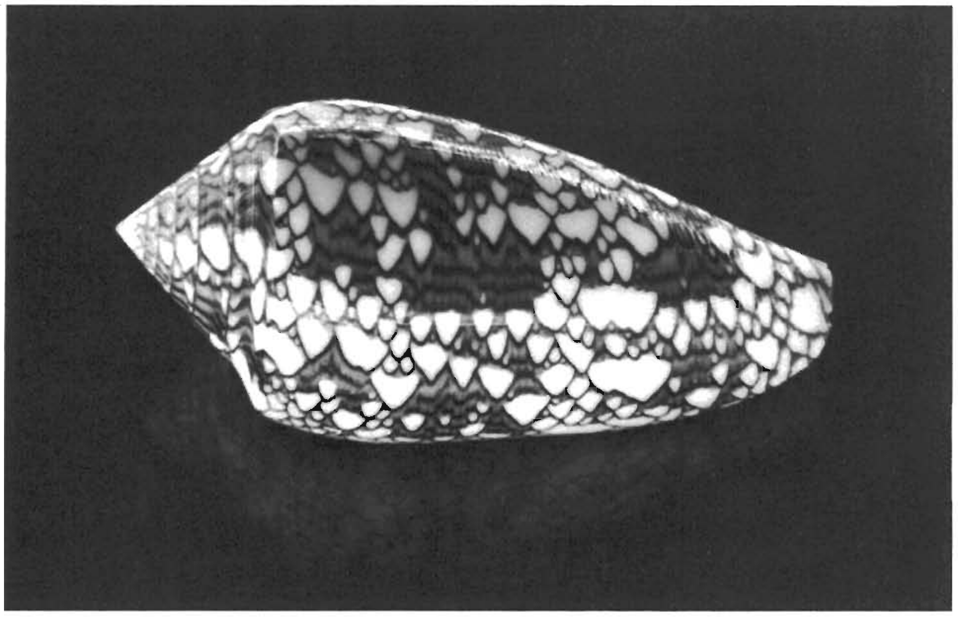


Fig. 5. A "cone shell" with a pigmentation pattern reminiscent of the pattern generated by the cellular automaton of Fig. 4. (Shell courtesy of P. Hut.)

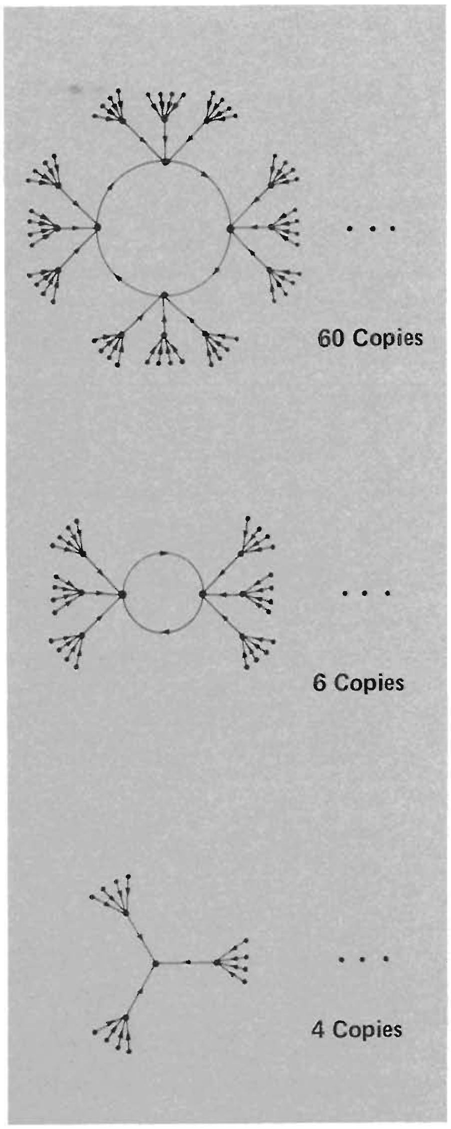


Fig. 6. The global state transition graph for a finite cellular automaton consisting of twelve sites arranged around a circle and evolving according to the simple rule of Eq. 1. Each node in the graph represents one of the 4096 possible states, or sequences of the twelve site values, of the cellular automaton. Each node is joined by a directed line to a successor node that corresponds to the state obtained by one time step of cellular automaton evolution. The state transition graph consists of many disconnected pieces, many of identical structure. Only one copy of each structurally identical piece is shown explicitly. Possible paths through the state transition graph represent possible trajectories in the state space of the cellular automaton. The merging of these trajectories reflects the irreversibility of the cellular automaton evolution. Any initial state of this cellular automaton ultimately evolves to an "attractor" represented in the graph by a cycle. For this particular cellular automaton all configurations evolve to attractors in at most three time steps. (From O. Martin, A. Odlyzko, and S. Wolfram, "Algebraic Properties of Cellular Automata," Bell Laboratories report (January 1983) and to be published in Communications in Mathematical Physics.*)*

automaton from the initial state represented by that particular node. The state transition graph of Fig. 6 shows all possible trajectories in state space for a cellular automaton with twelve sites evolving according to the simple rule of Eq. 1.

A notable feature of Fig. 6 is the presence of trajectories that merge with time. While each state has a unique successor in time, it may have several predecessors or no predecessors at all (as for states on the periphery of the state transition graph). The merging of trajectories implies that information is lost in the evolution of the cellular automaton: knowledge of the state attained by the system at a particular time is not sufficient to determine its history uniquely, so that the evolution is irreversible. Starting with an initial ensemble in which all configurations occur with any distribution of probabilities, the irreversible evolution decreases the probabilities for some configurations and increases those for others. For example, after just one time step the probabilities for states on the periphery of the state transition graph in Fig. 6 are reduced to zero; such states may be given as initial conditions, but may never be generated through evolution of the cellular automaton. After many time steps only a small number of all the possible configurations actually occur. Those that do occur may be considered to lie on "attractors" of the cellular automaton evolution. Moreover, if the attractor states have special "organized" features, these features will appear spontaneously in the evolution of the cellular automaton. The possibility of self-organization is therefore a consequence of the irreversibility of the cellular automaton evolution, and the structures obtained through self-organization are determined by the characteristics of the attractors.

The irreversibility of cellular automaton evolution revealed by Fig. 6 is to be contrasted with the intrinsic reversibility of systems described by conventional thermodynamics. At a microscopic level the trajectories representing the evolution of states in such systems never merge: each state has a unique predecessor, and no information is lost with time. Hence a completely disordered ensemble, in which all possible states occur with equal probabilities, remains disordered forever. Moreover, if nearby states are grouped (or "coarse-grained") together, as by imprecise measurements, then with time the probabilities for different groups of states will tend to equality, regardless of their initial values. In this way such systems tend with time to complete disorder and maximum entropy, as prescribed by the second law of thermodynamics. Tendency to disorder and increasing entropy are universal features of intrinsically reversible systems in statistical mechanics. Irreversible systems, such as the cellular automaton of Figs. 2, 3, and 4, counter this trend, but universal laws have yet to be found for their behavior and for the structures they may generate. One hopes that such general laws may ultimately be abstracted from an investigation of the comparatively simple examples provided by cellular automata.

While there is every evidence that the fundamental microscopic laws of physics are intrinsically reversible (information-preserving, though not precisely time-reversal invariant), many systems behave irreversibly on a macroscopic scale and are appropriately described by irreversible laws. For example, while the microscopic molecular interactions in a fluid are entirely reversible, macroscopic descriptions of the average velocity field in the fluid, using, say, the Navier-Stokes equations, are irreversible and contain dissipative terms. Cellular automata provide mathematical models at this macroscopic level.

Mathematical Analysis of a Simple Cellular Automaton

The cellular automaton rule of Eq. 1 is particularly simple and admits a rather complete mathematical analysis.

The fractal patterns of Figs. 2 and 3 may be characterized in a simple algebraic manner. If no reduction modulo 2 were performed, then the values of sites generated from a single nonzero initial site would simply be the integers appearing in Pascal's triangle of binomial coefficients. The pattern of nonzero sites in Figs. 2 and 3 is therefore the pattern of odd binomial coefficients in Pascal's triangle. (See Stephen Wolfram, "Geometry of Binomial Coefficients," to be published in *American Mathematical Monthly.*)

This algebraic approach may be extended to determine the structure of the state transition diagram of Fig. 6. (See O. Martin, A. Odlyzko, and S. Wolfram, "Algebraic Properties of Cellular Automata," Bell Laboratories report (January 1983) and to be published in *Communications in Mathematical Physics.*) The analysis proceeds by writing for each configuration a characteristic polynomial

$$A(x) = \sum_{i=0}^{N-1} a_i x^i ,$$

where x is a dummy variable, and the coefficient of x^i is the value of the site at position i. In terms of characteristic polynomials, the cellular automaton rule of Eq. 1 takes on the particularly simple form

$$A^{(t+1)}(x) = T(x)A^{(t)}(x) \mod (x^N-1) ,$$

where

$$T(x) = (x + x^{-1})$$

and all arithmetic on the polynomial coefficients is performed modulo 2. The reduction modulo x^N-1 implements periodic boundary conditions. The structure of the state transition diagram may then be deduced from algebraic properties of the polynomial $T(x)$. For even N one finds, for example, that the fraction of states on attractors is $2^{-D_2(N)}$, where $D_2(N)$ is defined as the largest integral power of 2 that divides N (for example, $D_2(12) = 4$).

Since a finite cellular automaton evolves deterministically with a finite total number of possible states, it must ultimately enter a cycle in which it visits a sequence of states repeatedly. Such cycles are manifest as closed loops in the state transition graph. The algebraic analysis of Martin *et al.* shows that for the cellular automaton of Eq.1 the maximal cycle length Π (of which all other cycle lengths are divisors) is given for even N by

$$\Pi_{N=2^j} = 1$$

or

$$\Pi_{N=2(2k+1)} = 2\Pi_{N=2k+1} .$$

For odd N, Π may be shown to divide

$$2^{\mathrm{sord}_N(2)} - 1$$

and in fact is almost always equal to this value (the first exception occurs for $N = 37$). Here $\mathrm{sord}_N(2)$ is a number theoretical function defined to be the minimum positive integer j for which $2^j = \pm 1$ modulo N. The maximum value of $\mathrm{sord}_N(2)$, typically achieved when N is prime, is $(N-1)/2$. The maximal cycle length is thus of order $2^{N/2}$, approximately the square root of the total number of possible states 2^N.

An unusual feature of this analysis is the appearance of number theoretical concepts. Number theory is inundated with complex results based on very simple premises. It may be part of the mathematical mechanism by which natural systems of simple construction yield complex behavior.

More General Cellular Automata

The discussion so far has concentrated on the particular cellular automaton rule given by Eq. 1. This rule may be generalized in several ways. One family of rules is obtained by allowing the value of a site to be an arbitrary function of the values of the site itself and of its two nearest neighbors on the previous time step:

$$a_i^{(t+1)} = F(a_{i-1}^{(t)}, a_i^{(t)}, a_{i+1}^{(t)}) .$$

A convenient notation illustrated in Fig. 7 assigns a "rule number" to each of the 256 rules of this type. The rule number of Eq. 1 is 90 in this notation.

Further generalizations allow each site in a cellular automaton to take on an arbitrary

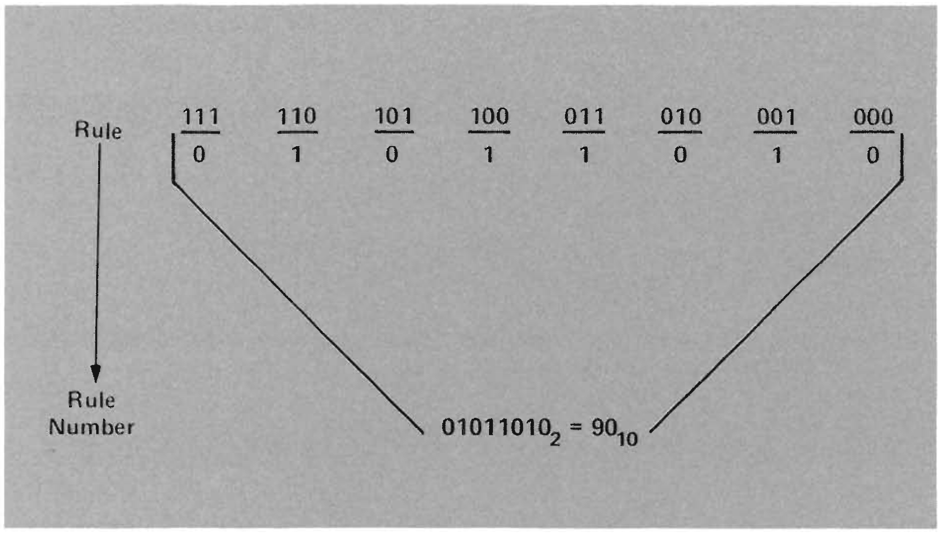


Fig. 7. Assignment of rule numbers to cellular automata for which k = *2 and* r = *1. The values of sites obtained from each of the eight possible three-site neighborhoods are combined to form a binary number that is quoted as a decimal integer. The example shown is for the rule given by Eq. 1.*

number k of values and allow the value of a site to depend on the values of sites at a distance up to r on both sides, so that

$$a_i^{(t+1)} = F(a_{i-r}^{(t)}, \ldots, a_{i+r}^{(t)}) .$$

The number of different rules with given k and r grows as $k^{k^{2r+1}}$ and therefore becomes immense even for rather small k and r.

Figure 8 shows examples of evolution according to some typical rules with various k and r values. Each rule leads to patterns that differ in detail. However, the examples suggest a very remarkable result: all patterns appear to fall into only four qualitative classes. These basic classes of behavior may be characterized empirically as follows:

○ Class 1—evolution leads to a homogeneous state in which, for example, all sites have value 0;
○ Class 2—evolution leads to a set of stable or periodic structures that are separated and simple;
○ Class 3—evolution leads to a chaotic pattern;
○ Class 4—evolution leads to complex structures, sometimes long-lived.

Examples of these classes are indicated in Fig. 8.

The existence of only four qualitative classes implies considerable universality in the behavior of cellular automata; many features of cellular automata depend only on the class in which they lie and not on the precise details of their evolution. Such universality is analogous, though probably not mathematically related, to the universality found in the equilibrium statistical mechanics of critical phenomena. In that case many systems with quite different detailed construction are found to lie in classes with critical exponents that depend only on general, primarily geometrical features of the systems and not on their detailed construction.

Universality Classes in Cellular Automata

To proceed in analyzing universality in cellular automata, one must first give more quantitative definitions of the classes identified above. One approach to such definitions is to consider the degree of predictability of the outcome of cellular automaton evolution, given knowledge of the initial state. For class 1 cellular automata complete prediction is trivial: regardless of the initial state, the system always evolves to a unique homogeneous state. Class 2 cellular automata have the feature that the effects of particular site values propagate only a finite distance, that is, only to a finite number of neighboring sites. Thus a change in the value of a single initial site affects only a finite region of sites around it, even after an infinite number of time steps. This behavior, illustrated in Fig. 9, implies that prediction of a particular final site value requires knowledge of only a finite set of initial site values. In contrast, changes of initial site values in class 3 cellular automata, again as illustrated in Fig. 9, almost always propagate at a finite speed forever and therefore affect more and more distant sites as time goes on. The value of a particular site after many time steps thus depends on an ever-increasing number of initial site values. If the initial state is disordered, this dependence may lead to an apparently chaotic succession of values for a particular site. In class 3 cellular automata, therefore, prediction of the value of a site at infinite time would require knowledge of an infinite number of initial site values. Class 4 cellular automata are distinguished by an even greater degree of unpredictability, as discussed below.

Class 2 cellular automata may be considered as "filters" that select particular features of the initial state. For example, a class 2 cellular automata may be constructed in which initial sequences 111 survive, but sites not in such sequences eventually attain

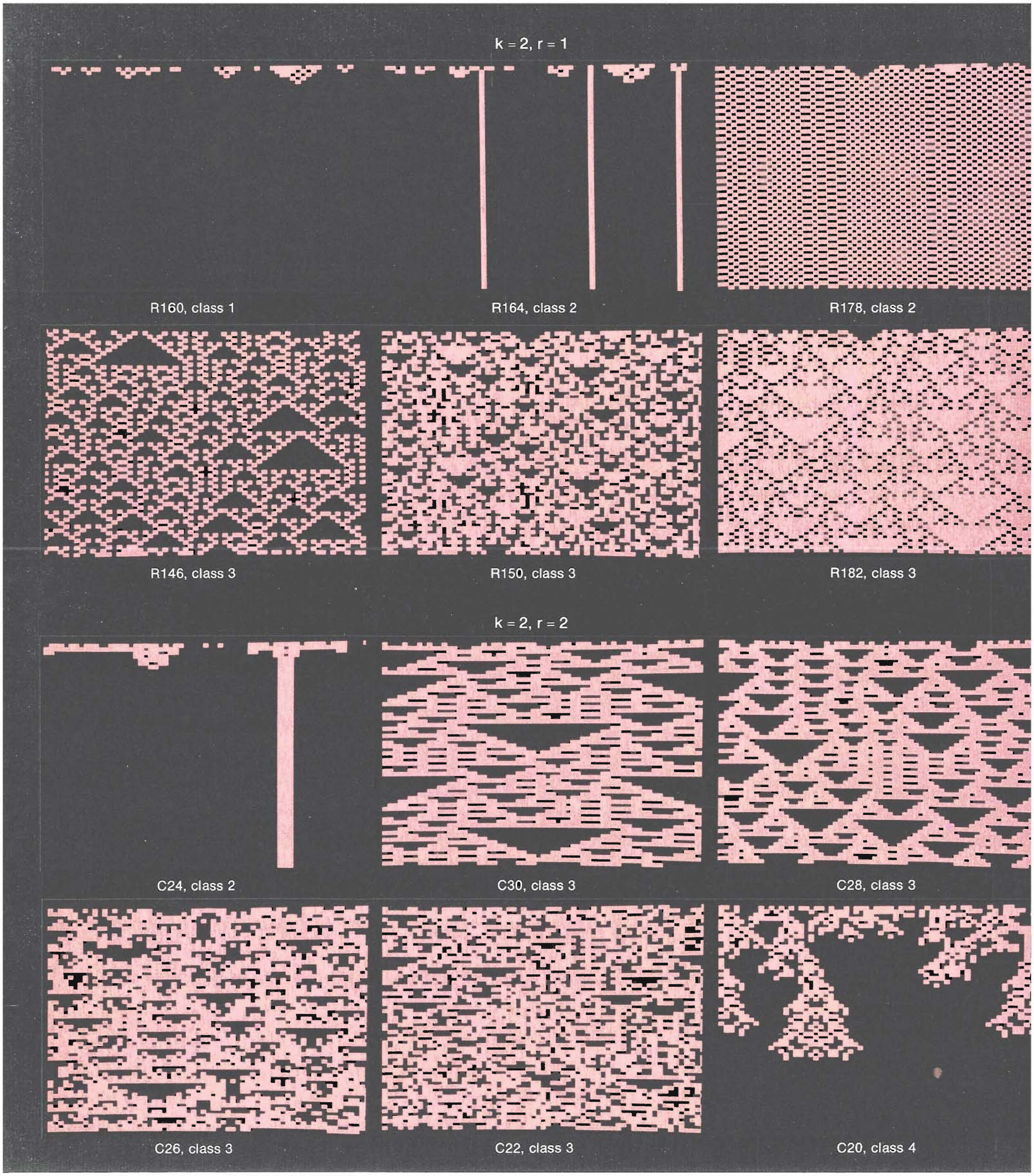
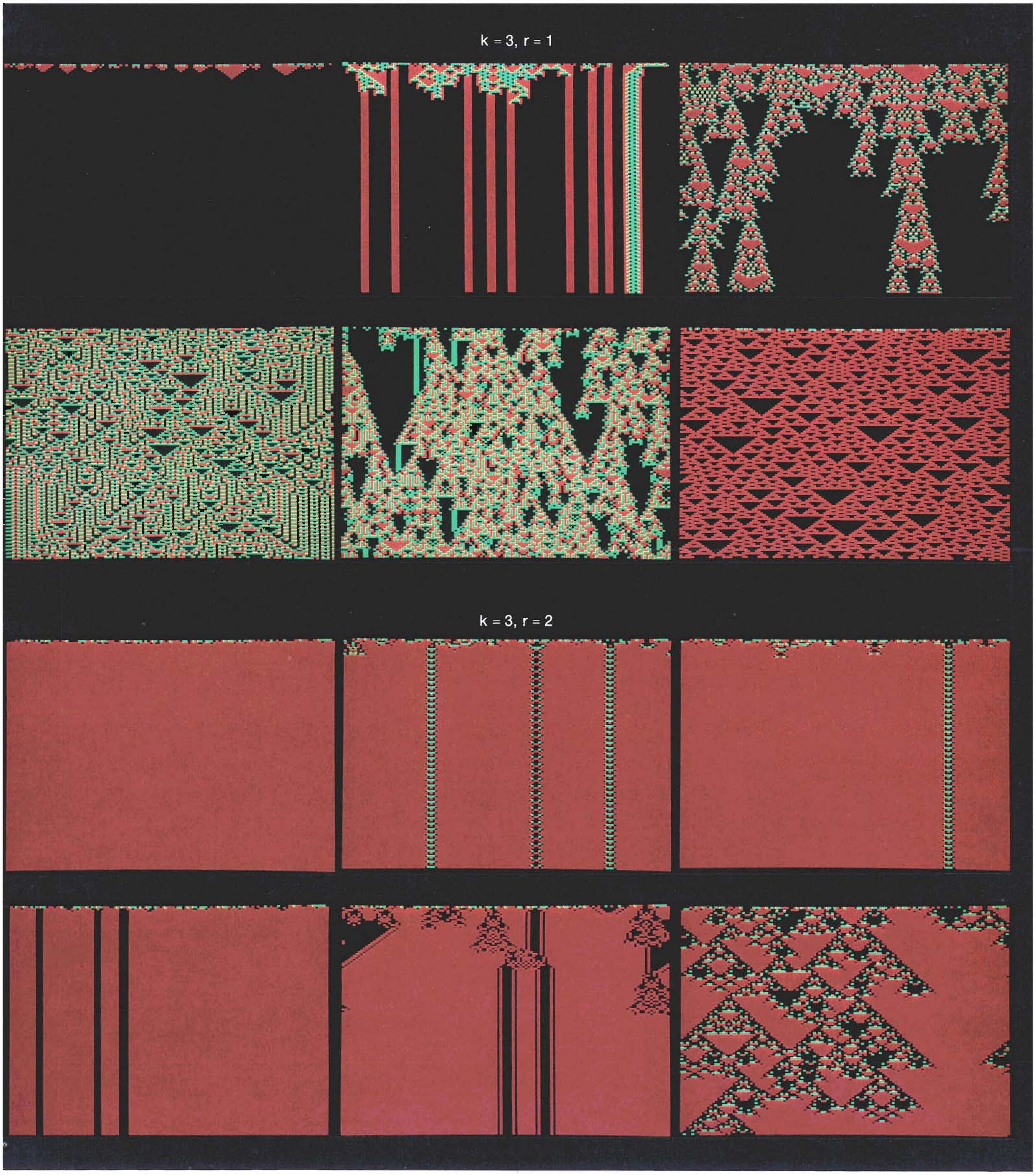
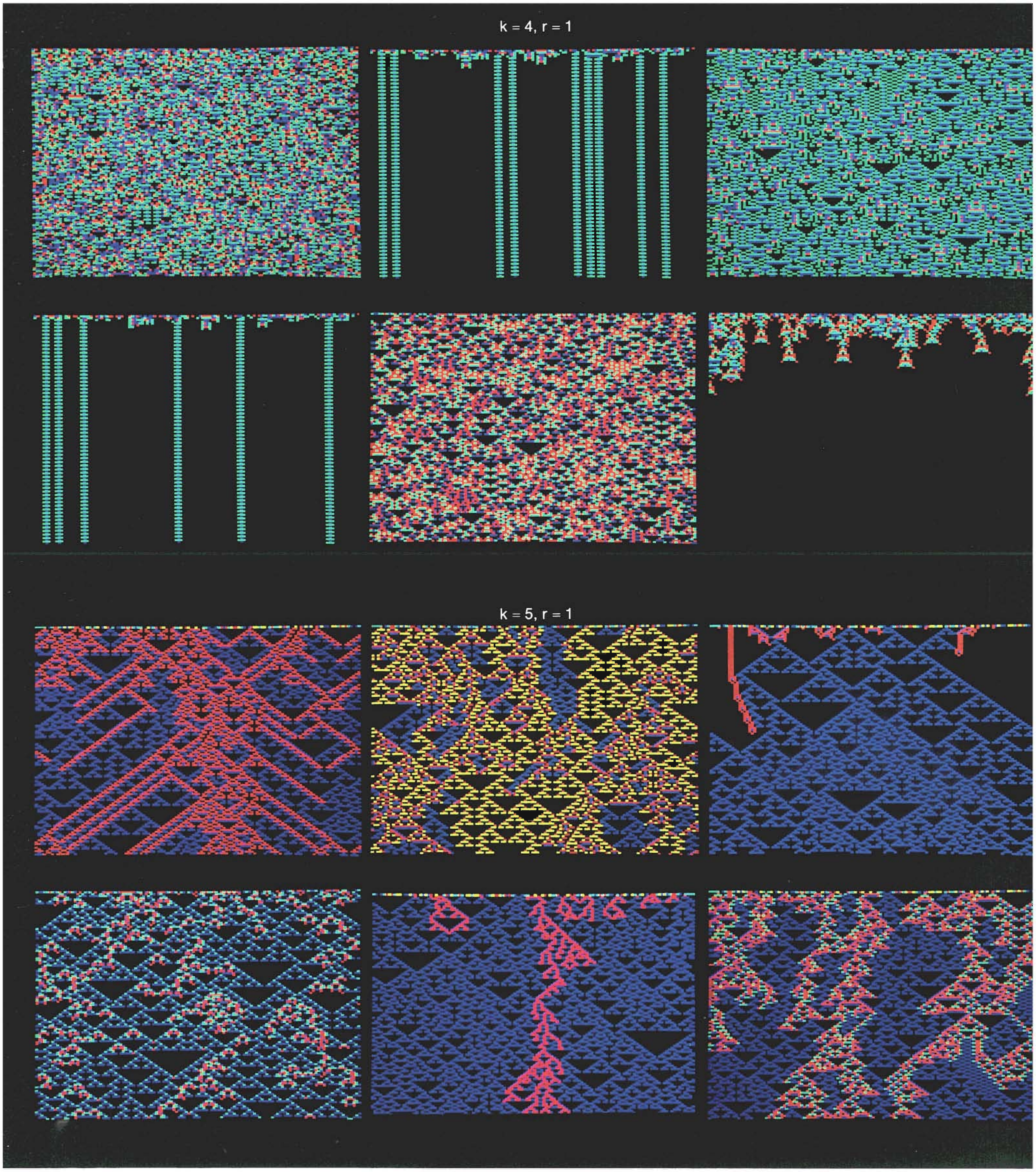


Fig. 8. Evolution of some typical cellular automata from disordered initial states. Each group of six patterns shows the evolution of various rules with particular values of k *and* r. *Sites take on* k *possible values, and the value of a site depends on the values of sites up to* r *sites distant on both sides. Different colors represent different site values: black corresponds to a value of 0, red to 1, green to 2, blue to 3, and yellow to 4. The fact that these and other examples exhibit only*



four qualitative classes of behavior (see text) suggests considerable universality in the behavior of cellular automata. The examples on page 10 for which r = *1 are labeled by rule number (in the notation of Fig. 7) and behavior class. The examples on page 10 for which* r = *2 evolve according to rules in which the value of a site depends only on the sum of the values of the 2*r + *1 sites in its neighborhood on the previous time step. Such rules may be specified by numerical codes* C



such that the coefficient of 2^j in the binary decomposition of C *gives the value attained by a site if its neighborhood had total value* j *on the previous time step. These examples are labeled by code number and behavior class. (I am grateful to R. Pike and J. Condon of Bell Laboratories for their help in preparing these and other color pictures of cellular automata.)*

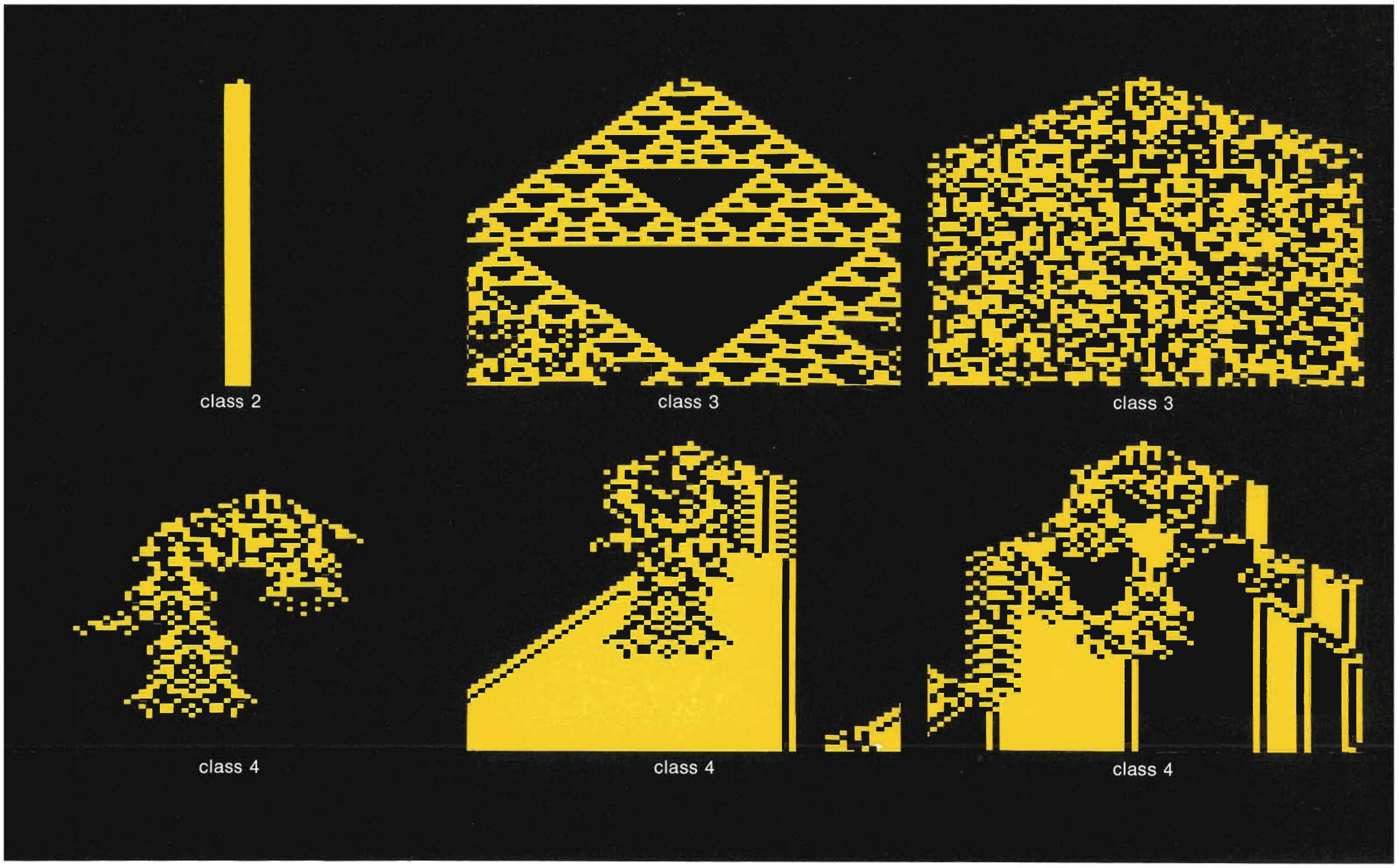


Fig. 9. Difference patterns showing the differences between configurations generated by evolution, according to various cellular automaton rules, from initial states that differ in the value of a single site. Each difference pattern is labeled by the behavior class of the cellular automaton rule. The effects of changes in a single site value depend on the behavior class of the rule: for class 2 rules the effects have finite range; for class 3 rules the effects propagate to neighboring sites indefinitely at a fixed speed; and for class 4 rules the effects also propagate to neighboring sites indefinitely but at various speeds. The difference patterns shown here are analogues of Green's functions for cellular automata.

value 0. Such cellular automata are of practical importance for digital image processing: they may be used to select and enhance particular patterns of pixels. After a sufficiently long time any class 2 cellular automaton evolves to a state consisting of blocks containing nonzero sites separated by regions of zero sites. The blocks have a simple form, typically consisting of repetitions of particular site values or sequences of site values (such as 101010...). The blocks either do not change with time (yielding vertical stripes in the patterns of Fig. 8) or cycle between a few states (yielding "railroad track" patterns).

While class 2 cellular automata evolve to give persistent structures with small periods, class 3 cellular automata exhibit chaotic aperiodic behavior, as shown in Fig. 8. Although chaotic, the patterns generated by class 3 cellular automata are not completely random. In fact, as mentioned for the example of Eq. 1, they may exhibit important self-organizing behavior. In addition and again in contrast to class 2 cellular automata, the statistical properties of the states generated by many time steps of class 3 cellular automaton evolution are the same for almost all possible initial states. The large-time behavior of a class 3 cellular automaton is therefore determined by these common statistical properties.

The configurations of an infinite cellular automaton consist of an infinite sequence of site values. These site values could be considered as digits in a real number, so that each complete configuration would correspond to a single real number. The topology of the real numbers is, however, not exactly the same as the natural one for the configurations (the binary numbers 0.111111... and 1.00000... are identical, but the corresponding configurations are not). Instead, the configurations of an infinite cellular automaton form a Cantor set. Figure 10 illustrates two constructions for a Cantor set. In construction (a) of Fig. 10, one starts with the set of real numbers in the interval 0 to 1. First one excludes the middle third of the interval, then the middle third of each interval remaining, and so on. In the limit the set consists of an infinite number of disconnected points. If positions in the interval are represented by ternimals (base 3 fractions, analogous to base 10 decimals), then the construction is seen to retain only points whose positions are represented by ternimals containing no 1's (the point 0.2202022 is therefore included; 0.2201022 is excluded). An important feature of the limiting set is its self-similarity, or fractal form: a piece of the set, when magnified, is indistinguishable from the whole. This self-similarity is mathematically analogous to that found for the limiting two-dimensional pattern of Fig. 3.

In construction (b) of Fig. 10, the Cantor set is formed from the "leaves" of an infinite binary tree. Each point in the set is reached by a unique path from the "root" (top as drawn) of the tree. This path is specified by an infinite sequence of binary digits, in which successive digits determine whether the left- or right-hand branch is taken at each successive level in the tree. Each point in the Cantor set corresponds uniquely to one infinite sequence of digits and thus to one configuration of an infinite cellular automaton. Evolution of the cellular automaton then corresponds to iterated mappings of the Cantor set to itself. (The locality of cellular automaton rules implies that the mappings are continuous.) This interpretation of cellular automata leads to analogies with the theory of iterated mappings of intervals of the real line. (See Mitchell J. Feigenbaum, "Universal Behavior in Nonlinear Systems," *Los Alamos Science*, Vol. 1, No. 1(1980): 4-27.)

Cantor sets are parameterized by their "dimensions." A convenient definition of dimension, based on construction (a) of Fig. 10, is as follows. Divide the interval from 0 to 1 into k^n bins, each of width k^{-n}. Then let $N(n)$ be the number of these bins that contain points in the set. For large n this number behaves according to

$$N(n) \sim k^{dn} , \qquad (2)$$

and d is defined as the "set dimension" of the Cantor set. If a set contained all points in the interval 0 to 1, then with this definition its dimension would simply be 1. Similarly, any finite number of segments of the real line would form a set with dimension 1. However, the Cantor set of construction (a), which contains an infinite number of disconnected pieces, has a dimension according to Eq. 2 of $\log_3 2 \simeq 0.63$.

An alternative definition of dimension, agreeing with the previous one for present

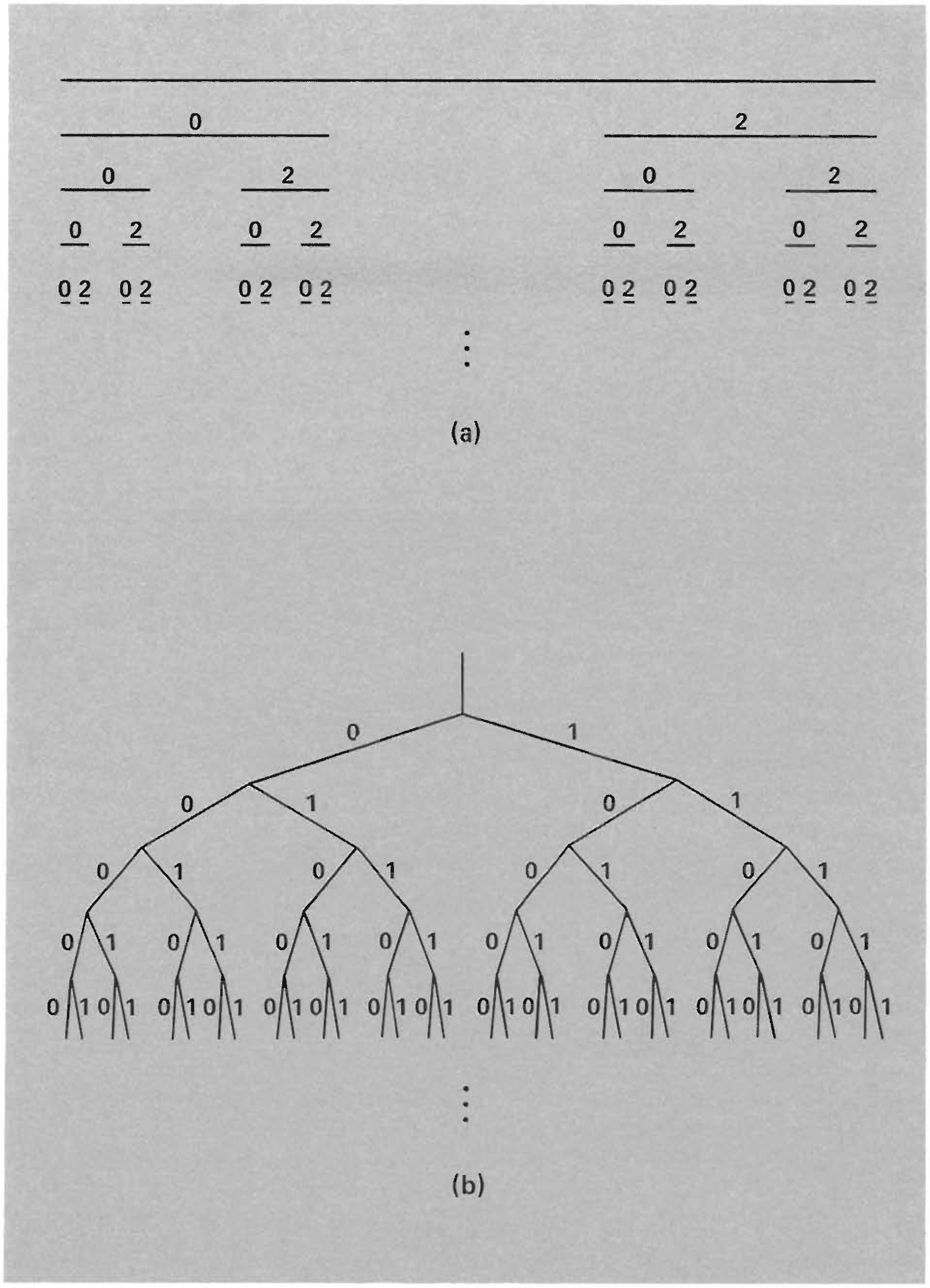


Fig. 10. Steps in two constructions of a Cantor set. At each step in construction (a), the middle third of all intervals is excluded. The first step thus excludes all points whose positions, when expressed as base 3 fractions, have a 1 in the first "ternimal place" (by analogy with decimal place), the second step excludes all points whose positions have a 1 in the second ternimal place, and so on. The limiting set obtained after an infinite number of steps consists of an infinite number of disconnected points whose positions contain no 1's. The set may be assigned a dimension, according to Eq. 2, that equals $\log_3 2 \simeq 0.63$. Construction (b) reflects the topological structure of the Cantor set. Infinite sequences of digits, representing cellular automaton configurations, are seen to correspond uniquely with points in the Cantor set.

