COMPUTER RECREATIONS

Exploring the field of genetic algorithms in a primordial computer sea full of flibs

by A. K. Dewdney

Imagine an abstract sea inhabited by abstract organisms called finite living blobs, or flibs. Each flib is equipped with the simplest decision-making apparatus possible. This is the biological equivalent of what computer scientists call a finite automaton. Each flib also contains a single chromosome consisting of a string of symbols that encodes the automaton. The flibs inhabit a primordial, digital soup in constant flux. These changes must be predicted accurately by the flib if it is to survive.

In the primordial soup I recently set simmering in my computer, flibs that predicted poorly died out. The best predictors left progeny that sometimes improved on ancestral performance. Eventually a line of perfect predictors evolved.

Flibs and their evolutionary tendencies illustrate nicely a form of programming known as the genetic algorithm. Pioneered by John H. Holland of the University of Michigan in the 1960's, the technique is sometimes able to solve difficult problems by evolving a sequence of approximate solutions. New solutions are produced by mating the best of the old solutions with one another. Before long a new solution that is superior to its parents appears and joins the list of preferred breeders. Genetic algorithms have been applied with some success to pattern recognition, classifier systems, pipeline operation, symbolic layout and a small number of other problems. In my computer soup the technique yielded superior flibs. Was this success due to the general efficacy of the genetic-algorithm method or to the simplicity of the predictive task facing the flibs? The question is hard to answer. It can be pondered and the underlying phenomenon can be reproduced by any interested reader who has a computer within reach.

A finite automaton has a finite number of states; an input signal causes it to change automatically from one state to another. The kind of automaton used in a flib also generates signals. Incoming and outgoing signals are represented within the automaton by symbols. When a signal is received, the automaton changes state and emits a second signal.

A state-transition table is useful for representing the process. For example, a finite automaton that is capable of assuming three states, *A, B* and *C,* and that can handle afferent and efferent 0's and 1's fits nicely into a 3-by-4 table. For each state the automaton might find itself in, and for each symbol it might receive, there are two entries. The first entry gives the corresponding output symbol; the second entry gives the state that the automaton next assumes:

	0		1	
A	1	*B*	1	*C*
B	0	*C*	0	*B*
C	1	*A*	0	*A*

The automaton represented by this table might well find itself in state *C* at some time. If the automaton receives a 1, the table tells us the automaton will generate a 0 and enter state *A.*

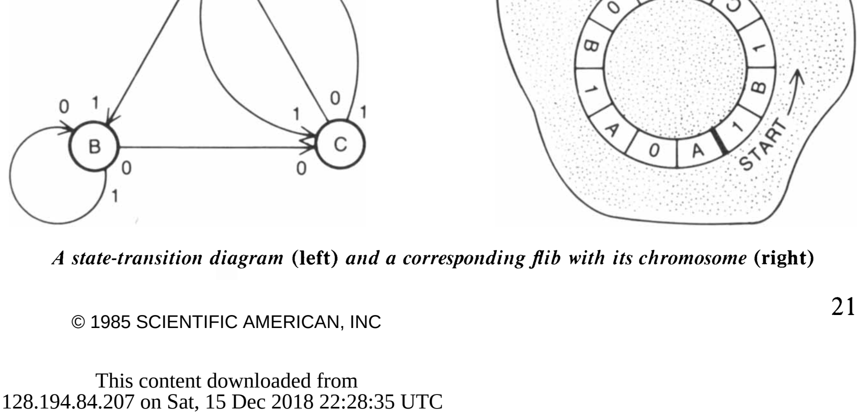
Another representation, easier for humans to read, is the state-transition diagram, in which circles represent states and arrows represent transitions. If an automaton goes from one state to another when it receives a specific symbol, an arrow should be drawn from one state circle to another. The arrow should be labeled both with the input symbol that caused the transition and with the resulting output symbol [*see illustration below*].

A finite automaton always begins its operations in a specific state, called the initial state. At each tick of an imaginary clock a new symbol arrives, a new symbol leaves, and a new state is entered. The automata used in my flibs all send and receive the same two symbols, 0 and 1.

How is one to interpret the behavior of a flib if so little is known about the creature's biology? Therein lies the joy of abstraction. The symbols received by the automaton are merely sensory messages from the environment. In corresponding fashion, an output symbol can be viewed as a response by the organism to the environment's most recent condition.

The concept of a flib is so flexible that input and output can represent a great variety of specific biological phenomena. For example, an input signal could represent a chemical or temperature gradient. The corresponding output symbol could be a command to an effector that controls cilia, or a spore-forming mechanism. A task of great importance to a creature wishing to evolve to some minimally acceptable level (say that of a university professor) is to predict the environment. To a flib the environment is a seemingly unending sequence of 0's and 1's. Insofar as symbols received indicate significant events, there is clearly some advantage in the ability of a flib to predict the next symbol, particularly if under some more specific interpretation of flib functioning the flib's survival were enhanced.

Most flibs are rather poor at predicting their environment in this sense. For example, the flib described by the



A state-transition diagram **(left)** *and a corresponding flib with its chromosome* **(right)** state-transition table given above responds to the environmental sequence

0111000010110...

with the outputs

1000011001000...

At each stage of its operation the flib's output is its prediction of the next symbol to arrive from the environment. To find the number of correct predictions shift the output sequence one symbol to the right and compare it bit by bit with the input sequence. Count the number of matching symbols. In this case the flib predicted correctly only six of the 12 incoming symbols, a score that is no better than might result from random guessing.

One can easily demand too much from a finite automaton. Indeed, it is unfair to ask a flib to predict any nonperiodic environment. Readers might like to ponder this point for a moment. Why must a perfectly predicted sequence of input symbols consist of the same basic string endlessly repeated? For example, the 3-state flib that failed the prediction test just set for it succeeds brilliantly on the following environmental sequence:

010011010011010011...

Here the environment marches to the beat of a simple repetition, 010011.

There are several dozen 3-state flibs, but only a few of them can predict this sequence perfectly. Among flibs that have more than three states perfect predictors for a given environmental sequence are rare and become more so as the number of states increases. Predictability depends heavily on the period of the sequence: no n-state flib will ever be able to predict the sequence that results from continued repetition if the basic string of symbols is too long. There is evidently a relation between the number of states a flib can have and the largest period in a sequence that it predicts perfectly. Readers might enjoy discovering the relation for themselves. What is the longest period an n-state flib can predict?

A flib is more than a finite automaton trying to predict its environment; it has a chromosome. Flibs periodically breed (by some unknown method). An examination of the chromosome in its relation to a flib's finite automaton shows how the inherited genes determine the behavior of the offspring. Start with the state-transition table and strip away the rows, one at a time, from top to bottom. Join the rows together end to end and then join the beginning of the string to its end. The result is a circular chromosome.

Before the final joining operation, the chromosome of our 3-state exemplar appears as a string of 12 genes:

1B1C0C0B1A0A

Strictly speaking, the symbols in this string are alleles. An allele is a specific form of a gene that appears at a given locus. As such, a gene can be specified either by its name or by its locus. Thus the seventh symbol from the left controls a flib's output symbol when it is in state B and a 1 is received from the environment. The locus here is 7.

I recently set up a primordial soup containing 10 4-state flibs in my personal computer. Before 1,000 of the time units I call chronons had passed none of the original flibs was alive. All had been replaced by superior predictors. The display screen showed the highest and lowest scores attained in the current population. The lowest score fluctuated a good deal; the highest score crept slowly upward [*see illustration at left*]. Just when I was beginning to give up hope that a perfect predictor would evolve, one suddenly appeared, whereupon the highest score jumped to 100.

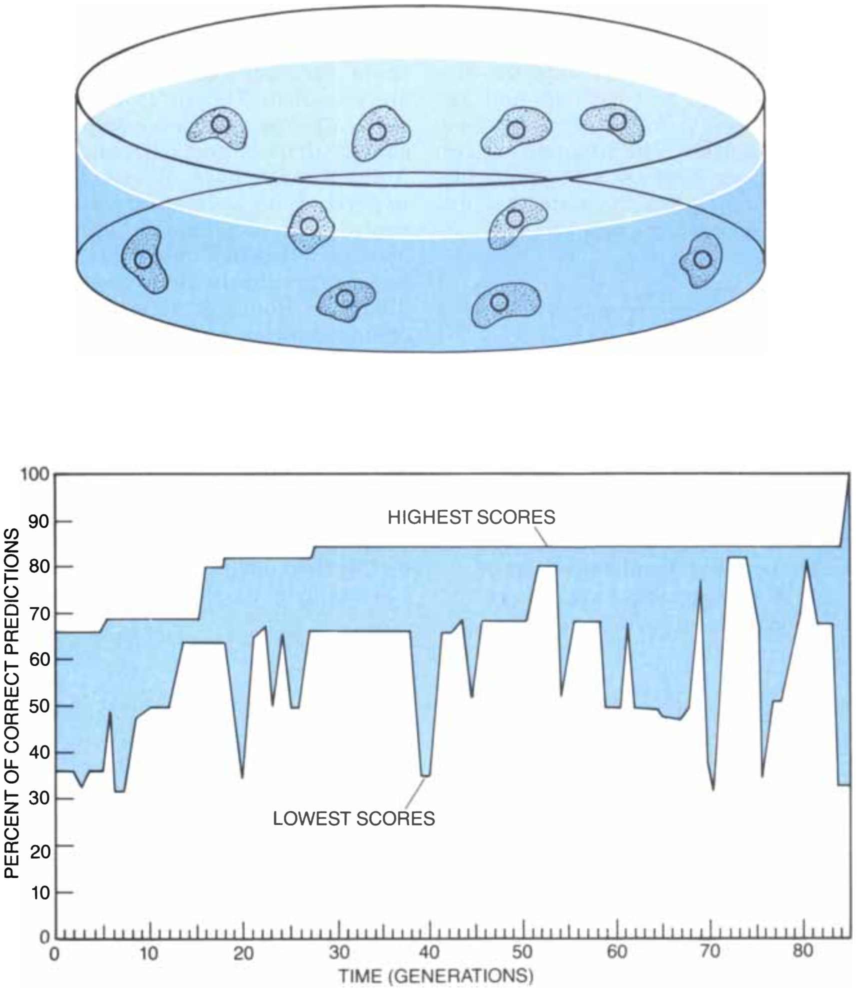
All of this raises the question of just how flibs evolve in my computer soup. Periodically a cosmic ray zips through the broth and strikes a random chromosome at a random locus; the result is that a specific gene is changed from one allele to another. For example, in the following 4-state flib chromosome the gene at locus 3 controls the output symbol for the transition from state A, when the creature receives a 1:

0D1C0D0B1A0C1B1A

A cosmic ray striking this gene changes the chromosome slightly:

0D0C0D0B1A0C1B1A

Mating is the other source of variation in the flib gene pool. During the mating season the highest-scoring flib shuffles genes with a randomly selected flib. The offspring bears a composite chromosome. One part comes from the superior parent, the other from



A soup of 10 flibs **(top)** *evolves a perfect predictor* **(bottom)**

Silent partners in world health

Schistosomiasis affects as many as 200 million people in Africa, Asia, the Middle East, Puerto Rico and Latin America. It is often called "snail fever" because at one stage of their life cycle, *Schistosoma* worms infect snails that live on the bottom of rivers and streams. These parasites invade the skin of humans who drink, wash or swim in contaminated waters. They can cause severe itching, fever, diarrhea, and eventually irreversible damage to the liver. For 16 years, researchers visiting the island of St. Lucia in the Caribbean have been testing the practicality of various methods of control. Three approaches have proven to be most effective.

First, a public health team sprayed the rivers and streams of St. Lucia to get rid of infested snails. New plumbing facilities were constructed to assure a supply of uncontaminated water. Finally, treatment of people carrying the parasite was greatly facilitated by a drug developed and supplied by Pfizer. While previous treatments had to be given by injection, this drug was given orally only once, making it much simpler to reach a large number of people. The total control and elimination of the parasite is not yet a reality, but this combined medical and environmental program has done much to make life better for the people of the island.

Developing a drug such as this is a significant task that takes a decade or more and tens of millions of dollars. It generally involves the synthesis of hundreds of compounds in the organic chemistry laboratory. These compounds are then screened for antiparasitic activity. If one or more of them shows promise, the next step is to do toxicity studies and learn all about how the potential new drugs behave in laboratory animals. Only after completion of extensive, time-consuming animal studies can the drug be tested for safety and effectiveness in humans. And clinical trials in human patients can last for several years. If the clinical trials indicate that the drug should be made available, new technology must be developed to produce it on a mass basis, and in cases like this, with little if any profitability for the developer.

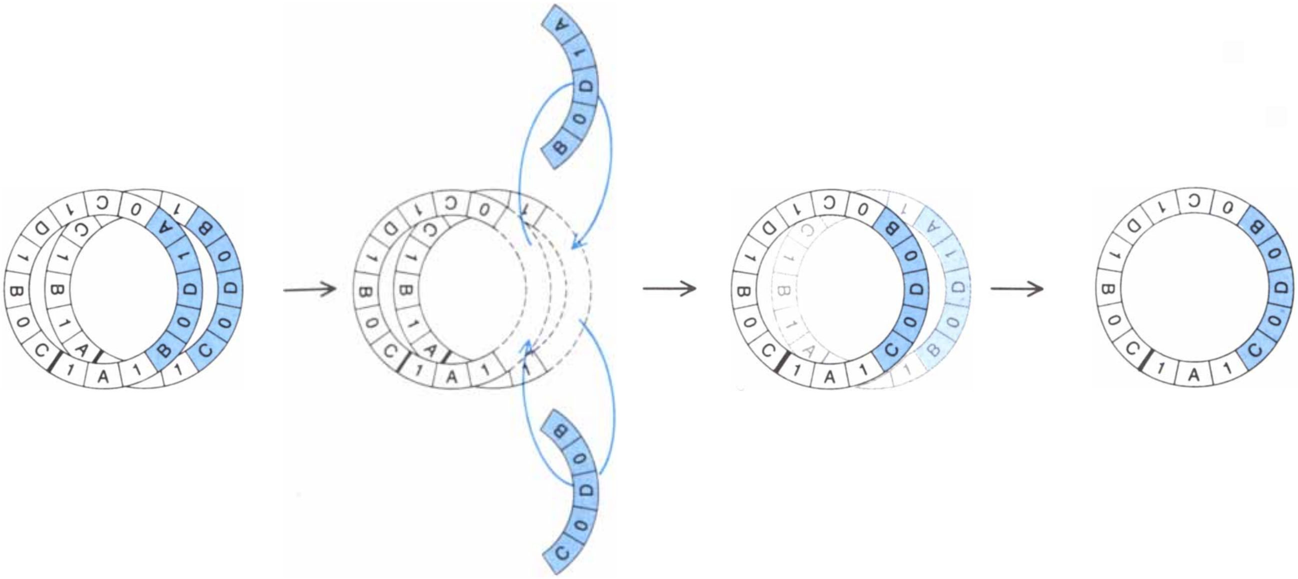
Drug research and development isn't always "good theater." And it's largely a team endeavor generally without charismatic heroes. The days of Paul Ehrlich and his "magic bullet" are long past. The work of the pharmaceutical industry isn't usually the stuff of TV documentaries. More often, the industry has been the silent partner of government agencies, physicians, nurses and their associates working together to improve public health in St. Lucia and other developing countries.