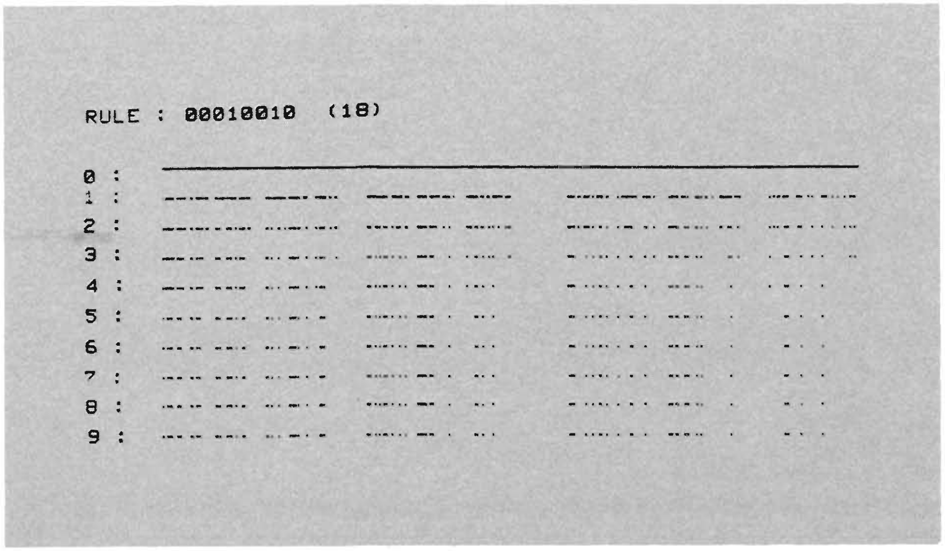


Fig. 11. Time evolution of the probabilities for each of the 1024 possible configurations of a typical class 3 cellular automaton with k = 2 ***and*** r = 1 ***and of size 10, starting from an initial ensemble in which each possible configuration occurs with equal probability. The configurations are specified by integers whose binary digits form the sequence of site values. The probability for a particular configuration is given on successive lines in a vertical column: a dot appears at a particular time step if the configuration occurs with nonzero probability at that time step. In the initial ensemble all configurations occur with equal nonzero probabilities, and dots appear in all positions. The cellular automaton evolution modifies the probabilities for the configurations, making some occur with zero probability and yielding gaps in which no dots appear. This "thinning" is a consequence of the irreversibility of the cellular automaton evolution and is reflected in a decrease of entropy with time. In the limit of cellular automata of infinite size, the configurations appearing at large times form a Cantor set. For the rule shown (rule 18 in the notation of Fig. 7) the limiting dimension of this Cantor set is found to be approximately 0.88.***

purposes, is based on self-similarity. Take the Cantor set of construction (a) in Fig. 10. Contract the set by a magnification factor k^{-m}. By virtue of its self-similarity, the whole set is identical to a number, say $M(m)$, of copies of this contracted copy. For large m, $M(m) \approx k^{dm}$, where again d is defined as the set dimension.

With these definitions the dimension of the Cantor set of all possible configurations for an infinite one-dimensional cellular automaton is 1. A disordered ensemble, in which each possible configuration occurs with equal probability, thus has dimension 1. Figure 11 shows the behavior of the probabilities for the configurations of a typical cellular automaton as a function of time, starting from such a disordered initial ensemble. As expected from the irreversibility of cellular automaton evolution, exemplified by the state transition graph of Fig. 6, different configurations attain different probabilities as evolution proceeds, and the probabilities for some configurations decrease to zero. This phenomenon is manifest in the "thinning" of configurations on successive time steps apparent in Fig. 11. The set of configurations that survive with nonzero probabilities after many time steps of cellular automaton evolution constitutes the "attractors" for the evolution. This set is again a Cantor set; for the example of Fig. 11 its dimension is $\log_2 \kappa \simeq 0.88$, where $\kappa \simeq 1.755$ is the real solution of the polynomial equation $z^3 - z^2 + 2z - 1 = 0$. (See D. A. Lind, "Applications of Ergodic Theory and Sofic Systems to Cellular Automata," University of Washington preprint (April 1983) and to be published in *Physica D*; see also Martin *et al.*, *op. cit.*) The greater the irreversibility in the cellular automaton evolution, the smaller is the dimension of the Cantor set corresponding to the attractors for the evolution. If the set of attractors for a cellular automaton has dimension 1, then essentially all the configurations of the cellular automaton may occur at large times. If the attractor set has dimension less than 1, then a vanishingly small fraction of all possible configurations are generated after many time steps of evolution. The attractor sets for most class 3 cellular automata have dimensions less than 1. For those class 3 cellular automata that generate regular patterns, the more regular the pattern, the smaller is the dimension of the attractor set; these cellular automata are more irreversible and are therefore capable of a higher degree of self-organization.

The dimension of a set of cellular automaton configurations is directly proportional to the limiting entropy (or information) per site of the sequence of site values that make up the configurations. (See Patrick Billingsley, *Ergodic Theory and Information*, John Wiley & Sons, 1965.) If the dimension of the set was 1, so that all possible sequences of site values could occur, then the entropy of these sequences would be maximal. Dimensions lower than 1 correspond to sets in which some sequences of site values are absent, so that the entropy is reduced. Thus the dimension of the attractor for a cellular automaton is directly related to the limiting entropy attained in its evolution, starting from a disordered ensemble of initial states.

Dimension gives only a very coarse measure of the structure of the set of configurations reached at large times in a cellular automaton. Formal language theory may provide a more complete characterization of the set. "Languages" consist of a set of words, typically infinite in number, formed from a sequence of letters according to certain grammatical rules. Cellular automaton configurations are analogous to words in a formal language whose letters are the k possible values of each cellular automaton site. A grammar then gives a succinct specification for a set of cellular automaton configurations.

Languages may be classified according to the complexity of the machines or computers necessary to generate them. A simple class of languages specified by "regular grammars" may be generated by finite state machines. A finite state machine is represented by a state transition graph (analogous to the state transition graph for a finite cellular automaton illustrated in Fig. 6). The possible words in a regular grammar are generated by traversing all possible paths in the state transition graph. These words may be specified by "regular expressions" consisting of finite length sequences and arbitrary repetitions of these. For example, the regular expression 1(00)*1 represents all sequences containing an even number of 0's (arbitrary repetition of the sequence 00) flanked by a pair of 1's. The set of configurations obtained at large times in class 2 cellular automata is found to form a regular language. It is likely that attractors for other classes of cellular automata correspond to more complicated languages.

Analogy with Dynamical Systems Theory

The three classes of cellular automaton behavior discussed so far are analogous to three classes of behavior found in the solutions to differential equations (continuous dynamical systems). For some differential equations the solutions obtained with any initial conditions approach a fixed point at large times. This behavior is analogous to class 1 cellular automaton behavior. In a second class of differential equations, the limiting solution at large times is a cycle in which the parameters vary periodically with time. These equations are analogous to class 2 cellular automata. Finally, some differential equations have been found to exhibit complicated, apparently chaotic behavior depending in detail on their initial conditions. With the initial conditions specified by decimals, the solutions to these differential equations depend on progressively higher and higher order digits in the initial conditions. This phenomenon is analogous to the dependence of a particular site value on pro-

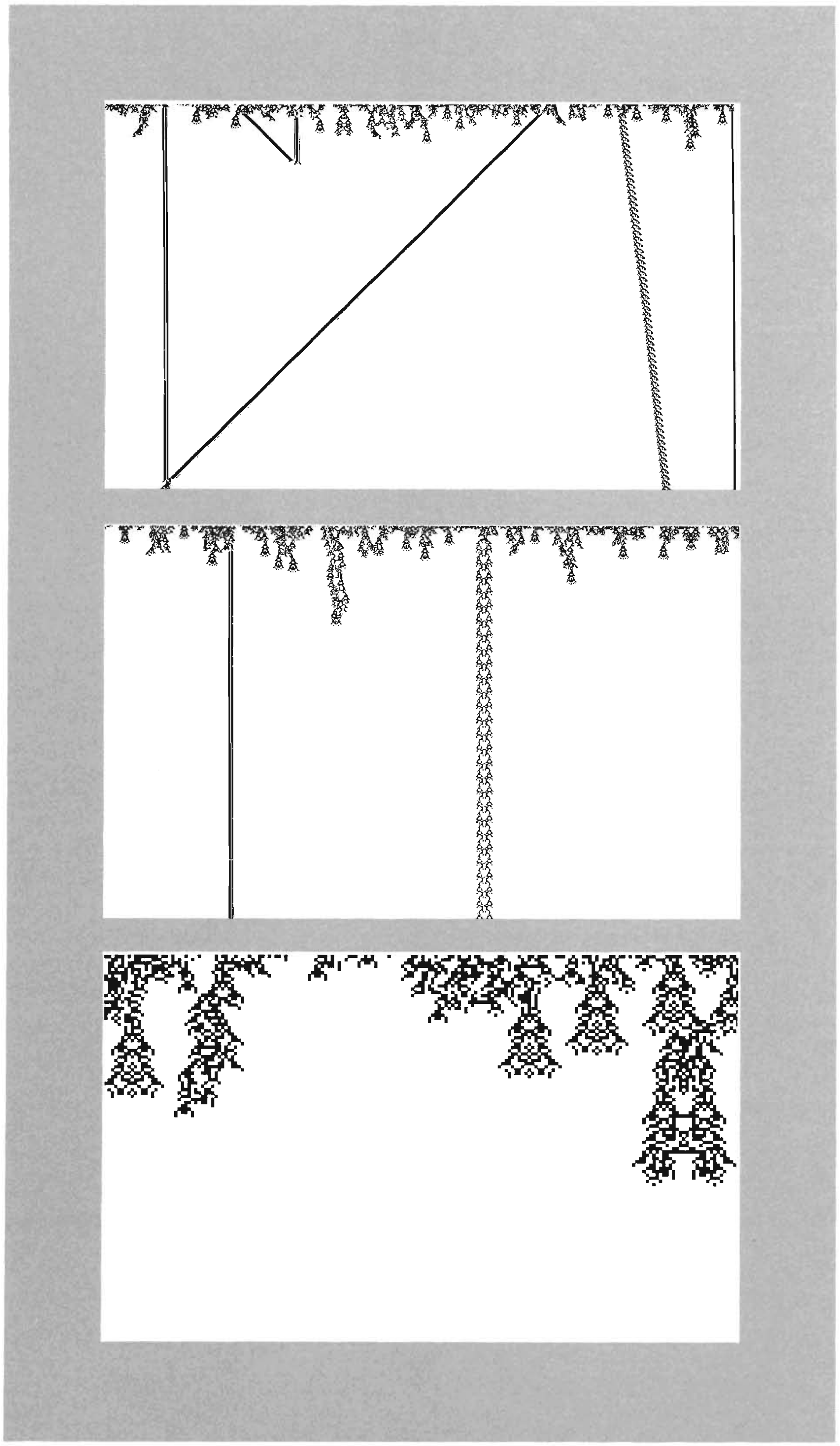


Fig. 12. Evolution of a class 4 cellular automaton from several disordered initial states. The bottom example has been reproduced on a larger scale to show detail. In this cellular automaton, for which k = 2 *and* r = 2*, the value of a site is 1 only if a total of two or four sites out of the five in its neighborhood have the value 1 on the previous time step. For some initial states persistent structures are formed, some of which propagate with time. This cellular automaton is believed to support universal computation, so that with suitable initial states it may implement any finite algorithm.*

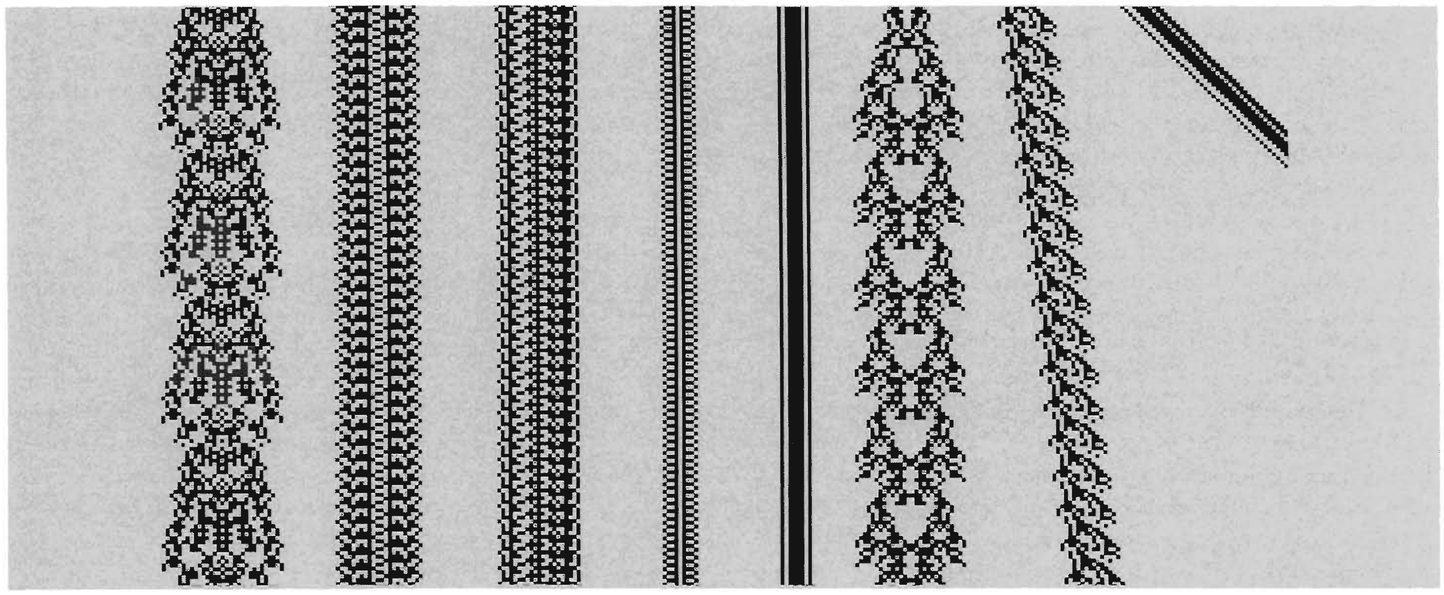


Fig. 13. Persistent structures exhibited by the class 4 cellular automaton of Fig. 12 as it evolves from initial states with nonzero sites in a region of twenty or fewer sites. These structures are almost sufficient to demonstrate a universal computation capability for the cellular automaton.

gressively more distant initial site values in the evolution of a class 3 cellular automaton. The solutions to this final class of differential equations tend to "strange" or "chaotic" attractors (see Robert Shaw, "Strange Attractors, Chaotic Behavior, and Information Flow," *Zeitschrift für Naturforschung* 36A(1981):80), which form Cantor sets in direct analogy with those found in class 3 cellular automata. The correspondence between classes of behavior found in cellular automata and those found in continuous dynamical systems supports the generality of these classes. Moreover, the greater mathematical simplicity of cellular automata suggests that investigation of their behavior may elucidate the behavior of continuous dynamical systems.

A Universal Computation Class of Cellular Automata

Figure 12 shows patterns obtained by evolution from disordered initial states according to a class 4 cellular automaton rule. Complicated behavior is evident. In most cases all sites eventually "die" (attain value 0). In some cases, however, persistent structures that survive for an infinite time are generated, and a few of these persistent structures propagate with time. Figure 13 shows all the persistent structures generated from initial states with nonzero sites in a region of twenty or fewer sites. Unlike the periodic structures of class 2 cellular automata, these persistent structures have no simple patterns. In addition, the propagating structures allow site values at one position to affect arbitrarily distant sites after a sufficiently long time. No analogous behavior has yet been found in a continuous dynamical system.

The complexity apparent in the behavior of class 4 cellular automata suggests the conjecture that these systems may be capable of universal computation. A computer may be regarded as a system in which definite rules are used to transform an initial sequence of, say, 1's and 0's to a final sequence of 1's and 0's. The initial sequence may be considered as a program and data stored in computer memory, and part of the final sequence may be considered as the result of the computation. Cellular automata may be considered as computers; their initial configurations represent programs and initial data, and their configurations after a long time contain the results of computations.

A system is a universal computer if, given a suitable initial program, its time evolution can implement any finite algorithm. (See Frank S. Beckman, *Mathematical Foundations of Programming*, Addison-Wesley Publishing Co., 1980.) A universal computer need thus only be "reprogrammed," not "rebuilt," to perform each possible calculation. (All modern general-purpose electronic digital computers are, for practical purposes, universal computers; mechanical adding machines were not.) If a cellular automaton is to be a universal computer, then, with a fixed rule for its time evolution, different initial configurations must encode all possible programs.

The only known method of proving that a system may act as a universal computer is to show that its computational capabilities are equivalent to those of another system already classified as a universal computer. The Church-Turing thesis states that no system may have computational capabilities greater than those of universal computers. The thesis is supported by the proven equivalence of computational models such as Turing machines, string-manipulation systems, idealized neural networks, digital computers, and cellular automata. While mathematical systems with computational power beyond that of universal computers may be imagined, it seems likely that no such systems could be built with physical components. This conjecture could in principle be proved by showing that all physical systems could be simulated by a universal computer. The main obstruction to such a proof involves quantum mechanics.

A cellular automaton may be proved capable of universal computation by identifying structures that act as the essential components of digital computers, such as wires, NAND gates, memories, and clocks. The persistent structures illustrated in Fig. 13 provide many of the necessary components, strongly suggesting that the cellular automaton of Figs. 12 and 13 is a universal computer. One important missing component is a "clock" that generates an infinite sequence of "pulses"; starting from an initial configuration containing a finite number of nonzero sites, such a structure would give rise to an ever-increasing number of nonzero sites. If such a structure exists, it can undoubtedly be found by careful investigation, although it is probably too large to be found by any practical exhaustive search. If the cellular automaton of Figs. 12 and 13 is indeed capable of universal computation, then, despite its very simple construction, it is in some sense capable of arbitrarily complicated behavior.

Several complicated cellular automata have been proved capable of universal computation. A one-dimensional cellular automaton with eighteen possible values at each site (and nearest neighbor interactions) has been shown equivalent to the simplest known universal Turing machine. In two dimensions several cellular automata with just two states per site and interactions between nearest neighbor sites (including diagonally adjacent sites, giving a nine-site neighborhood) are known to be equivalent to universal digital computers. The best known of these cellular automata is the "Game of Life" invented by Conway in the early 1970s and simulated extensively ever since. (See Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy, *Winning Ways*, Academic Press, 1982 and Martin Gardner, *Wheels, Life, and Other Mathematical Amusements*, W. H. Freeman and Company, October 1983. The Life rule takes a site to have value 1 if three and only three of its eight neighbors are 1 or if four are 1 and the site itself was 1 on the previous time step.) Structures analogous to those of Fig. 13 have been identified in the Game of Life. In addition, a clock structure, dubbed the glider gun, was found after a long search.

By definition, any universal computer may in principle be simulated by any other universal computer. The simulation proceeds by emulating the elementary operations in the first universal computer by sets of operations in the second universal computer, as in an "interpreter" program. The simulation is in general only faster or slower by a fixed finite factor, independent of the size or duration of a computation. Thus the behavior of a universal computer given particular input may be determined only in a time of the same order as the time required to run that universal computer explicitly. In general the behavior of a universal computer cannot be predicted and can be determined only by a procedure equivalent to observing the universal computer itself.

If class 4 cellular automata are indeed universal computers, then their behavior may be considered completely unpredictable. For class 3 cellular automata the values of particular sites after a long time depend on an ever-increasing number of initial sites. For class 4 cellular automata this dependence is by an algorithm of arbitrary complexity, and the values of the sites can essentially be found only by explicit observation of the cellular automaton evolution. The apparent unpredictability of class 4 cellular automata introduces a new level of uncertainty into the behavior of natural systems.

The unpredictability of universal computer behavior implies that propositions concerning the limiting behavior of universal computers at indefinitely large times are formally undecidable. For example, it is undecidable whether a particular universal computer, given particular input data, will reach a special "halt" state after a finite time or will continue its computation forever. Explicit simulations can be run only for finite times and thus cannot determine such infinite time behavior. Results may be obtained for some special input data, but no general (finite) algorithm or procedure may even in principle be given. If class 4 cellular automata are indeed universal computers, then it is undecidable (in general) whether a particular initial state will ultimately evolve to the null configuration (in which all sites have value 0) or will generate persistent structures. As is typical for such generally undecidable propositions, particular cases may be decided. In fact, the halting of the cellular automaton of Figs. 12 and 13 for all initial states with nonzero sites in a region of twenty sites has been determined by explicit simulation. In general, the halting probability, or fraction of initial configurations ultimately evolving to the null configuration, is a noncomputable number. However, the explicit results for small initial patterns suggest that for the cellular automaton of Figs. 12 and 13, this halting probability is approximately 0.93.

In an infinite disordered configuration all possible sequences of site values appear at some point, albeit perhaps with very small probability. Each of these sequences may be considered to represent a possible "program"; thus with an infinite disordered initial state, a class 4 automaton may be considered to execute (in parallel) all possible programs. Programs that generate structures of arbitrarily great complexity occur, at least with indefinitely small probabilities. Thus, for example, somewhere on the infinite line a sequence that evolves to a self-reproducing structure should occur. After a sufficiently long time this configuration may reproduce many times, so that it ultimately dominates the behavior of the cellular automaton. Even though the *a priori* probabililty for the occurrence of a self-reproducing structure in the initial state is very small, its *a posteriori* probability after many time steps of cellular automaton evolution may be very large. The possibility that arbitrarily complex behavior seeded by features of the initial state can occur in class 4 cellular automata with indefinitely low probability prevents the taking of meaningful statistical averages over infinite volume (length). It also suggests that in some sense any class 4 cellular automaton with an infinite disordered initial state is a microcosm of the universe.

In extensive samples of cellular automaton rules, it is found that as k and r increase, class 3 behavior becomes progressively more dominant. Class 4 behavior occurs only for $k > 2$ or $r > 1$; it becomes more common for larger k and r but remains at the few percent level. The fact that class 4 cellular automata exist with only three values per site and nearest neighbor interactions implies that the threshold in complexity of construction necessary to allow arbitrarily complex behavior is very low. However, even among systems of more complex construction, only a small fraction appear capable of arbitrarily complex behavior. This suggests that some physical systems may be characterized by a capability for class 4 behavior and universal computation; it is the evolution of such systems that may be responsible for very complex structures found in nature.

The possibility for universal computation in cellular automata implies that arbitrary computations may in principle be performed by cellular automata. This suggests that cellular automata could be used as practical parallel-processing computers. The mechanisms for information processing found in most natural systems (with the exception of those, for example, in molecular genetics) appear closer to those of cellular automata than to those of Turing machines or conventional serial-processing digital computers. Thus one may suppose that many natural systems could be simulated more efficiently by cellular automata than by conventional computers. In practical terms the homogeneity of cellular automata leads to simple implementation by integrated circuits. A simple one-dimensional universal cellular automaton with perhaps a million sites and a time step as short as a billionth of a second could perhaps be fabricated with current

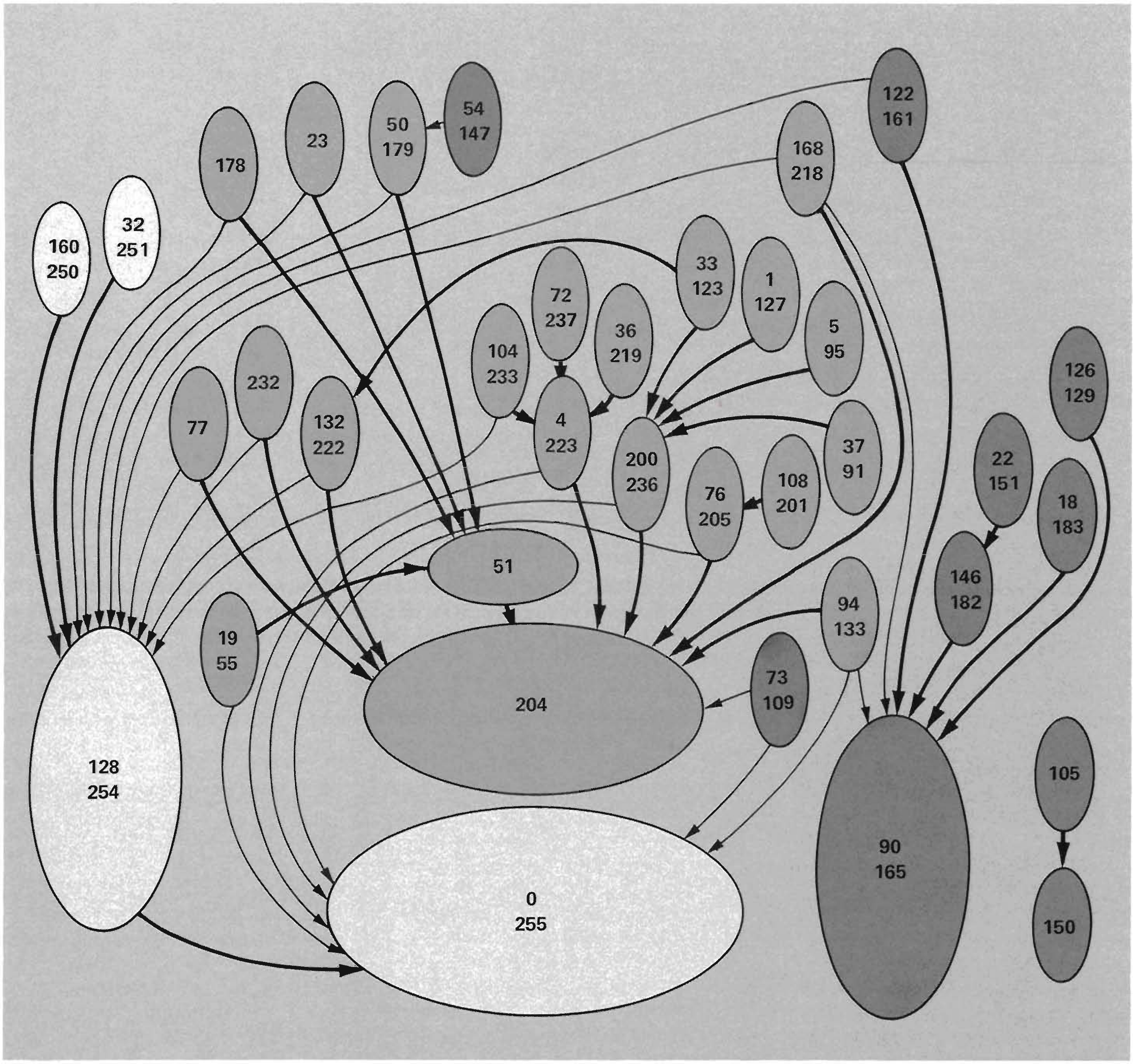


Fig. 14. Simulation network for symmetric cellular automaton rules with k = *2 and* r = *1. Each rule is specified by the number obtained as shown in Fig. 7, and its behavior class is indicated by shades of gray: light gray corresponds to class 1, medium gray to class 2, and dark gray to class 3. Rule A is considered to simulate rule B if there exist blocks of site values that evolve under rule A as single sites would evolve under rule B. Simulations are included in the network shown only when the necessary blocks are three or fewer sites long. Rules 90 and 150 are additive class 3 rules, rule 204 is the identity rule, and rules 170 and 240 are left- and right-shift rules, respectively. Attractive simulation paths are indicated by bold lines. (Network courtesy of J. Milnor.)*

technology on a single silicon wafer (the one-dimensional homogeneous structure makes defects easy to map out). Conventional programming methodology is, of course, of little utility for such a system. The development of a new methodology is a difficult but important challenge. Perhaps tasks such as image processing, which are directly suitable for cellular automata, should be considered first.

A Basis for Universality?

The existence of four classes of cellular automata was presented above as a largely empirical result. Techniques from computation theory may provide a basis, and ultimately a proof, of this result.

The first crucial observation is that with special initial states one cellular automaton may behave just like another. In this way one cellular automaton may be considered to "simulate" another. A single site with a particular value in one cellular automaton may be simulated by a fixed block of sites in another; after a fixed number of time steps, the evolution of these blocks imitates the single time-step evolution of sites in the first cellular automaton. For example, sites with value 0 and 1 in the first cellular automaton may be simulated by blocks of sites 00 and 11, respectively, in the second cellular automaton, and two time steps of evolution in the second cellular automaton correspond to one time step in the first. Then, with a special initial state containing 11 and 00 but not 01 and 10 blocks, the second cellular automaton may simulate the first.

Figure 14 gives the network that represents the simulation capabilities of symmetric cellular automata with $k = 2$ and $r = 1$. (Only simulations involving blocks of length less than four sites were included in the construction of the network.) If a cellular automaton is computationally universal, then with a sufficiently long encoding it should be able to simulate any other cellular automaton, so that a path should exist from the node that represents its rule to nodes representing all other possible rules.

An example of the simulation of one cellular automaton by another is the simulation of the additive rule 90 (Eq. 1) by the class 3 rule 18. A rule 18 cellular automaton behaves exactly like a rule 90 cellular automaton if alternate sites in the initial configuration have value 0 (so that 0 and 1 in rule 90 are represented by 00 and 01 in rule 18) and alternate time steps are considered. Figure 15 shows evolution according to rule 18 from a disordered initial state. Two "phases" are clearly evident: one in which sites at even-numbered positions have value 0 and one in which sites at odd-numbered positions have value 0. The boundaries between these regions execute approximately random walks and eventually annihilate in pairs, leaving a system consisting of blocks of sites that evolve according to the additive rule 90. (*Cf.* P. Grassberger, "Chaos and Diffusion in Deterministic

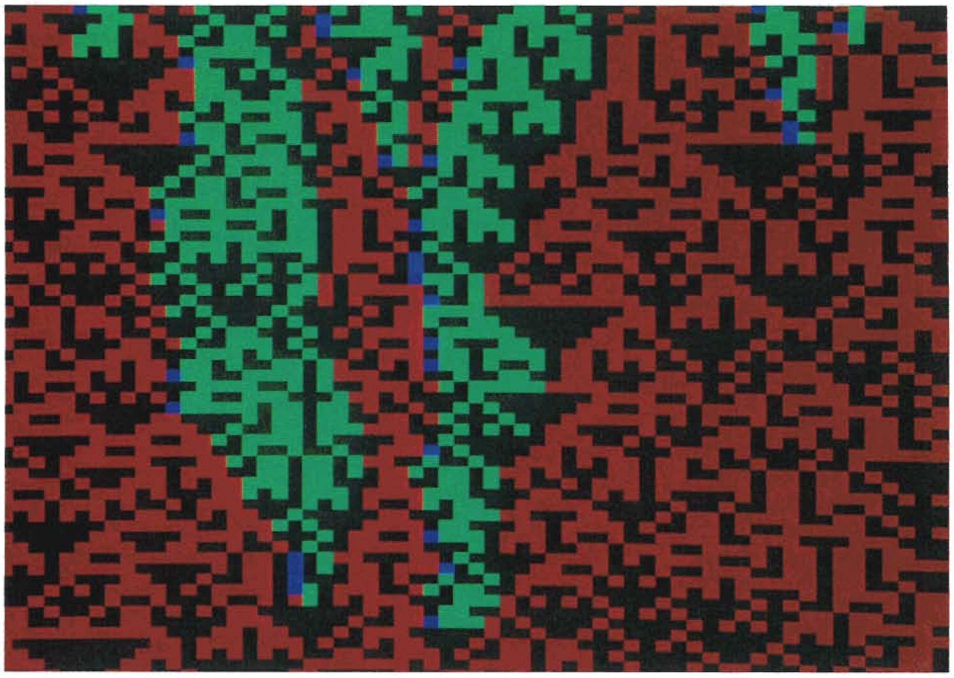


Fig. 15. Evolution of the class 3 cellular automaton rule 18 from a disordered initial state with pairs of sites combined. The pair of site values 00 is shown as black, 01 as red, 10 as green, and 11 as blue. At large times two phases are clearly evident, separated by "defects" that execute approximately random walks and ultimately annihilate in pairs. In each phase alternate sites have value 0, and the other sites evolve according to the additive rule 90. Thus for almost all initial states rule 18 behaves like rule 90 at large times. Rule 18 therefore follows an attractive simulation path to rule 90.

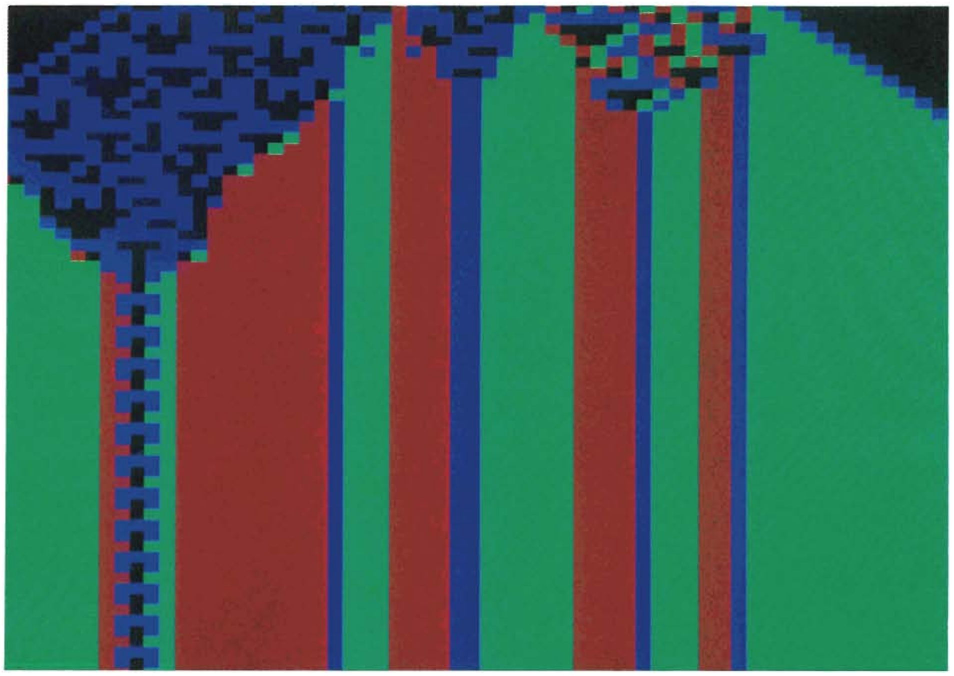
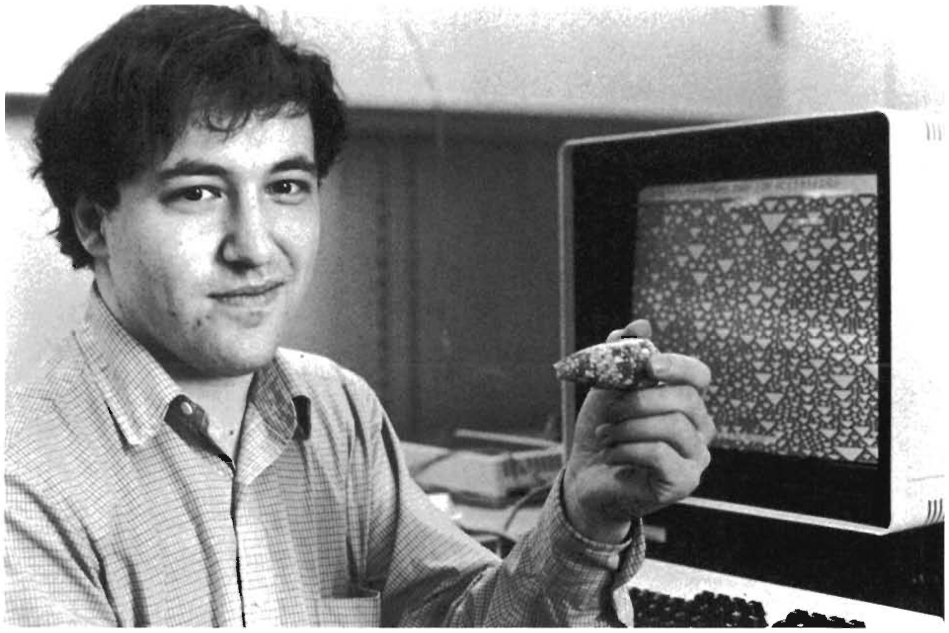


Fig. 16. Evolution of the class 2 cellular automaton rule 94 from an initial state in which the members of most pairs of sites have the same values, so that the digrams 00 and 11 predominate and the sequences 010 and 101 are nearly absent. (Color designations are the same as in Fig. 15.) Class 3 behavior occurs, but is unstable; class 2 behavior is "seeded" by 10 and 01 digrams and ultimately dominates. Rule 94 exhibits a repulsive simulation path to the class 3 additive rule 90 and an attractive path to the identity rule 204.

Cellular Automata," to be published in *Physica D*.) Thus the simulation of rule 90 by rule 18 may be considered an "attractive" one: starting from almost all initial states, rule 18 evolves toward states in which it simulates rule 90. In general, one expects that some paths in the network of Fig. 14 are attractive, while the rest are repulsive. The consequences of a repulsive simulation path are illustrated in Fig. 16: with special initial states rule 94 behaves like rule 90, but any impurities in the initial states grow and eventually dominate the evolution of the system.

Class 1 cellular automata have an attractive simulation path to rule 0 (or its equivalents). Class 2 cellular automata have attractive simulation paths to the identity rule 204. A conjecture for which some evidence exists is that all class 3 rules exhibit attractive simulations has to additive rules such as 90 or 150. Simulation by blocking of site values is analogous to a block spin or renormalization group transformation; additive rules have the special property that they are invariant under such transformations. As mentioned earlier, class 4 cellular automata are distinguished by the presence of simulation paths leading to every other cellular automaton rule. It is likely that no specific path is distinguished as attractive.

Cellular automata of different classes may thus be distinguished by their limiting behavior under simulation transformations. This approach suggests that classification of the qualitative behavior of cellular automata may be related to determinations of equivalence of systems and problem classes in computation theory. In general, one may hope for fundamental connections between computation theory and the theory of complex nonequilibrium statistical systems. Information theory forms a mathematical basis for equilibrium statistical mechanics. Computation theory, which addresses time-dependent processes, may be expected to play a fundamental role in nonequilibrium statistical mechanics. ■



Stephen Wolfram was born in London, England in 1959. He was educated at Eton College, Oxford University, and the California Institute of Technology where he received his Ph.D. in theoretical physics in 1979. He was a member of the faculty at Caltech from 1980 until he resigned in 1982. At that time he joined the Institute for Advanced Study in Princeton, New Jersey. He has worked in various areas of theoretical physics, including high-energy physics, cosmology, and statistical mechanics and has also worked in computer science, particularly in the area of symbolic computation. He received a MacArthur Fellowship in 1981 and since 1982 has been a Visiting Staff Member of the Theoretical Division at Los Alamos.

Acknowledgments

I am grateful to many people for discussions and suggestions. I thank in particular my collaborators in various cellular automaton investigations: O. Martin, J. Milnor, and A. Odlyzko. The research described here was supported in part by the Office of Naval Research under contract number N00014-80-C-0657.

Further Reading

John von Neumann. Edited and completed by Arthur W. Burks. *Theory of Self-Reproducing Automata.* Urbana: University of Illinois Press, 1966.

Arthur W. Burks, editor. *Essays on Cellular Automata.* Urbana: University of Illinois Press, 1970.

Stephen Wolfram. "Statistical Mechanics of Cellular Automata." *Reviews of Modern Physics* 55(1983):601.

Stephen Wolfram. "Universality and Complexity in Cellular Automata." The Institute for Advanced Study preprint (May 1983) and to be published in *Physica D.*

Stephen Wolfram, J. Doyne Farmer, and Tommaso Toffoli, editors. "Cellular Automata: Proceedings of an Interdisciplinary Workshop (Los Alamos; March 7-11, 1983)." To be published in *Physica D* and to be available separately from North-Holland Publishing Company.