In the Third World, pharmaceuticals are perhaps even more important than in advanced industrial countries. Often they are the only form of advanced medical technology which is practicable. Other forms of care, such as surgery, are often too cumbersome and too demanding of scarce resources. Drugs, by comparison, are portable, relatively inexpensive and comparatively simple to use.

The vast majority of drugs for the Third World and also for developed countries originate in the pharmaceutical industry. The government agencies do not have the broad expertise or resources for drug development, and medical schools and universities have different missions. Only the major research/pharmaceutical companies have the necessary skills and resources. Most manufacturers of generic drugs lack the research capabilities to create new drugs and test them for safety and efficacy. And that's only one reason an economically viable research-based pharmaceutical industry is important to all of us.

Pfizer is pleased to have been a partner in helping to reduce the hazards of one of the world's more widespread health problems. Pfizer is also pleased to be working on other solutions to similar health problems around the world.

Pfizer PHARMACEUTICALS • A PARTNER IN HEALTHCARE



Crossover of two flib chromosomes, and the resulting chromosome of the offspring **(right)**

the winner of the mating lottery. The composition resembles a phenomenon called crossover that takes place in real chromosomes. In flib chromosomes crossover can be illustrated by combining the first (unaltered) chromosome listed above with another:

$$\begin{array}{c}1A1B0D1A0C1D1B0C\\ \quad\uparrow\qquad\uparrow\end{array}$$

Arrows indicate randomly selected crossover points. The offspring's chromosome is identical with that of the second parent as far as the first crossover point. Between points it is identical with the first parent's chromosome. After the second point it is again identical with the chromosome of the second parent [*see illustration above*]:

1A1C0D0B0C1D1B0C

Before actually writing and testing the primordial program, I was somewhat skeptical of the value of crossover breeding. I was surprised to find, however, that if the first parent is a reasonably good predictor, the offspring tends to be as well.

Readers may judge the issue themselves by writing a program called AUTOSOUP. Listed, the program does not extend much beyond a single page. It consists of four modules embedded in a loop. A limit that defines the top score should be set. As long as the top score is less than the limit the program should continue to run through the four modules.

In the first module the 10 flibs are scored on a sequence of 100 environmental symbols. The second module identifies the flibs with the highest and lowest scores that result. In the third module the top-scoring flib is bred with a randomly selected mate. The offspring of this union replaces the bottom-scoring flib. In the fourth and last module a cosmic ray arrives, strikes a random flib and causes a mutation. Just before the program invokes the third (breeding) module a random number is selected. If the number falls below a certain threshold, the program will skip around the breeding module and execute the mutation module immediately. The threshold can be set to any level. Certain settings, however, are better than others; if the breeding module is executed too often, the small population quickly becomes dominated by the genes of the top-scoring flib. The gene pool loses diversity and evolution slows to a painful crawl if not to a downright standstill. Evolution slows down, in any event, as the scores get higher. The top-scoring flib remains on the scene for a lengthening period because it becomes increasingly unlikely that flibs superior to it will evolve.

Four arrays are useful in AUTOSOUP. They are called *chrom, state, score* and *e. Chrom* is a two-dimensional array of 10 flibs and 16 genes. *Chrom*(*i,j*) refers to the *j*th gene of the *i*th flib. *State* and *score* contain the current state and score of the 10 flibs. The fourth array, *e,* contains the basic string used to generate environmental symbols. This array is received from the keyboard at the beginning of the program.

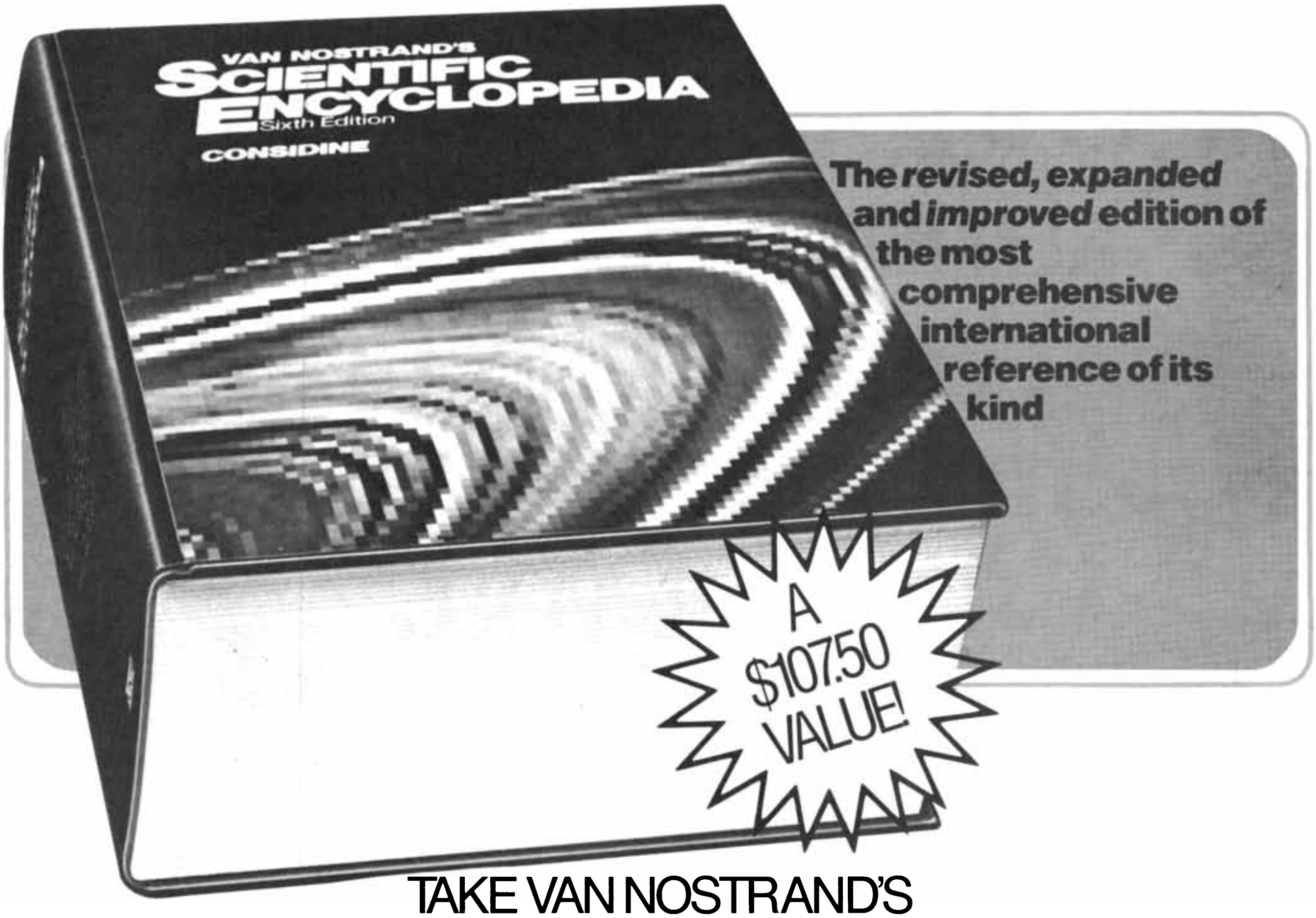
Flibs are evaluated by means of a double loop. The outer loop generates 100 environmental symbols and the inner loop increases the score of each flib if it correctly predicts the next symbol. One can test 4-state flibs adequately on environments of period 6, a challenge of intermediate difficulty. Perfect predictors may require a day's run to evolve in a period-8 environment, whereas period-4 environments are almost no challenge at all. Two tricks are useful in this module. The first trick retrieves the next environmental symbol from the outer-loop index *i* by computing *i* modulo 6; the result is the remainder of *i* on division by 6. The number can be used to index the array *e*. As *i* runs from 1 to 100 the computed index runs through the array repeatedly, producing the proper sequence of environmental symbols. Given the index of the current symbol, the next symbol is easy to compute and look up. This symbol is compared with the prediction made by each flib in turn.