From Turing and von Neumann to the Present

by Necia G. Cooper

automaton—a mechanism that is relatively self-operating; a device or machine designed to follow automatically a predetermined sequence of operations or respond to encoded instructions.

The notion of automata in the sense of machines that operate on their own from encoded instructions is very ancient, and one might say that mechanical clocks and music boxes fall under this category. The idea of computing machines is also very old. For instance, Pascal and Leibnitz outlined various schematics for such machines. In the latter part of the 18th century Baron de Kempelen built what was alleged to be the first chess-playing machine. Remarkable as it appeared, alas, it was a fake operated by a person hidden within it!

The modern theory of automata can be traced to two giants in the field of mathematics, Alan Turing and John von Neumann. These two men laid much of the logical foundation for the development of present-day electronic computers, and both were involved in the practical design of real computing machines.

Before World War II Turing had proved the logical limits of computability and on the basis of this work had designed in idealized terms a universal computer, a machine that could perform all possible numerical computations. This idealized machine is now known as a Turing machine. (All modern computers have capabilities equivalent to some of the universal Turing machines.) During World War II Turing successfully applied his logical talent to the real and urgent problem of breaking the Nazi intelligence code, a feat that played a crucial role in the Allied victory.

Prior to World War II von Neumann was aware of Turing's work on computing machines and realized how useful such machines would be for investigating nonlinear problems in mathematical physics, in particular, the fascinating problem of turbulence. Numerical calculations might, for example, elucidate the mysterious role of the Reynolds number in turbulent phenomena. (The Reynolds number gives roughly the ratio of the inertial forces to the viscous forces. A flow that is regular becomes turbulent when this number is about 2000.) He was convinced that the best mathematics proceeds from empirical science and that numerical calculation on electronic computers might provide a new kind of empirical data on the properties of nonlinear equations. Stan Ulam suggests that the final impetus for von Neumann to work energetically on computer methods and design came from wartime Los Alamos, where it became obvious that analytical work alone was often not sufficient to provide even qualitative answers about the behavior of an atomic bomb. The best way to construct a computing machine thus presented a practical as well as a theoretical problem.

Starting in 1944 von Neumann formulated methods of translating a set of mathematical procedures into a language of instructions for a computing machine. Before von Neumann's work on the logical design of computers, the few existing electronic machines had to be rewired for each new problem. Von Neumann developed the idea of a fixed "flow diagram" and a stored "code," or program, that would enable a machine with a fixed set of connections to solve a great variety of problems.

Von Neumann was also interested, as was Turing, in discovering the logical elements

HISTORICAL PERSPECTIVE

and organization required to perform some of the more general types of functions that human beings and other life forms carry out and in trying to construct, at least at an abstract level, machines that contained such capabilities. But whereas Turing was primarily interested in developing "intelligent" automata that would imitate the thinking and decision-making abilities of the human brain, von Neumann focused on the broader problem of developing a general theory of complicated automata, a theory that would encompass both natural automata (such as the human nervous system and living organisms) and artificial automata (such as digital computers and communication networks).

What is meant by the term "complicated"? As von Neumann put it, it is not a question of how complicated an object is but rather of how involved or difficult its purposive operations are. In a series of lectures delivered at the University of Illinois in 1949, von Neumann explored ideas about what constitutes complexity and what kind of a theory might be needed to describe complicated automata. He suggested that a new theory of information would be needed for such systems, one that would bear a resemblance to both formal logic and thermodynamics. It was at these lectures that he explained the logical machinery necessary to construct an artificial automaton that could carry out one very specific complicated function, namely, self-reproduction. Such an automaton was also logically capable of constructing automata more complex than itself. Von Neumann actually began constructing several models of self-reproducing automata. Based on an inspired suggestion by Ulam, one of these models was in the form of a "cellular" automaton (see the preceding article in this issue by Stephen Wolfram for the definition of a cellular automaton).

From the Illinois lectures it is clear that von Neumann was struggling to arrive at a correct definition of complexity. Although his thoughts were still admittedly vague, they do seem, at least in some respects, related to the present efforts of Wolfram to find universal features of cellular automaton behavior and from these to develop new laws, analogous to those of thermodynamics, to describe self-organizing systems.

Von Neumann suggested that a theory of information appropriate to automata would build on and go beyond the results of Turing, Gödel, Szilard, and Shannon.

Turing had shown the limits of what can be done with certain types of information—namely, anything that can be described in rigorously logical terms can be done by an automaton, and, conversely, anything that can be done by an automaton can be described in logical terms. Turing constructed, on paper, a universal automaton that could perform anything that any other automaton could do. It consisted of a finite automaton, one that exists in a finite number of states, plus an indefinitely extendible tape containing instructions. "The importance of Turing's research is just this:" said von Neumann, "that if you construct an automaton right, then any additional requirements about the automaton can be handled by sufficiently elaborate instructions. This is true only if [the automaton] is sufficiently complicated, if it reaches a certain minimum level of complexity" (John von Neumann, *Theory of Self-Reproducing Automata*, edited and completed by Arthur W. Burks, University of Illinois Press, 1966, p. 50).

Turing also proved that there are some things an automaton cannot do. For example, "You cannot construct an automaton which can predict in how many steps another automaton which can solve a certain problem will actually solve it.... In other words, you can build an organ which can do anything that can be done, but you cannot build an organ which tells you whether it can be done" (*ibid.*, p. 51). This result of Turing's is connected with Gödel's work on the hierarchy of types in formal logic. Von Neumann related this result to his notion of complexity. He suggested that for objects of low complexity, it is easier to predict their properties than to build them, but for objects of high complexity, the opposite is true.

Von Neumann stated that the new theory of information should include not only the strict and rigorous considerations of formal logic but also statistical considerations. The reason one needs statistical considerations is to include the possibility of failure. The actual structure of both manmade and artificial automata is dictated by the need to achieve a state in which a majority of all failures will not be lethal. To include failure, one must develop a probabilistic system of logic. Von Neumann felt that the theory of entropy and information in thermodynamics and Shannon's information theory would be relevant.

Szilard had shown in 1929 that entropy in a physical system measures the lack of information; it gives the total amount of missing information on the microscopic structure of the system. Entropy defined as a physical quantity measures the degree of degradation suffered by any form of energy. "There are strong indications that information is similar to entropy and that the degenerative processes of entropy are paralleled by degenerative processes in the processing of information" (*ibid.*, p. 62).

Shannon's work focused on the problem of transmitting information. He had developed a quantitative theory of measuring the capacity of a communication channel, a theory that included the role of redundancy. Redundancy makes it possible to correct errors and "is the only thing which makes it possible to write a text which is longer than, say, ten pages. In other words, a language which has maximum compression would actually be completely unsuited to conveying information beyond a certain degree of complexity, because you could never find out whether a text is right or wrong" (*ibid.*, p. 60).

Von Neumann emphasized the ability of living organisms to operate across errors. Such a system "is sufficiently flexible and well organized that as soon as an error shows up in any one part of it, the system automatically senses whether this error matters or not. If it doesn't matter, the system continues to operate without paying any attention to it. If the error seems to be important, the system blocks that region out, by-passes it and proceeds along other channels. The system then analyzes the region separately at leisure and corrects what goes on there, and if correction is impossible the system just blocks the region off and bypasses it forever. . . .

"To apply the philosophy underlying natural automata to artificial automata we must understand complicated mechanisms better than we do, we must have elaborate statistics about what goes wrong, and we must have much more perfect statistical information about the milieu in which a mechanism lives than we now have. An automaton cannot be separated from the milieu to which it responds" (*ibid.*, pp. 71-72).

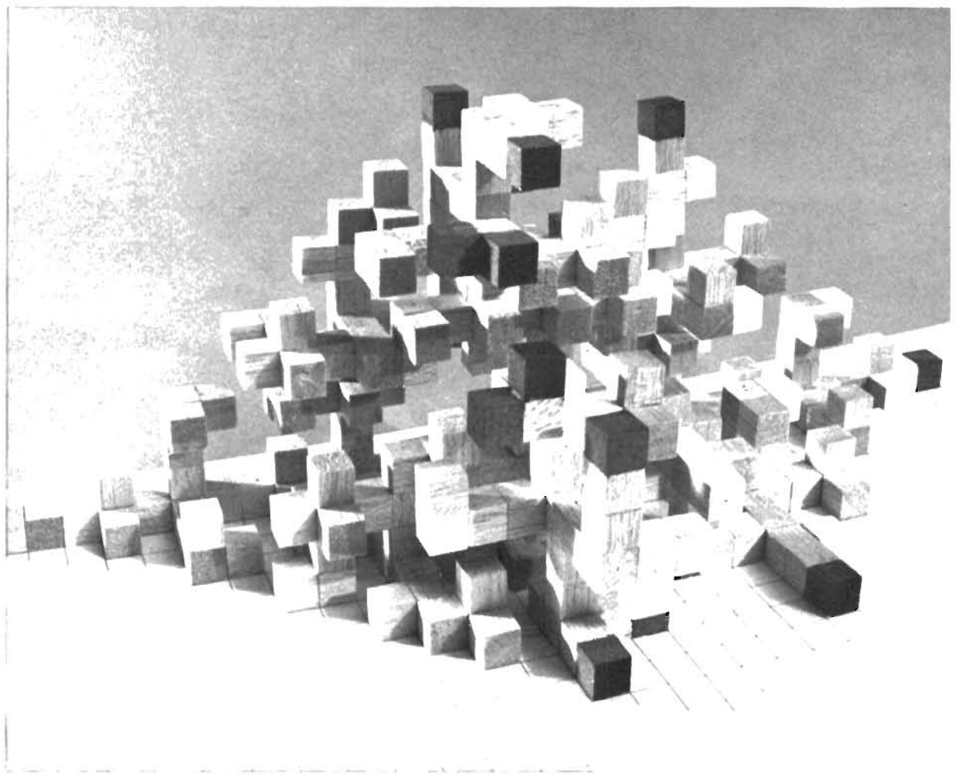
From artificial automata "one gets a very strong impression that complication, or productive potentiality in an organization, is degenerative, that an organization which synthesizes something is necessarily more complicated, of a higher order, than the organization it synthesizes" (*ibid.*, p. 79).

But life defeats degeneracy. Although the complicated aggregation of many elementary parts necessary to form a living organism is thermodynamically highly improbable, once such a peculiar accident occurs, the rules of probability do not apply because the organism can reproduce itself provided the milieu is reasonable—and a reasonable milieu is thermodynamically much less improbable. Thus probability leaves a loophole that is pierced by self-reproduction.

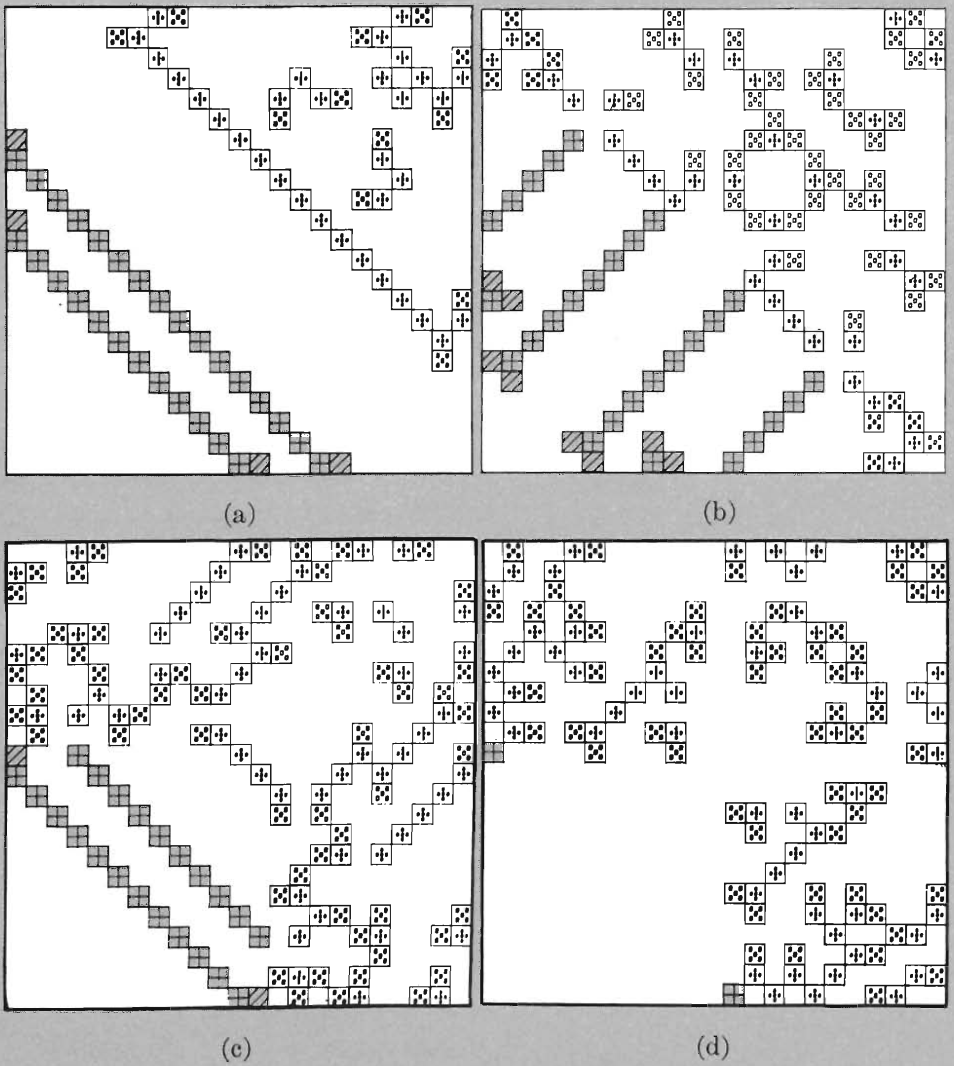
Is it possible for an artificial automaton to reproduce itself? Further, is it possible for a machine to produce something that is more complicated than itself in the sense that the offspring can perform more difficult and involved tasks than the progenitor? These questions arise from looking at natural automata. In what sense can a gene contain a description of the human being that will come from it? How can an organism at a low level in the phylogenetic order develop into a higher level organism?

From his comparison of natural and artificial automata, von Neumann suggested that complexity has one decisive property, namely, a critical size below which the process of synthesis is degenerative and above which the process is explosive in the sense that an automaton can produce others that are more complex and of higher potentiality than itself. However, to get beyond the realm of vague statements and develop a correct formulation of complexity, he felt it was necessary to construct examples that exhibit the "critical and paradoxical properties of complication" (*ibid.*, p. 80).

To this end he set out to construct, in principle, self-reproducing automata, automata "which can have outputs something like themselves" (*ibid.*, p. 75). (All artificial automata discussed up to that point, such as Turing machines, computing machines, and


A three-dimensional object grown from a single cube to the thirtieth generation (dark cubes). The model shows only one octant of the three-dimensional structure. This figure and the two others illustrating this article are from R. G. Schrandt and S. M. Ulam, "On Recursively Defined Geometrical Objects and Patterns of Growth," Los Alamos Scientific Laboratory report LA-3762, November 1967 and are also reprinted in Arthur W. Burks, editor, Essays on Cellular Automata, ***University of Illinois Press, 1970.***

HISTORICAL PERSPECTIVE



A "contest" between two patterns, one of lines within squares (shaded) and one of dots within squares, growing in a 23 by 23 checkerboard. Both patterns grow by a recursive rule stating that the newest generation (represented by diagonal lines or by dots in an* x *shape) may occupy a square if that square is orthogonally contiguous to one and only one square occupied by the immediately preceding generation (represented by perpendicularly bisecting lines or by dots in a + shape). In addition, no piece of either pattern may survive more than two generations. Initially, the line pattern occupied only the lower left corner square, and the dot pattern occupied only the square immediately to the left of the upper right corner square. (a) At generation 16 the two patterns are still separate. (b) At generation 25 the two patterns engage. (c) At 32 generations the dot pattern has penetrated enemy territory. (d) At 33 generations the dot pattern has won the contest.

the network of abstract neurons discussed by McCulloch and Pitts ("A Logical Calculus of the Ideas Immanent in Nervous Activity," *Bulletin of Mathematical Biophysics*, 1943), had inputs and outputs of completely different media than the automata themselves.)

"There is no question of producing matter out of nothing. Rather, one imagines automata which can modify objects similar to themselves, or effect syntheses by picking up parts and putting them together, or take synthesized entities apart" (*ibid.*, p. 75).

Von Neumann drew up a list of unambiguously defined parts for the kinematic model of a self-reproducing automaton. Although this model ignored mechanical and chemical questions of force and energy, it did involve problems of movement, contact, positioning, fusing, and cutting of elements.

Von Neumann changed his initial approach after extensive discussions with Ulam. Ulam suggested that the proof of existence and construction of a self-reproducing automaton might be done in a simpler, neater way that retained the logical and combinatorial aspects of the problem but eliminated complicated aspects of geometry and motion. Ulam's idea was to construct the automaton in an indefinitely large space composed of cells. In two dimensions such a cellular structure is equivalent to an infinite checkerboard. The elements of the automaton are a set of allowable states for each cell, including an empty, or quiescent, state, and a transition rule for transforming one state into another. The rule defines the state of a cell at time interval t+1 in terms of its own state and the states of certain neighboring cells at time interval t. Motion is replaced by transmitting information from cell to cell; that is, the transition rule can change a quiescent cell into an active cell.

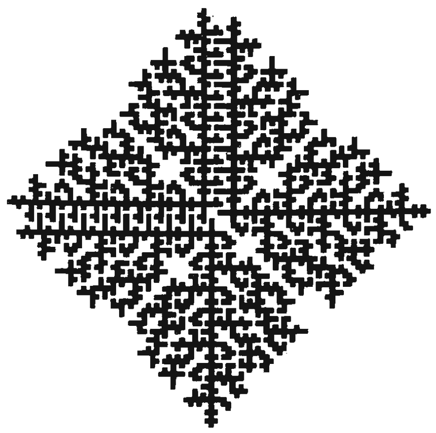
Von Neumann's universal self-reproducing cellular automaton, begun in 1952, was a rather baroque construction in which each cell had twenty-nine allowable states and a neighborhood consisting of the four cells orthogonal to it. Influenced by the work of McCulloch and Pitts, von Neumann used a physiological simile of idealized neurons to help define these states. The states and transition rules among them were designed to perform both logical and growth operations. He recognized, of course, that his construction might not be the minimal or optimal one, and it was later shown by Edwin Roger Banks that a universal self-reproducing automaton was possible with only four allowed states per cell.

The logical trick employed to make the automaton universal was to make it capable of reading any axiomatic description of any other automaton, including itself, and to include its own axiomatic description in its memory. This trick was close to that used by Turing in his universal computing machine. The basic organs of the automaton included a tape unit that could store information on and read from an indefinitely extendible linear array of cells, or tape, and a constructing unit containing a finite control unit and an indefinitely long constructing arm that could construct any automaton whose description was stored in the tape unit. Realization of the 29-state self-reproducing cellular automaton required some 200,000 cells.

Von Neumann died in 1957 and did not complete this construction (it was completed by Arthur Burks). Neither did he complete his plans for two other models of self-reproducing automata. In one, based on the 29-state cellular automaton, the basic element was to be neuron-like and have fatigue mechanisms as well as a threshold for excitation. The other was to be a continuous model of self-reproduction described by a system of nonlinear partial differential equations of the type that govern diffusion in a fluid. Von Neumann thus hoped to proceed from the discrete to the continuous. He was inspired by the abilities of natural automata and emphasized that the nervous system was not purely digital but was a mixed analog-digital system.

Much effort since von Neumann's time has gone into investigating the simulation capabilities of cellular automata. Can one define appropriate sets of states and transition rules to simulate natural phenomena? Ulam was among the first to use cellular automata in this way. He investigated growth patterns of simple finite systems, simple in that each cell had only two states and obeyed some simple transition rule. Even very simple growth rules may yield highly complex patterns, both periodic and aperiodic. "The main feature of cellular automata," Ulam points out, "is that simple recipes repeated many times may lead to very complicated behavior. Information analysts might look at some final pattern and infer that it contains a large amount of information, when in fact the pattern is generated by a very simple process. Perhaps the behavior of an animal or even ourselves could be reduced to two or three pages of simple rules applied in turn many times!" (private conversation, October 1983). Ulam's study of the growth patterns of cellular automata had as one of its aims "to throw a sidelight on the question of how much 'information' is necessary to describe the seemingly enormously elaborate structures of living objects" (*ibid.*). His work with Holladay and with Schrandt on an electronic computing machine at Los Alamos in 1967 produced a great number of such patterns. Properties of their morphology were surveyed in both space and time. Ulam and Schrandt experimented with "contests" in which two starting configurations were allowed to grow until they collided. Then a fight would ensue, and sometimes one configuration would annihilate the other. They also explored three-dimensional automata.

Another early investigator of cellular automata was Ed Fredkin. Around 1960 he began to explore the possibility that all physical phenomena down to the quantum mechanical level could be simulated by cellular automata. Perhaps the physical world is a discrete space-time lattice of information bits that evolve according to simple rules. In other words, perhaps the universe is one enormous cellular automaton.


A pattern grown according to a recursive rule from three noncontiguous squares at the vertices of an approximately equilateral triangle. A square of the next generation is formed if (a) it is contiguous to one and only one square of the current generation, and (b) it touches no other previously occupied square except if the square should be its "grandparent." In addition, of this set of prospective squares of the (n+1)th generation satisfying condition (b), all squares that would touch each other are eliminated. However, squares that have the same parent are allowed to touch.

There have been many other workers in this field. Several important mathematical results on cellular automata were obtained by Moore and Holland (University of Michigan) in the 1960s. The "Game of Life," an example of a two-dimensional cellular automaton with very complex behavior, was invented by Conway (Cambridge University) around 1970 and extensively investigated for several years thereafter.

HISTORICAL PERSPECTIVE

Cellular automata have been used in biological studies (sometimes under the names of "tesselation automata" or "homogeneous structures") to model several aspects of the growth and behavior of organisms. They have been analyzed as parallel-processing computers (often under the name of "iterative arrays"). They have also been applied to problems in number theory under the name "stunted trees" and have been considered in ergodic theory, as endomorphisms of the "dynamical" shift system.

A workshop on cellular automata at Los Alamos in March 1983 was attended by researchers from many different fields. The proceedings of this workshop will be published in the journal *Physica D* and will also be issued as a book by North-Holland Publishing Co.

In all this effort the work of Stephen Wolfram most closely approaches von Neumann's dream of abstracting from examples of complicated automata new concepts relevant to information theory and analogous to the concepts of thermodynamics. Wolfram has made a systematic study of one-dimensional cellular automata and has identified four general classes of behavior, as described in the preceding article.

Three of these classes exhibit behavior analogous to the limit points, limit cycles, and strange attractors found in studies of nonlinear ordinary differential equations and transformation iterations. Such equations characterize dissipative systems, systems in which structure may arise spontaneously even from a disordered initial state. Fluids and living organisms are examples of such systems. (Non-dissipative systems, in contrast, tend toward disordered states of maximal entropy and are described by the laws of thermodynamics.) The fourth class mimics the behavior of universal Turing machines. Wolfram speculates that his identification of universal classes of behavior in cellular automata may represent a first step in the formulation of general laws for complex self-organizing systems. He says that what he is looking for is a new concept—maybe it will be complexity or maybe something else—that like entropy will be always increasing (or decreasing) in such a system and will be manifest in both the microscopic laws governing evolution of the system and in its macroscopic behavior. It may be closest to what von Neumann had in mind as he sought a correct definition of complexity. We can never know. We can only wish Wolfram luck in finding it. ■

Acknowledgment

I wish to thank Arthur W. Burks for permission to reprint quotations from *Theory of Self-Reproducing Automata*. We are indebted to him for editing and completing von Neumann's manuscripts in a manner that retains the patterns of thought of a great mind.

Further Reading

John von Neumann. *Theory of Self-Reproducing Automata*. Edited and completed by Arthur W. Burks. Urbana: University of Illinois Press, 1966. Part I is an edited version of the lectures delivered at the University of Illinois. Part II is von Neumann's manuscript describing the construction of his 29-state self-reproducing automaton.

Arthur W. Burks, editor. *Essays on Cellular Automata*. Urbana: University of Illinois Press, 1970. This volume contains early papers on cellular automata including those of Ulam and his coworkers.

Andrew Hodges, "Alan Turing: Mathematician and Computer Builder." *New Scientist*, 15 September 1983, pp. 789-792. This contains a wonderful illustration, "A Turing Machine in Action."

Martin Gardner. "On Cellular Automata, Self-Reproduction, the Garden of Eden, and the Game 'Life.'" *Scientific American*, October 1971.

The following publications deal with complexity *per se*:

W. A. Beyer, M. L. Stein, and S. M. Ulam. "The Notion of Complexity." Los Alamos Scientific Laboratory report LA-4822, December 1971.

S. Winograd. *Arithmetic Complexity of Computations*. Philadelphia: Society of Industrial and Applied Mathematics, 1980.

Gloria Sharp

There is a gene cluster common to algae and man.
It codes for proteins that defend against cadmium poisoning.
It contains a switch that can be used to regulate other genes.
The cluster is the metallothionein locus, a model system for the study of

Gene Expression

by Carl E. Hildebrand, Brian D. Crawford, Ronald A. Walters, and M. Duane Enger
in collaboration with Roger Eckhardt

The massive tangle of DNA making up our chromosomes holds the plans, or genes, for tens of thousands of proteins. But if all these proteins were produced at the same time, the cell would be a madhouse out of control. What causes a particular gene at a given moment to express itself by directing the synthesis of the protein it encodes? What causes specific cells containing identical sets of genes to specialize—one to become a nerve cell, another a blood cell? How are specific proteins enlisted in defense of threats ranging from heavy-metal poisoning to viral infections?

We must reach into this tangle of DNA if we are to study gene expression. But how can we locate the site of a particular gene along a strand of DNA, then explore that site for both the blueprint of the protein and the switches that regulate gene expression? Two conditions must be satisfied if such a study is to be manageable: We need genes that can be switched on and off easily, and we need specific molecular probes that mark only those parts of the DNA strand where the gene and its switches are located.

Scientists at Los Alamos have been studying such a gene site—the metallothionein locus—and have made the necessary molecular probes for that site. The result is a model system for gene expression that is revealing a great deal about the various levels at which gene expression can be regulated. And even more exciting, scientists, both at Los Alamos and elsewhere, are using a switch from this site to regulate other genes. Thus, we have gained a powerful tool for manipulating specific gene expression in the laboratory and in the animal.

Our work at the Laboratory grew out of research in trace-metal toxicology. It was known from nutritional studies that certain metalloproteins played important roles in the cellular metabolism of essential trace metals. This work was buttressed by toxicological research on trace metals—of special concern because of industrial exposure to metals and growing problems with metal pollution of the environment. For example, the occurrence in Japan of the tragedy of Minamata and Itai-itai ("ouch-ouch") disease stimulated increased research on the molecular mechanisms by which the body detoxifies metals such as mercury and cadmium.

At Los Alamos we concentrated on the metallothioneins, which consist of at least two distinct proteins (MT I and II) with two corresponding genes defining the metallothionein locus. These similar, low-molecular-weight proteins have a strong affinity for several trace-metal ions and play important roles in the homeostatic regulation of the essential trace metals copper and zinc and in the control of zinc's toxic counterpart, cadmium. Cadmium is ubiquitous in the earth's crust and is transferred normally to animals and man through the food chain. Thus, even in uncontaminated environments, significant amounts of Cd^{2+} can gradually accumulate in the body where it is dealt with by binding to the metallothioneins and being stored, primarily in the liver and kidneys.

Our goal was to move from the cellular response to metal ions down to the underlying mechanisms that generate this response on a molecular level. We were thus faced with understanding the regulation of gene expression at a particular site in the specific context of heavy metal poisoning. However, our studies revealed the metallothionein locus to be an exciting system for exploring gene expression in general.

For one thing, the locus offered a gene system that could be easily switched on and off. There are two general modes of gene expression: the *constitutive* mode in which the gene is locked on and synthesizes its protein continuously, and the *inducible* mode in which the gene can switch on and off according to specific conditions in the cell. Metallothionein synthesis can be induced in a variety of cells grown in culture by adding appropriate concentrations of Cu^{2+}, Zn^{2+}, or Cd^{2+} ions. Such straightforward control is important.

We also learned that we could develop cell lines that overproduced metallothionein. These were variant cell lines, more resistant to cadmium, that could provide us with a valuable source of the molecules involved in gene expression such as metallothionein genes, gene products, and regulatory factors. Of course, how cells accomplished this overproduction of metallothionein and the mechanisms underlying genetic expression were fascinating problems in their own right.

We could reach into the tangle of DNA with strands of radioactively tagged DNA that would selectively attach themselves to portions of the metallothionein locus. The advent of cloning of DNA sequences through recombinant DNA technology had made it possible to isolate pieces of particular genes. These cloned genes could be tagged to facilitate detection and be used as probes that hybridize, or form a double strand, at the site of the gene. The radioactivity of the probe would serve as a marker for the gene, regardless of whether that gene remained in its intact form in the chromosome or was a fragment sliced out by special enzymes.

In the Genetics Group of the Laboratory's Life Sciences Division, we prepared such probes by isolating functional pieces of the metallothionein locus. The probes enabled a variety of experiments, eventually revealing two mechanisms for metallothionein gene expression, the order of the DNA coding units at the locus, and the location of the gene site in its chromosome.

Another aspect that made the metallothionein locus an exciting model system for the study of gene expression was its universality. Metallothioneins have been found in species ranging from blue-green algae to man. Research in our laboratory and others has now shown that the DNA regions that code for the two metallothioneins have been highly conserved from species to species. We estimate that the presence of paired metallothionein genes, MT I and II, has been maintained for many tens or hundreds of millions of years in species that otherwise diverged from each other. The conservation of this inducible system highlights its crucial cellular role.

Finally, it was realized that once the switch regulating metallothionein synthesis was located, it could be joined by recombinant DNA methods to other, unrelated genes, then reintroduced into cells by gene-transfer techniques. It was hoped the expression of these recombinant genes could then be induced by exposing the cells to Zn^{2+} or Cd^{2+}. We would thus take advantage of the clearly defined switching properties of the metallothionein gene to manipulate the expression of other, perhaps normally constitutive, genes. Already, despite an incomplete understanding of how the regulatory switch of the metallothionein locus operates, such experiments have been performed successfully. Scientists at the University of Washington and the University of Pennsylvania have used the metallothionein switch to regulate the gene for growth hormone in mice. Now, in collaboration with several laboratories, we are using the switch to study the regulation of oncogenes—genes whose expression is implicated in human carcinogenesis.

Although research has now revealed much about gene expression at the metallothionein locus, many questions remain that are only partially answered. In some ways the work is just beginning. But a model system for gene expression has been defined, appropriate tools have been made, and what began as the study of cellular defense against metal poisoning has now grown into a multi-pronged assault on both the mechanisms and the effects of gene expression.

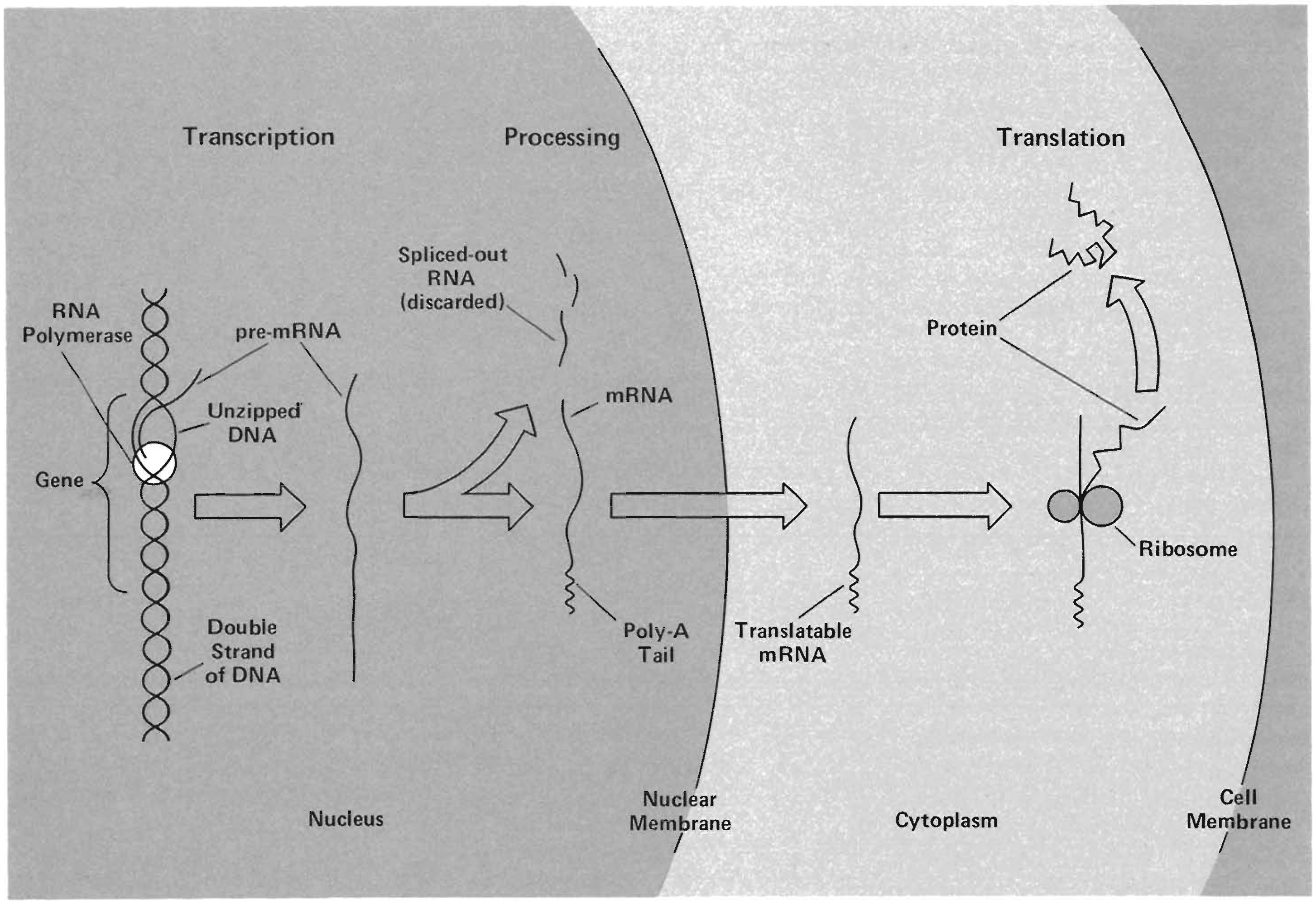


Fig. 1. Gene expression involves three major steps in cells with a nucleus. Transcription: ***One strand of the DNA at the gene acts as a template for the formation of messenger RNA (mRNA). The enzyme RNA polymerase moves along the double helix, unzipping part of the DNA and synthesizing a complementary RNA strand.*** Processing: ***Many of the transcribed gene regions not coding for protein are spliced out, and a polyadenylate tail is added. This tail distinguishes most mRNA from other RNA in the cell. Processed mRNA then passes through the nucleus into the cytoplasm.*** Translation: ***At complex particles called ribosomes, the mRNA is drawn through, the triplet code is read, and the corresponding set of amino acids are linked together into the protein.***

From Gene to Protein

A gene holds its plan as a particular sequence of monomer units, called nucleotides, linked together in long, polymeric strands of DNA. Each nucleotide provides one of four possible base moieties, projecting sideways from the phosphodiester backbone of the molecule. Each base pairs with its complementary base on the nucleotide of a second strand, thus forming the double helix of DNA (see the figure and caption on page 54). The complementary nature of the two strands is the basis not only of the hybridization techniques used in identifying genes but of the DNA-directed synthesis of RNA that must occur before the information in the DNA can be used to synthesize a protein.

The key steps in the process of gene expression are *transcription* of a primary strand of DNA into a complementary strand of RNA, and *translation* of the RNA into protein. In cells without a nucleus, these steps are rather simple and are not compartmentalized. In eukaryotes (cells with a nucleus), transcription takes place in the nucleus, whereas translation takes place outside the nucleus in the cytoplasm at sites for protein synthesis called ribosomes (Fig. 1). As a result, the process of gene expression is quite complicated, and regulation of this process may occur at several levels.

The DNA in eukaryotes holds the genetic blueprint, yet only certain segments along the strands encode protein sequences. The noncoding regions mark, among other things, the start and stop positions for transcription and probably play a role in turning some genes on and off. During transcription an enzyme called RNA polymerase unzips part of the DNA double strand and moves down through the gene. A growing strand of RNA complementary to the original gene is copied using the one-to-one complementary base-pairing rules.

At this stage the synthesized RNA mirrors both the protein coding regions of the gene and noncoding regions that flank or interrupt the coding sequences. Such RNA is *processed* before it passes through the nuclear membrane of the cell. During processing many of the coding regions are spliced together, and the noncoding regions are discarded. One other modification that occurs during processing is the addition of a poly-A tail, consisting of about 200 consecutive adenylate units. (Adenylate is one of the four nucleotides that make up the RNA or DNA polymeric chains.) We now have messenger RNA (mRNA) ready for protein synthesis.

The mRNA that passes into the cytoplasm is translatable, in that it may move to the ribosomes and have its genetic message translated into protein. In this step each group of three consecutive nucleotides in the protein-encoding regions of the mRNA delineates the introduction of a specific amino acid into the protein.

The Cells

Because of the hierarchy of complex processes governing protein synthesis, we could hope to begin understanding the regulation of gene expression in eukaryotes only by using purified components and an easily controlled cellular system. Cultured mammalian cells offered several advantages in this regard. Various stimulatory or inhibitory agents, such as metal ions and metabolic inhibitors, could be added or withdrawn rapidly. Also, the number and viability of the cells could be watched closely and interesting variants or mutants removed to be grown as a new cloned subline. And, of course, complicating interactions that occur between tissues *in vivo* would be eliminated with cultured cells.

Our initial studies used Chinese hamster ovary cells, a cell line developed by Ted Puck at the University of Colorado in the early days of cell culture. The strategy adopted to derive cadmium-resistant cells was long-term growth of the hamster ovary cells in marginally toxic levels of Cd^{2+} (Fig. 2). Most cells died, but survivors with an increased resistance to cadmium were cloned and developed into sublines. Clonal isolation, because it uses the progeny of a single cell, allowed us to study genetically homogeneous populations.

By repeating the procedure as shown in Fig. 2, we were able to derive a series of variant sublines that were resistant to in-

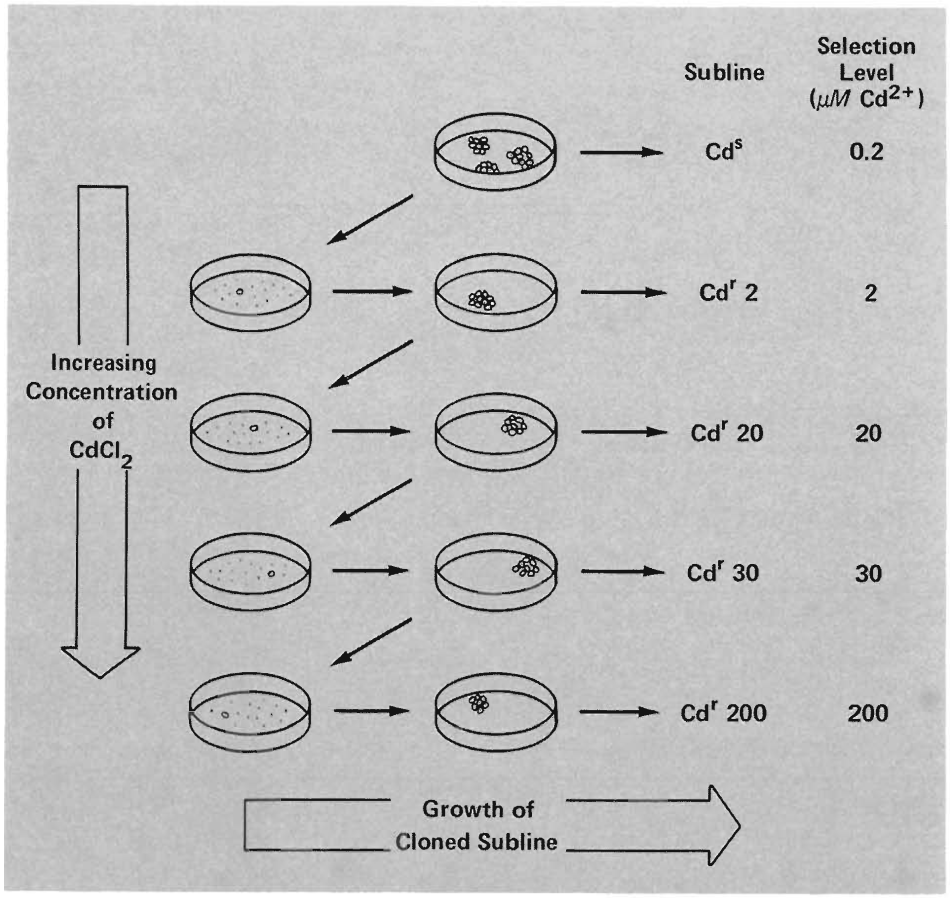


Fig. 2. Cells that are resistant to cadmium (Cd^r) are generated by exposing the original cells (Cd^s) to marginally toxic levels of the metal ion. Repeated treatment using increasing concentrations of Cd^{2+} generates increasingly resistant sublines. At each stage a variant cell that survives is cloned, becoming a genetically homogenous population of cells.