The second trick enables the program to find the flib's next state quickly and determine its output merely by scanning the chromosome. Instead of representing the four states by *A, B, C* and *D,* the numbers 0, 1, 2 and 3 are used as entries in the array *state*. If the environmental symbol is called *symb,* the output of the *i*th flib can be found by first using a simple formula:

$$l = 4 \times state(i) + 2 \times symb\,.$$

Then *chrom*(*i,l*) should be identified. The locus *l* on the *i*th flib's chromosome yields its output when the creature is in state *i* and is receiving input *symb*. The next state occupies the locus $l + 1$.

The module that determines the top and bottom flibs uses an exercise common in elementary programming courses: given an array of *n* numbers, write a program that finds the largest number. The solution involves setting a variable called *top* to 0 and then scan-


VAN NOSTRAND'S
SCIENTIFIC ENCYCLOPEDIA
Sixth Edition
CONSIDINE
The *revised, expanded* and *improved* edition of the most comprehensive international reference of its kind
A $107.50 VALUE!

TAKE VAN NOSTRAND'S SCIENTIFIC ENCYCLOPEDIA for only $12.95!

When you join **The Library of Science**—*the oldest and most respected science book club in the world.* You simply agree to buy 3 more books—at handsome discounts—in the next 12 months.

For over 40 years, this unique single-volume sourcebook has been acclaimed for the breadth and depth of its knowledge, accuracy, clarity and practicality. Now its coverage has been dramatically expanded to bring you comprehensive information on *7,300 entries,* including the latest discoveries in science and technology.

And to give you "instant access" to this vital data, the format of the SCIENTIFIC ENCYCLOPEDIA has been revised so it's easier than ever before to read and cross reference. You'll learn terms and definitions; theories and their practical applications—on a vast range of topics such as:

- animal life
- astronomy
- biosciences
- chemistry
- computer science
- earth science
- energy sources
- mathematics
- medicine and anatomy
- physics
- plant sciences

This entire 3,100 page ENCYCLOPEDIA—a $107.50 value—can be yours for only $12.95.

What's *new* about the new edition of the SCIENTIFIC ENCYCLOPEDIA?

- **500 additional pages, for a total count of 3,100**
- **Comprehensive coverage of 7,300 topics**
- **2,550 photographs, diagrams and charts, plus 600 tables to give you vital details at a glance**
- **Contributions by over 200 international experts**
- **Over 2.3 million words**
- **Extra-sturdy binding for durability**

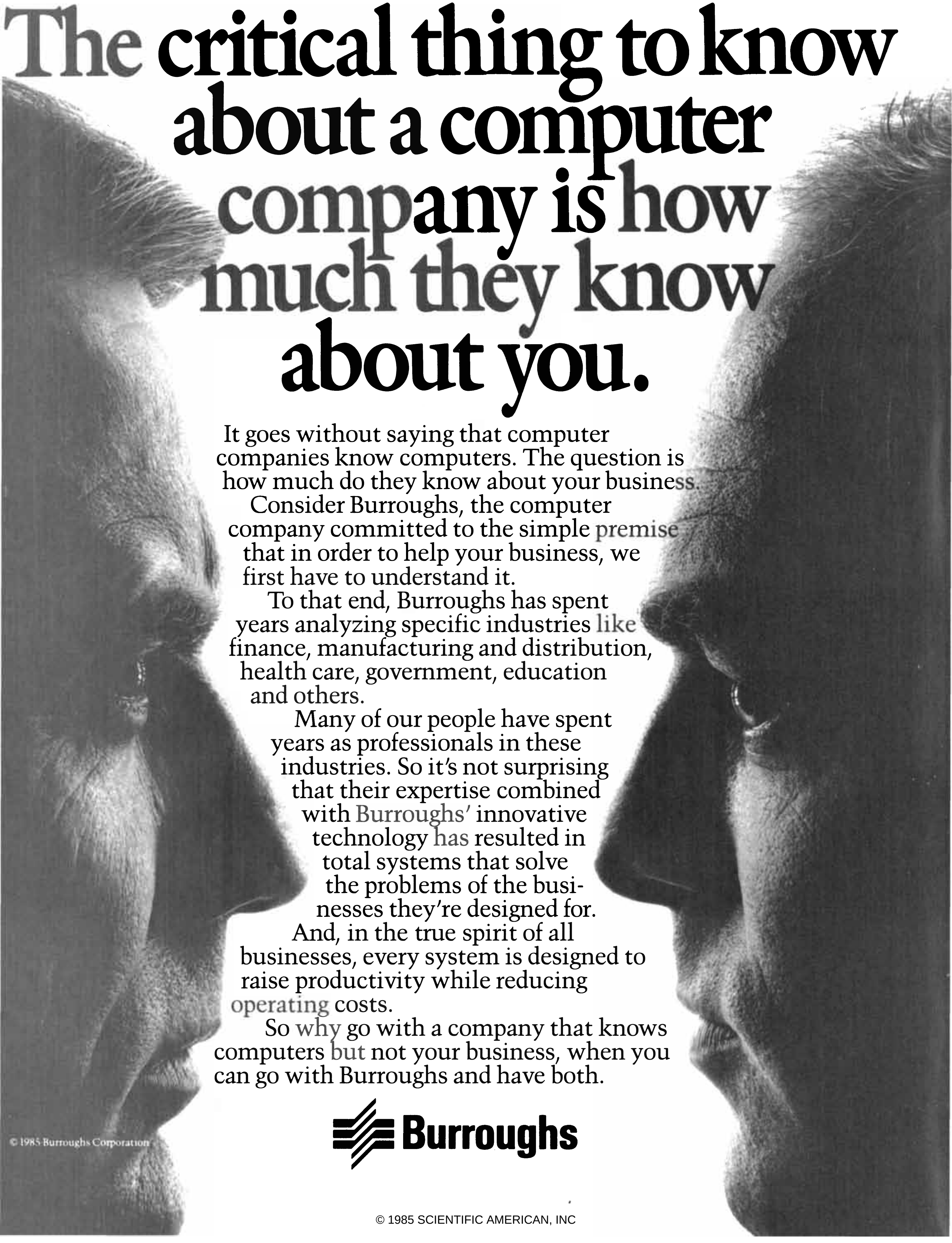
About The Library of Science . . .

The Library of Science is the nation's oldest and most respected science book club. Dedicated to serving both professionals and serious amateurs, it surveys the entire field of published sources on scientific subjects to make the very finest works available to club members at substantial savings.

MEMBERSHIP BENEFITS • In addition to getting VAN NOSTRAND'S SCIENTIFIC ENCYCLOPEDIA for only $12.95 when you join, you keep saving substantially on the books you buy—up to 30% and occasionally even more. • Also, you will immediately become eligible to participate in our Bonus Book Plan, with savings up to 70% off publishers' prices. • At 3-4 week intervals (16 times per year), you will receive The Library of Science News, describing the coming Main Selection and Alternate Selections together with a dated reply card. • If you want the Main Selection, do nothing and it will be sent to you automatically. • If you prefer another selection, or no book at all, simply indicate your choice on the card, and return it by the date specified. • You will have at least 10 days to decide. If, because of late mail delivery of the News, you should receive a book you do not want, we guarantee return postage.

Reply card removed? Please write: **The Library of Science, Dept. 2-BD6-85755, Riverside, N.J. 08075** for membership information and an application.

Scientific American 11/85



The critical thing to know about a computer company is how much they know about you.

It goes without saying that computer companies know computers. The question is how much do they know about your business.

Consider Burroughs, the computer company committed to the simple premise that in order to help your business, we first have to understand it.

To that end, Burroughs has spent years analyzing specific industries like finance, manufacturing and distribution, health care, government, education and others.

Many of our people have spent years as professionals in these industries. So it's not surprising that their expertise combined with Burroughs' innovative technology has resulted in total systems that solve the problems of the businesses they're designed for.

And, in the true spirit of all businesses, every system is designed to raise productivity while reducing operating costs.

So why go with a company that knows computers but not your business, when you can go with Burroughs and have both.

Burroughs

ning the array within a simple loop. Each entry is then compared with *top*. If it is larger than *top, top* should be replaced by that entry. The index should be saved as well. The same procedure can be inverted and used to find the bottom score. This time a variable, *bot,* should be set at 100 and replaced by entries that are smaller.

The third module breeds the top-scoring flib with a randomly selected member of the population. The only difficulty in writing this segment arises in the selection of the two crossover points. I think it is easiest to select two random integers c_1 and c_2 from the range 1 through 16. If c_1 is greater than c_2, their values should be exchanged. With just a little finesse readers will discover how three loops that range from 1 to c_1, c_1 to c_2 and c_2 to 16 supply the machinery to move elements of *chrom* on the breeding rows into the destination row, which is occupied by the doomed flib with the lowest score.

In the fourth module a random flib index and a random *locus* should be selected. The parity of *locus* determines whether a state gene or an output gene is to mutate. If the value is 0, then 1 modulo 2 should be added to the number already stored there. This flips the bit, so to speak. If the *locus* value modulo 2 is 1, then 1 modulo 4 should be added to the array entry. This changes the state stored there.

Have I cheated? Surely a systematic change of state from 0 to 1 to 2 to 3 and back again is hardly a random effect. My answer is that it is random enough: the number of states is small enough so that one cannot expect the final outcome of the program to be much different from the outcome when more randomly selected states prevail. Indeed, I also cheated in a mild way by choosing c_1 and c_2 so carelessly: the method guarantees an advantage for certain substrings in relation to others. Again, I think differences between AUTOSOUP and a statistically corrected crossover selection procedure would be slight. Either way there is so much juggling of genes and cracking of chromosomes that the top flib is hard put to recognize its own grandchildren.

The only parts of AUTOSOUP as yet unspecified are its beginning and its end. The flibs initially occupying the soup should be selected randomly. For each gene in each flib an integer should be selected from the appropriate range and assigned to that gene. Finally, when a flib first exceeds the limit set in the outer loop, AUTOSOUP should print it.

Readers embarking on this genetic adventure are warned that there is much exploring to do. Perhaps some explorers will become addicted. Questions to be answered concern the presence of evolution and its speed. When an environment period is too long for a 4-state perfect predictor to evolve, how fit do the flibs get? How do changes in the length of the period affect the amount of time it takes a perfect predictor to evolve? Nothing in the AUTOSOUP description prevents extending the program to 5- and 6-state flibs. One can even alter the program to explore nonperiodic environments or ones that occasionally change their basic string.

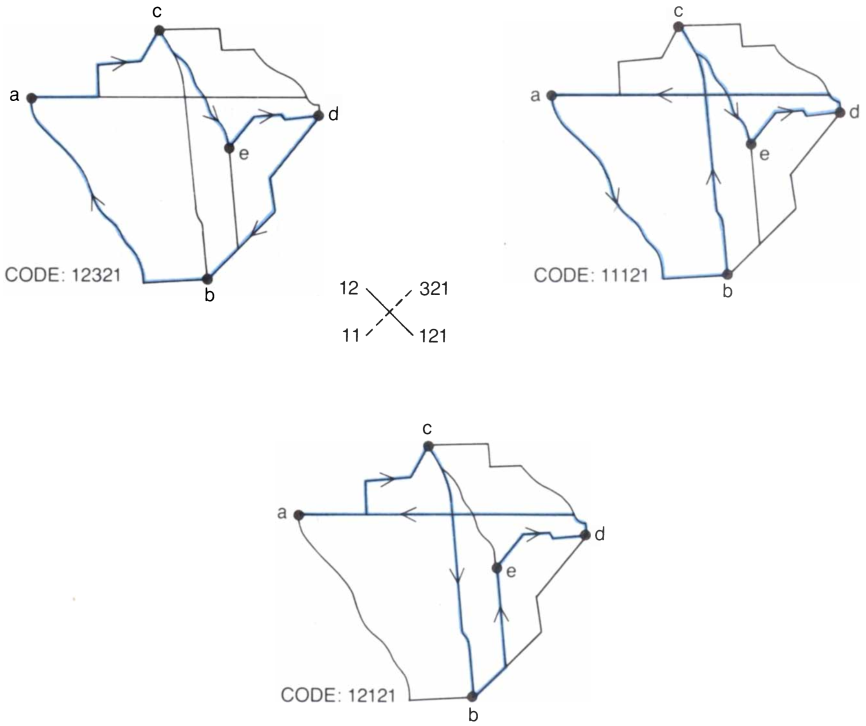
Automaton soup was inspired by a book that appeared in the early 1960's. Titled *Artificial Intelligence through Simulated Evolution,* the book describes a series of experiments in the evolution of automata by Lawrence J. Fogel, Alvin J. Owens and Michael J. Walsh. Automata were asked to predict periodic sequences and were allowed to evolve much the same way as our flibs. No breeding or crossover was allowed in this austere study, however.

It was Holland who suggested that I add the crossover facility to the automaton soup. As noted above, Holland is the acknowledged father of the genetic algorithm. Practitioners of the discipline, growing steadily in number, met at their first large-scale, funded conference, held at Carnegie-Mellon University. They discussed a wide range of theory and applications. A problem explored in several papers serves as an interesting introduction to the subject of genetic programming.

Called the traveling salesman problem, it poses the following challenge: Given a map of *n* cities connected by a network of roads, find the shortest tour of the *n* cities. Such a tour can then be used by a salesman or saleswoman to minimize travel expenses. In spite of my inclusion of salespeople of both sexes, the foregoing description has a 1940's ring to it. But more modern methods of travel and actual costs are easily incorporated without changing the mathematical skeleton implicit in the statement.

It is entirely possible that a tour of minimum length can be made to evolve just as perfect flib predictors evolved from lesser flibs. Each tour should be encoded in a chromosome. The shortest tours should be bred in the hope of obtaining yet shorter offspring. Crossover yields the chromosomes of the progeny.