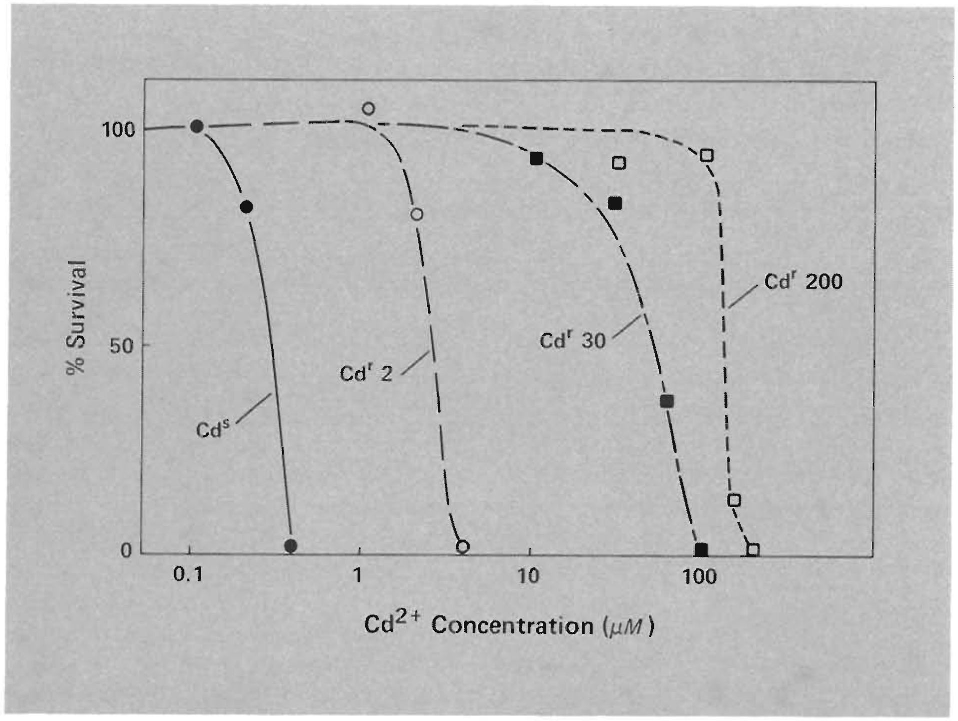
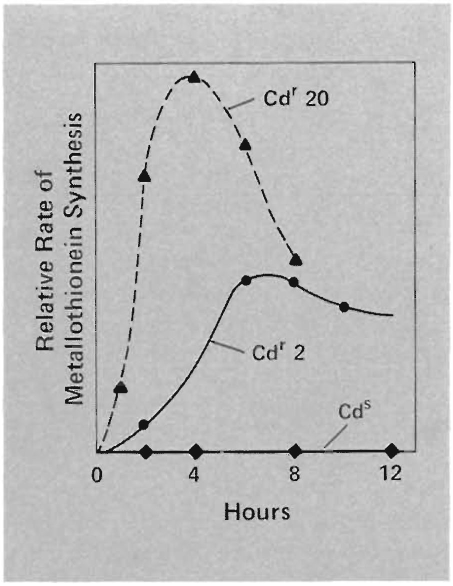


Fig. 3. The percentage of cells that survive cadmium exposure as a function of concentration for several sublines.



creasing levels of Cd^{2+} in the culture medium. Although a Cd^{2+} concentration of only 0.2 micromolar is enough to begin killing cells from the original line of hamster ovary cells, sublines were derived with Cd^{2+} resistance levels of about 2, 20, 30, and 200 micromolar. The percentage survival as a function of Cd^{2+} concentration of both the original cells and several of the sublines is shown in Fig. 3.

Fig. 4. Rate of metallothionein synthesis. Cells were exposed to the Cd^{2+} concentration that induces maximum synthesis of metallothionein for their subline, and then the rate of synthesis was measured as a function of time from exposure. The cadmium-sensitive cells (Cd^s) produced undetectable levels of the protein compared to the two resistant sublines.

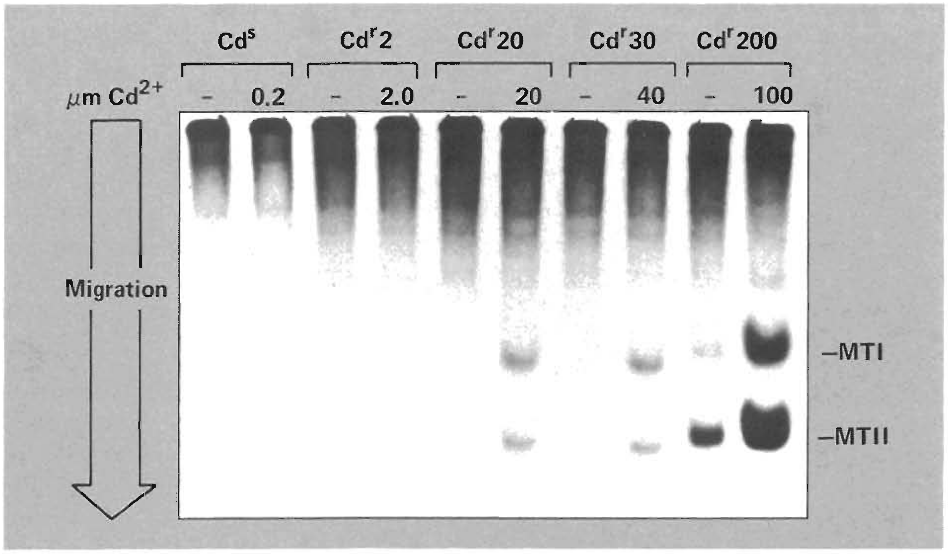


Fig. 5. Separation of the two metallothioneins, MT I and II, by electrophoresis. Cellular protein, labeled with sulfur-35, was extracted from the sublines shown at the top. The left column of each pair is for no cadmium exposure, the right is for the Cd^{2+} exposure shown. The relatively small metallothionein molecules migrate rapidly, and differences in their amino acid composition cause them to separate. Autoradiography then reveals their respective positions and relative amounts. These data show that increased synthesis of metallothionein occurs coordinately; that is, there is a proportionate increase in the activity of both genes. Also note that considerable metallothionein is synthesized in the Cd^r 200 subline, even when synthesis is not being induced by Cd^{2+} exposure.

Over the past several years these cadmium-sensitive (Cd^s) and cadmium-resistant (Cd^r) cells have provided a useful framework for a variety of studies in both the Genetics and the Toxicology groups at Los Alamos. In particular, the effect of Cd^{2+} on growth and survival, the manner in which the cells take up and compartmentalize the Cd^{2+}, and the extent of Cd^{2+} binding to metallothionein have been studied. For gene expression, of course, it was important to examine the induction of metallothionein synthesis.

Figure 4 shows what happens to the rate of metallothionein synthesis when the concentration of Cd^{2+} that induces maximum synthesis for a given subline is added to those cells. Normal Cd^s cells produce undetectable levels of the protein, whereas the Cd^r variants show a much increased rate of synthesis. Several lines of evidence from cellular, biochemical, and molecular genetic studies showed that synthesis of this protein definitely ameliorates the toxic effects of Cd^{2+}. Typically, the rapidity with which Cd^r variants begin to synthesize metallothionein and, to some extent, the relative synthesis levels correspond to their respective resistance.

The dramatic increase in metallothionein production in Cd^r cells as one progresses from Cd^s cells to the most resistant subline (Cd^r 200) is revealed in Fig. 5. Here, one of the amino acids used in the synthesis of the metallothioneins, cysteine, was labeled with the radioisotope sulfur-35. Protein extracts were taken from the cytoplasm of all sublines before and after Cd^{2+} exposure, then separated by size using a standard technique called electrophoresis. In this technique, molecules migrate through a gel under the influence of an applied electric field, the smaller molecules migrating faster. Next, we used autoradiography, in which radiation from the sulfur-35 exposes film at the locations of the spatially separated proteins in

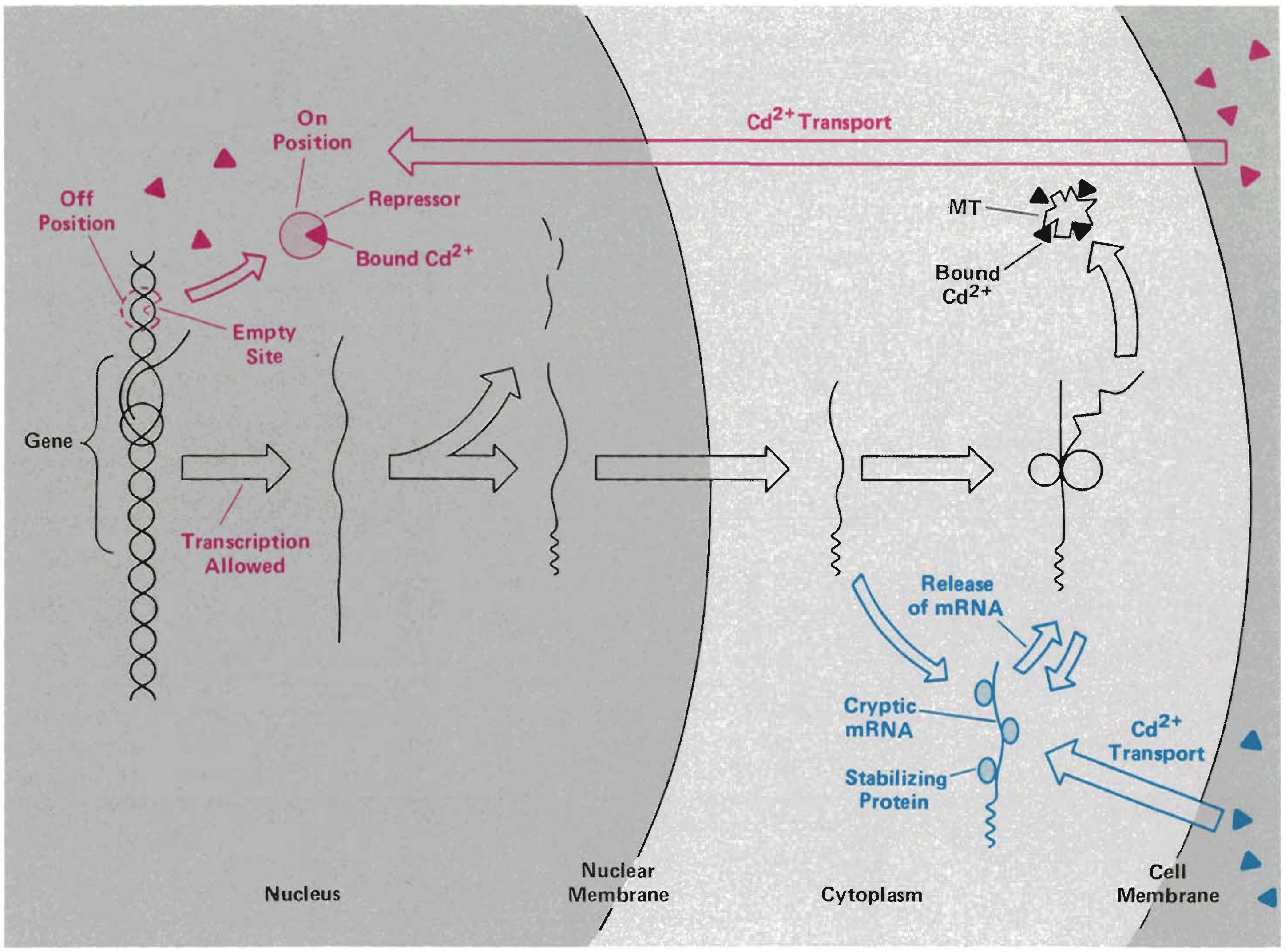


Fig. 6. Alternative control mechanisms for metallothionein synthesis: transcriptional control *(red) versus* cryptic mRNA *(blue). The basic steps of gene expression are the same in both cases. However, for transcriptional control the regulation occurs at the gene. The figure shows one possibility: a repressor binds to the DNA, preventing transcription, until Cd^{2+} binds to it, causing the repressor to release the DNA and allow transcription to take place. For cryptic mRNA control occurs in the cytoplasm. In this case, although the mRNA has already been synthesized, it is stored before translation to protein can occur. Cd^{2+} exposure leads to the release of this mRNA, followed by rapid synthesis of the metallothionein.*

the gel. The intensities of these spots are a measure of metallothionein concentrations.

We see for the Cd^r sublines that not only is more metallothionein produced for Cd^{2+} exposures but also the two metallothioneins, MT I and II, are expressed coordinately. In other words, an increase in metallothionein production reflects a proportionate increase in the activity of not just one, but both genes. Note that even when cells in the Cd^r 200 subline are grown without Cd^{2+} exposure, concentrations of metallothionein are relatively large. This reflects a high basal level of synthesis in these cells—an interesting observation that may reflect metallothionein gene regulation run awry in these cells.

Some of our data indicated that cellular accommodation of Cd^{2+} is not attributable solely to the metallothioneins. In Fig. 5 we see that the Cd^r 30 subline synthesizes about the same amount of metallothionein as does the Cd^r 20 subline, yet the former variant subline is more resistant to Cd^{2+}. Such data suggest there must be another protective mechanism distinct from the synthesis of metallothionein. We are pursuing this possibility in other research not discussed here.

Location of Control

Returning to the data of Fig. 4, the initial slopes of the kinetic curves show that the *rapidity* of the response is also greater in the more resistant variants. A resistant cell thus is able not only to synthesize more metallothionein, but it also accomplishes the synthesis more quickly.

This last observation leads naturally to the next important question: where is control of metallothionein synthesis located? For example, control might occur at the gene as the result of activation or repression of transcription. Or it might occur in the cytoplasm by the release of cryptic mRNA, that is, mRNA that has been processed and transported into the cytoplasm, but then stored in an inactive form (Fig. 6). The latter hypothesis would explain in a straightforward manner the quick mobilization of metallothionein. Evidence for such translational control exists in other genetic systems. It was obvious that we needed to extend our study at the molecular level to the link between gene and protein, that is, to messen-

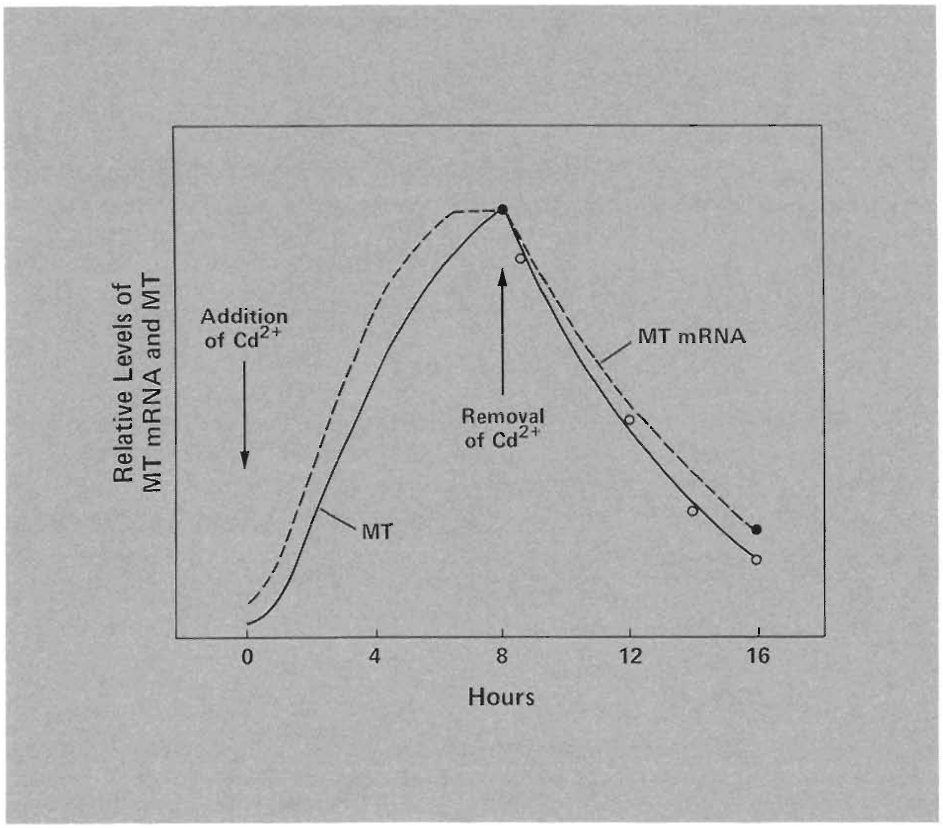


Fig. 7. A close temporal correlation exists between the cellular levels of metallothionein (MT) and its messenger RNA (MT mRNA). Both concentrations increase when Cd^{2+} is added to the cell and decrease when it is removed. These data show a half-life for MT mRNA in the cytoplasm of about two or three hours.

ger RNA coded specifically for metallothionein (MT mRNA).

Eventually, several lines of evidence were obtained in our laboratory and elsewhere that argued against a mechanism based on cryptic mRNA. First, previous reports had shown that actinomycin D, a chemical that is an inhibitor of mRNA synthesis, inhibited the induction of metallothionein in other systems. When used in Cd^r cells, it completely inhibited the synthesis of metallothionein, although general cellular protein synthesis continued at about 75 per cent of the normal rate. This result suggested that transcription of *new* MT mRNA is required for induction of metallothionein synthesis.

Unfortunately, considerable controversy surrounds the use of specific inhibitors of RNA synthesis. Demonstrating inhibition of metallothionein induction alone was insufficient to give a quantitative assessment of transcriptional control at the gene. To obtain a more precise assessment, we developed methods to measure the levels of translatable MT mRNA present in the cytoplasm. This involves a system in which total mRNA is extracted from the cell and used to direct translation of a specific protein *in vitro*, that is, in a reconstituted cell-free milieu using just those chemicals and components needed for protein synthesis. Extraction of the mRNA removes proteins that may be inhibiting translation *in vivo*. The amount of metallothionein generated is then a direct measure of the amount of translatable MT mRNA in the extract.

We compared the level of metallothionein translated *in vitro* with the level of translatable MT mRNA in the cytoplasm as a function of exposure to Cd^{2+} (Fig. 7). These data show that accumulation of MT mRNA slightly precedes (by about half an hour) metallothionein synthesis. Further, when the Cd^{2+} is removed, both metallothionein synthesis and the level of MT mRNA drop coordinately. This close temporal correlation is consistent with the hypothesis that metallothionein synthesis requires *de novo* mRNA synthesis. Our *in vitro* measurement of translatable MT mRNA would include cryptic mRNA stored in the cytoplasm, and our data show that MT mRNA accumulates in the cytoplasm only during times of exposure to Cd^{2+}. In fact, the half-life for MT mRNA in the cytoplasm estimated from these data appears to be very short (about two to three hours). This would be expected for a system that is required to respond rapidly to changing cellular levels of metal but that is regulated at the transcriptional step of gene expression. It remains possible, however, that MT mRNA is stored in the *nucleus* in an unprocessed form, then processed and exported quickly to the cytoplasm when the cell encounters Cd^{2+}. This possibility is a question we and other laboratories will continue to examine.

Gene Probes

With indirect evidence in hand that regulation was taking place at the gene, it was essential to isolate DNA sequences from the metallothionein locus in order to further define that regulation. Such isolated sequences would provide gene probes for the metallothionein locus. Our initial techniques used to generate these probes depended essentially on the hybridization, or reannealing reaction, between complementary strands of nucleic acids. We intended to generate probe strands of DNA that were specifically complementary to MT mRNA and, thus, were direct copies of MT I and MT II gene sequences.

Experiments with specific gene probes would circumvent two problems. First, our earlier measurements of metallothionein synthesis rates and MT mRNA levels had limited ranges of sensitivities. Gene probe experiments would provide more quantitative data. Second, there remained a question of cryptic MT mRNA that might be maintained in the cytoplasm in a *non*translatable state even following extraction. Nucleic acid hybridization experiments would determine cytoplasmic levels of mRNA that appeared only during exposure to Cd^{2+}. Additionally, we could investigate levels of unprocessed, *nuclear* RNA prior to or during exposure to Cd^{2+}.

We hoped these analyses would not just strengthen our hypothesis of *de novo* synthesis of MT mRNA. We hoped they

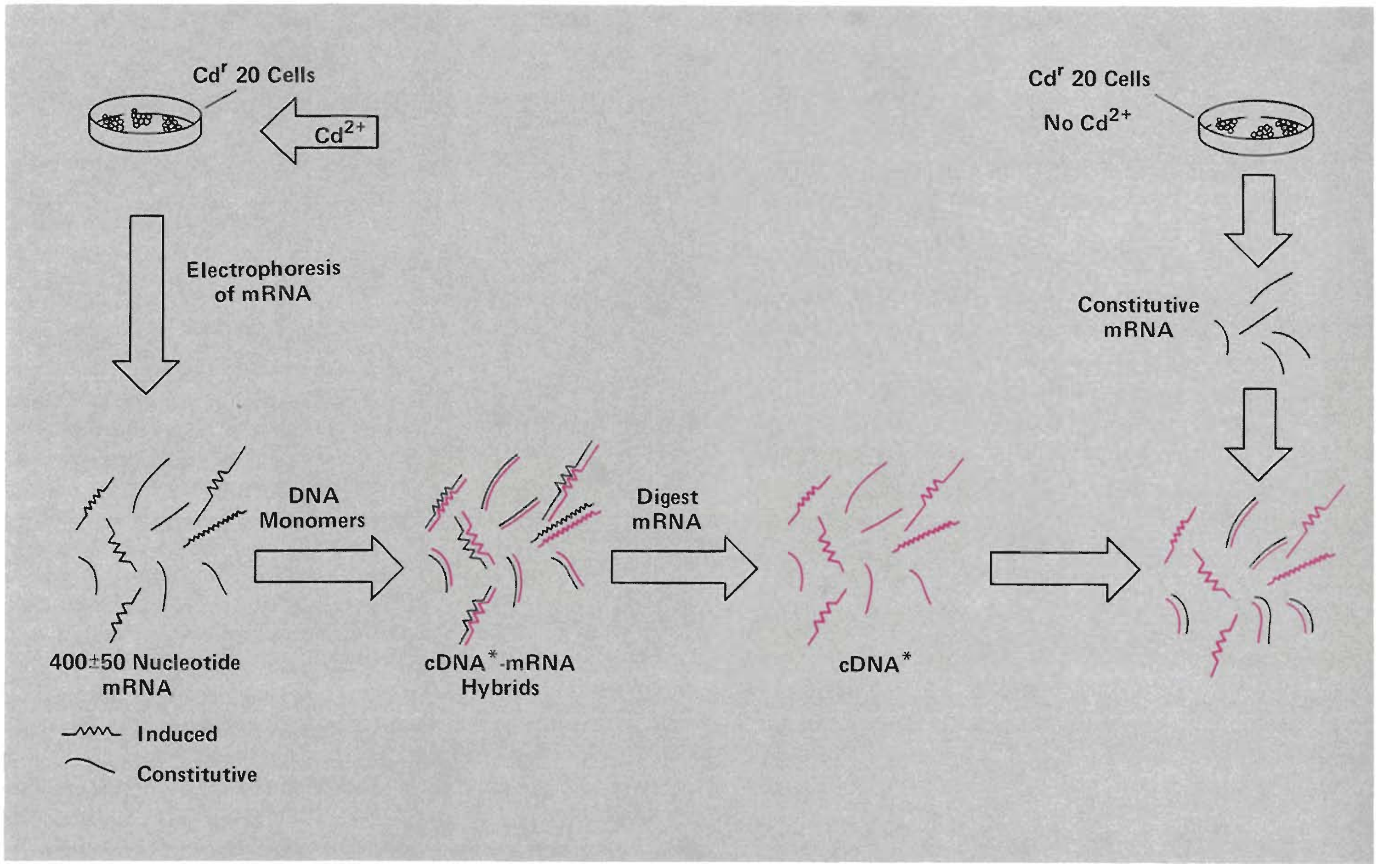


Fig. 8. Synthesis and isolation of cDNA probe. Cadmium-resistant cells exposed to Cd^{2+} (left) synthesize both induced messenger RNA and constitutive mRNA. The initial purification steps, including electrophoresis, isolate mRNA of the size (400±50 nucleotides) that includes induced mRNA coding specifically for metallothionein (MT mRNA). These mRNA strands are used as templates to construct a complementary set of DNA strands (typically labeled with tritium) called cDNA* (red). Messenger RNA isolated from cells not exposed to Cd^{2+} (center) will not include induced MT mRNA. Hybridization of such mRNA with cDNA* forms cDNA*-mRNA hybrids, the mRNA of which is expressed constitutively, and single strands*

would help answer the next question: Do the Cd^r sublines synthesize more metallothioneins because transcription occurs more frequently at a given gene, or do these cells have an increased number of operating genes?

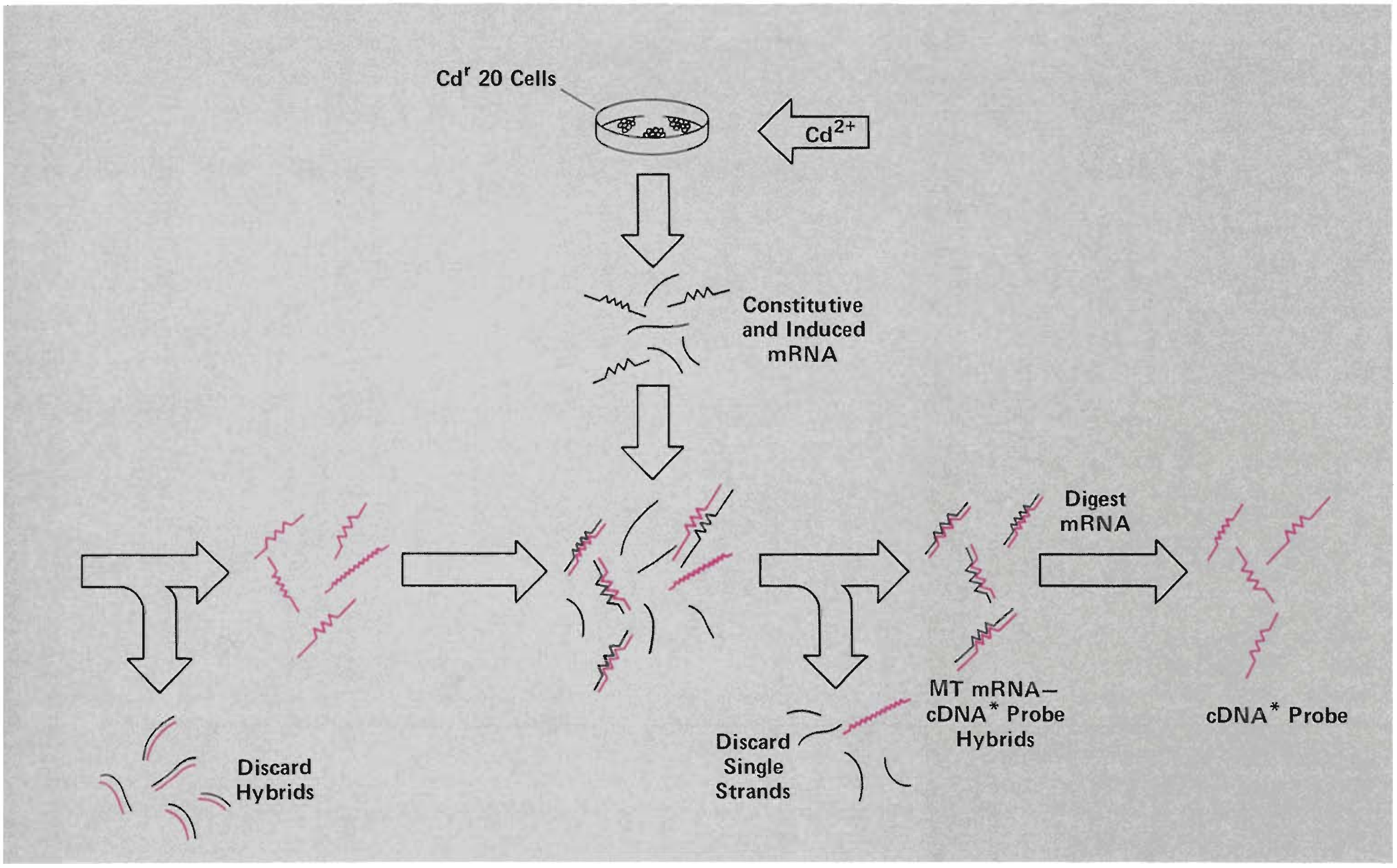
Purification. The first step was to isolate just the MT mRNA from the cell's total RNA. We started by using a standard affinity column chromatography separation technique in which the poly-A tail, attached only to mRNA, binds to the column substrate, thus separating mRNA from other RNA in the cytoplasm. (There is only about 2 per cent mRNA in the cytoplasm; most of the RNA is ribosomal and transfer RNA.)

We then used electrophoresis to separate the mRNA on the basis of size. Because the metallothioneins are small proteins (only 61 amino acids), the corresponding MT mRNA should be of relatively low molecular weight (183 nucleotides in the regions coding for metallothionein plus a similar number of nucleotides in the leading and trailing untranslated regions). Independent experiments had revealed an interesting class of mRNA extracted from Cd^r cells with only 400±50 nucleotides, and this class increased to approximately 5 per cent of the total cytoplasmic mRNA after Cd^r cells were exposed to Cd^{2+}.

The following experiment further demonstrated that this class contained the processed MT mRNA. We tagged mRNA that was being synthesized in Cd^r cells with different radioisotopes depending on whether the cells were turned on or off with respect to the synthesis of metallothionein. Specifically, during a 6-hour exposure to Cd^{2+}, we permitted cells to synthesize mRNA using uridine tagged with tritium (uridine contains uracil, the base unique to RNA). Separately, during a 6-hour period with *no* exposure to Cd^{2+}, uridine tagged with carbon-14 was used. Thus, only the mRNA population tagged with tritium included significant amounts of cadmium-induced mRNA. The two mRNA populations were mixed and separated by size by electrophoresis. There was a striking increase in the tritium to carbon-14 ratio at the position in the gel for the 400-nucleotide mRNA, demonstrating that this class included the mRNA induced by Cd^{2+} exposure.

Next, we pulled copies of the MT mRNA out of this 400-nucleotide class using a multistep scheme based on a DNA-RNA hybridization reaction (Fig. 8). In fact, at the penultimate step, copies of MT mRNA are in double-strand form, the other strand in the hybrid being our desired gene probe.

The first step was to synthesize, with an enzyme called reverse transcriptase, what we call cDNA*, that is, single-strand DNA complementary (c) to all the mRNA in the



of induced mRNA. The hybrid group is then eliminated by hydroxylapatite column chromatography that separates single-strand from double-strand molecules. Messenger RNA from cells exposed to Cd^{2+} (right) consists of the full set of mRNA, including MT mRNA. This mRNA is allowed to react with what is left in the cDNA set. This time the single strands are discarded. The remaining hybrids contain only the induced MT mRNA and strands of labeled DNA specifically complementary to it. The mRNA is digested, leaving this highly specific DNA, called cDNA* probe.*

400-nucleotide class and labeled with tritium (*) or some other radioisotope. Thus, cDNA* is actually a set of sequences. The members in the set are present in the same proportional amounts as each corresponding mRNA sequence in the 400-nucleotide class. The set contains sequences complementary to the mRNA induced by Cd^{2+} exposure as well as sequences complementary to the constitutive mRNA always present in the cell.

To separate the induced and constitutive cDNA*, we took advantage of the second-order kinetics of the DNA-RNA hybridization reaction (the bimolecular reaction rate is proportional to the concentrations of both species). We first mixed tracer cDNA* with mRNA taken from cells whose metallothionein synthesis was turned off. In this case, little or no MT mRNA was present, and the constitutive cDNA* sequences were the ones that hybridized. The reaction time was kept relatively long to help ensure complete hybridization of these unwanted sequences, then all hybrids were discarded. What was left was almost all single-strand cDNA* complementary to the mRNA that is only present when the cells are turned on, that is, complementary to the induced mRNA. We then performed the inverse step by mixing the remaining cDNA* with mRNA taken from cells in which metallothionein synthesis was turned on. Concentrations were kept low and times short so that only those abundant sequences with excellent matches to our tracer cDNA* were likely to hybridize. In this case, single-strand cDNA* was discarded, leaving only double strands of the desired MT mRNA and its DNA complement.

We had separated the mRNA induced by exposure to Cd^{2+} from the rest of the cell's RNA but now in hybrid or double-strand form. Our last step removed this mRNA, leaving us with the more important *cDNA* probe*, that is, DNA specific to MT mRNA and labeled with tritium or some other radioisotope.

Now, MT mRNA is complementary to the DNA at the metallothionein locus, and cDNA* probe is, in turn, complementary to MT mRNA. Thus, the synthetic cDNA* probe reflects the coding sequence of nucleotides on the transcribed strand of the original genetic DNA (except for those intervening, noncoding sequences deleted when the mRNA was processed). This cDNA* probe was the key to what we would learn about regulation of gene expression and about the organization of multiple genes at the metallothionein locus.

Quantitative Measurements. To begin with, the radioactivity of the cDNA* probe allowed us to quantitatively measure amounts of MT mRNA by the following technique. A sample with a known total concentration of mRNA is incubated with a low concentration of the probe. Conditions that can alter the hybridization rate, such as temperature and ionic strength of the reaction mixture, are adjusted to a given set of criteria that optimize specific hybridization of DNA and RNA. During the incubation samples are drawn at various times, the reaction quenched in the sample, and any single-strand DNA digested by an appropriate enzyme. Any cDNA* probe that hybridized with MT mRNA remains undigested and can be separated and quantitatively measured by liquid scintillation counting. The rate at which the cDNA* probe forms hybrids with MT mRNA can be used to calculate the concentration of those sequences in the total sample.

Figure 9 shows the results of an experiment using this technique to measure the concentrations of MT mRNA in extracts from different cells. Here relative time is the time of sample withdrawal corrected for anything that would have altered the hybridization reaction rate, such as total concentration of mRNA and ionic strength. These corrections allow a quantitative comparison that yields relative concentrations of MT mRNA. For example, the two extreme curves in Fig. 9 tell us that the MT mRNA concentration in cells from the Cd^r 20 subline that are turned on is greater than 10,000 times the concentration in uninduced, normal cells (Cd^s). Also, note that when the Cd^r 20 cells are turned off, their MT mRNA concentrations are only about 10 times larger than the normal cells.

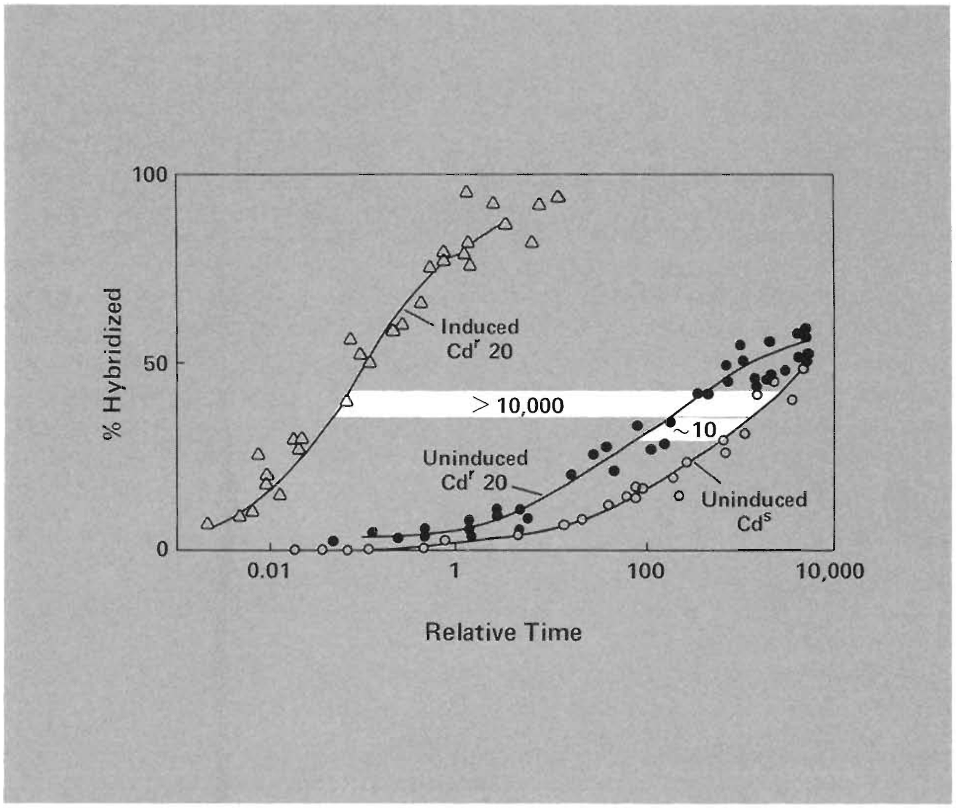


Fig. 9. The concentration of MT mRNA can be determined by the rate at which a given amount of cDNA probe hybridizes with it. Messenger RNA taken from Cd^r 20 cells exposed to Cd^{2+} hybridizes more than 10,000 times faster than mRNA taken from normal Cd^s cells not exposed to Cd^{2+} and, thus, must contain more than 10,000 times the concentration of MT mRNA. When these same Cd^r 20 cells are not exposed to Cd^{2+}, the hybridization rate and corresponding MT mRNA concentration drops to only a factor of 10 greater than for Cd^s cells. Here relative time is actually the reaction time multiplied by the sample's total concentration of mRNA and then corrected for other factors that also alter the reaction rate.*

Large-Scale Purification. To carry out further experiments, we needed larger amounts of the important cDNA* probe. Fortunately, recombinant DNA cloning can be used to generate specific DNA sequences in large amounts. In our case, we incorporated metallothionein cDNA sequences into plasmids, extrachromosomal genetic elements found in various bacteria (see the figure on page 56). Our recombinant plasmids were then placed in the *Escherichia coli* bacterium where they self-replicated, producing amplified amounts of cDNA probe sequences.

Of course, there are two metallothionein proteins (MT I and II), two corresponding types of MT mRNA, and two genes in the metallothionein locus. We succeeded in isolating separate cloned cDNA copies corresponding to the two genes and here called cDNA probe I and cDNA probe II. As mentioned earlier, except for sequences deleted during mRNA processing, the nucleotide sequence of each of the two probes should be identical to the corresponding sequence of the transcribed strand at the gene. With sufficient cloned material we were able to determine, by biochemical analyses, actual nucleotide sequences along cDNA probes I and II. By comparing our results to the known amino acid sequences in the proteins, we saw that our cloned, recombinant DNA did, indeed, represent the two major Chinese hamster metallothioneins. Each probe had a central region coding for its respective protein with adjacent sequences on either side representing part of what are called the 5′ and 3′ untranslated regions (Fig. 10). There were only small differences between the I and II sequences in the coded regions (81 per cent homology), but large differences in the 3′ untranslated regions (around 25 or 30 per cent homology, which, with four bases, is essentially random).