It is a beguiling task to choose a good representation for a tour. For example, if one simply uses a list of the cities in the order visited, the offspring may not even be a tour. To get around this difficulty the authors of one paper, John Grefenstette, Rajeev Gopal, Brian Rosmaita and Dirk Van Gucht of Vanderbilt University, propose an ingenious chromosome. The representation for a five-city tour such as *a, c, e, d, b* turns out to be 12321. To obtain such a numerical string reference is made to some standard order for the cities, say *a, b, c, d, e*. Given a tour such as *a, c, e, d, b,* systematically remove cities from



Two parent salesman tours (**top**), *and an offspring* (**bottom**) *resulting from genetic crossover*

A THOROUGH EXPLANATION OF THE 16-VALVE SAAB 900 ENGINE.

This is for all those people who were not far-sighted enough to take Auto Shop as an elective in high school.

Where do we start? Well, let's begin with the familiar: the car you're presently driving.

The overwhelming odds are that for each cylinder your car has, it has two valves. One to let the fuel in and run the engine; the other to let the gases out after they've been burned.

Hence the rule: Make the valves bigger and you increase what's called the *volumetric* efficiency. (In plainer English, the larger the valve, the easier it is for gas to come in and exhaust to go out.)

Unfortunately, the bigger you make the valves, the farther away you move the spark plugs from the center of the combustion chamber. And that starts to wreck what's called the *thermal* efficiency.

The problem: How to increase one efficiency (the volumetric) without decreasing the other (the thermal).

Engineers had been solving that problem for years in their competition and rally cars. Instead of just making the valves bigger, they simply doubled the number of valves.

Each cylinder, then, gets two valves to ingest the fuel and two valves to throw off the exhaust.

Getting back to your car. If it has a four-cylinder engine, it probably has eight valves.

1986 SAAB 900S

Power output:	*125 HP/92 kw @ 5500 rpm*
Max. torque:	*123 ft. lbs./166 NM @ 3000 rpm*
Compression ratio:	*10.1:1*
Fuel injection:	*Bosch LH electronic*
Ignition:	*Bosch electronic with knock detector*

1986 SAAB 900 TURBO

Power output:	*160 HP/118 kw @ 5500 rpm*
Max. torque:	*188 ft. lbs./255 NM @ 3000 rpm*
Compression ratio:	*9.0:1*
Fuel injection:	*Bosch LH electronic*
Ignition:	*Bosch electronic, Hall effect*

Whereas a Saab 900S or Turbo (the descendants of rally and competition cars) has sixteen.

Back in the days of cheap gas and free glassware, none of this engineering cleverness was needed. (Well, it *was* needed; it just wasn't called for.)

All you had to do then was worry about performance (V-8 engines and the like) and throw efficiency to the wind.

Today, Saab throws nothing to the wind.

Engine weight: On a football field, 180-pound cornerbacks are faster than 290-pound defensive tackles. They have less weight to carry around. So, too, the four-cylinder, two-liter Saab engine block. Everything is attuned to keeping that weight down: The cylinder head, camshaft cover, and intake manifold are built from lightweight aluminum alloys instead of clunkier materials.

Fuel injection: The fuel is not burned mindlessly; the computer-like Bosch fuel injection always maintains the most efficient mixture of gas and air.

Even the stroke the piston makes is only this long:

3.07 inches. So hours and hours and hours and hours of high cruising speed don't create unnecessary wear and tear on the engine.

Saab offers both turbocharged and 16-valve naturally aspirated engines.

In the legendary Saab Turbo. And the not-yet-legendary new Saab 900S.

They prove that there are more ways than one to achieve what is more appropriate to poets than to engineers: the picture of driving exhilaration you see on the right.

SAAB

The most intelligent cars ever built.

Saabs range in price from $12,285 for the 900 3-door 5-speed to $18,695 for the 900 3-door 5-speed, 16-valve Turbo. Manufacturer's suggested retail prices. Not including taxes, license, freight, dealer charges or options. There are a limited number of Turbo models available with Saab's Exclusive Appointments group, which includes: leather upholstery, fog lights, and electric sunroof at additional cost.

A MORE THOROUGH EXPLANATION.

Expected.

It's come up before.

It will come up again.

And that's what's wonderful about it. Some of the best things in life are predictable, repetitive, expected. Thank heaven they are.

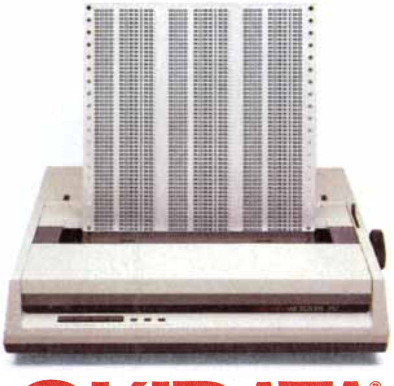
Even in business, it's nice to have some things you can count on. Things that work so dependably, so expectedly, they're completely forgettable.

Well, let's hope not completely forgettable.

When it comes to "expected" performance in printers for personal computers, the name is Okidata. Our record for reliability—for dependable, day-in, day-out performance—has set the industry standard.

Or, to put it more expectedly, we think it's the best.

Okidata printers work expectedly with IBM®, Apple®, and all other personal computers.



OKIDATA®
an OKI AMERICA company

We put business on paper.

Okidata is a registered trademark of OKI AMERICA, Inc. IBM is a registered trademark of International Business Machines Corp. Apple is a registered trademark of Apple Computer, Inc.

the standard list in the order of the given tour: remove *a*, then *c*, *e* and so on. As each city is removed from the special list, note its position just before removal: *a* is first, *c* is second, *e* is third, *d* is second and, finally, *b* is first. Hence the chromosome 12321 emerges. Interestingly, when two such chromosomes are crossed over, the result is always a tour [*see illustration on page 27*]. With such a representation tours can now be bred, so to speak, for fitness.

Most practitioners of the art of genetic algorithms readily admit that the traveling salesman problem is one of their greatest challenges. Although experiments with the representation just described were not very encouraging, there are other genetic algorithms that perform better on the problem.

Still, no genetic algorithm has ever been able to conquer the traveling salesman problem in any well-accepted sense. This is undoubtedly owing to the difficulty of the problem itself. Because it is what theorists call NP-complete, it may be doomed to eternal insolubility in the practical sense.

Since its appearance in this column last August the MANDELZOOM program has been incarnated in hundreds of homes, schools and workplaces. Although the program has apparently awed adults, intrigued teenagers and frightened a few small children, to my surprise the mail on iteration diagrams, the secondary topic, nearly equals that on the Mandelbrot set.

Many readers have lost themselves in the colored intricacies of the Mandelbrot set by zooming ever deeper. Some readers who are determined to have their own color images but who lack color-display equipment have been ordering pictures from John H. Hubbard of Cornell University. I have been told by Heinz-Otto Peitgen, the Mandelbrot explorer whose images graced the August pages, that those images too are for sale; they are included with dozens of others in a color catalogue from Design-Büro-Weisser, Lothringer Strasse 23, D-2800 Bremen 1, West Germany. Hubbard and Peitgen together learned the art of image generation from Mandelbrot.

For readers whose equipment is limited to black and white, I should have thought of shades of gray. Such pictures can be nearly as inspiring as their colored counterparts. The best gray images were produced by David W. Brooks, who works with equipment at Prime Computer, Inc., in Framingham, Mass., to compute and plot his pictures. In his fabulous and delicate riots of halftones each shade of gray is rendered by tiny black squares of a certain size; the squares are made by a laser printer. Brooks has been searching for the tiny filaments that are thought to connect miniature Mandelbrots to the main set. So far they have not appeared at any magnification used by Brooks. Mandelbrot has advised him that they are probably infinitesimal.

Those with less sophisticated equipment can still work with shades of gray on a black-and-white monitor. John B. Halleck of Salt Lake City varies the density of points per pixel to indicate different shades.