With these facts firmly established, we were ready to use our two probes to further analyze the mechanisms underlying gene expression for metallothionein. Labeled cDNA* probes I and II or their complements could be easily synthesized starting with the unlabeled material cloned in plasmid form. We typically used the complementary version of the labeled probes to identify the fate of various fragments of cellular DNA

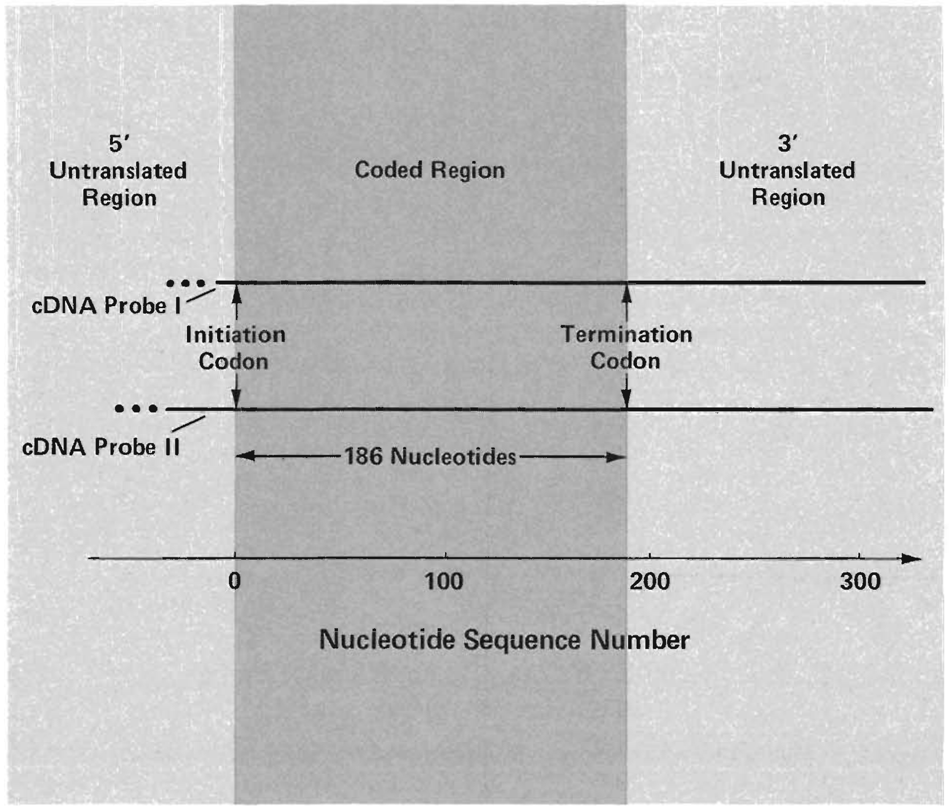


Fig. 10. Features of the two metallothionein gene probes. Recombinant DNA techniques were used to isolate DNA sequences complementary to MT mRNA for the I and II metallothionein genes. Both sequences had a central coded region of 186 nucleotides, including the nucleotide triplets, or codons, that initiate and terminate protein synthesis. Also, both had adjacent untranslated regions that do not code for synthesized protein, but which may contain sequences used to regulate gene expression. The enzyme used in the initial synthesis of these fragments became inactive before either of the 5′ untranslated regions were complete. (See "Sequencing the Genes" for the actual base sequences.)

after experimental treatment. In fact, we eventually fabricated probes, complementary to several specific locations along cDNA probes I or II, that would identify unique portions of the metallothionein locus.

The Role of DNA Methylation

As we pointed out earlier, regulation of gene expression is critical for the programmed development and cellular differentiation of any complex organism. Several mechanisms have evolved to provide such regulation. One, found in higher eukaryotes, is called DNA methylation. It involves a specific chemical modification: the covalent linkage of a methyl group at carbon 5 in cytosine, the base in one of the DNA monomers. This modification does not alter the coding capacity of the DNA because this carbon is not involved in the hydrogen-bonding interactions required for faithful DNA replication or RNA transcription.

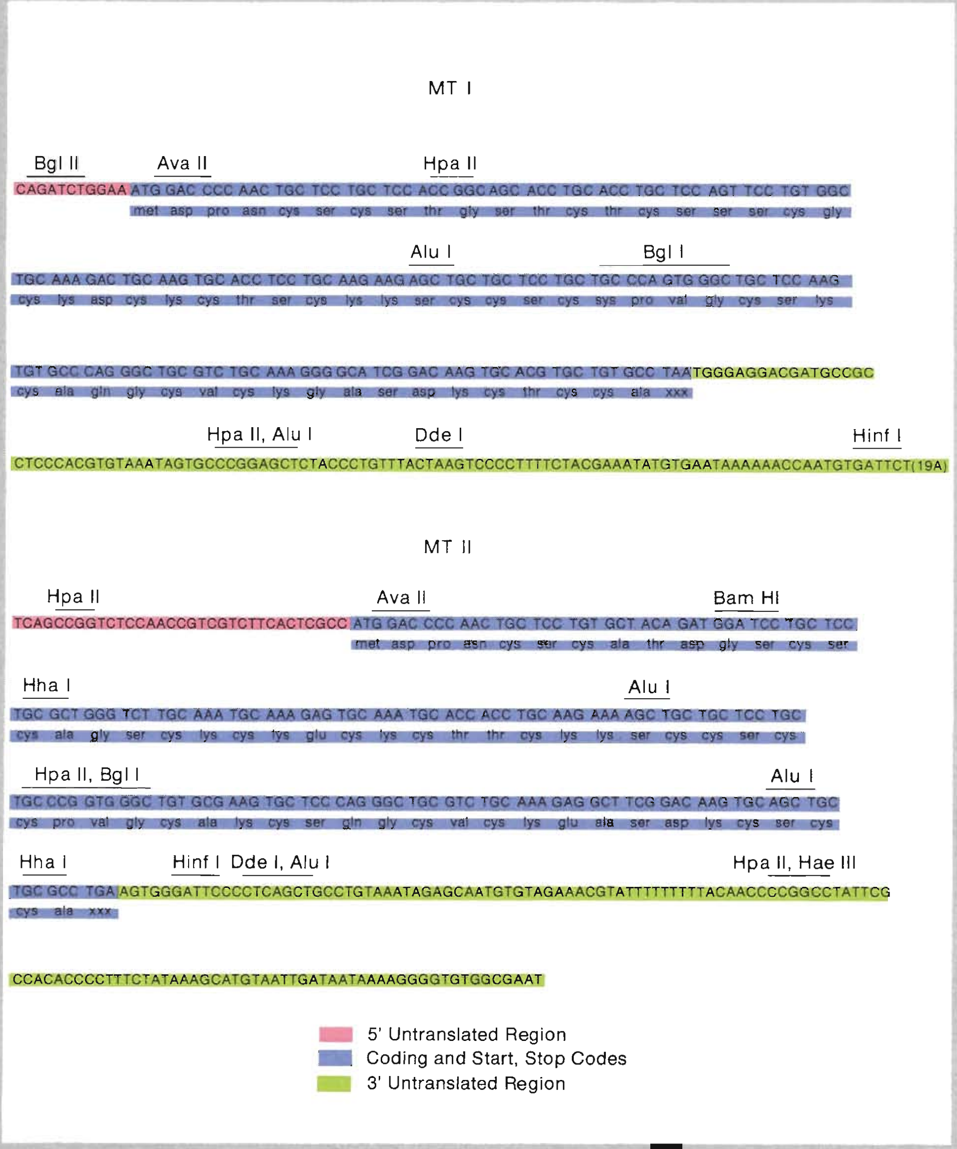
In most cases studied, increased methylation (hypermethylation) of cytosine within or close to a protein-encoding DNA sequence of a gene suppresses expression of that gene's RNA product. Moreover, the hypermethylated state of a specific gene is maintained, that is, it is inherited by succeeding generations of the cell. The precise molecular mechanism by which DNA methylation suppresses gene expression is not yet known. However, several lines of evidence indicate that DNA sequences rich with methylated cytosine can undergo a change from the normal, right-handed helical configuration to a left-handed helical configuration. It is speculated that this conformational change alters the interaction of the gene with the nucleoproteins needed to make transcription work.

Is DNA methylation a mechanism controlling the metallothionein genes? Studies have been performed both in our laboratory and that of Richard Palmiter at the University of Washington that were designed to examine the correlation between this mechanism and metallothionein gene expression.

First, we tested directly the effect of a lack of methylation (hypomethylation) on gene activity. The DNA is generated with monomers that contain, instead of the normal cytidine, the modified nucleoside 5-azacytidine. This DNA behaves in most respects like regular DNA, except that it cannot be methylated because a nitrogen blocks the carbon atom that normally accepts the methyl group. Treatment of Cd^s cells with this hypomethylating compound converts a relatively high fraction (1 to 5 per cent) of these cells to a cadmium-resistant type. Such cells express both metallothioneins and are resistant to low (1 to 2 micromolar) levels of Cd^{2+}. Further, this resistance is heritable; that is, the sublines generated by this treatment are stably maintained during prolonged growth in the absence of Cd^{2+}. The activation presumably is the consequence of the random substitution of 5-azacytidine for cytidine in DNA, but once achieved randomly, a hypomethylated site is inherited in all further generations even though cytidine is used. Palmiter's group obtained similar results studying mouse cells that initially lacked metallothionein gene expression.

With this approach we could develop, at will, other cadmium-resistant sublines of cells whose resistance was apparently due to hypomethylation. But what was the extent to which metallothionein gene activity corresponded to hypomethylation? DNA was purified from normal Cd^s cells, Cd^r cells generated spontaneously, and Cd^r cells generated by 5-azacytidine treatment. These different sets of DNA were treated with restriction enzymes, special enzymes that recognize defined sequences and then cleave the DNA at those sites. For our experiment, we chose two restriction enzymes that cleave DNA at the sequence of bases cytosine-cytosine-guanine-guanine (CCGG). However, one enzyme cleaves no matter what the

Sequencing the Genes


MT I
Bgl II
Ava II
Hpa II
CAGATCTGGAA ATG GAC CCC AAC TGC TCC TGC TCC ACC GGC AGC ACC TGC ACC TGC TCC AGT TCC TGT GGC
met asp pro asn cys ser cys ser thr gly ser thr cys thr cys ser ser ser cys gly
Alu I
Bgl I
TGC AAA GAC TGC AAG TGC ACC TCC TGC AAG AAG AGC TGC TGC TCC TGC TGC CCA GTG GGC TGC TCC AAG
cys lys asp cys lys cys thr ser cys lys lys ser cys cys ser cys sys pro val gly cys ser lys
TGT GCC CAG GGC TGC GTC TGC AAA GGG GCA TCG GAC AAG TGC ACG TGC TGT GCC TAA TGGGAGGACGATGCCGC
cys ala gln gly cys val cys lys gly ala ser asp lys cys thr cys cys ala xxx
Hpa II, Alu I
Dde I
Hinf I
CTCCCACGTGTAAATAGTGCCCGGAGCTCTACCCTGTTTACTAAGTCCCCTTTTCTACGAAATATGTGAATAAAAAACCAATGTGATTCT(19A)
MT II
Hpa II
Ava II
Bam HI
TCAGCCGGTCTCCAACCGTCGTCTTCACTCGCC ATG GAC CCC AAC TGC TCC TGT GCT ACA GAT GGA TCC TGC TCC
met asp pro asn cys ser cys ala thr asp gly ser cys ser
Hha I
Alu I
TGC GCT GGG TCT TGC AAA TGC AAA GAG TGC AAA TGC ACC ACC TGC AAG AAA AGC TGC TGC TCC TGC
cys ala gly ser cys lys cys lys glu cys lys cys thr thr cys lys lys ser cys cys ser cys
Hpa II, Bgl I
Alu I
TGC CCG GTG GGC TGT GCG AAG TGC TCC CAG GGC TGC GTC TGC AAA GAG GCT TCG GAC AAG TGC AGC TGC
cys pro val gly cys ala lys cys ser gln gly cys val cys lys glu ala ser asp lys cys ser cys
Hha I
Hinf I
Dde I, Alu I
Hpa II, Hae III
TGC GCC TGA AGTGGGATTCCCCTCAGCTGCCTGTAAATAGAGCAATGTGTAGAAACGTATTTTTTTTACAACCCCGGCCTATTCG
cys ala xxx
CCACACCCCTTTCTATAAAGCATGTAATTGATAATAAAAGGGGTGTGGCGAAT
🟥 5' Untranslated Region
🟦 Coding and Start, Stop Codes
🟩 3' Untranslated Region

Genetic information is stored as particular sequences of nucleotides or bases along a strand of DNA. What are the base sequences encoding the metallothioneins? To answer this question we isolated cDNA probes I and II, which encode the two major metallothioneins (MT I and II) of Chinese hamster cells. We then cloned in plasmids the double-stranded form of each of these probes, generating sufficient material to determine base composition and sequence by biochemical cleavage with restriction enzymes. The nucleotide sequence determined for each probe included both the complete protein-coding region and portions of the adjacent 3′ and 5′ untranslated regions present in processed mRNA.

The accompanying figure gives the base sequence for each of the probe molecules, with C standing for cytosine, A for adenine, G for guanine, and T for thymine. Under each base triplet, or codon, in the coding regions (blue) is the decoded amino acid (abbreviated). These amino acids make up the protein molecules. The first triplet in each coding region (ATG) also codes for the start of protein synthesis; the last triplet (TAA or TGA) codes for the stop of synthesis. The abbreviations above the base sequences represent restriction enzymes that recognize and cleave at the sequence indicated by the line.

The metallothionein sequences can be compared by computer-assisted analyses with each other and with sequences that encode the metallothioneins of other species. Once sequence data are stored, information can be retrieved and homologies calculated using programs such as those developed at Los Alamos for the Genetic Sequence Data Bank. Such homology searches were essential to the development of sequence-specific hybridization experiments, as described in the text. Additionally, computer analysis can be used to define potential restriction-enzyme cleavage sites that can be used in a variety of experiments. Clearly, the establishment of a DNA sequence data base and associated software for sequence analysis is essential to the problem of information handling in molecular biology—especially when one notes the complexity of the sequence data for the metallothioneins, which are relatively short proteins. ■

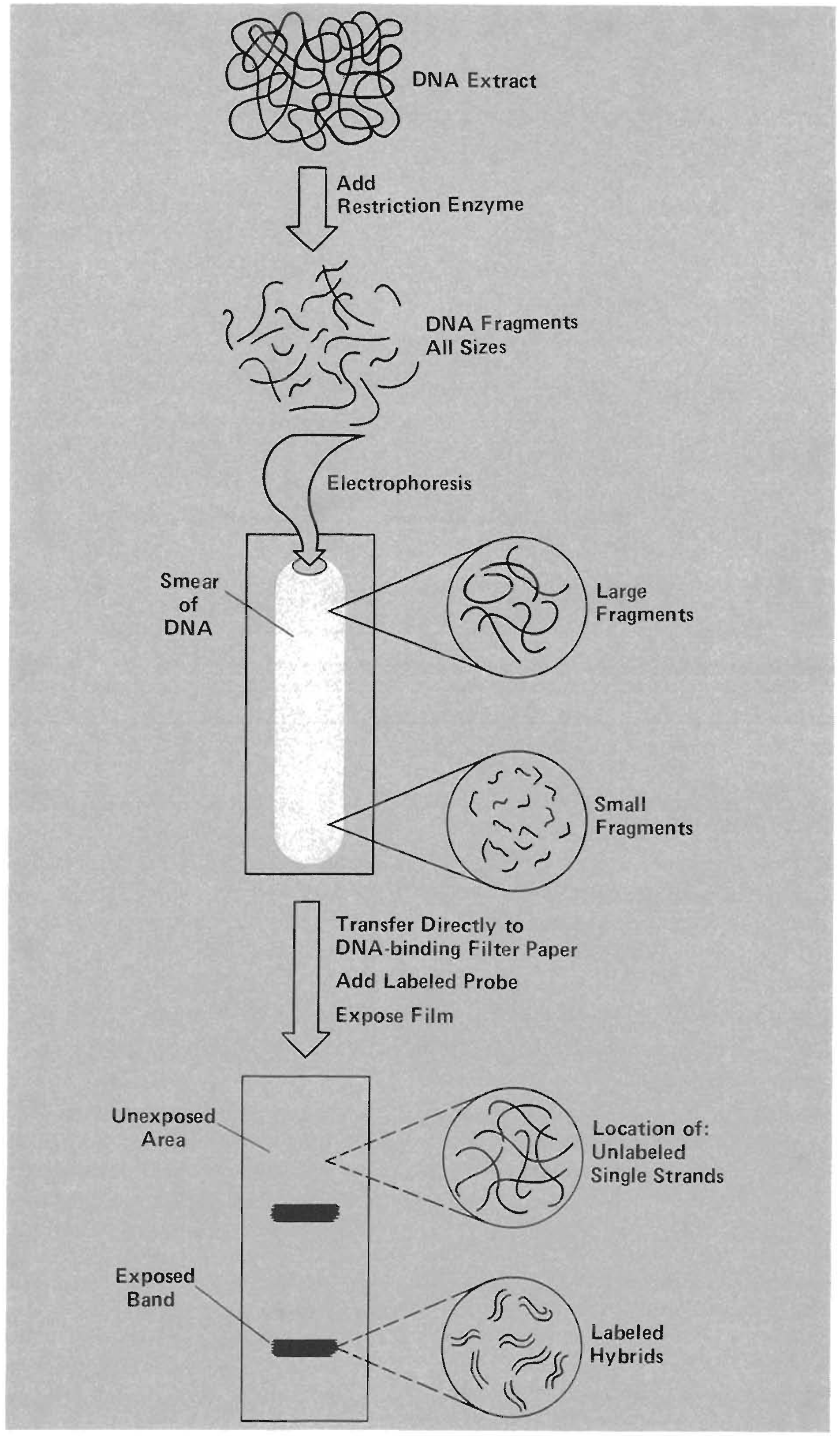


Fig. 11. Filter hybridization analysis of DNA. After the sample of DNA has been treated with a restriction enzyme, the fragments are separated by size using electrophoresis. The smear of single-strand DNA is transferred to special DNA-binding filter paper in such a way that the spatial arrangement of the fragments is preserved. To locate those fragments containing, say, parts of the metallothionein locus, a radioactively labeled gene probe is added to the filter paper. Autoradiography generates exposure bands at the locations where the labeled probe hybridized with fragments.

state of methylation, whereas the other is more selective and will *not* cleave if the internal cytosine is methylated. These two enzymes allowed us to determine the presence or absence of genes modified by methylation in the manner described next and in Fig. 11.

The DNA fragments obtained after treatment with the restriction enzymes were separated on the basis of size by electrophoresis and analyzed with a technique called filter hybridization. In this method the fragments are denatured to single-strand form, transferred directly to special DNA-binding filter paper, and then the appropriate labeled probe is added to the filter paper and allowed to hybridize with the spatially resolved fragments. We used labeled probe that was complementary to cellular DNA at the metallothionein locus, and autoradiography of the filter paper revealed bands wherever gene fragments had hybridized with this probe. We found that the metallothionein locus in DNA from Cd^r sublines capable of metallothionein expression was cleaved by *both* restriction enzymes and thus was hypomethylated. However, the metallothionein locus in DNA from Cd^s cells was resistant to cleavage by the more selective enzyme and thus had methylated gene sequences. In this way, use of our gene probes demonstrated that DNA methylation is one mechanism controlling metallothionein gene expression with, once again, a lack of methylation corresponding to the ability of that gene to express itself.

There are several exciting aspects to the research on DNA methylation. For one, a change in methylation is a nonmutational alteration in gene expression. Knowledge of the mechanisms that control methylation of DNA in the vicinity of specific genes can ultimately be used to study how chemicals or other environmental insults might disrupt the heritable pattern of methylation, and thereby alter the normal pattern of gene expression in cells. Such nonmutational alterations may be involved in carcinogenesis or abnormal development. The metallothionein locus provides a convenient, defined genetic locus at which to study how different conditions elicit such alterations.

As mentioned earlier, DNA methylation may prevent transcription by causing a conformational change in the DNA from one type of helix to another. Further analysis of the metallothionein locus and its control by DNA methylation may help delineate structural and functional hierarchies in the molecular organization of genes through the impact of one or the other on gene expression. Much of this will become possible in the near future as we define more precisely the molecular organization of the metallothionein locus.

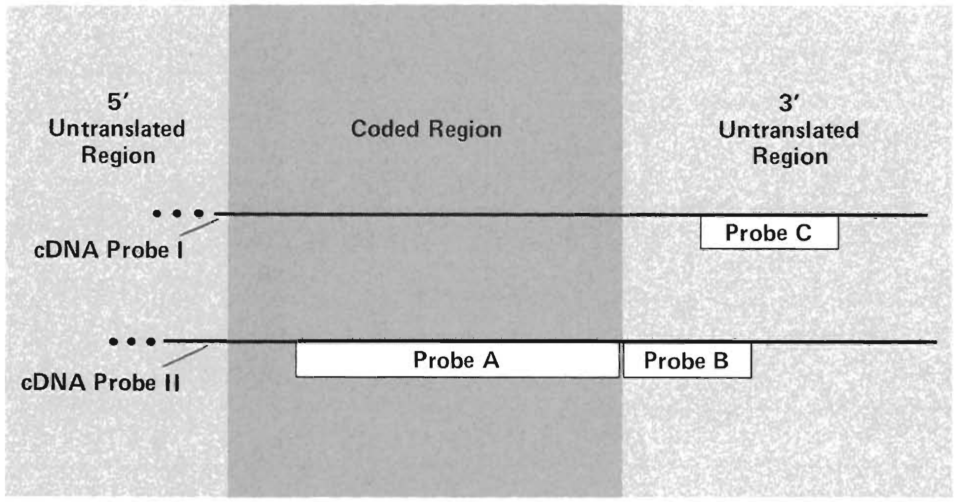


Fig. 12. Specialized gene probes. The original cDNA probes I and II were cut with restriction enzymes, and three of the fragments served as templates to synthesize labeled DNA probes A, B, and C. Probe A is complementary to the major part of the coded region of cDNA probe II and approximately complementary (81 per cent homology) to the same region of cDNA probe I. Thus, probe A hybridizes with the coded region of either metallothionein gene and can be used to identify fragments of cellular DNA that include pieces of either. Because of the differences in nucleotide sequences in the 3′ untranslated regions of cDNA probe I and II, probe B is specific to the MT II gene whereas probe C is specific to the MT I gene.

Gene Amplification

Although we'd shown that DNA methylation was a mechanism regulating the extent of expression of the metallothionein locus, we knew it couldn't be the only one, because the Cd^r cells generated by treatment with 5-azacytidine showed resistance to only low concentrations of Cd^{2+}. Many of the sublines derived by long-term culturing in the presence of Cd^{2+} showed resistance at much higher concentrations and, thus, must be regulated by an additional mechanism. One interesting possibility was *gene amplification*, in which there is an increase in the number of metallothionein genes in the chromosome.

It was known that mammalian cells could respond to environmental stress, such as toxic agents, by a variety of mechanisms, including a remarkable ability to increase the number of copies of specific genes. The most widely studied example of such a genetic response is the development of resistance to methotrexate, an anticancer drug. In a fashion reminiscent of metallothionein production, the cells that develop an increased resistance to methotrexate overproduce a protein, the enzyme dihydrofolate reductase. In this case, the ameliorating protein being overproduced (the enzyme) is also the direct target of the toxic agent (the drug). June Biedler at The Memorial Sloan-Kettering Cancer Institute and Robert Schimke and his collaborators at Stanford University had first shown that one mechanism for this increased production was a capability on the part of the resistant cells to expand a specific portion of a chromosome, generating an amplified number of genes encoding the enzyme. Could such gene amplification also be happening at the metallothionein locus?

To answer this question, we analyzed cellular DNA for gene organization and gene copy number. We started by constructing several probes smaller and more specific than cDNA probes I and II. We did this by slicing particular fragments from these probes with restriction enzymes according to the scheme in Fig. 12. One fragment selected (A) corresponded to a large portion of the coding region of the gene for MT II. Two other fragments corresponded to separate portions of the untranslated regions, but with one of these (C) from probe I and the other (B) from probe II. We then used the fragments as templates to synthesize radioactively labeled probes A, B, and C that were complementary to the corresponding regions of the gene locus in cellular DNA.

We checked the efficiency of probes A, B, and C with several control cross-hybridization experiments between I and II sequences. For example, probe A, derived from the coding region for the MT II gene, cross-hybridized with cDNA probe I under conditions in which hybridization would only take place between strands that had 80 per cent homology or greater. This result was expected because the nucleotide sequences for the two separate coding regions have 81 per cent homology. In contrast, under the same

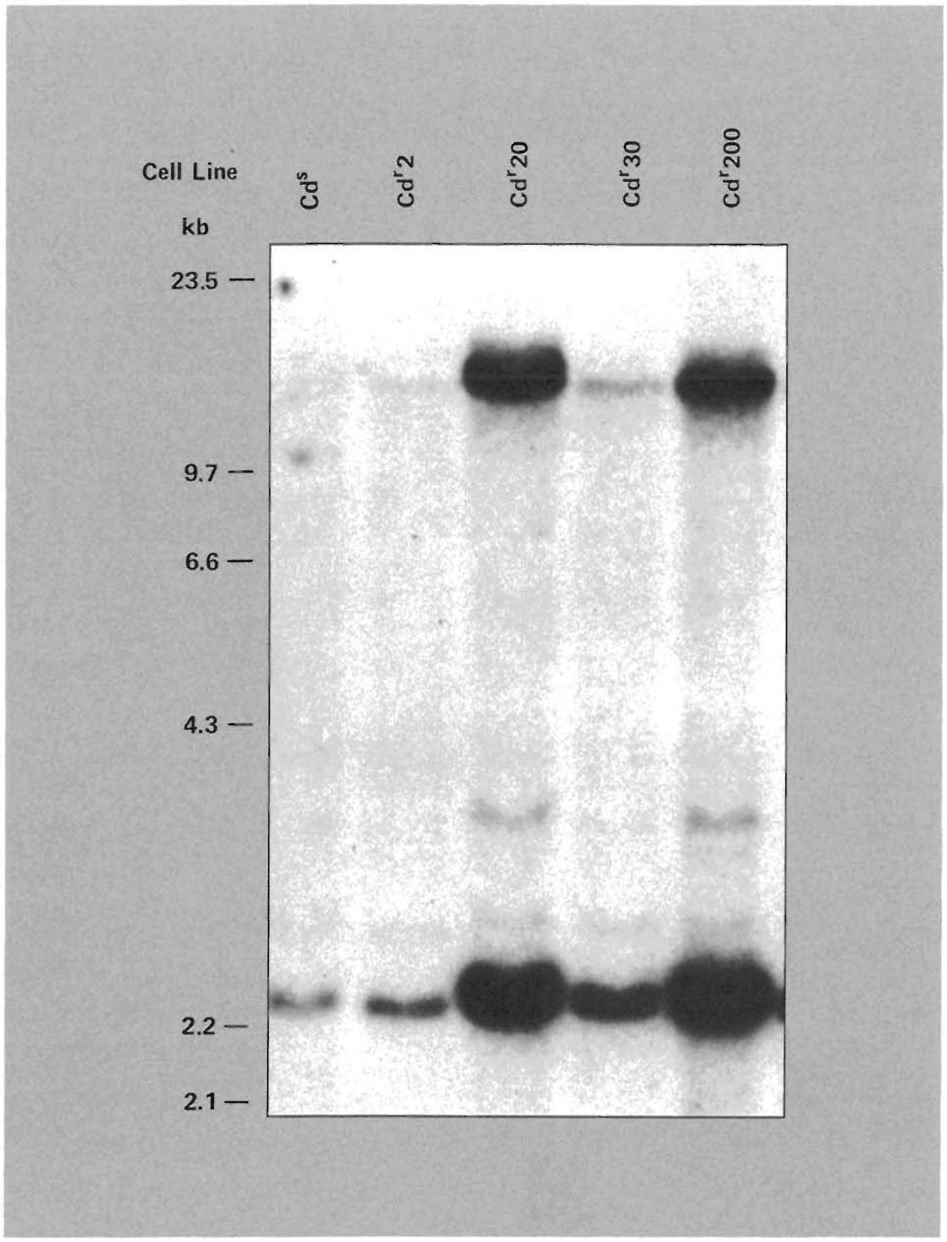


Fig. 13. Treatment of cellular DNA with the restriction enzyme Hind*III results in two gene fragments that hybridize with probe A, the probe specific to the coding region of either metallothionein gene. The concentrations of the two fragments (here revealed with electrophoresis, filter hybridization, and autoradiography) change proportionately from one subline to another, suggesting coordinate amplification of both genes. The numbers on the left are fragment lengths in kilobases (kb). These numbers were determined from the positions of molecules, called markers, of known length that were separated at the same time by the electrophoresis.*

hybridization conditions, probes B and C did hybridize with their homologous cDNA probe (that is, with their original template) but did not *cross*-hybridize with their nonhomologous cDNA probe. Again, this was expected because the homology of the two separate 3′-untranslated regions is only 25 per cent for the B fragment and 30 per cent for the C fragment. These properties of probes A, B, and C then permitted us to analyze both the organization and the gene copy number of metallothionein genes in normal Cd^s cells and the Cd^r sublines.

For example, because probe A is complementary to the DNA of the MT II coding region, the hybridization of probe A with cellular DNA is driven by the "concentration" or gene copy number of the MT II gene. We isolated DNA from each Cd^r subline, measured the kinetics of the hybridization reactions, then compared the data to that for Cd^s cells to estimate an amplification factor. Although we found the Cd^r 200 subline had an amplification factor of 14, the Cd^r 2 subline had a factor of 1, implying that the Cd^{2+} resistance of this latter subline is due solely to DNA methylation. We also found a drop in amplification factor from 7 for the Cd^r 20 subline to 3 for the Cd^r 30 subline, which is consistent, once again, with the idea that another mechanism besides metallothionein synthesis is also operating in the Cd^r 30 subline. Also, an independent study compared the hybridization between probe A and DNA from Cd^s cells with the hybridization between probe A and dilutions of DNA from Cd^r 200 cells. This study revealed that the gene amplification factor for the Cd^r 200 subline may be as high as 40 to 50.

Next we used a particular DNA restriction enzyme called *Hind*III to cut into pieces the DNA from each of the Cd^s and Cd^r cells. We analyzed these fragments using electrophoresis, filter hybridization, and autoradiography and found two major bands (Fig. 13). Because the MT I and II coding regions share extensive homology, probe A

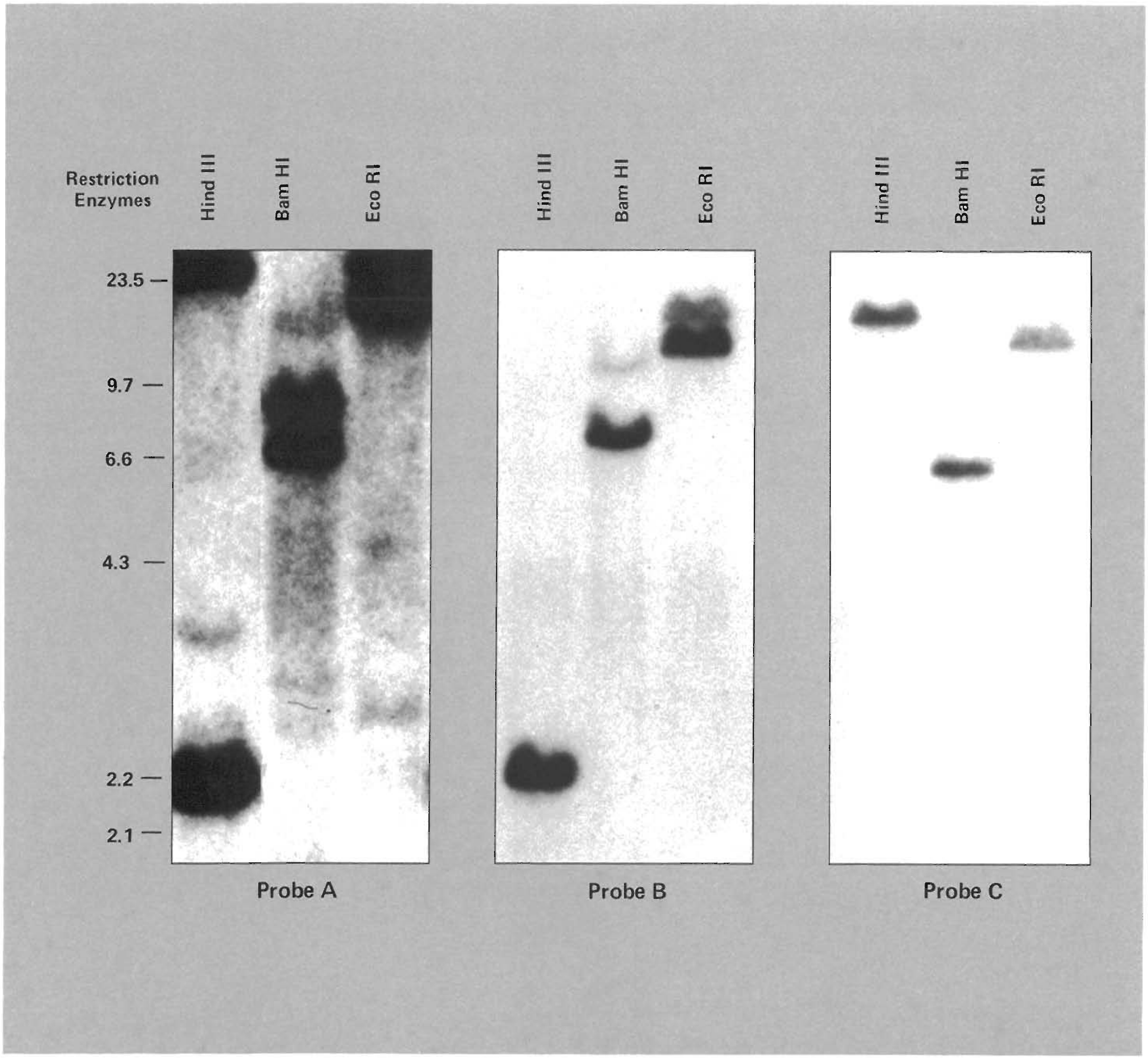


*Fig. 14. Differential hybridization. Extracts of DNA from Cd^r 200 cells were treated with three different restriction enzymes (*Hind*III,* Bam*HI, and* Eco*RI). After electrophoresis, probe A in each case hybridized primarily with two fragments. However, probes B and C, for the* Hind*III and* Bam*HI digests, hybridized differentially. That is, probe C (specific only to the 3′ untranslated region of gene I) hybridized with one fragment, whereas probe B (specific only to the 3′ untranslated region of gene II) hybridized with the other. The* Eco*RI digest did not show this differential behavior.*

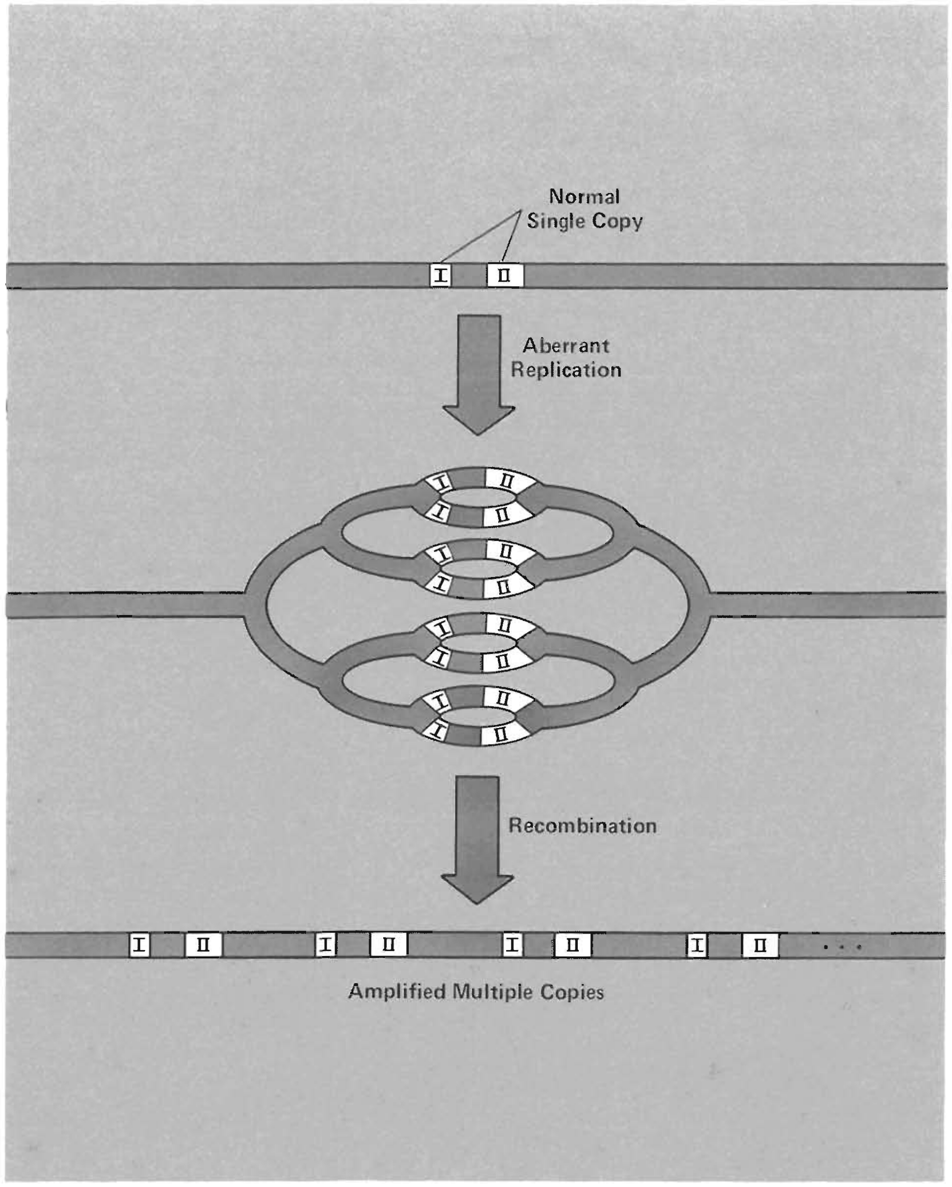


Fig. 15. A possible scheme for gene amplification in which the multiple copies of the two metallothionein genes (I and II) remain localized to a specific chromosome.

probably hybridized with two fragments, each containing one of the two genes. Moreover, the concentrations of each of the two bands increased proportionately in all Cd^r sublines, suggesting that the MT I and II genes are amplified coordinately.

To test such an hypothesis, we digested Cd^s and Cd^r cellular DNA with three different restriction enzymes (*Hind*III, *Bam*HI, and *Eco*RI) and analyzed the fragments in the same manner as before using all three probes A, B, and C. Figure 14 shows the results for DNA from Cd^r 200 cells. When probe A was used, all three digests produced two major bands of approximately equal intensity. If the fragments in each of these two bands did, indeed, contain only one of the two MT genes, then probes B and C should hybridize differentially, that is, probe B only with the fragment containing MT II and probe C only with the one containing MT I. In fact, we found this behavior for two of the three restriction enzymes, *Hind*III and *Bam*HI. For these enzymes, probe C produced a single band at the location of one of the earlier probe A hybridization bands; likewise, probe B produced a single band at the other location. The third enzyme, *Eco*RI, did not show differential hybridization, indicating that the pattern of cuts in the DNA did not separate the two genes. *Hind*III and *Bam*HI digests with other Cd^r sublines gave similar results, showing that amplification in other sublines is also coordinate.

In some cases reported, amplified genes have been found to be extrachromosomal. In our studies coordinate amplification suggested the two metallothionein genes are closely linked physically; that is, they form a gene cluster. This idea, coupled with the observed stability of the Cd^r sublines, further suggests that the phenomenon of gene amplification may be localized to a specific chromosome, obeying a scheme such as the one in Fig. 15.

With Raymond Stallings of the University of Texas, we tested this hypothesis by *in situ* hybridization. Mitotic chromosomes (that is, chromosomes that had assumed their characteristic shape in preparation for cell division) from the Cd^r 200 subline were mildly denatured so that the DNA double strands would separate enough to allow hybridization with probe A. Figure 16 shows these chromosomes after hybridization: the black spots reveal the radioactive probe concentrated at a single chromosomal site. Interestingly, structural analysis of the chromosomes of this subline have shown that this site corresponds to a translocation breakpoint on a large, rearranged chromosome. Further molecular studies in progress in our laboratory have confirmed that the MT I and II genes are, indeed, linked in cellular DNA.

Several exciting aspects of these findings are prompting further research. The metallothionein locus with its two genes provides an opportunity to study the organization of an amplified gene cluster as well as the mechanisms that lead to duplication and rearrangement of that cluster. The metallothionein locus is an example of gene amplification in which the products of the amplified genes are not *directly* related to the toxic action of the agent being resisted. There is also the question of whether or not there are genes coding for the molecules that regulate transcription at the "switch" region flanking the MT genes. Further knowledge about these points will greatly aid our understanding of how environmental insults, such as exposure to chemicals or ionizing radiations, might cause alterations in genes that are not classically defined gene mutations.

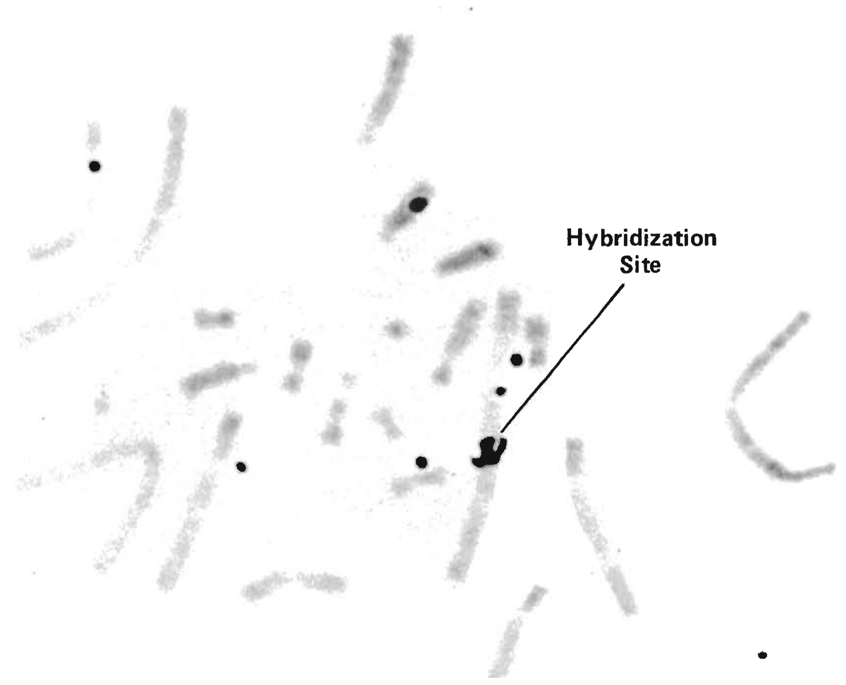


Fig. 16. Gene amplification site. Radioactively labeled probe A, specific to the coded regions of the metallothionein genes, was allowed to hybridize with the mildly denatured chromosomes of the Cd^r 200 subline. The hybridization site revealed by the probe shows that gene amplification is localized on a single chromosome rather than, say, being extrachromosomal. The site corresponds to a translocation breakpoint, that is, the point at which a chromosome has broken and then reformed with a segment normally found on another chromosome. (Photograph contributed by Raymond L. Stallings, University of Texas System Cancer Center, Smithville, Texas.)

The Organization of Cellular DNA

So far, we've ignored the noncoding regions in cellular DNA that interrupt the coding sequences and are spliced out during the RNA processing step. Because these noncoding regions are not present in mRNA, their sequences are not copied into cDNA. Thus, the analysis of cDNA sequences only reveals information about the protein-coding regions of a gene and those parts of the 3′ and 5′ untranslated regions that survive processing, purification, and probe synthesis. As a result, important data are missing on the fine-structure organization of the original cellular DNA.

Also important is the fact that sequences immediately adjacent to the gene in cellular DNA may be involved in the control of gene expression. But the knowledge we'd gained from cDNA sequence analysis alone about these 3′ and 5′ untranslated regions was incomplete. This was especially true of the 5′ region because the enzyme used in the reaction that generates cDNA molecules generally became inactive too soon.

To fill these gaps in our knowledge we turned once again to recombinant DNA cloning techniques. This time, however, our starting material was the cellular DNA itself. We chopped this DNA into almost random fragments with restriction enzymes, selected fragments of appropriate size, and inserted these into the DNA of bacteriophage. The phage were then allowed to infect and multiply in host bacteria. Labeled cDNA probe specific to the metallothionein genes was used to locate (in discrete lytic plaques in bacterial lawns, caused by phage lysis) the cloned fragments of interest.

Because of the manner in which the cellular DNA is cut up, numerous recombinant phage clones are generated. The different fragments thus obtained represent a "library" of sequences representing the cellular genetic material. A few of these are sequences of interest, and contain overlapping sequences that should allow one to "walk" along the DNA defining and flanking a given genetic locus, for example the metallothionein locus. The Genetics Group at Los Alamos is presently attempting such a walk. Hopefully, we will eventually define the nucleotide sequence all the way from one metallothionein gene to the next.

Switching on Other Genes

Already, it has been demonstrated that DNA sequences adjacent to the coding region control the induction of the metallothioneins by Zn^{2+} or Cd^{2+}. In an elegant series of experiments, Richard Palmiter and Ralph Brinster (University of Pennsylvania) located in the DNA of mice a "promoter" region, immediately flanking the MT I gene in the 5′ region, that provided this regulation. In control experiments, deletion of DNA sequences within this promoter region abolished gene expression.

Further proof for the promoter then came from recombinant DNA cloning experiments. The basic idea was to fuse the promoter to genes unrelated to the metallothioneins to see if such chimeric genes could be regulated with Zn^{2+} or Cd^{2+}. Palmiter, Brinster, and their colleagues microinjected the fused DNA into the pronuclei of fertilized eggs and then inserted the eggs into the reproductive tracts of mice serving as foster mothers. In the first experiment the gene for thymidine kinase (a particular "housekeeping" enzyme) was stably integrated so that, in certain tissues of the offspring, the enzyme was synthesized in response to Zn^{2+} or Cd^{2+} exposures. Next, the gene for growth hormone was introduced in a similar fashion. This time the offspring, when exposed to Zn^{2+} or Cd^{2+}, grew to larger than normal size.

These exciting results open a new arena to molecular biologists: the ability to manipulate gene expression in the laboratory and in the animal at a variety of specific gene sites using the metallothionein promoter as the switch. At Los Alamos, in collaboration with Esther Chang of the Uniformed Health Services University of the Health Sciences (Bethesda, Maryland) and Richard Palmiter, we are using the metallothionein promoter to regulate expression of an oncogene (special genes, originally found in certain tumor-causing viruses, whose expression is implicated in human carcinogenesis). By fusing the promoter to one such gene, we hope to manipulate the expression of an oncogene and see how the resulting cellular traits are associated with the malignant behavior of cells. Similarly, in the laboratory of geneticist Frank Ruddle (Yale University), metal-controlled oncogenes are being microinjected into developing mice to study the consequences of oncogene expression during embryogenesis.

In a more general vein, the discovery of this regulatory switch is allowing the experimental manipulation of gene expression in higher eukaryotes. Until recently, such manipulation was confined to simpler or less advanced cells. Use of this tool in mammalian genetics is certain to provide significant information about the biological and biochemical consequences of gene expression and has implications for the eventual correction of genetic diseases (see "Metallothionein Regulation and Menkes' Disease").

Our analysis of both the coding and flanking regulatory regions of the metallothionein locus also permits the intra- and inter-species comparison of the metallothionein genes at the nucleotide sequence level. Such studies may reveal the significance of evolutionary conservation of metallothionein genes. In this way, we and others hope also to find the commonalities in DNA sequences that permit RNA transcription to be induced by metals. For example, we need to determine if the promoter region operates when certain regulatory components present in the cell recognize either a particular base sequence along the DNA or a higher order structure in the DNA. Such nucleotide sequence studies are being aided greatly by the Genetic Sequence Data Bank organized and managed by members of the Laboratory's Theoretical Biology and Biophysics Group.

The range of studies we have discussed emphasizes the flexibility of the metallothionein locus as a model system. It is an excellent system for studying multiple levels of gene expression. It is a unique set of regulated sequences. It is helping to uncover the complex pattern of evolutionary development of genes controlled by metal ions, and it has become a tool for exploring the effects of the expression of other genes. ■

Acknowledgment

The Genetics Group has had a long-standing interest in the study of the metallothioneins, and a number of investigators have contributed and continue to contribute to the project whom we would like to acknowledge. They include Paul Jackson, Robert Moyzis, Robert Tobey, John Hanners, Jean Clare Seagrave, Jon Longmire, Chris Munk, Raymond Stallings, Judy Tesmer, Judy Buckingham, Deborah Grady, and Barbara and Jeff Griffith. Funding for the research described in this article has been provided by the Department of Energy's Office of Health and Environmental Research and the Laboratory's Institutional Supporting Research Program through the Chemistry, Earth, and Life Sciences directorate.

Metallothionein Regulation and Menkes' Disease

Studies of the regulation of metallothionein synthesis have provided insight concerning the molecular basis of human genetic diseases. Several serious human diseases involve altered copper metabolism. One of these, Menkes' disease, is an inherited disorder linked to the X chromosome in which copper is distributed throughout the body in an abnormal fashion. Some tissues, such as the intestine and kidney, accumulate abnormally large amounts of copper. Other tissues, such as those in blood vessels and brain, lack adequate amounts of copper. The results of this metabolic disorder are neural degeneration, abnormal vasculature, and early death.

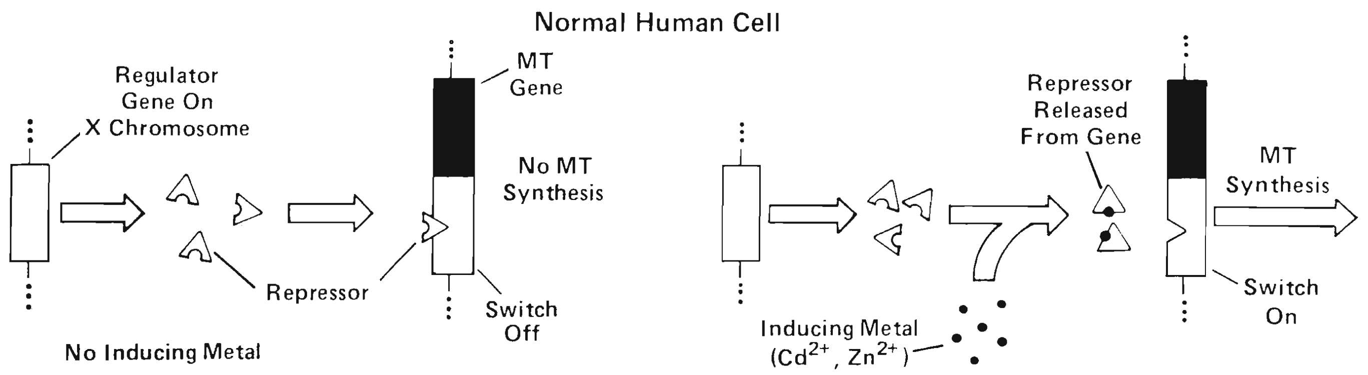
A clue to the molecular basis for this disease came from studies at Los Alamos and elsewhere of cells isolated from patients with Menkes' disease. These cells accumulate more copper than do normal cells when exposed to typical physiological levels of the metal. Copper is one of the metals whose ions bind to metallothionein. Moreover, high levels of copper bound to metallothionein correlated with enhanced cellular copper uptake and retention in Menkes' cells. Thus, in Menkes' cells, the synthesis of metallothionein appears to be "locked on," that is, in the constitutive mode. The disease apparently involves, not genetically altered MT genes, but altered *regulation* of gene expression for metallothionein.

Although Menkes' disease is inherited as a recessive X-linked trait, the MT gene is *not* on the X chromosome. This observation suggests that there is a gene on the X chromosome encoding a molecule that regulates copper uptake and, perhaps, MT gene expression (see part (a) of the figure). This latter gene is apparently the one altered in Menkes' cells so that an effective regulator is not synthesized (b).

To explore this idea further, we used somatic cell genetic techniques to fuse normal hamster and Menkes' human cells, creating hybrid cells with the genetic components of both species. The MT gene from the hamster cells was inactive (due to methylation) so that it could not be induced to synthesize hamster metallothionein (c). However, copper uptake and the synthesis of human metallothionein were found to be normal in the hybrid cells.

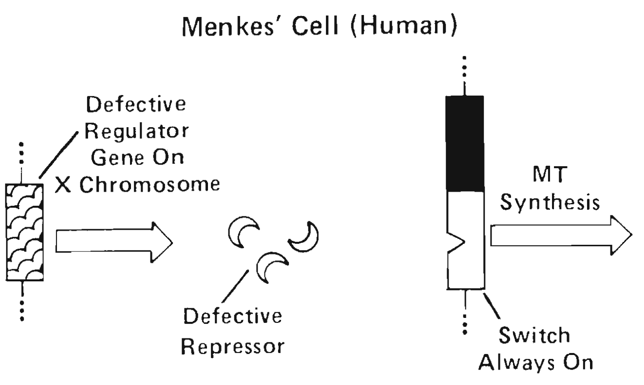
One hypothesis is that the hybrid cells now contained a nondefective gene on the X chromosome of the hamster cell that encoded the regulator. The hamster regulator was synthesized in the hybrid cells, interacted in the usual fashion with the appropriate metal ions, and then regulated the synthesis of the human gene for metallothionein normally (d). Once human metallothionein gene expression was controlled, the cell could maintain normal levels of copper ions. Alternatively, by correcting the abnormal copper uptake in Menkes' cells, the hybrid cells may have regained normal basal levels of metallothionein synthesis.

These results suggest that the regulator is actually a *repressor* of MT gene activity, because without a regulator, the synthesis of metallothionein is always turned on. The results also demonstrate that inter-species regulation of metallothionein synthesis is possible, an exciting result that further points out the evolutionary conservation of the metallothionein system. ●

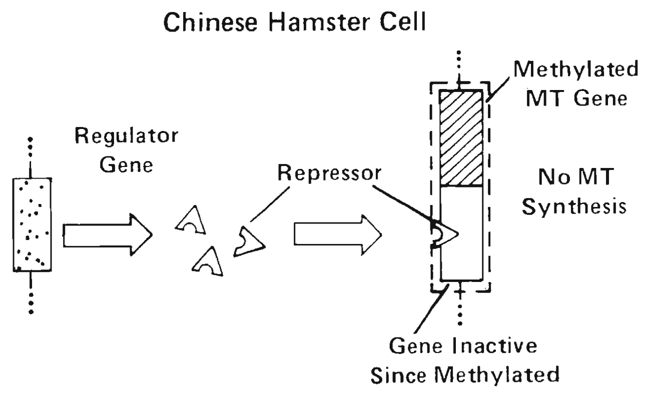


(a)

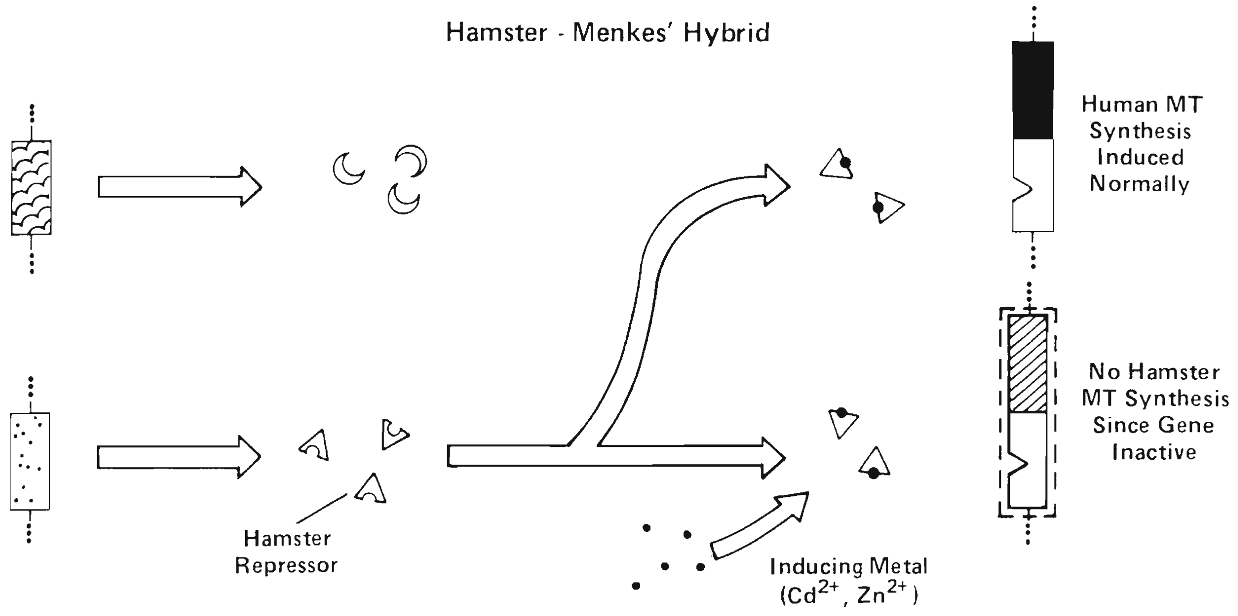
(a) A model for the normal regulation of metallothionein synthesis. Normal human cells control the synthesis of metallothionein with a regulator whose gene is located on the X chromosome. The regulator acts as a repressor in that metal ions induce synthesis of metallothionein by causing the repressor to leave the switch region adjacent to the MT gene, turning the switch on.



(b)



(c)


Hamster - Menkes' Hybrid
Human MT Synthesis Induced Normally
No Hamster MT Synthesis Since Gene Inactive
Hamster Repressor
Inducing Metal (Cd^{2+}, Zn^{2+})

(d)

(b) Menkes' cells have a defective X-chromosome regulator gene and metallothionein synthesis is always switched on. (c) Chinese hamster cells are similar to normal human cells except that methylation has made the MT gene inactive. Hamster metallothionein synthesis is thus turned off regardless of the presence or absence of inducing ions. (d) Hybrid cells that contain the genetic components of both hamster and Menkes' cells use the hamster repressor to control the synthesis of human metallothionein. Copper levels in such hybrid cells are normal.

AUTHORS

Carl Edgar (Ed) Hildebrand is a Pennsylvania native who received his B.A. in physics and mathematics from Gettysburg College. After receiving his M.S. and Ph.D. in biophysics from Pennsylvania State University, he accepted a postdoctoral fellowship at the Laboratory in 1971. In 1972 he joined the Cellular and Molecular Biology Group of the Health Division and subsequently became a member of the Genetics Group when the Life Sciences Division was formed. His scientific interests include the study of DNA replication, chromatin structure, and gene expression. He is Deputy Leader of the Genetics Group and is currently on sabbatical leave at the National Institutes of Health, Bethesda, Maryland.

Brian D. Crawford was born in Maryland and received his B.S. in biochemistry from the University of Maryland at College Park in 1976. In 1981 he received his Ph.D. from the Division of Biophysics, The Johns Hopkins University School of Hygiene and Public Health. He joined the Genetics Group of the Life Sciences Division in 1981 as a J. Robert Oppenheimer Fellow, a position he currently holds. His scientific interests include somatic cell genetics, carcinogenesis, chromosome and gene structure, and regulation of gene expression.

Ronald A. Walters, a native of Colorado, received his B.S. in chemistry and his M.S. and Ph.D. in radiology and radiation biology from Colorado State University. He came to Los Alamos in 1965 to do his dissertation research at the Laboratory under the auspices of the Associated Western Universities and joined the Laboratory in 1967 as a staff member of the Cellular and Molecular Biology Group in the Health Division. As Leader of the Genetics Group of the Life Sciences Division, his scientific interests include DNA replication, histone metabolism, gene expression, gene structure, and molecular radiobiology.

AUTHORS

M. Duane Enger, a native of North Dakota, received his B.S. in chemistry and his M.S. in bacteriology from North Dakota State University. He received his Ph.D. in biochemistry in 1964 from the University of Wisconsin, and that same year he came to the Laboratory as a member of the new teams being formed to study cellular and molecular biology. In addition to serving the Los Alamos community in a variety of roles, he has scientific interests in RNA metabolism and structure, gene expression, trace element metabolism, and genetic protective mechanisms. When the Life Sciences Division was formed, he became Leader of the Genetics Group and, following a sabbatical leave at the Mayo Foundation in Rochester, Minnesota, was appointed Deputy Leader of the Division, the position he currently holds.

Further Reading

J. Kägi, T. L. Coombs, J. Overnell, and M. Webb. "Synthesis and Function of Metallothioneins." *Nature* 292(1981):495-6.

Frank O. Brady. "The Physiological Function of Metallothionein." *Trends in Biochemical Sciences* 7(1982):143-5.

Sally J. Compere and Richard D. Palmiter. "DNA Methylation Controls the Inducibility of the Mouse Metallothionein-I Gene in Lymphoid Cells." *Cell* 25(1981):233-40.

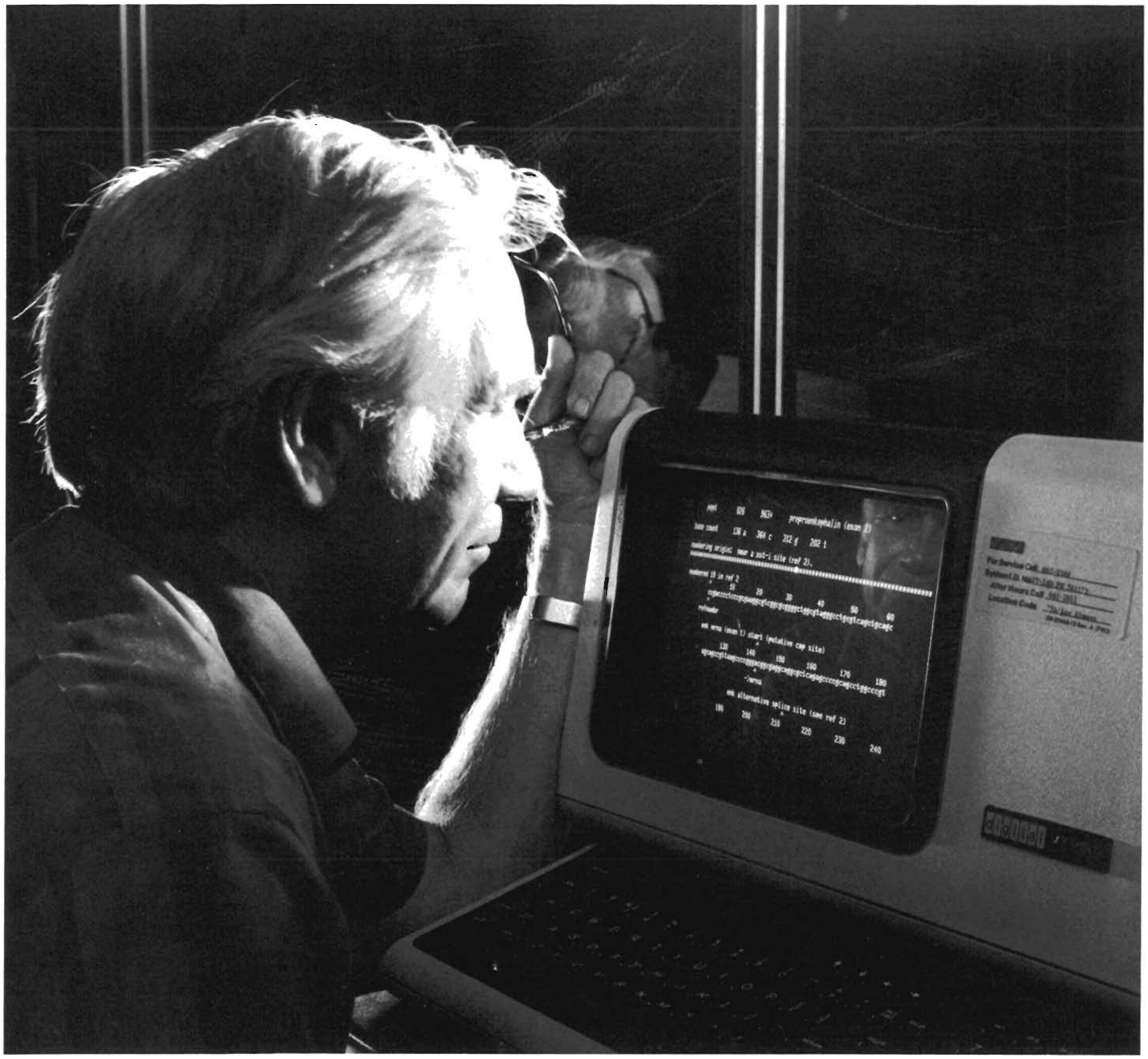
Richard D. Palmiter, Ralph L. Brinster, Robert E. Hammer, Myrna E. Trumbauer, Michael G. Rosenfeld, Neal C. Birnberg, and Ronald M. Evans. "Dramatic Growth of Mice That Develop from Eggs Microinjected with Metallothionein-Growth Hormone Fusion Genes." *Nature* 300(1982):611-5.

C. E. Hildebrand, J. K. Griffith, R. A. Tobey, R. A. Walters, and M. D. Enger. "Molecular Mechanisms of Cadmium Detoxification in Cadmium-Resistant Cultured Cells: Role of Metallothionein and Other Inducible Factors." In *Biological Roles of Metallothionein*, E. C. Foulkes, editor (Elsevier North Holland, Inc., 1982), pp. 279-303.

B. B. Griffith, R. A. Walters, M. D. Enger, C. E. Hildebrand, and J. K. Griffith. "cDNA Cloning and Nucleotide Sequence Comparison of Chinese Hamster Metallothionein I and II mRNAs." *Nucleic Acids Research* 11(1983):901-10.

C. E. Hildebrand, B. D. Crawford, M. D. Enger, B. B. Griffith, J. K. Griffith, J. L. Hanners, P. J. Jackson, J. Longmire, A. C. Munk, J. G. Tesmer, and R. A. Walters. "Coordinate Amplification of Metallothionein I and II Gene Sequences in Cadmium-Resistant CHO Variants." In *Gene Expression, CETUS: UCLA Symposia on Molecular and Cellular Biology*, D. Hamer and M. Rosenberg, editors, Vol. 8 (in press).

ACGGCGAGGCAGGCGCT............... So starts one of the sequences of bases in human DNA that encode enkephalin, an essential brain hormone and regulator of mood. Such sequences are being determined in great and growing numbers. A data bank that makes this information accessible to computer-aided analysis is becoming an important tool in the worldwide campaign to unravel the mechanisms of life and its evolution.



GenBank

by Walter B. Goad

To understand the significance of the information stored in GenBank, you need to know a little about molecular genetics. What that field deals with is self-replication—the process unique to life—and mutation and recombination—the processes responsible for evolution—at the fundamental level of the genes in DNA. This approach of working from the blueprint, so to speak, of a living system is very powerful, and studies of many other aspects of life—the process of learning, for example—are now utilizing molecular genetics.

Molecular genetics began in the early '40s and was at first controversial because many of the people involved had been trained in the physical sciences rather than the biological sciences, and yet they were answering questions that biologists had been asking for years. Max Delbrück, for instance, a very prominent early figure, was trained as a theoretical physicist. He and his group worked with bacteriophage as the simplest systems in which to study replication on a molecular level. Bacteriophage are just at the boundary of life and can replicate themselves only in host bacteria, but what Delbrück and his group learned about the mechanics of replication and the structure of the genes in these viruses turned out to be relevant to all living things.

It's amazing that so much of what today's biology major knows of genetics was discovered so recently. For example, it wasn't recognized until 1944 that DNA, which had been known since 1869, is the carrier of the genes. And not until 1953 was a structure suggested for DNA that explained its ability to transmit hereditary information. That year Watson and Crick proposed that DNA consists of two long nucleotide chains bound together as a double helix by the attractive forces between pairs of complementary bases. This binding of complementary bases is the key physical process in replication. It is physically a very weak association—in fact, two complementary bases bind to each other only a little more strongly than each binds to water when free in solution. Why this association should be such a very strong ordering factor in living systems is a puzzling question, but that it is so is an overwhelming fact.

Another important event in molecular genetics was the cracking of the genetic code, which you might say relates the stuff of memory—DNA—to the stuff of activity—proteins. The idea that somehow the bases in DNA determine the amino acids in proteins had been around for some time. In fact, George Gamow suggested in 1954, after learning about the structure proposed for DNA, that a triplet of bases corresponded to an amino acid. That suggestion was shown to be true, and by 1965 most of the genetic code had been deciphered. Also worked out in the '60s were many details of what Crick called the central dogma of molecular genetics—the now firmly established fact that DNA is not translated directly to proteins but is first transcribed to messenger RNA. This molecule, a nucleic acid like DNA, then serves as the template for protein synthesis.

These great advances prompted a very distinguished molecular geneticist to predict, in 1969, that biology was just about to end since everything essential was now known. It's ironic that within a year of that prediction, restriction enzymes were first isolated. These enzymes are the key to using recombinant DNA techniques to make billions or trillions of copies of a DNA segment. With that many copies and well-known chemical and physical techniques one can then sequence the segment, that is, determine the exact order of the bases it contains.

Before sequencing was possible, almost all the advances in molecular genetics were based on making a single change, with radiation or chemicals, in the DNA of an organism and seeing what happens. For example, you alter the DNA of a phage, allow the phage to multiply in its host bacterium, and examine the consequences of the change in the large population of identical descendants. Incidentally, phage are particularly convenient subjects because they multiply so rapidly. In these experiments you are essentially asking yes or no questions the way you do in the game of twenty questions. With twenty well-chosen questions you can narrow a million possibilities to just one very rapidly. But sequence data is a tool for analysis that is finer than asking yes or no questions.

People working on problems throughout biology—problems like hereditary diseases, cancer, evolutionary relationships—immediately saw that they could make very good use of this tool, and as soon as it became clear that sequencing could be done with facility, it also became clear that sequence data would accumulate at a very great rate. To discuss how these data could be managed and exploited, a meeting was organized at Rockefeller University in the summer of 1979. No one questioned but that computers and sequence data were made for each other. Transmitting a long, seemingly random

Walter Goad, leader of the group responsible for the Laboratory's role in GenBank, at work on one of the entries in this national repository for nucleic acid sequence data.

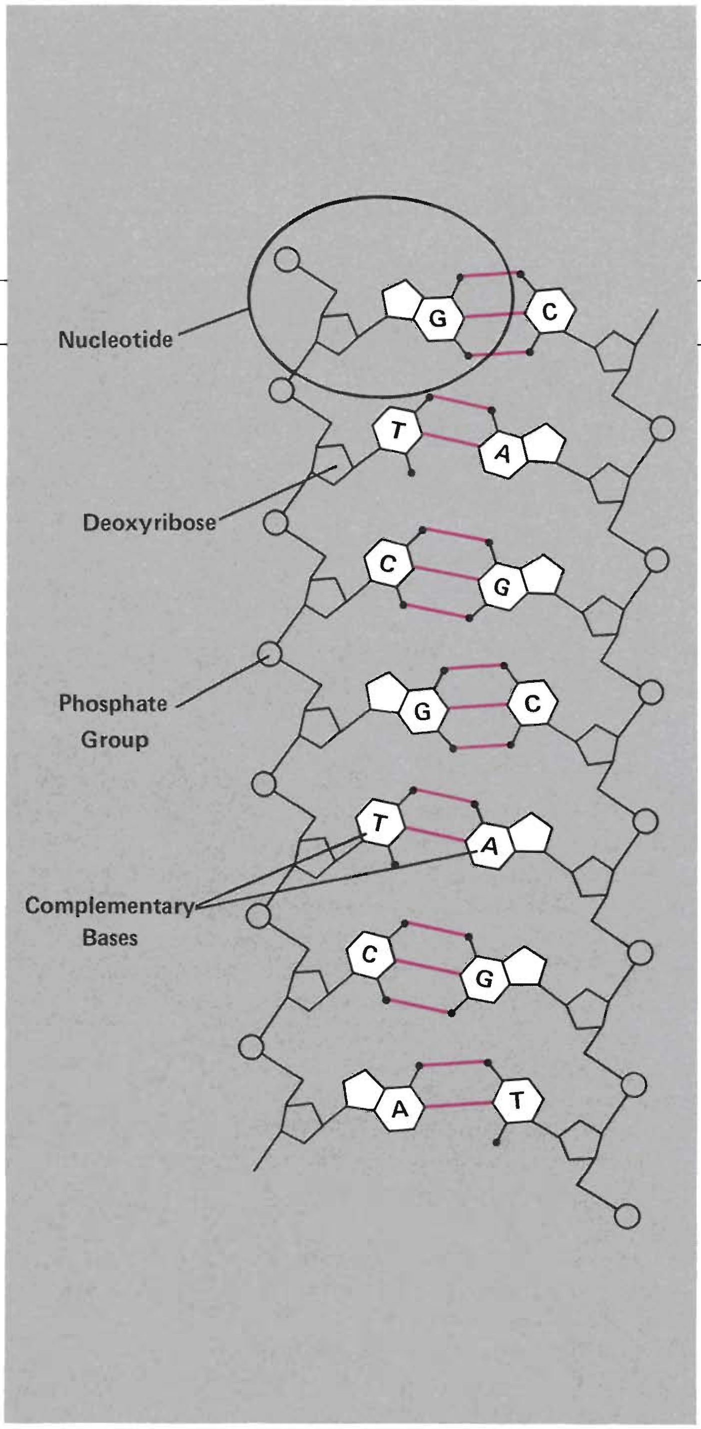
sequence of four letters from one person to another without errors is hardly possible except by putting the information on a computer-readable medium. The need for a data bank was in the air.

How We Got Involved

Mike Waterman and Temple Smith from Los Alamos went to that meeting because for a number of years they had been working with Bill Beyer and Stan Ulam on recognition of patterns in the sequences of amino acids in proteins. Nucleic acid sequencing was just breaking on the scene, but some hundreds of protein sequences were already known, at least partially. Mike and Temple came back from the meeting and talked to people here who were interested in sequence data and their analysis with computers.

I was one of those people, although my training had not been in biology. My early work at the Laboratory was in physics—neutron transport and hydrodynamics—and in molecular physics in the sense of computing equations of state for various materials. I first ventured into biology in 1960 or thereabout. Jim Tuck had met Leonard Lerman, one of the small pioneering group studying bacteriophage, at a cocktail party in Boulder and invited him to Los Alamos. Lerman had discovered that the simplest electronic excitation of DNA—by ultraviolet light—affected its genetic behavior in some more complicated way than would be expected if the ultraviolet light simply altered a single base. To explain this he suggested that perhaps the excitation energy migrated as excitons, which were somewhat the rage in solid-state physics at the time. I talked to Lerman and became quite fascinated by the opportunities for studying a living system on the molecular level. It was so unlike anything known in physics that a single molecular change in DNA could be duplicated faithfully and its consequences examined. You might say that biology offers to physics a molecular amplifier. I hadn't any idea that such a thing existed.

So Lerman and I did some work on the migration of excitons in DNA and its genetic interpretation. As a consequence, I became involved with Ted Puck's group at the University of Colorado Medical Center in Denver. They were looking at genetics at a higher level—at chromosomal abnormalities in newborns. There seemed to be epidemics of them, and I tried to determine how likely it was that we were seeing purely chance behavior instead of epidemics. Then in 1970 I spent a year with Crick at the Laboratory of Molecular Biology in Cambridge, where Max Perutz was working out the structural and functional details of hemoglobin. As a result I did some work on conformational changes in proteins. These changes in shape allow proteins to act as adapters to bring about interactions between molecules that have no specific chemical relation to each other. A classic example is the interaction that hemoglobin induces among four oxygen molecules. I also worked with several molecular



A small segment of a DNA molecule. For clarity the two strands of nucleotides are shown uncoiled from the usual double helical configuration. Each nucleotide is made up of a phosphate group, deoxyribose, and one of four nitrogenous bases. The two nucleotide strands are bound to each other by hydrogen bonds (red) between two possible pairs of structurally complementary bases: guanidine (G) and cytosine (C), or thymine (T) and adenine (A). This pairing of bases is the mechanism directing both replication and transcription of DNA. The DNA of one organism differs from that of another by the sequence of bases along the strands. DNAs of viruses contain tens of thousands of nucleotide pairs, those of bacteria a few million, and those of higher plants and animals billions.

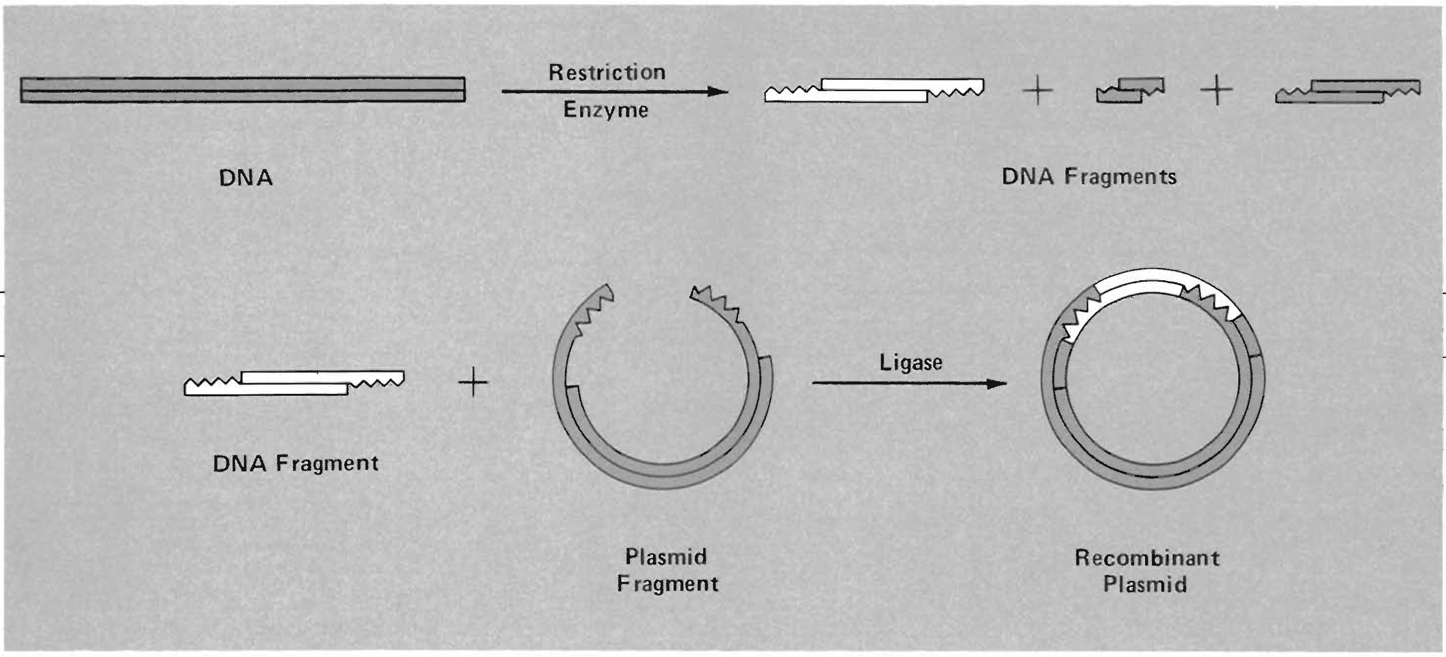
TTT	phenylalanine	TCT	serine	TAT	tyrosine	TGT	cysteine
TTC	phenylalanine	TCC	serine	TAC	tyrosine	TGC	cysteine
TTA	leucine	TCA	serine	TAA	end	TGA	end
TTG	leucine	TCG	serine	TAG	end	TGG	tryptophan
CTT	leucine	CCT	proline	CAT	histidine	CGT	arginine
CTC	leucine	CCC	proline	CAC	histidine	CGC	arginine
CTA	leucine	CCA	proline	CAA	glutamine	CGA	arginine
CTG	leucine	CCG	proline	CAG	glutamine	CGG	arginine
ATT	isoleucine	ACT	threonine	AAT	asparagine	AGT	serine
ATC	isoleucine	ACC	threonine	AAC	asparagine	AGC	serine
ATA	isoleucine	ACA	threonine	AAA	lysine	AGA	arginine
ATG	methionine	ACG	threonine	AAG	lysine	AGG	arginine
GTT	valine	GCT	alanine	GAT	aspartic acid	GGT	glycine
GTC	valine	GCC	alanine	GAC	aspartic acid	GGC	glycine
GTA	valine	GCA	alanine	GAA	glutamic acid	GGA	glycine
GTG	valine	GCG	alanine	GAG	glutamic acid	GGG	glycine

Dictionary of the genetic code, listing the codons, or triplets of bases, in DNA that code for amino acids in proteins. The code is degenerate in the sense that, with the exception of tryptophan and methionine, each amino acid is specified by more than one codon. Synthesis of a protein involves transcription of a DNA segment to a single-stranded chain of nucleotides known as messenger RNA (mRNA) and translation of the mRNA to the protein. The bases in mRNA are complementary to those in one strand of the DNA and are assembled in the order given by that strand. The transcribed codons are then translated to the sequence of amino acids in the protein. The codons TAA, TAG, and TGA act as signals for terminating protein synthesis, and the codon for methionine, ATG, acts as a signal for initiating the synthesis of nearly all proteins.

geneticists on the details of how bacterial DNA is replicated and how its expression is turned on and off. These experiences led rather naturally to my being interested in DNA sequence data.