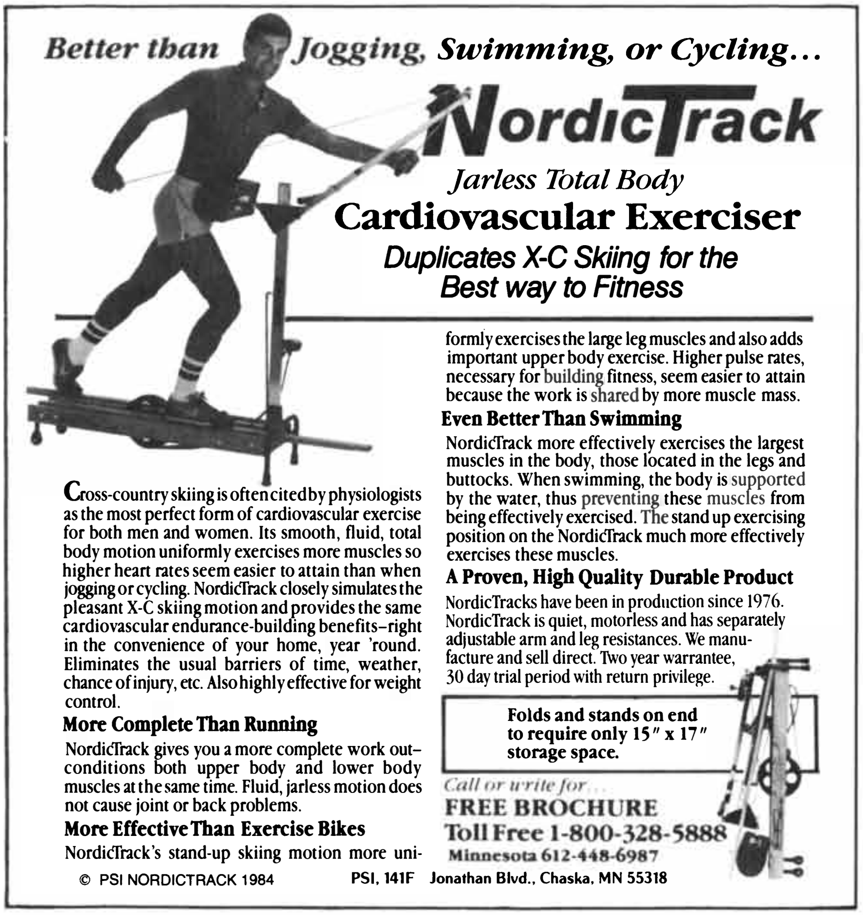
Another approach depends on black and white contours. Yekta Gursel of Cambridge, Mass., has generated views of the Mandelbrot set that rival the ones Brooks generates. Gursel replaces a discrete spectrum of colors with alternating bands of black and white. Gary J. Shannon of Grants Pass, Ore., suggested the same technique and Victor Andersen of Santa Clara, Calif., took it to an extreme. He suggested changing from black to white (or the converse) whenever the *count* variable changes from one pixel to its neighbor.

Two other explorations are worth mentioning. James A. Thigpenn IV of Pearland, Tex., uses height instead of color. The Mandelbrot set becomes an immense plateau seen from an angle, with a complicated arrangement of spiky hills approaching the plateau in various places. Richard J. Palmaccio of Fort Lauderdale dispenses with the set altogether. His interest is in tracking individual complex numbers in the course of iteration. Their choreography near the boundary can result in spiral ballets or circular jigs.

The function $z^2 + c$ gives rise to the Mandelbrot set. Naturally other functions are possible, but they produce other sets. For example, Bruce Ikenaga of Case Western Reserve University has been exploring what appears to be a cubic cactus. The function $z^3 + (c - 1)\ z - c$ produces a prickly and uncomfortable-looking set (at least in stark black and white) surrounded by mysterious miniature spiral galaxies.

There are mysteries in iteration diagrams as well: when the integers modulo *n* are squared, each number migrates to another, in effect. The iteration diagram appears when each number is replaced by a point and each migration is replaced by an arrow. I raised several questions about such diagrams. How many components do they have? Readers sent diagrams documenting their explorations for various values of *n*.

The largest diagrams were completed by Rosalind B. Marimont of Silver Spring, Md. She examined the integers



modulo 1000 and reported four pairs of components in the resulting iteration diagram. Each component sported a single attractor, as usual, and the largest attractors had 20 numbers. As a mathematician Marimont is allowed to conjecture that the integers modulo 10^k will produce $k + 1$ pairs of components and that the largest attractors will have $4 \times 5^{k-2}$ numbers.

Stephen Eberhart of Reseda, Calif., investigated the case where n is a Fermat prime (a prime number of the form $2^{2^k} + 1$). Here the number 0 forms an attractor by itself and the remaining numbers all lie in one single, grand tree. A number-theorist friend affirms that this will always be the case for Fermat primes and that the tree is binary: each internal point has two incoming arrows.

Iteration diagrams, like numbers, can be multiplied. If n is the product of two relatively prime numbers, say p and q, the iteration diagram for the integers modulo n is the product respectively of the diagrams for p and q. This interesting observation was made by Stephen C. Locke of Florida Atlantic University. Locke has also described a fascinating relation between the nth iteration diagram and a diagram of a seemingly different kind, one in which the numbers, instead of being squared, are merely doubled. When n is a prime, the latter diagram for the integers modulo $n - 1$ is the same as our nth iteration diagram, except for a single isolated number forming an attractor by itself. Much the same observation was made in number-theory terms by Noam Elkies of New York City.

A powerful tool for analyzing the (squared) iteration diagrams was developed by Frank Palmer of Chicago. Apparently all the trees attached to a given attractor are isomorphic. This means essentially that they have precisely the same form.

Finally, Bruce R. Gilson of Silver Spring, Md., and Molly W. Williams of Kalamazoo, Mich., examined a quite different generalization of the numbers from 0 through 99. These may be regarded as numbers to different bases. As numbers to the base 3, for example, one would count 00, 01, 02, 10, 11, 12, 20, 21, 22 before arriving once more at 00. Such numbers also produce iteration diagrams that look like those arising for integers modulo n. Gilson proved the diagrams always have paired components when n is even but not a multiple of 4.

There was an error in the iteration diagram presented in the August column for the integers from 0 through 99. Two arrows were missing from two components and the direction of one attractor was reversed.

Here are 147 reasons to buy at Elek-Tek, not to mention the fastest delivery anywhere.

Save 30% to 43%

off Manufacturer Suggested Retail prices on

EPSON • Okidata • Star • TOSHIBA • COMREX • OLYMPIA

HEWLETT-PACKARD

73.	**HP-71B Computer**	$ 399
74.	**HP-82400A** Series 70 card reader	125
75.	**HP-82401A** HP-IL interface	95
76.	**HP-82402A** 4K memory module	60
77.	**HP-82700A** 8K memory module	150
78.	**HP-82441A** Assembly/Forth	120
79.	**HP-82480A** HP-71 math pac	75
80.	**HP-82482A** HP-71 finance pac	60
81.	**HP-82483A** Surveying Pac	120
82.	**HP-82484A** Curve Fitting	75
83.	**HP-82485A** Text Editor	60
84.	**HP-82488A** Data Com. Pac	120

IOMEGA

BERNOULLI BOX (Modular)

1.	10 meg ½ height Drive for IBM-PC/XT/AT & compatibles	1675
2.	20 meg ½ height Drive for IBM-PC/XT/AT & compatibles	2335
3.	Non-Bootable Interface Card	104
4.	Bootable Interface Card	234
5.	10 meg cartridges for above (3 pak special)	125