That 1979 meeting set several of us to thinking about a data bank. It was clear that just accumulating the data in a computer was not particularly interesting. What was interesting were questions about organizing, managing, and analyzing the data. We started collecting sequence data and writing software for analysis. We worked up a proposal and presented it to NIH and NSF and anyone else who would read it. That proposal included, in addition to a data bank, an analysis center that would provide access to software running on our computers. Thousands of experts around the country have questions they would like to ask about sequence data. Many of them will find collaborators who are expert in applying computers to answer their questions. But the ideal situation is for the person with a question to sit down at a terminal and ask it himself. There are many barriers to doing that, but one that can be removed is the necessity of writing your own software or of rewriting some existing software to suit your own computer. It turns out that very little software is really portable despite all the talk about the issue. But given modern telecommunications, there is no reason not to use existing software on the machine it was developed for. We were excited about an analysis center because it would put us in touch with many imaginative, talented people.

We contacted other groups—in the States, in Europe, and in Israel—that were also interested in sequence data. We talked, for instance, with Margaret Dayhoff's group at the National Biomedical Research Foundation in Washington. Margaret for many years led their data bank for protein sequences; unfortunately she died not too long ago. Her group started collecting data on nucleic acid sequences about the same time we did. We exchanged ideas with Margaret's group about organizing and managing the data, which were



Many copies of a DNA fragment are needed to determine its sequence of bases by standard chemical or biochemical methods. These copies are provided by recombinant DNA techniques made possible by the action of restriction enzymes, which recognize certain short sequences of bases in a DNA molecule and cut the molecule at those sequences. Such an enzyme cuts each DNA molecule in a sample containing many molecules of the DNA into the same set of fragments. The desired fragments can then be separated from the others by length. Many restriction enzymes cut DNA so as to leave the fragments with "sticky" ends; that is, a short stretch of bases left on one end of a fragment is complementary to a short stretch of bases left on the other end. If the restriction enzyme used does not act in this way, sticky ends can be added to the fragments by other enzymes. The DNA fragments are then mixed with plasmids (small circular pieces of DNA found in bacteria) whose circles have been opened by an appropriate restriction enzyme and equipped, one way or the other, with complementary sticky ends. Some of the opened plasmid circles associate with and are closed by the added DNA fragments, whereas others simply close again. Treatment with still another enzyme, a ligase, re-establishes the covalently linked circles of DNA, which can now infect their bacterial hosts and be replicated rapidly. Finally the added DNA fragments are removed from the multiplied plasmids, again by action of an appropriate restriction enzyme and separation by length. A great many ingenious refinements have been devised to enhance the efficiency and ease of these procedures. For example, some of the bacteria will lack an infecting plasmid, and some of the plasmids will lack an added DNA fragment. Those bacteria infected by a "recombinant" plasmid can be made to replicate to a greater degree by arranging that the recombinant plasmids carry some property advantageous to the bacteria.

particularly important issues in light of the anticipated volume of the data. And we were in touch with people at Stanford who had feet in both the computer science department and the biochemistry and molecular genetics departments, some of whom were especially interested in the question of interfaces between people with questions and the computers and software that could answer them.

Many people we talked with agreed that a data bank would be most useful if combined with an analysis center. But partly because of the ever-present shortage of funds, partly because of the difficulty of specifying what an analysis center should be like, and partly because of politics, NIH decided to put off an analysis center and to invite proposals only for a data bank to be cosponsored by a number of organizations, including NSF, DOE, and DOD.

Oddly enough, we submitted two proposals, each a joint proposal with one of two firms we were already involved with. One of the firms was IntelliGenetics, which was formed by the Stanford group to offer analytical services to the biotechnology industry; the other was Bolt Beranek and Newman Inc. BBN had distributed our collection of nucleic acid sequence data through their PROPHET system, which provides dial-up software to bench biochemists and biologists.

Both BBN and IntelliGenetics approached us independently and pointed out that a joint proposal would make sense. We would collect and manage and organize the data, and the company would handle the distribution. That sounded good to us. For one thing, a joint proposal with a commercial firm solved the practical problem of collecting user fees, which are probably necessary if for no other reason than to discourage abuse of the system.

However, as part of a national laboratory we could not offer our collaboration to one firm and not the other. So we ended up with two joint proposals to write. They were quite different, though, because BBN is a spin-off from MIT and IntelliGenetics is a spin-off from Stanford. Anyone familiar with the artificial intelligence community knows that MIT and Stanford differ a great deal in style.

After having the proposals reviewed—there were three major ones, our two and a third by Margaret Dayhoff's group—and asking a great many questions, NIH selected our joint proposal with BBN. We heard the news in September of last year and felt nicely rewarded for all the effort on the proposals. But then the work began in earnest.

How the Data Bank Operates

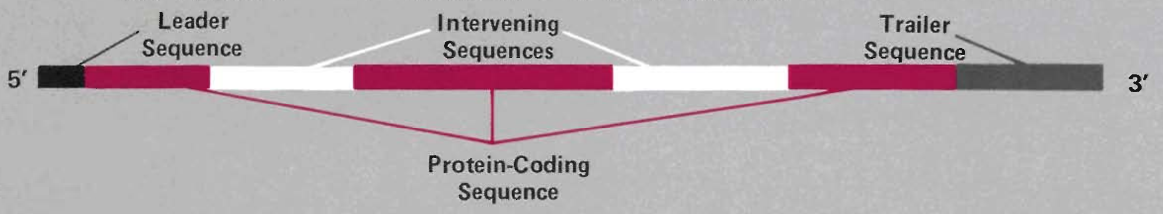
The first job we face is collecting the sequence data. This is straightforward although physically very demanding. We've been working very hard just to get caught up with the data that was in existence when we started. Our estimate of that was based on scanning some journals and looking through lists of titles. Unfortunately, the estimate was low because the titles of many articles with sequences in them don't reveal that. Recently, more and more sequences are being sent directly to us by the authors, and we are trying to get support from the journals to encourage or enforce that practice.

Our contract calls for us to collect all nucleic acid sequences containing more than fifty bases. The data bank now contains about two million bases, and that number is increasing at the rate of about a million bases per year, roughly what we expected. Most of the sequences are on the order of a thousand bases long, but the longest—the entire genome of the lambda bacteriophage—contains about 50,000 bases. About 200 species are represented, although the usual laboratory species, drawn from viruses, bacteria, fruit flies, and small rodents, predominate. We do have a few unusual species—some plants, apes, and an East Indian deer that happens to have a very small number of chromosomes. However, for only a very, very few of these species is the entire genome known. In fact, even for that most studied of all bacteria, *E. coli*, only about 3.5 percent of its genome has been sequenced, and for *Homo sapiens* the fraction is less by a factor of about a thousand.

The data bank entry for a sequence includes, in addition to the sequence itself, a name up to ten characters long that tries to speedily identify the sequence by at least suggesting species and function, the

The weekly meeting of the group is a time for discussing problems about the GenBank entries.

```
locus          humhba2      1138 bp                        updated    06/30/82
definition     human alpha2-globin gene and flanks. 1138bp
source         human.
reference      1  (bases 1 to 1138)
  authors      liebhaber,s.a., goossens,m.j. and kan,y.w.
  journal      proc nat acad sci usa 77, 7054-7058 (1980)
reference      2  (bases 1 to 1054)
  authors      proudfoot,n.j. and maniatis,t.
  journal      cell 21, 537-544 (1980)
reference      3  (bases 98 to 929)
  authors      orkin,s.h., goff,s.c. and hechtman,r.l.
  journal      proc nat acad sci usa 78, 5041-5045 (1981)
reference      4  (bases 802 to 837)
  authors      proudfoot,n.j. and longley,j.i.
  journal      cell 9, 733-746 (1976)
comment        formerly humanhba2. compared with humhbaps in ref 2. ref 3 reports
               sequence of alpha-thalassemic individual with complete absence of
               alpha2 mrna. ref 3 suggests that cttgg rather than cctgg at base
               660 is inconsistent with the presence of a bstni restriction site.
               ref 4 compares with rabbit alpha- and beta-globins.
features          from       to        description
    pept           135       230       alpha2-globin
                   348       551
                   692       820
base count         183 a     411 c    351 g    193 t
origin         approximately 100bp 5-prime to bstni site
          1 aggccgcgcc ccgggctccg cgccagccaa tgagcgccgc ccggccgggc gtgcccccgc
         61 gccccaagca taaaccctgg cgcgctcgcg gcccggcact cttctggtcc ccacagactc
        121 agagagaacc caccatggtg ctgtctcctg ccgacaagac caacgtcaag gccgcctggg
        181 gtaaggtcgg cgcgcacgct ggcgagtatg gtgcggaggc cctggagagg tgaggctccc
        241 tcccctgctc cgacccgggc tcctcgcccg cccggaccca caggccaccc tcaaccgtcc
        301 tggccccgga cccaaacccc acccctcact ctgcttctcc ccgcaggatg ttcctgtcct
        361 tccccaccac caagacctac ttcccgcact tcgacctgag ccacggctct gcccaagtta
        421 agggccacgg caagaaggtg gccgacgcgc tgaccaacgc cgtggcgcac gtggacgaca
        481 tgcccaacgc gctgtccgcc ctgagcgacc tgcacgcgca caagcttcgg gtggacccgg
        541 tcaacttcaa ggtgagcggc gggccgggag cgatctgggt cgaggggcga gatggcgcct
        601 tcctctcagg gcagaggatc acgcgggttg cgggaggtgt agcgcaggcg gcggcgcggc
        661 ttgggccgca ctgaccctct tctctgcaca gctcctaagc cactgcctgc tggtgaccct
        721 ggccgcccac ctccccgccg agttcacccc tgcggtgcac gcttccctgg acaagttcct
        781 ggcttctgtg agcaccgtgc tgacctccaa ataccgttaa gctggagcct cggtagccgt
        841 tcctcctgcc cgctgggcct cccaacgggc cctcctcccc tccttgcacc ggcccttcct
        901 ggtctttgaa taaagtctga gtgggcggca gcctgtgtgt ggctgggttc tctctgtccc
        961 ggaatgtgcc aacaatggag gtgtttacct gtctcagacc aaggacctct ctgcagctgc
       1021 atggggctgg ggagggagaa ctgcagggag tatgggaggg gaagctgagg tgggcctgct
       1081 caagagaagg tgctgaacca tcccctgtcc tgagaggtgc cagcctgcag gcagtggc
sites          key         span  description
⬛        98 ->mrna         1   mrna cap site
          98 refnumbr       1   numbered 1 in ref 1; zero not used
🟥       135 ->pept         1   hba2 cds start
         138 pept/pept      0   hba2 mature protein start
         139 refnumbr       1   numbered 1 in ref 2
⬜       231 pept/ivs       0   ivs1 start
         231 variation      5   tgagg deleted in ref 3
🟥       348 ivs/pept       0   ivs1 end
         416 conflict       1   a in refs 1 & 2; g in ref 3
⬜       552 pept/ivs       0   ivs2 start
         655 conflict       1   c in refs 1 & 2; ct in ref 3
         659 conflict       4   gctt in refs 1 & 2; ggcct in ref 3
🟥       692 ivs/pept       0   ivs2 end
         763 conflict       1   t in refs 1 & 2; c in ref 3
▨        820 pept<-         1   hba2 cds end
         930 mrna<-         1   poly a addition site
         930 refnumbr       1   numbered 1 in refs 4;3'to 5'
```



The GenBank entry for a gene specifying the alpha-2 polypeptide chain of human hemoglobin. This aberrant gene is responsible for a particular form of thalassemia, that is, for one of a group of hereditary anemias. We have added to the entry a schematic representation of the gene and its environs, which is color-keyed to the sites of interest and their descriptions given in the entry. The sequence specifying the leader region of the mRNA begins at base 98 with a site to which is added a "cap" (containing a methylated guanine bound to the mRNA by phosphate groups) that may promote translation of the mRNA. The polypeptide coding, which begins at base 135 with the initiation codon ATG, is interrupted by two intervening sequences beginning at bases 231 and 552. These sequences are transcribed but are spliced out before translation. The sequence specifying the trailer region of the mRNA ends with a site to which is added a "poly-A tail" (of many adenylate residues) that may protect the mRNA from degradation.

source of the sequence data, and a succinct description of the biological function and setting of the sequence. The sequences are catalogued according to these three items, which are roughly equivalent to title, author, and subject of a book. We also note features that have been determined to be of biological interest—but not those that are only speculated to be so. We update the entries, of course, as new information comes our way. For example, many of the sequences determined early in the game have since been redone. We make no judgment about the accuracy or reliability of the data; that is a matter for investigators, referees, and users to thrash out. We are, however, very concerned about handling the entries consistently because the data are to be used primarily as input for computer programs, which are not at all tolerant of inconsistency. At our weekly meetings problems of consistency take a great fraction of our time.

In some cases we have to decide whether two sequences thought to be from different genetic regions are actually from the same region and vice versa. These situations can arise from sequencing errors, from working with different strains of the same organism or with different alleles of the same gene, or from the fact that DNA is not always copied exactly and some of the inexact copies are still being replicated. We try to resolve conflicts of this nature with the authors, and if it is determined that two sequences are indeed the same, we combine them in a single entry and note the differences.

The work involved in collecting and annotating the sequences will soon be somewhat reduced because of an arrangement we have worked out with the data bank for nucleic acid sequences at the European Molecular Biology Laboratory in Heidelberg. That data bank and ours have been communicating since the beginning but until now have been randomly duplicating rather than sharing each other's efforts. But the sharing must be done in a way that preserves the integrity of each data bank, since each has a different constituency and different support. The Japanese are also moving toward creating a data bank, and we believe they will be a third collaborating entity. We hope this collaboration will permit us to spend less time just hurrying to keep up and more on improving our operation and getting on with our research.

One improvement we have been able to implement recently is the transfer of data entry and management activities to a microcomputer. Previously we had been using the Laboratory's mainframe computers, which certainly have their virtues but offer only line-editing capability. With screen editing we should all be more productive. Using a microcomputer has another advantage—it puts us more in touch with the people who are doing the sequencing, who seldom use big computers but are turning increasingly to microcomputers. We can easily make the software developed for our microcomputer compatible with other commonly used ones. The authors will, we think, find this software useful and will be more likely to send their sequence data to us already in our format.

But we love big computers, too. For detailed comparisons of sequences, a Cray is the machine to use because it's twenty times faster and, more important for us, ten times cheaper. We feel that each new entry should be routinely compared with existing entries and think we can develop a much cheaper way to do that. But unfortunately, our preoccupation with the primary job delays work on many such improvements.

The sequence data are available to users primarily through BBN. Once a month we send the data base to them, and they issue it on tape. A few people subscribe to the data base on a regular basis, but most receive it only now and again. The data base is also available over the telephone through a computer at BBN. National Biomedical and IntelliGenetics distribute the data base to their customers, and several computer centers at universities maintain it for a large number of users. We also offer our software to users on a dial-up basis. About seventy people around the country take advantage of this service now and then. We regard this as a way of encouraging people to submit data to us directly, to find errors in the entries, and to criticize and make suggestions about our operation. We would like to have more funds to support this aspect of GenBank, but it smacks of an analysis center, which is still hanging fire.

I'd like to mention our splendid crew, which includes people from mathematics and from biology and certainly represents one of the few instances where these two fields really coalesce. The staff members are Jim Fickett, Christian Burks, and Minoru Kanehisa. Minoru helped create the data bank; he is now physically at NIH and works only half-time for us. Temple Smith and Ruth Nussinov have been involved as visiting staff members. And Gerry Myers, a molecular biologist and tutor at St. John's College, is a consultant and major contributor. We rely for help on Graduate Research Assistants—young people between college and graduate school—with majors in biology or computer science. Since they write the first drafts of the sequence annotations, they learn quite a bit about molecular genetics, not to mention about computers. They generally stay for about a year, and all have gone on to either graduate or medical school. Right now we have with us Bryan Bingham, Ute Elbe, Leslie Kay, Randy Linder, and Debra Nelson. Carol England is both secretary and coordinator of data entry, and Mia McLeod has just joined us as a data analyst.

What to Look for in the Data

Analysis of the sequences is, of course, the ultimate objective of their being determined and our collecting them. But those of you who may have seen sequences in journals will agree that picking out features of interest or making comparisons requires a specialist—or a computer fed very clever software. Since sequence data have neither the internal order of numerical data—the order of the numbers themselves—nor that of textual data—the grammar and syntax of the language—their analysis calls for a different approach. We have developed some interesting programs for analysis. For example, Jim Fickett discovered that the portions of a sequence that code for proteins have a regularity, a periodicity in the statistical sense, that is absent in noncoding portions. This fact can be used to find the protein-coding segments, and then it is relatively trivial to translate the bases in the segments to the amino acids in the proteins. Incidentally, more and more protein sequences are being predicted from nucleic acid sequences because it's much easier determining them that way than from the proteins themselves. People are particularly interested in the predicted amino acid sequences of membrane proteins because these proteins are of special importance in cellular activities.

As I mentioned before, comparison of sequences is a significant aspect of sequence analysis. [See the sidebar "Quantitative Comparison of DNA Sequences" for more detail about this subject.] What one wants to know is in what way and by how much two sequences differ, as a whole or over certain portions. Generally sequences differ a great deal, but the lowest level of difference between sequences from closely related DNAs is the replacement of one base by another here and there along the sequences. Such base replacements may in some instances be inconsequential because of the degeneracy of the genetic code. A higher level of difference arises from additions or deletions of bases, and we and other people have developed very efficient algorithms for spotting such differences.

Another type of analysis involves searching for strings of complementary bases along an RNA molecule that would permit it to fold and bind to itself—to form hairpin-like structures. A good deal of experimental evidence indicates that in many circumstances it is such a structure, rather than the sequence itself, that is recognized by an enzyme or a protein as a signal for some activity. Similarly, one can look for sequences of bases in a DNA molecule that would cause irregularities in its structure. Even today the structures of various DNAs are not precisely known because of the difficulty of obtaining large crystals. One idealizes the structure as perfectly regular, but considering that the sugar-phosphate backbone encloses four different bases, two of which are twice as large as the other two, certain sequences of bases are bound to cause irregularities in shape or mechanical properties. These irregularities may serve as signals for initiating certain processes or may play a role in the packaging of DNA. In higher organisms DNA does not exist simply as a random coil. Instead it is superwound around some proteins, and this superwound structure is itself superwound into a very complex structure.

In terms of analysis, one thing is clear: as more and more sequence data become available, more and more ideas about what to look for will be proposed. And there's no question but that this aspect of GenBank is the most challenging and the most rewarding.

What the Sequence Data Offer

There's also no question but that sequence data will answer—and raise—more and more questions about the mechanisms of life and its evolution. I don't mean, however, to imply that sequence data displaces everything else. Biochemistry, cell biology, organismic biology—these fields are as important as ever, but they are increasingly being propelled and unified by insights and techniques from molecular genetics. And much has been and can be learned about DNA without actually determining sequences. It is relatively simple experimentally to determine, for example, how many times a certain segment of DNA is repeated or the degree to which different organisms share a given gene. But sequence data provide the finest detail.

Classical evolutionary studies, for instance, rely on comparing characteristics such as anatomy or geographical distribution to find out how organisms are related. At the level of molecular genetics, you compare the sequences of bases in the genes of the organisms and the proteins dictated by those sequences. One of the things you find is that there is a tremendous range in how exactly the sequences of bases are conserved even between organisms that had a common ancestor not so long ago. What apparently is true is that if a small change in a gene causes a big disturbance in the organism, then the sequences differ by very little. But if a small change disturbs the organism very little or not at all—perhaps makes only a trivial change of an amino acid in a protein or no change whatever—then

the sequences will drift a lot. One of the exciting lines of inquiry is to try to understand, in terms of these big or small changes in the sequences, what evolutionary pressures were involved.

Comparison of sequences can give the history of evolution over various time scales. Over the whole of evolution, one looks at widely different organisms, say yeast and man, for genes that are common and genes that are entirely different. Over shorter evolutionary periods one compares, say, mice and rats. And over even shorter times one can examine the differences among the sequences of human ethnic and racial groups. In terms of the evolutionary tree, one can compare branches that are close together or one can compare the root with the top of the tree. Finding out what the similarities and differences are is very interesting for the light it sheds not only on how organisms evolve but also on how they had to evolve, that is, on what functions are essential.

Another intriguing aspect of DNA that sequence data has revealed with great clarity is how little of it codes for proteins—only about 20 percent in most organisms. In human DNA, for example, one short sequence of about 300 noncoding bases is repeated about 300,000 times, and at least four or five others are repeated a comparable number of times. Some noncoding sequences are repeated tens of thousands of times, some thousands of times, and some hundreds of times. The sequences are repeated not exactly but with a very high degree of fidelity. What is all that DNA doing? Some of it may be reproducing itself because it can, parasitically so to speak. And clearly some of it must be involved in controlling gene expression and, thereby, morphogenesis—the transformation of a single fertilized egg cell to a complex three-dimensional organism containing an enormous number of specialized cells.

I should point out that bacteria seem to be much more economical—only about 10 percent of their DNA does not code for proteins. Perhaps their high rate of reproduction precludes wasting energy by replicating nonessentials. But most of the DNA in all higher organisms does not code for protein. An extreme example is a certain crab, 95 percent of whose DNA consists solely of repetitions of the sequence AT.

Questions remain even about the DNA that does code for proteins: human DNA probably contains codes for about 100,000 different proteins and yet less than 1000 proteins have actually been identified. Many of the coded proteins are probably produced in very small numbers, a few molecules per cell. It is likely that different kinds of cells produce quite different arrays of these minor proteins, as is true for many proteins that are abundant enough that one can tell. We would like to know under what circumstances the minor proteins are synthesized and what their functions are. Undoubtedly, in a general sense they serve as control devices.

A most unexpected fact learned from sequence data is that DNA undergoes much more change in the course of development of an organism and in short-term evolution than anyone had anticipated. Classical genetics had established that genetic traits are very stable, changing only occasionally from one generation to another. One would expect this stability to be reflected on the molecular level. But that seems not always to be the case. It is clear that pieces of DNA move about from one part of the genome to another. Viruses may be responsible for some of this dynamism. It is certainly true that bacteria pass DNA around from one to another and indeed from one species to another. Another example of the fluidity of DNA is the difference between the DNA in those white blood cells that produce antibodies and the DNA in other cells of the same organism. In the course of becoming antibody-producing cells, they snip and splice portions of their DNA to form the code for the particular antibodies they produce.

I've mentioned only a few specific examples of what the sequence data offer to biology and, more broadly, to human thought. Hardly anything affects the way people think about their world more than detailed understanding of how living systems work according to the ordinary laws of physics and chemistry. My outlook is that mysticism about life is being crowded out by the greater joy of knowledge—thanks to molecular genetics and molecular biology in general. There is, after all, an immense difference between speculating about the way things *might* work and knowing how they *do* work. ■

Quantitative Comparison

SCIENCE IDEAS

by William A. Beyer, Christian Burks, and Walter B. Goad

Although DNA sequences are replicated and passed on to future generations with great fidelity, changes do, of course, occur. They provide mutations, the raw material for evolution, as well as causation for disease and death. Three kinds of localized change can occur: replacement of one base by another, deletion of a base, or insertion of a base. In addition, a number of adjacent bases may be simultaneously deleted or inserted. The probabilities of these various changes are not known in general, and their determination is an outstanding problem.

The idea of comparing sequences quantitatively—in this case the sequences of amino acids in proteins—goes back to 1963. Then Linus Pauling and Emile Zuckerkandl suggested the possibility of reconstructing the course of evolution by examining the relations among the sequences of hemoglobin proteins in extant vertebrate organisms. And in 1967 W. M. Fitch and E. Margoliash constructed an evolutionary tree by measuring "distances" among the cytochrome c proteins of various organisms. Unfortunately, some of the distances in the tree were negative! Then in 1968 Stan Ulam, in conversation with Temple Smith, both at the University of Colorado, suggested that the relatedness of two sequences be measured by use of a distance that fulfills the criteria of a metric: a binary relation that is real-valued, positive-definite, symmetric, and satisfies the triangle inequality. In terms of the changes that occur in the evolution of protein or nucleic acid sequences, these properties of a metric make biological sense, excepting perhaps the symmetry property. This distance between two sequences was defined as the minimum total of localized changes—replacements, insertions, and deletions—that would transform one sequence into the other.

Another measure of relatedness of sequences is called similarity. The properties of similarity have never been made precise. Presumably similarity should be a binary, positive-valued, symmetric relation and should in some unspecified sense be complementary to a metric distance. That is, a small distance should correspond to a high similarity and a large distance to a low similarity.

Now if you imagine comparing two sequences by, say, writing them on paper tapes and sliding one along relative to the other, you will quickly see that to find by trial and error the minimum number of changes—an optimal alignment of the two sequences—generally requires considerable effort. You have to be prepared to snip out a base from one tape or the other, see whether the resulting alignment is improved, and repeat this operation many times. In 1970 two biologists, Needleman and Wunsch, then at Northwestern University, devised a procedure for finding the optimal alignment (calculating the similarity) on a computer. Their method proceeds by induction, that is, by assuming that the optimal alignment of the first *n* bases of one sequence with the first *m* bases of another is constructible from the optimal alignments of shorter segments of the two sequences. The resulting algorithm requires on the order of *nm* operations.

Also in 1970 Bill Beyer of Los Alamos, Smith, and Ulam commenced work on refinements of the idea of distance between sequences and on applications of those distances to studies of evolution. They developed a mathematical theory in which biological sequences were regarded as words of finite length over a finite alphabet. (The alphabets for DNA and protein sequences consist of four bases and twenty amino acids, respectively.) Smith made use of a suggestion by Fitch that local closeness of two sequences could be detected by comparing all possible subsequences of one sequence with all possible subsequences of the other sequence and then comparing the sums of certain differences with those expected for two random sequences. Beyer developed a method for applying linear programming to the construction of evolutionary trees based on distances between contemporary protein sequences. This method, together with a metric of Smith's, was used to produce evolutionary trees based on cytochrome c sequences. Most of the computer calculations were done by Myron Stein on the MANIAC computer.

In 1974 Peter Sellers, a mathematician at Rockefeller University, after hearing a talk there by Ulam, developed a theory of metrics among sequences and an algorithm, related to a 1972 algorithm by David Sankoff of Université de Montréal, to calculate one of Ulam's metrics. (It was not until 1981 that Smith and Mike Waterman showed that, under a certain relation between similarity and distance, the Needleman-Wunsch and the Sellers algorithms are equivalent.)

The Needleman-Wunsch algorithm, and its refinements, finds the optimal overall alignment of two fixed sequences. However, one of the key discoveries of recent work in molecular genetics is the frequency and great biological importance of events in which substantial pieces of DNA are moved from one place to another in the genome of an organism or from one organism to another. To locate such DNA segments, algorithms are needed that find locally close subsequences embedded within otherwise unrelated sequences. Sellers devised one solution to this problem in 1979, and later in the same year Goad and Minoru Kanehisa and, independently, Smith and Waterman devised another that provides a more controlled "sieve." The latter finds all pairs of subse-

# of DNA Sequences

quences whose distances fall below a prescribed threshold.

When insertion and deletion of bases is allowed, any two sequences can be aligned in some way. To distinguish biologically important relationships, it becomes important to study the frequency with which subsequences of a given closeness occur in unrelated sequences—that is, by chance alone. Such a study was begun by Goad and Kanehisa in 1982 and is being continued by them. Earlier this year, Smith, Waterman, and Christian Burks completed an investigation of the statistics of close subsequences in the entire GenBank database. The results of this investigation provide an empirical basis for assessing the statistical significance of calculated similarities. However, establishing a biologically proper measure for statistical significance remains a critical problem.

The combination of the GenBank database and methods for determining similarities between sequences will provide a very useful tool to molecular biologists. For example, screening the database for similarities to a newly sequenced segment of DNA can reveal, in the case of an extremely high similarity, that the new segment has been sequenced previously in either the same or a different genetic context. High similarity here means that the two sequences being compared are almost identical over a span of greater than fifty to one hundred nucleotides. A lower, though still statistically significant similarity may indicate that the two sequences share a common functional role in living cells, despite originating in different genetic locations. The distance algorithm can also be fruitful in comparing the sequence for one strand of a DNA segment with that for its own complementary strand. High similarities in this type of comparison can be used to trace regions of potential "hairpin" structures on the RNA transcribed from the DNA. Such structures, where the RNA folds and binds to itself, are in some cases known to be the basis for recoginition by an enzyme. Kanehisa and Goad have developed an elaboration of the distance algorithm for this purpose. Self-comparison of sequences has also proved useful in catching the evidence left behind by a particular kind of experimental error, called loop-back, that often occurs during the process of biochemically determining nucleic acid sequences.

To enable and encourage searches of the entire database for similarities to a "query" sequence, Smith and Burks have worked on developing an implementation of the distance algorithm that will make such comparisons, which have not been practicable by hand or even on most computers, possible now and as the database continues to grow. The current program employs the following strategy. For every comparison of the query sequence with another sequence, the similarity score for the best local alignment of the two sequences is saved; after a run through the database, the statistically significant scores are printed out, together with the names of the corresponding sequences. This list can then guide a more focused examination of the similarity of the query sequence to others in the database. The program was written to take advantage of the vector architecture of Cray computers, and a recent run involving about 44,000 comparisons between pairs of vertebrate sequences, each several hundred nucleotides long, took 170 minutes on a Cray-1 at Los Alamos.

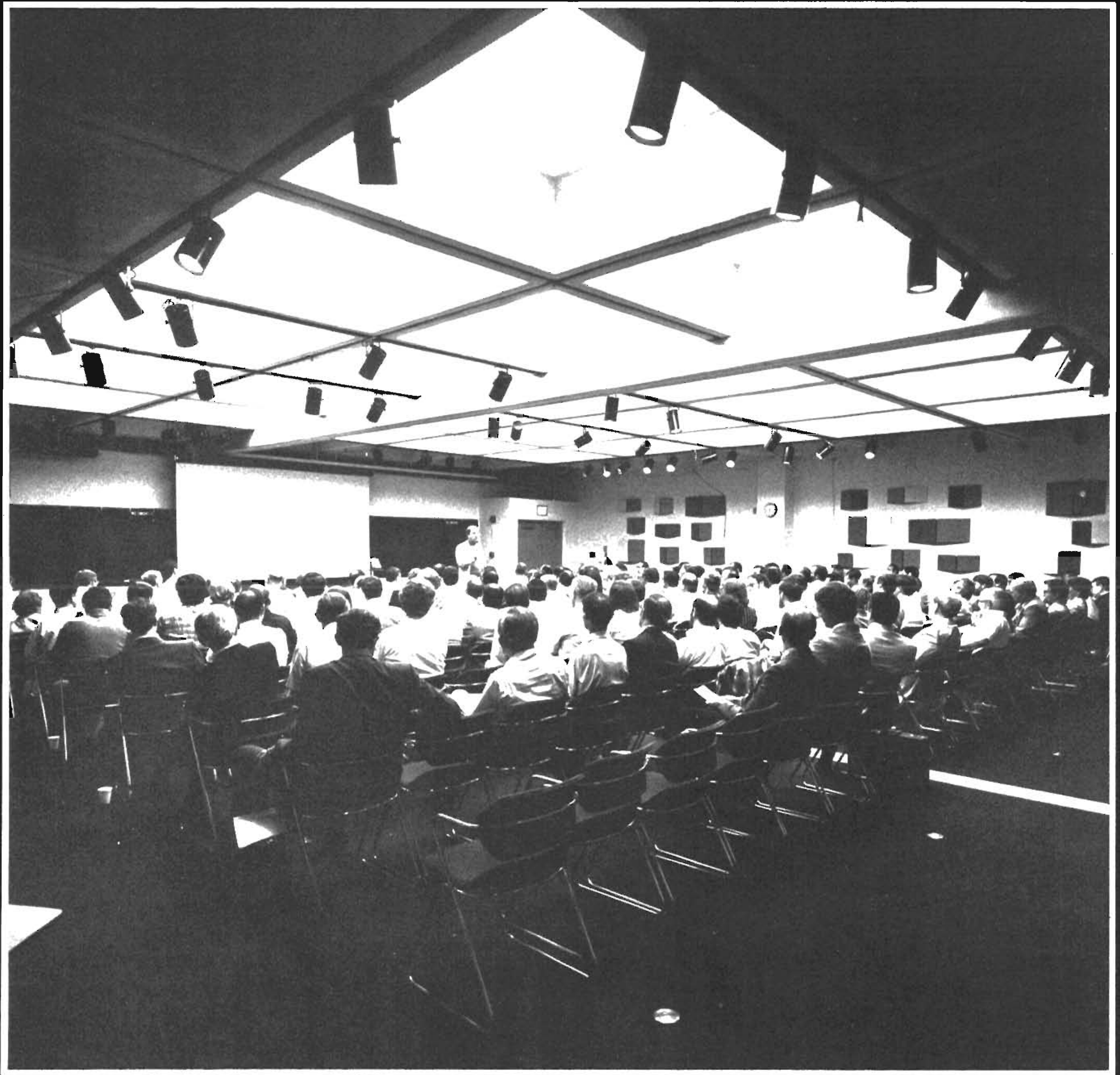
Scientists will continue to increase the speed of comparisons based on the concept of distance between sequences by developing more efficient algorithms and computer programs. For instance, Jim Fickett has developed an algorithm that, in most cases, increases the speed of the distance calculation by a factor of ten. Efforts in this direction will, of course, become more and more essential as the sequence data expand. But a more exciting direction now being explored is that of making the transition from basing the characterization of distance on the symbolic, or alphabetic, representations of sequences to basing this characterization on the physical structures of the DNA segments. An analogy with human language illustrates the need to extend the distance concept in this way.

Consider the words "leek" and "leak"; if we were comparing only the letters in this pair of homonyms, we would judge them to be almost identical. Or consider the words "sanguine" and "cheerful"; on the same basis of comparison, these synonyms would be judged quite dissimilar. Of course, in terms of the role of words in allowing communication between people, the meaning of a word is a much more appropriate criterion for comparison than the symbols for that meaning. Now consider the following nucleotide sequences:

(1) ACACAC,
(2) ACAAAC,
(3) GTGTGT.

The distance algorithm discussed above would classify (1) and (2) as quite close (only a single mismatch among the six bases) and (1) and (3) as quite distant (six mismatches). However, extrapolation from recent x-ray crystallographic studies of DNA by Dickerson and coworkers at Caltech and by Rich and coworkers at MIT indicate that although (2) is found in the right-handed B-form double-helical structure suggested by Watson and Crick, (1) and (3) are both found in radically different left-handed Z-form double-helical structures. From the point of view of the proteins in living cells that have to communicate with DNA by making chemical contact with its nucleotide strings, (1) and (3) would be almost identical sequences, both quite different from (2). Thus, current attempts to extend the distance algorithm are anticipating and incorporating a variety of spectroscopic, crystallographic, and biochemical data that identify, on the basis of structure and function, homonyms and synonyms in nucleic acid sequences.

This work is an example of the evolution of biology itself from the qualitative studies of the pre-DNA days to the mathematical, highly quantitative studies of today. ■



Frontiers of Supercomputing

by B. L. Buzbee, N. Metropolis, and D. H. Sharp

For many years scientific computing has been a major factor in maintaining U.S. leadership in many areas of science and technology and in the application of science to defense needs. Electronic computers were developed, in fact, during and after World War II to meet the need for numerical simulation in the design of nuclear weapons, aircraft, and conventional ordnance. Today, the availability of supercomputers ten thousand times faster than the first electronic devices is having a profound impact on all branches of science and engineering—from astrophysics to elementary particle physics, from fusion energy research to automobile design. The reason is clear: supercomputers extend enormously the range of problems that are effectively solvable.

Although the last forty years have seen a dramatic increase in computer performance, the number of users and the range of applications have been increasing at an even faster rate, to the point that demands for greater performance now far outstrip the improvements in hardware. Moreover, the broadened community of users is also demanding improvements in software that will permit a more comfortable interface between user and machine.

Scientific supercomputing is now at a critical juncture. While the demand for higher speed grows daily, computers as we have known them for the past twenty to thirty years are about as fast as we can make them. The growing demand can only be met by a radical change in computer architecture, a change from a single serial processor whose logical design goes back to Turing and von Neumann to an aggregation of up to a thousand parallel processors that can perform many independent operations concurrently. This radical change in hardware will necessitate improvements in programming languages and software. If made, they could significantly reduce the time needed to translate difficult problems into machine-computable form. What now takes a team of scientists two years to do may be reduced to two months of effort. When this happens, practice in many fields of science and technology will be revolutionized.

These radical changes would also have a large and rapid impact on the nation's economy and security. The skill and effectiveness with which supercomputers can be used to design new and more economical civilian aircraft will determine whether there is employment in Seattle or in a foreign city. Computer-aided design of automobiles is already playing an important role in Detroit's effort to recapture its position in the automobile market. The speed and accuracy with which information can be processed will bear importantly on the effectiveness of our national intelligence activities.

Japan and other foreign countries have already realized the significance of making this quantum jump in supercomputer development. Japan is, in fact, actively engaged in a highly integrated effort to realize it. At this juncture the United States is in danger of losing its long-held leadership in supercomputing.

Various efforts are being made in this country to develop the hardware and software necessary for the change to massively parallel computer architecture. But these efforts are only loosely coordinated.

How can the United States maintain its worldwide supremacy in supercomputing? What policies and strategies are needed to make the revolutionary step to massively parallel supercomputers? How can individual efforts in computer architecture, software, and languages be best coordinated for this purpose?

To discuss these questions, the Laboratory and the National Security Agency sponsored a conference in Los Alamos on August 15-19, 1983. Entitled "Frontiers of Supercomputing," the conference brought together leading representatives from industry, government, and universities who shared the most recent technical developments as well as their individual perspectives on the tactics needed for rapid progress.

Before presenting highlights of the conference we will review in more depth the importance of supercomputing and the trends in computer performance that form the background for the conference discussions.

The Importance of Supercomputers

The term "supercomputer" refers to the most powerful scientific computer available at a given time. The power of a computer is measured by its speed, storage capacity (memory), and precision. Today's commercially available supercomputers, the Cray-1 from Cray Research and the CYBER 205 from Control Data Corporation, have a peak speed of over one hundred million operations per second, a memory of about four million 64-bit "words," and a precision of at least 64 bits.

There are presently about seventy-five of these supercomputers in use throughout the world. They are being used at national laboratories to solve complex scientific problems in weapon design, energy research, meteorology, oceanography, and geophysics. In industry they are being used for design and simulation of very large-scale integrated circuits, design of aircraft and automobiles, and exploration for oil and minerals. To demonstrate why we need even greater speed, we will examine some of these uses in more detail.

The first step in scientific research and engineering design is to model the phenomena being studied. In most cases the phenomena are complex and are modeled by equations that are nonlinear and singular and, hence, refractory to analysis. Nevertheless, one must somehow discern what the model predicts, test whether the predictions are correct, and then (invariably) improve the model. Each of these steps is difficult, time-consuming, and expensive. Supercomputers play an important role in each step, helping to make practical what would otherwise be impractical. The demand for improved performance of supercomputers stems from our constant desire to bring new problems over the threshold of practicality.

As an example of the need for supercomputers in modeling complex phenomena, consider magnetic fusion research. Magnetic fusion requires heating and compressing a magnetically confined plasma to the extremes of temperature and density at which thermonuclear fusion will occur. During this process unstable motions of the plasma may occur that make it impossible to attain the required final conditions. In addition, state variables in the plasma may change by many orders of magnitude. Analysis of this process is possible only by means of large-scale numerical computations, and scientists working on the problem report the need for computers one hundred times faster and with larger memories than those now available.

Another example concerns computing the equation of state of a classical one-component plasma. A one-component plasma is an idealized system of one ionic species immersed in a uniform sea of electrons such that the whole system is electrically neutral. If the equation of state of this "simple" material could be computed, it would provide a framework from which to study the equations of state of more complicated substances. Before the advent of the Cray-1, the equation of state of a one-component plasma could not be computed throughout the parameter range of interest.

Using the Cray-1, scientists developed a better, more complex approximation to the interparticle potential. Improved software and very efficient algorithms made it possible for the Cray-1 to execute the new calculation at about 90 million operations per second. Even so, it took some seven hours; but, for the first time, the equation of state for a one-component plasma was calculated throughout the interesting range.

There are numerous ways in which supercomputers play a crucial role in testing models. Models of the circulation of the oceans or the atmosphere or the motion of tectonic plates cannot be tested in the laboratory, but can be simulated on a large computer. Many phenomena are difficult to investigate experimentally in a way that will not modify the behavior being studied. An example is the flow of reactants and products within an internal combustion engine. Supercomputers are currently being used to study this problem.

Testing a nuclear weapon at the Nevada test site costs several million dollars. Running a modern wind tunnel to test airfoil designs costs 150 million dollars a year. Supercomputers can explore a much larger range of designs than can actually be tested, and expensive test facilities can be reserved for the most promising designs.

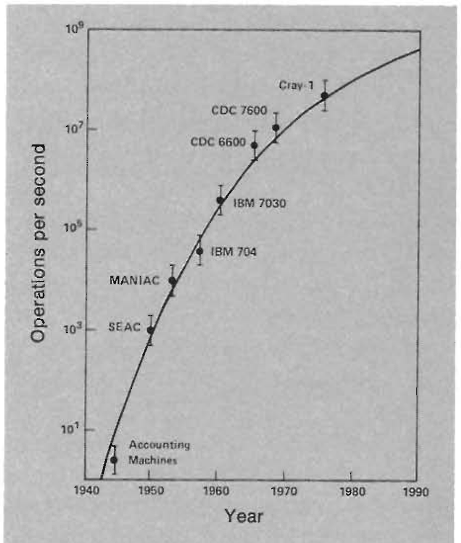
Supercomputers are needed to interpret test results. For example, in nuclear weapons tests many of the crucial physical parameters are not accessible to direct observation, and the measured signals are only indirectly related to the underlying physical processes. D. Henderson of Los Alamos characterized the situation well: "The crucial linkage between these signals, the physical processes, and the device design parameters is available only through simulation."

Finally, supercomputers permit scientists and engineers to circumvent some real-world constraints. In some cases environmental considerations impose constraints. Clearly, the safety of nuclear reactors is not well suited to experimental study. But with powerful computers and reliable models, scientists can simulate reactor accidents, either minor or catastrophic, without endangering the environment.

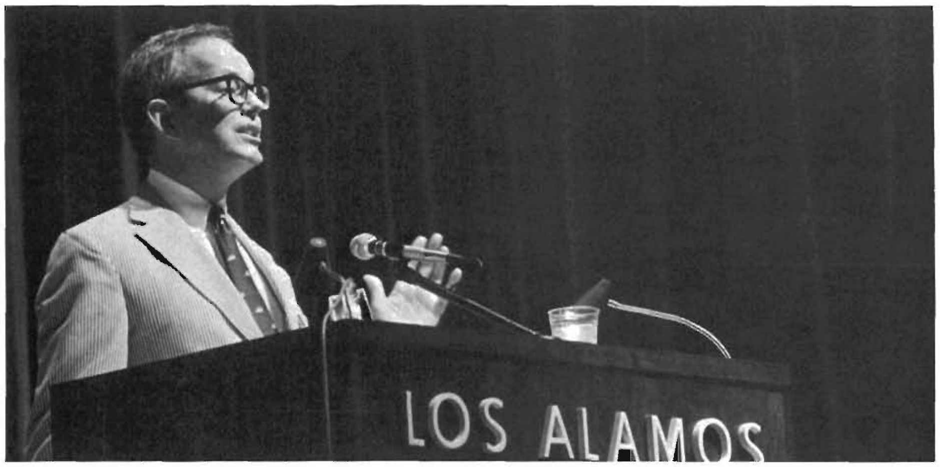
Time can also impose severe constraints on the scientist. Consider, for example, chemical reactions that take place within microseconds. One of the advantages of large-scale numerical simulation is that the scientist using it can "slow the clock" and, with graphic display, observe the associated phenomena in slow motion. At the other end of the spectrum are processes that take years or decades to complete. Here numerical simulation can provide an accelerated picture that may help, for example, in assessing the long-term effects of increasing the percentage of carbon dioxide in our atmosphere.

Trends in Performance and Architecture

Most scientists engaged in solving the complex problems outlined above feel that an increase of speed of *at least* two orders of magnitude is required to make significant progress. The accompanying figure shows that the increase in speed, while very rapid at the start, now appears to be leveling off.



The rapid growth has been due primarily to advances in microelectronics. First came the switch from cumbersome and capricious vacuum tubes to small and reliable semiconductor transistors. Then in 1958 Jack Kilby invented a method for fabricating many transistors on a single silicon chip a fraction of an inch on a side, the so-called integrated circuit. In the early 1960s computer switching circuits were made of chips each containing about a dozen transistors. This number increased to several thousand (me-



Admiral B. R. Inman in his keynote address stressed that progress in the information industry will be a critical factor in meeting our security and economic needs.

dium-scale integration) in the early 1970s and to several hundred thousand (very large-scale integration, or VLSI) in the early 1980s. Furthermore, since 1960 the cost of transistor circuits has decreased by a factor of about 10,000.

The increased circuit density and decreased cost has had two major impacts on computer power. First, it became possible to build very large, very fast memories at a tolerable cost. Large memories are essential for complex problems and for problems involving a large data base. Second, increased circuit density reduced the time needed for each cycle of logical operations in the computer.

Until recently a major limiting factor on computer cycle time has been the gate, or switch, delays. For vacuum tubes these delays are 10^{-5} second, for single transistors 10^{-7} second, and for integrated circuits 10^{-9} second. With gate delays reduced to less than a nanosecond, cycle times are now limited by the time required for signals to propagate from one part of the machine to another. The cycle times of today's supercomputers, which contain VLSI components, are between 10 and 20 nanoseconds and are roughly proportional to the linear dimensions of the computer, that is, to the length of the longest wire in the machine.

The figure summarizes the history of computer operation, and the data have been extrapolated into the future by approximation with a modified Gompertz curve. The asymptote to that curve, which represents an upper limit on the speed of a single-processor machine, is about three billion operations per second. Is this an accurate forecast in view of forthcoming developments in integrated circuit technology? The technology with the greatest potential for high speed is Josephson-junction technology. However, it is estimated that a supercomputer built with that technology would have a speed of *at most* one billion operations per second, which is greater than the speed of the Cray-1 or the CYBER 205 by only a factor of ten.

Thus, supercomputers appear to be close to the performance maximum based on our experience with single-processor machines. If we are to achieve an increase in speed of two orders of magnitude or more, we must look to machines with multiple processors arranged in parallel architectures, that is, to machines that perform many operations concurrently.

Three types of parallel architecture hold promise of providing the needed hundredfold increase in performance:

- lockstep vector processors.
- tightly coupled parallel processors, and
- massively parallel devices.

The first type may be the least promising. It has been shown that achieving maximum performance from a vector processor requires vectorizing at least 90 percent of the operations involved, but a decade of experience with vector processors has revealed that achieving this level of vectorization is difficult.

The second type of architecture employs tightly coupled systems of a few high-performance processors. In principle, collaboration of these processors on a common task can produce the desired hundredfold increase in speed. But important architectural issues, such as the communication geometry between processors and memories, remain unresolved. Analysis of the increase in speed possible from parallel processing shows that the research challenge is to find algorithms, languages, and architecture that, when used as a system, allow a large percentage of work to be processed in parallel with only a minimum number of additional instructions. (See "The Efficiency of Parallel Processing.") However, formulating algorithms for this second type is somewhat easier than for lockstep vetor processors.

Recent work on tightly coupled parallel processors has concentrated on systems with two to four vector processors sharing a large memory. Such machines have been used successfully for parallel processing of scientific computation. Logically, the next steps are systems with eight, sixteen, even sixty-four processors; however, scientists may not be able to find sufficient concurrent tasks to achieve high parallelization with sixty-four processors. The problem lies in the granularity of the task, that is, the size of the pieces into which the problem can be broken. To achieve high performance on a given processor, granularity should be large. However, to provide a sufficient number of concurrent tasks to keep a large number of processors busy, granularity will have to decrease, and high performance may be lost.

The situation is even more challenging when we consider a massively parallel system with thousands of processors communicating with thousands of memories. In general, scientists cannot find and manage parallelism for such large numbers of processors. Rather, the software must find it, map it onto the architecture, and manage it. Therein lies a formidable research issue. In fact, the issues of concurrency are so great that some scientists are suggesting that we will have to forego the familiar ordering of computation intrinsic in sequential processing. This has revolutionary implications for algorithms.

In summary, the architecture of supercomputers is likely to undergo fundamental changes within the next few years, and these changes may affect many aspects of large-scale computation. To make this transition successfully, a substantial amount of basic research and development will be needed.

Robert S. Cooper
DARPA

Sidney Fernbach
Control Data Corporation

Jacob T. Schwartz
Courant Institute

Conference Highlights

National and Industry Perspectives. In his opening comments, Laboratory Director Donald Kerr characterized the present state of affairs as presenting challenges on two frontiers—the intellectual frontier concerned with scientific and technological progress and a leadership frontier concerned with national policies and strategies aimed at maintaining U.S. leadership in supercomputing.

U.S. Senator Jeff Bingaman (New Mexico) raised several questions that were a focus of attention during the conference. How should our overall national effort in supercomputing be coordinated? What is the proper role of government? Are recent initiatives such as the formation of SRC and MCC and the DARPA project sufficient? (SRC, the Semiconductor Research Corporation, and MCC, the Microelectronics and Computer Technology Corporation, are nonprofit research cooperatives drawn from private industry. DARPA, the Pentagon's Defense Advanced Research Projects Agency, has recently inaugurated a Strategic Computing and Survivability project.) Senator Bingaman also stressed that Congress needs to be made more keenly aware of the importance of supercomputing.

In his keynote address Admiral B. R. Inman (U.S. Navy, retired, and now president of MCC) outlined challenges and opportunities facing the United States in coming decades. In the military sphere the USSR will continue to pose a formidable challenge, especially in view of their increasingly effective and mobile conventional forces. (This point was also stressed in the remarks of DARPA director R. S. Cooper and Undersecretary of Defense R. DeLauer.) Robust economic prosperity, however, offers the opportunity for progress toward world stability. Admiral Inman stressed that a critical factor in meeting our security and economic needs will be progress in the information industry: information processing, supercomputing, automation, and robotics. He cited numerous actions that are required to put the United States in a better position to seize and exploit opportunities as they arise. These include greater investments at universities for graduate training in science and mathematics, new organizations for pooling scarce talent and resources, revised governmental procedures (such as three-year authorization bills) to facilitate commitments to longer range research projects, new antitrust legislation, in part to enable the pooling of talent, and a consensus on national security policy.

Both Senator Bingaman and Admiral Inman stressed that the key to rapid progress in the supercomputing field lies in effective collaboration among the academic, industrial, and governmental sectors. Several speakers addressed the question of how the different components of this triad could best support and reinforce each other's activities.

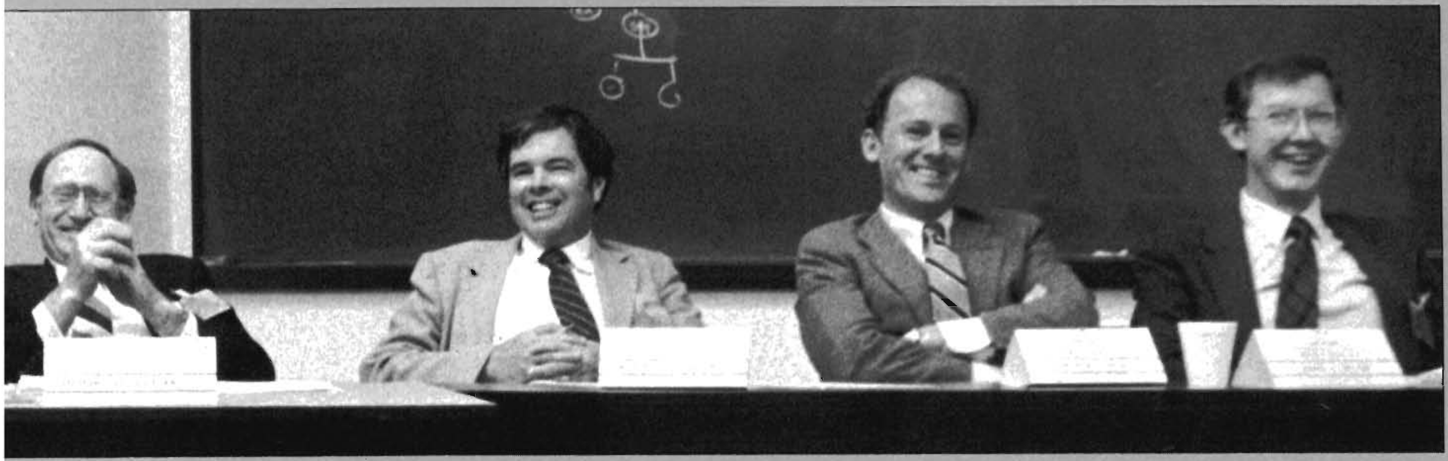
The viewpoint of the supercomputer manufacturers was presented by William Norris of Control Data Corporation and by John Rollwagen of Cray Research. Both speakers called attention to the thinness of the current supercomputer market and pointed out that a larger market is required to sustain an accelerated research and development program. Mr. Norris called for pooling of research and revisions in antitrust laws. He took the occasion to announce the formation of a new firm, ETA Systems, Inc., whose goal is to develop, for delivery by the end of 1986, a computer capable of executing ten billion floating-point operations per second.

Mr. Rollwagen called attention to the fact that the market for supercomputers might increase substantially if software were available that made them easier to use. Norris and Rollwagen also suggested that a government program to place supercomputers in universities would not only help the market but would produce other benefits as well.

This point was strongly reinforced from the university standpoint by Nobel laureate Kenneth Wilson. He stressed that the potential market for supercomputers was much greater than was generally supposed and that the key to developing this market was adequate software. Researchers at universities can make a major contribution in this area by, for example, expanding the use of modular programming for large, complex problems. In addition, he emphasized that one of the best ways to create a market for supercomputers is to train many people in their use. This can be done if the universities have supercomputers.

Wilson also pointed out that at present universities are largely isolated from the supercomputer community. For example, while supercomputer manufacturers are planning evolutionary steps in machine design, workers at universities are concentrating on the revolutionary massively parallel architectures and appropriate software and algorithms to make such systems usable. Would not each sector gain from closer interaction? How should this be accomplished?

Several speakers were concerned about the fact that highly reliable VLSI components, which form an essential element of


Richard DeLauer
dersecretary of Defense

Kenneth G. Wilson
Cornell University

John Rollwagen
Cray Research

James F. Decker
DOE

modern supercomputers, are often difficult to procure from domestic vendors. The result is that more and more of these components are being purchased from Japanese manufacturers. In responding to the suggestion that the U.S. government provide more support to the domestic semiconductor industry, DeLauer pointed out that his agency was already furnishing some help. Moreover, there are numerous domestic industries calling for aid, and available funds are limited. Hence the semiconductor industry as well as the supercomputer industry must find effective ways to leverage the support that the DOD can give.

There was a consensus at the conference on the need for clearly defined national supercomputer goals and on a strategic plan to reach them. The Federal Coordinating Committee for Science and Engineering Technology has asked the DOE to prepare such a plan. The current status of this plan was reviewed by James Decker of the DOE. The plan provides for the government to accelerate its use of supercomputers, to acquire and use experimental systems, and to encourage long-range research and development with tax incentives and increased support.

Throughout the conference the Japanese initiatives in supercomputing were much on people's minds. J. Worlton reviewed Japan's activities in the supercomputer field and assessed the strengths and weaknesses of their development strategies. The effective cooperation that they have achieved between government, industry, and academia was viewed as a substantial asset. While no one suggested that Japanese organizational methods could provide a detailed role model for the United States, a concerted effort to achieve comparable cooperation among the various sectors of the U.S. supercomputing community is certainly in order.

Scientific Developments in Architecture, Software, Algorithms, and Applications. The scientific talks presented at the conference were organized into sessions covering advanced architectures, supercomputing at Los Alamos, software, and algorithms and applications.

The trend toward parallel computing architectures was abundantly evident in presentations by computer manufacturers and academic researchers. By the end of the decade, commercially supplied systems with eight or more processors will be available. B. Smith of Denelcor presented a new industrial entry, the first attempt, *ab initio*, at modest parallelism.

Most academic projects make extensive use of VLSI, exploiting 32-bit microprocessors and 256K memory chips. Many of them can be classified as "dancehall" machines. Dancehall systems have processors aligned along one side, memories on the other, and a communication geometry to bind them together. High performance on these systems necessitates algorithms that keep all of the processors "dancing" all of the time.

James Browne of the University of Texas, Austin, observed that there are three general models of massively parallel systems based on whether decisions concerning the communication, synchronization, and granularity of parallel operations are fixed at design time, compile time, or execution time. These decisions are made at design time in fixed-net architectures. Systolic arrays containing hardware components designed to carry out specific algorithms are one example of a fixed-net architecture. Such systems provide fast execution of the algorithms for which they are designed, but other algorithms may have to be adapted to fit the architecture.

In bind-at-compile-time architectures the topology of the interconnection of processors and memory can be reconfigured to suit a particular algorithm but remains in that configuration until explicitly reconfigured. The TRAC machine at the University of Texas at Austin is an example of bind-at-compile-time architecture. Reconfigurable architectures must have more complex control systems than those that bind at design time, and the reconfiguration process does take time. The trade-off is that reconfigurable architectures can be used for many different algorithms.

Bind-at-execution-time architectures do not explicitly specify topology but, through use of control structures and shared memory, allow any topology. These systems are the most flexible since no algorithm is constrained by an inappropriate architecture. The price is the overhead required to synchronize and communicate information. Dataflow machines are an example of bind-at-execution-time systems.

Today, effective use of a multiprocessor system (even a vectorizing system) requires intimate knowledge of the hardware, the compiler's mapping of code onto the hardware, and the problem at hand. On systems with one hundred or more processors, software must free the user from these details because their complexity will defy human management. Many issues involving algorithms, models, architectures, and languages for massively parallel systems have yet to be resolved.

John Armstrong of IBM, in his discussion of VLSI technology, indicated that as components become more integrated, the need for basic research and development in materials science and packaging technology increases. Further, he believes that the current level of research in these areas is inadequate.

Several speakers noted that supercomputer systems require high-speed peripherals (in particular, disks) and that, in general, little work is being done to advance their performance. Resolution of this problem will require collaboration between manufacturers and government laboratories.

The current status of large-scale computation in nuclear weapons design (D. Henderson, Los Alamos), fluid dynamics (J. Glimm, Courant Institute), fusion reactor design (D. Nelson, DOE), aircraft and spacecraft design (W. Ballhaus, NASA-Ames Research Center), and oil reservoir simulation (G. Byrne, Exxon Research) was reviewed. An observation of general applicability was made by Glimm. Problems in all these areas are three-dimensional, time-dependent, nonlinear, and singular. Simple algorithms converge slowly when applied to such problems. Hence, most interesting problems are undercomputed. To achieve an increase in resolution by an order of magnitude in a three-dimensional, time-dependent problem requires an increase in computing power by a factor of 10,000, far beyond projected hardware improvements. Thus, to bridge the gap between needs and projected hardware capabilities, one must look to the development of better algorithms and more powerful software for their implementation.

Several new applications of supercomputing were also discussed. These included design of special-purpose, very large-scale integrated circuits (D. Rose, Bell Laboratories), robotics (J. Schwartz, Courant Institute), CAD/CAM (J. Heiney, Ford Motor Co.) and finally, an unexpected topic, the study of cooperative phenomena in human behavior (F. Harlow, Los Alamos). Also presented was an exhibition of animated computer graphics, an application whose full realization demands far more of computers than today's supercomputers can provide. Quite apart from the dazzle, animated graphics represents a key mode of interaction between the computer and the brain, which if properly exploited could progress far beyond the already impressive development.

Left to right: Admiral B. R. Inman, Senator Jeff Bingaman, and Laboratory Director Donald Kerr at the opening of the conference.

K. H. Speierman of the National Security Agency delivering the conference summary.

Conference Summary

A broad spectrum of interests and points of view was expressed during the week. The participants concurred on many general points, and these were well summarized by K. Speierman of the National Security Agency. Among them were the following critical issues.

- *Systems Approach.* All aspects of massively parallel systems—architecture, algorithms, and software—must be developed in concert.
- *VLSI Supply.* Highly reliable VLSI components are often difficult to procure from domestic vendors. Further, associated research and development in materials science and packaging technology are inadequate.
- *High-Speed Peripherals.* High-speed peripherals are essential to supercomputing systems, and very little research and development is being done in the United States in this area.
- *Future Market.* Many of the participants foresee a much larger market for supercomputers due to growing industrial use.

Nearly forty years ago the art of computing made a discontinuous leap from electromechanical plodding to crisp, electronic swiftness. Many aspects of life have been both broadened and rendered sophisticated by the advent of electronic computation. The quantum jump envisioned today is from the ultimate in serial operation to a coordinated parallel mode. ■

The proceedings of the conference are to be published by the University of California Press as a volume in the Los Alamos Series in Basic and Applied Sciences.

The Efficiency of Parallel Processing

by B. L. Buzbee

Parallel processing, or the application of several processors to a single task, is an old idea with a relatively large literature. The advent of very large-scale integrated technology has made testing the idea feasible, and the fact that single-processor systems are approaching their maximum performance level has made it necessary. We shall show, however, that successful use of parallel processing imposes stringent performance requirements on algorithms, software, and architecture.

The so-called asynchronous systems that use a few tightly coupled high-speed processors are a natural evolution from high-speed single-processor systems. Indeed, systems with two to four processors will soon be available (for example, the Cray X-MP, the Cray-2, and the Control Data System 2XX). Systems with eight to sixteen processors are likely by the early 1990s. What are the prospects of using the parallelism in such systems to achieve high speed in the execution of a single application? Early attempts with vector processing have shown that plunging forward without a precise understanding of the factors involved can lead to disastrous results. Such understanding will be even more critical for systems now contemplated that may use up to a thousand processors.

The key issue in the parallel processing of a single application is the speedup achieved, especially its dependence on the number of processors used. We define speedup (S) as the factor by which the execution time for the application changes; that is,

$$S = \frac{\text{execution time for one processor}}{\text{execution time for } p \text{ processors}} .$$

To estimate the speedup of a tightly coupled system on a single application, we use a model of parallel computation introduced by Ware. We define α as the fraction of work in the application that can be processed in parallel. Then we make a simplifying assumption of a two-state machine; that is, at any instant either all p processors are operating or only one processor is operating. If we normalize the execution time for one processor to unity, then

$$S(p,\alpha) = \frac{1}{(1-\alpha) + \alpha/p} .$$

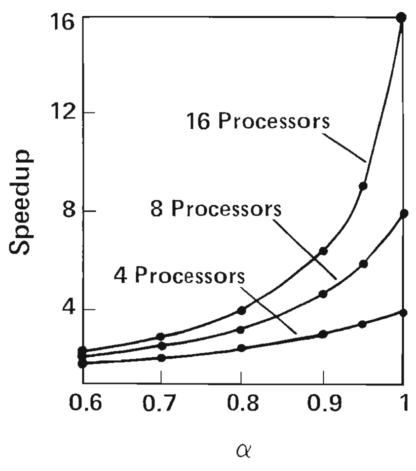
Note that the first term in the denominator is the execution time devoted to that part of the application that *cannot* be processed in parallel, and the second term is the time for that part that can be processed in parallel. How does speedup vary with α? In particular, what is this relationship for $\alpha = 1$, the ideal limit of complete parallelization? Differentiating S, we find that

$$\left.\frac{\partial S(p,\alpha)}{\partial \alpha}\right|_{\alpha=1} = p^2 - p .$$

The accompanying figure shows the Ware model of speedup as a function of α for a 4-processor, an 8-processor, and a 16-processor system. The quadratic dependence of the derivative on p results in low speedup for α less than 0.9. Consequently, to achieve *significant* speedup, we must have highly parallel algorithms. It is by no means evident that algorithms in current use on single-processor machines contain the requisite parallelism, and research will be required to find suitable replacements for those that do not. Further, the highly parallel algorithms available must be implemented with care. For example, it is not sufficient to look at just those portions of the application amenable to parallelism because α is determined by the entire application. For α close to 1, changes in those few portions less amenable to parallelism will cause small changes in α, but the quadratic behavior of the derivative will translate those small changes in α into large changes in speedup.

Those who have experience with vector processors will note a striking similarity between the Ware curves and plots of vector processor performance versus the fraction of vectorizable computation. This similarity is due to the assumption in the Ware model of a two-state machine since a vector processor can also be viewed in that manner. In one state it is a relatively slow, general-purpose machine, and in the other state it is capable of high performance on vector operations.

Ware's model is inadequate in that it assumes that the instruction stream executed on a parallel system is the same as that executed on a single processor. Seldom is this the case because multiple-processor systems usually require execution of instructions dealing with synchronization of the processes and communication between



Speedup as a function of parallelism (α) and number of processors.

processors. Further, parallel algorithms may inherently require additional instructions. To correct for this inadequacy, we add a term, $\sigma(p)$, to the execution time for parallel implementation that is at best nonnegative and usually monotonically increasing with p. Actually, σ is a function not only of p but of the algorithm, the architecture, and even of α. Let $S(p,\alpha,\sigma)$ denote speedup for this modified model. Then

$$S(p,\alpha,\sigma) = \frac{1}{(1-\alpha) + \alpha/p + \sigma(p)} .$$

If the application can be put completely in parallel form, then

$$\left. S(p,\alpha,\sigma)\right|_{\alpha=1} = \frac{p}{1 + p\sigma(p)} .$$

In other words, the maximum speedup of a real system is less than the number of processors p, and it may be significantly less. Also note that, whatever the value of α, S will have a maximum for sufficiently large p because α/p becomes insignificant while $\sigma(p)$ continues to increase.

Thus the research challenge in parallel processing involves finding algorithms, programming languages, and parallel architectures that, when used as a system, yield a large amount of work processed in parallel (large α) at the expense of a minimum number of additional instructions (small σ). ■

The HEP Parallel Processor

by James W. Moore

Although there is an abundance of concepts for parallel computing, there is a dearth of experimental data delineating their strengths and weaknesses. Consequently, for the past three years personnel in the Laboratory's Computing Division have been conducting experiments on a few parallel computing systems. The data thus far are uniformly positive in supporting the idea that parallel processing may yield substantial increments in computing power. However, the amount of data that we have been able to collect is small because the experiments had to be conducted at sites away from the Laboratory, often in "software-poor" environments.

We recently leased a Heterogeneous Element Processor (HEP) manufactured by Denelcor, Inc. of Denver, Colorado. This machine (first developed for the Army Ballistic Research Laboratories at Aberdeen) is a parallel processor suitable for general-purpose applications. We and others throughout the country will use the HEP to explore and evaluate parallel-processing techniques for applications representative of future supercomputing requirements. Only the beginning steps have been taken, and many difficulties remain to be resolved as we move from experiments that use one or two of this machine's processors to those that use many. But what are the principles of the HEP?

Parallel processing can be used separately or concurrently on two types of information: instructions and data. Much of the early parallel processing concentrated on multiple-data streams. However, computer systems such as the HEP can handle both multiple-instruction streams and multiple-data streams. These are called MIMD machines.

The HEP achieves MIMD with a system of hardware and software that is one of the most innovative architectures since the advent of electronic computing. In addition, it is remarkably easy to use with the FORTRAN language. In its maximum configuration it will be capable of executing up to 160 million instructions per second.

The Architecture

Figure 1 indicates the general architecture of the HEP. The machine consists of a number of process execution modules (PEMs), each with its own data memory bank, connected to the HEP switch. In addition, there are other processors connected to the switch, such as the operating system processor and the disk processor. Each PEM can access its own data memory bank directly, but access to most of the memory is through the switch.

In a MIMD architecture, entire programs or, more likely, pieces of programs, called *processes*, execute in parallel, that is, concurrently. Although each process has its own independent instruction stream operating on its own data stream, processes cooperate by sharing data and solving parts of the same problem in parallel. Thus, throughput can be increased by a factor of N, where N is the average number of operations executed concurrently.

The HEP implements MIMD with up to sixteen PEMs, each PEM capable of executing up to sixty-four processes concurrently. It should be noted, however, that these upper limits may not be the most efficient configuration for a given, or even for most, applications. Any number of PEMs can cooperate on a job, or each PEM may be running several unrelated jobs. All of the instruction streams and their associated data streams are held in main memory while the associated processes are active.

How is parallel processing handled within an individual PEM? This is done by "pipelining" instructions so that several are in different phases of execution at any one moment. A process is selected for execution each machine cycle, a single instruction for that process is started, and the process is made unavailable for further execution until that instruction is complete. Because most instructions require eight cycles to complete, at least eight processes must be executed concurrently in order to use a PEM fully. However, memory access instructions require substantially more than eight cycles. Thus, in practice, about twelve concurrent processes are needed for full utilization, and a single HEP PEM can be considered a "virtual" 8- to 12-processor machine. If a given application is formulated and executed using p processes (where p is an integer from 1 to 12), then execution time for the application will be inversely proportional to p.

Now what happens when individual PEMs are linked together? Each PEM has its own program memory to prevent conflicts in accessing instructions, and all PEMs are connected to the large number of other data memory banks through the HEP switch (a high-speed, packet-switched network). One result of these connections is that the number of switch nodes increases more rapidly than the number of PEMs. As one changes from a 1-PEM system toward the maximum 16-PEM configuration, the transmittal time through the HEP switch, called latency,

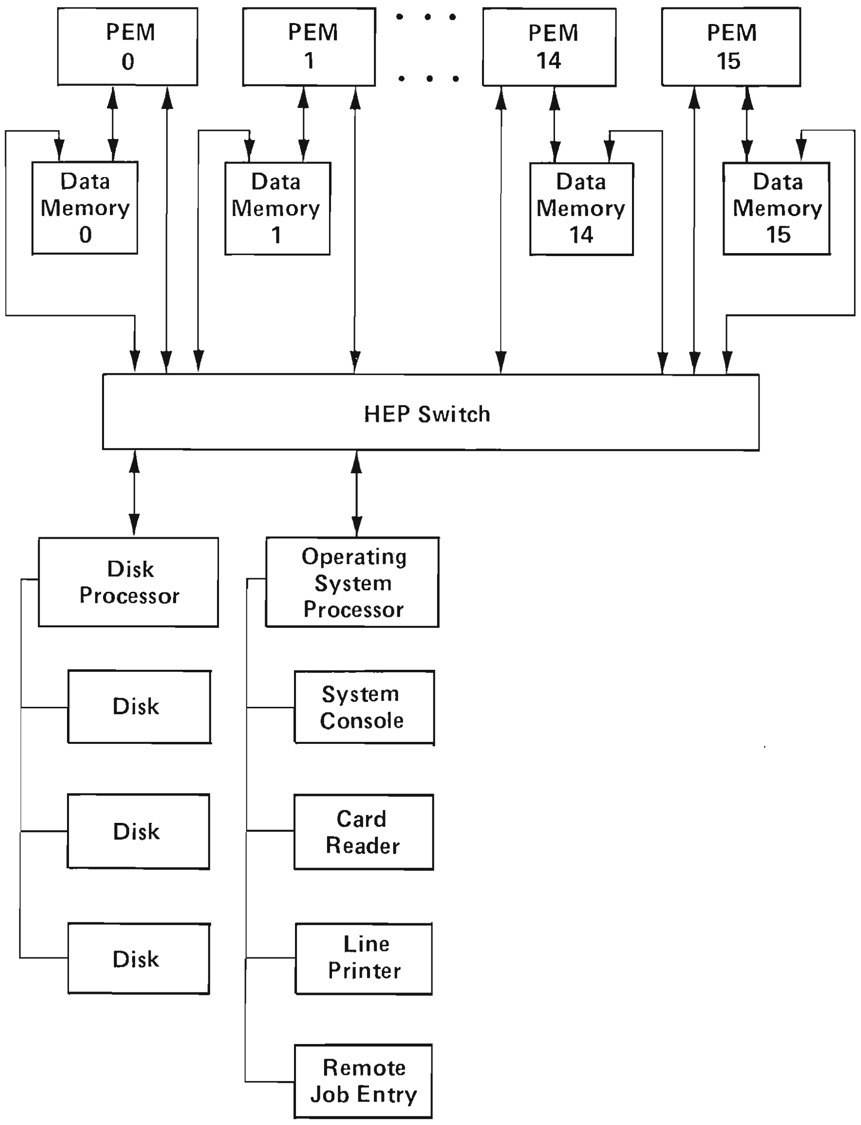


Fig. 1. The HEP parallel processor.

Asynchronous Variables — Normal FORTRAN Variable

```
      COMMON /ASYNC/ $PROCS, $FIN, $COL, MAXCOL
      MAXCOL = 100
      PURGE $PROCS, $FIN, $COL
      $COL = 1
      $PROCS = 12
```

This portion of the program sets up the variables to process an 100-column array using 12 concurrent processes. $FIN remains set at empty throughout the computations, $COL will count up through the 100 columns, and $PROCS will count down as the 12 processes die off.

```
      DO 1 I = 1, 12
      CREATE COL( arguments )
 1    CONTINUE
```

This DO loop CREATEs the 12 processes. Each process does its computations using SUBROUTINE COL.

The main program continues here.

```
      $FIN = $FIN
```

This statement, otherwise trivial, stops the main program if $FIN is not yet set to full.

```
      SUBROUTINE COL( arguments )
      COMMON /ASYNC/ $PROCS, $FIN, $COL, MAXCOL
 1    L = $COL
      $COL = L + 1
      IF ( L .GT. MAXCOL ) GO TO 2
```

Column L is processed here.

```
      GO TO 1
```

This portion gives each process a column to work on in the array and, as long as the column index L is not greater than 100, will send each completed process back for another column, regardless of the order in which the processes finish. Use of the asynchronous variable $COL in the first two statements prevents a second process from starting until the column index has been reset.

```
 2    J = $PROCS - 1
      $PROCS = J
      IF ( J .EQ. 0 ) $FIN = 1
      RETURN
      END
```

This portion terminates each process, counting down with $PROCS, and then setting $FIN to full as the last process is killed.

Fig. 2. An example of HEP FORTRAN.

quickly becomes substantial. Such latency increases the number of processes that must be running in each PEM to achieve full utilization. Although there is not enough data yet to provide good estimates on how fast latency actually increases with the number of PEMs, experience with a 2-PEM system suggests that a 4-PEM system will require about twenty concurrent processes in each PEM.

Process Synchronization

A critical issue in MIMD machines is the synchronization of processes. The HEP solves this problem in a simple and elegant manner. Each 64-bit data word has an extra bit that is set to full each time a datum is stored and is cleared to empty each time a datum is fetched. In addition, two sets of memory instructions are employed. One set is used normally throughout most of the program. This set ignores the extra bit and will fetch or store a data word regardless of whether the bit is full or empty. Data may then be used as often as required in a process.

The second set of instructions is defined through the use of asynchronous variables. Typically, this set is used only at the start or finish of a process, acting as a barrier against interference from other processes. The set will not fetch from an empty word or store into a full word. Thus, to synchronize several processes an asynchronous variable is defined that can only be accessed, using the second set of instructions, at the appropriate time by each process. A process that needs to fetch an asynchronous variable will not do so if the extra bit is empty and will not proceed until another process stores into the variable, setting it full. Because the full and empty properties of this extra bit are implemented in the HEP hardware at the user level, requiring no operating system intervention, the usual synchronization methods (semaphores, etc.) can be used, and process synchronization is very efficient.

FORTRAN Extensions to Support Parallelism

Only two extensions to standard FORTRAN are required to exploit the parallelism inherent in the HEP: process creation and asynchronous variables. Standard FORTRAN can, in fact, handle both, but the current HEP FORTRAN has extensions specifically tailored to do so.

These extensions allow the programmer to create processes in parallel as needed and then let them disappear once they are no longer needed. Also the number of PEMs being used will vary with the number of processes that are created at any given moment.

Process creation syntax is almost identical to that for calling subroutines: CALL is replaced by CREATE. However, in a normal program, once a subroutine is CALLed, the main program stops; in HEP FORTRAN, the main program may continue while a CREATEd process is being worked on. If the main program has no other work, it may CALL a subroutine and put itself on equal footing with the other processes until it exits the subroutine. A process is eliminated when it reaches the normal RETURN statement.

We can illustrate these techniques by showing how the HEP is used to process an array in which each column needs to be processed in the same manner but independently of the other columns (Fig. 2). First, one defines a subroutine that can process a single column in a sequential fashion. We could use this subroutine by creating a process for each column and then scheduling all the processes in parallel, but there is a limit on the number of processes that each PEM can handle. A better technique would be to CREATE eight to twelve processes per PEM and let the processes *self-schedule*. Each process selects a column from the array, does the computation for that column, then looks for additional columns to work on. Several asynchronous variables are the key to this technique. Each process that is not computing checks the first of these variables both to see if it can start a computation and, if so, which column is next in line. At the end of that computation and regardless of what stage any other process has reached, the process checks again to see if there are further columns to be dealt with. If not, the process is terminated. A second asynchronous variable counts down as the processes die off. When the last operating process completes its computation, a number is stored in a third, previously empty asynchronous variable, setting its extra bit to full. This altered variable is a signal to the main program that it may use the recently generated data. This method tends to smooth irregularities in process execution time arising from disparities in the amount of processing done on the individual columns and, further, does not require changes if the dimension of the array is changed.

More elegant syntactic constructs can be devised, but the HEP extensions are workable. For well-structured code, conversion to the HEP is quite easy. For more complex programs the main difficulty is verifying that the parallel processes defined are in fact independent. No tools currently exist to help with this verification.

Early Experience with the HEP

Several relatively small FORTRAN codes have been used on the HEP at Los Alamos. One such code is SIMPLE, a 2000-line, two-dimensional, Lagrangian, hydro-diffusion code. By partitioning this code into processes, about 99 per cent of it can be executed in parallel. The speedup on a 1-PEM system is close to linear for up to eleven processes and then flattens out, indicating that the PEM is fully utilized. To achieve this degree of speedup on any MIMD machine requires a very high percentage of parallel code. How difficult it will be to achieve a high percentage of parallelism on full-scale production codes is an open question, but the potential payoff is significant. ■

☆ U.S. GOVERNMENT PRINTING OFFICE: 1983 — 776-101/1007