EPSON®

15.	LX 80	$ 215
16.	LX 90	220
17.	RX 100+	300
18.	FX 85	340
19.	FX 185	475
20.	LQ 1500 parallel	CALL
21.	DX10 Daisy Wheel 10CPS	230
22.	DX20 Daisy Wheel 20CPS	CALL

TOSHIBA

45.	Toshiba P341	CALL
46.	Toshiba P351	CALL
47.	Toshiba P1340	430

OKIDATA

48.	Oki 182	CALL
49.	Oki 192/193	CALL
50.	Oki 84	CALL

star

51.	SG 10	$ 235
52.	SD 10/15	CALL
53.	SR 10/15	CALL

HP-ThinkJet

HP-2225
$375

85.	HP-41CV	$ 168
86.	HP-41CX	245
87.	Optical Wand	95
88.	Card Reader	145
89.	Printer	283
90.	Quad R.A.M. (for HP41C)	60
91.	Ext. Memory Module	60
92.	Ext. Function Module	60
93.	Time Module	60
94.	HP-IL Loop Module	95
95.	Digital Cassette Drive	400
96.	Printer/Plotter (HP-IL)	335
97.	**HP-9114A** Disk Drive	600

SLIMLINE Shirtpocket Styled Power Packed Programmable LCD PROBLEM SOLVERS

98.	**HP-11C** Scientific	58
99.	**HP-15C** Scientific	90
100.	**HP-12C** Financial	90
101.	**HP-16C** Programmer	90

HEWLETT PACKARD PERSONAL COMPUTATION DEALER

HARD DRIVES

Dysan CORPORATION

Internal Hard Disk Subsystems

6.	**10 MEGABYTE**	500
7.	**20 MEGABYTE**	600

Each system includes:

- Micro Science Drive
- Western Digital Controller
- 2 cables
- Documentation
- Dysan 1yr limited warranty
- For use in IBM PC, XT or most compatibles

PRINTER BUFFERS

8.	**DDP-16** 64K Par./Par.	70

QUADRAM

9.	**QRMP8** 8K Parallel/Parallel	125
10.	**QRMPS8** 8K Parallel/Serial	139
11.	**QRMSP8** 8K Serial/Parallel	139
12.	**QRMSS8** 8K Serial/Serial	139

POWER PROTECTORS

13.	**PowerMite PG600-S9** 6 outlet Surge Suppressor	30
14.	**ISOBAR 8** 8 outlet Surge Suppressor	50

IBM is a trademark of International Business Machines Corp.

PRODUCTS FOR IBM-PC

23.	**Amdek 310A** Amber Monitor	$ 150
24.	**Generic Multi** Multifunction Board, 64K	129
25.	**Generic Multi 384K** Multifunction Board, 384K	169
26.	**AST Six Pak +** Multifunction Board, 64K	225
27.	**AST Six Pak + (loaded)** Multifunction Board, 384K	279
28.	**AST Megaplus II** Multifunction Board, 64K	270
29.	**Quadram Quadboard** Multifunction Board, 0K	195
30.	**Quadram Quadboard** Multi. Board, 64K/384K	210/267
31.	**Orchid Tech.** PC Turbo 186	570
32.	**Paradise** Modular Graphics Card	290
33.	**Hercules** Monochrome Card	329
34.	**Hercules Color** Color Graphic Card	170
35.	**Novation 490605-1** 2400BPS inc. Mite Software	620
36.	**Novation 490603** ½ Card Modem 2400BPS No software	425
37.	**Novation 490603-1** As above inc. MS-DOS Software	490
38.	**Hayes 1200B** Internal modem w/software	359
39.	**AT&T 4000** 300/1200 Ext. Modem	335
40.	**Hayes 1200** External modem	399
41.	**Hayes 2400** External modem	599
42.	**US Robotics Courier 2400** Ext. 2400B Smart Modem	460
43.	**US Robotics Telpac** Telecomm. Software	75
44.	**TEAC FD55B** ½ ht. DSDD Disk Drive	90

PRODUCTS FOR APPLE

54.	**Toshiba HND 04DT** ½ Height DSDD Disk Drive	$ 70
55.	**U—Print IIc-16K** Par Interface external	85
56.	**Grappler PLUS** Graphics Interface	80
57.	**Buffered Grappler** 16K Buffer (Expand to 64K)	150
58.	**Kensington System Saver** Fan and Surge Suppr.	69
59.	**Super Cooling Fan** Fan and Surge Suppr.	29
60.	**Gamma** Disk Controller Card	48
61.	**Gamma II** SS Disk Drive for II, IIe, IIplus	125
62.	**Gamma IIc** SS Disk Drive for IIc	125
63.	**Rana ELITE 2** DSSD Disk Drive	335
64.	**Rana ELITE 3** DSDD Disk Drive	420
65.	**ERAM 80** 80 col; 64K RAM for IIe	115
66.	**Novation Apple Cat II** 300 Baud Modem	200
67.	**Novation 212** 300/1200 Baud Modem	390
68.	**Nov 49059112** Mod. & Soft. for MacIntosh	CALL
69.	**Hayes Micromodem IIe** 300 Baud Modem	145
70.	**Hayes Smartmodem IIc** 300 Baud Modem for IIc	240
71.	**Wico 501030** Analog Joystick	36
72.	**Amdek Monitors**	CALL
73.	**Zenith Monitors**	CALL

HUGE SAVINGS ON DISKETTES

	Dysan	maxell	3M	SONY	wabash	MEMOREX	BASF
3½″ SSDD	23.00	25.00	25.00	25.00	20.00	—	—
DSDD	29.00	37.00	33.00	37.00	24.00	—	—
5¼″ SSDD	17.00	14.00	14.00	—	11.50	11.50	11.00
DSDD	21.00	18.00	18.00	—	12.50	14.00	12.00
SSDD96TPI	24.00	24.00	24.00	—	—	—	—
DSDD96TPI	30.00	29.00	29.00	—	—	—	—
5¼″ DSDDHD (For IBM AT)	38.00	39.00	35.00	—	24.00	—	24.00
8″ SSDD**	22.00	29.00	25.00	—	19.00	—	—
8″ DSDD**	26.00	32.00	29.00	—	20.00	—	—

**Unformatted

Call for Quantity pricing for 10 boxes or more.

3M DATA CARTRIDGES

142.	DC100	$ 14.00
143.	DC300A	18.00
144.	DC300XL	21.00
145.	DC300XL/P	22.00
146.	DC600A	23.50
147.	DC1000	15.00

Call for Quantity pricing for 10 cartridges or more

CALL TOLL FREE 800-621-1269 EXCEPT Illinois, Alaska

ELEK-TEK, inc. 6557 N. Lincoln Ave. Chicago, IL 60645
(312) 631-7800 (312) 677-7660

Corp. Accts. Invited. Min. Ord. $15.00. Visa or MasterCard by Mail or Phone. Mail Cashier's Check, Mon. Ord., Personal Check (2 wks. to clear) Add $4.00 1st item. (AK, HI, P.R., Canada add $10.00 first item) $1.00 ea. add'l shpg. & handl. Shipments to IL address add 7% tax. Prices subj. to change. WRITE for free catalog. RETURN POLICY: Defectives Only: Most products replaced within 30 days of purchase with identical merchandise only. Computer and large peripherals replaced only when defective on arrival (within 3 work days of delivery). Other problems covered by mfr. warranty. ALL ELEK-TEK MERCHANDISE IS BRAND NEW, FIRST QUALITY AND COMPLETE. Delivery subject to availability. DUNS #06-716-0517