by Shawn Carlson

Algorithm of the Gods

I used to spend my days (and I do mean my days) hunting for supernovae. The astrophysics group I worked for dusted off an old 30-inch telescope, then used only for teaching, and converted it into a fully automated research instrument. When night arrived, this computer-controlled marvel woke up, checked its systems and set about the business of discovery. On clear nights it scanned hundreds of galaxies, searching for objects that appeared as bright stars not detected in earlier images. Sometimes it found them.

It was wonderful! And I didn't feel too guilty about not freezing all night in a remote observatory. I had toiled in cyberhell for months teaching the telescope's computer how to decide which galaxies to image and in what order it should observe them. Because large excursions from horizon to horizon sent the telescope's 40-year-old drive system into shock, it was vital that the feeble old veteran be moved as little as possible. That meant ordering the galaxies into a sequence that, totaled over the whole night, required the telescope to move through the smallest possible angle. Computer aficionados will recognize my galaxy conundrum as a variant of the classic traveling salesman problem, in which a huckster seeks an itinerary that will let him travel the smallest possible distance through a list of cities.

The problem of minimizing the telescope's motion appeared intractable. There are nearly 10^{375} different ways to sort 200 galaxies—a fairly typical evening's caseload for our telescope. (For you math types, 10^{375} is 200 factorial.) To find the one way of ordering the galaxies for a search that put the absolute least strain on the telescope, we would have had to check every possible ordering. Unfortunately, this job could not be accomplished even by all the world's supercomputers working day and night for *sextillions* of years.

For most real-world problems, a solution that comes within a few percent of the ideal one is quite acceptable. Fortunately, computer scientists have devised an algorithm that can find such solutions to many seemingly impossible problems. This amazing algorithm enables ordinary laptop computers to come up with these solutions in just a few minutes, making it a powerful addition to any amateur scientist's tool kit.

Called simulated annealing, this algorithm is a remarkable melding of human cleverness and natural efficiency. A mathematical physicist named Nicholas Metropolis and his co-workers developed the procedure back in 1953, but it has only recently come into wide use. Metropolis's procedure was later used to find useful solutions to traveling-salesman-type problems by mimicking on a computer the natural process by which the crystal lattices of glass or metal relax when heated. This process is called annealing.

Although the procedure is modeled on annealing, the relevant fundamentals are the same, and perhaps a bit more easily explained, for crystal growth. The molecules in a solution of hot sugar water wander about randomly. If the temperature drops quickly, the sugar molecules solidify into a complicated jumble. But if the temperature drops slowly, they form a highly ordered crystal that can be billions of times larger than the individual molecules.

With each molecule not only immobile but also at its lowest possible energy level in the crystal's lattice, this crystal is in its minimum energy state. (It corresponds, in my problem, to the galaxy ordering that demands the least movement from the telescope.) The system naturally progresses toward this minimum in a rather unexpected and remarkable way. The molecules are naturally distributed over a range of kinetic energies. As the temperature cools, the average kinetic energy drops. But some individual molecules remain in high-energy states, even when most are moving slow enough to bind to the crystal. When the lower-energy molecules get "hung up," they often bind in states with excess energy rather than in the lowest-energy states where they belong.

The higher-energy molecules, howev-

SLIM FILMS

GALAXY SEARCH was made possible by a heavenly algorithm.

Algorithm of the Gods Revealed

Here is a procedure for implementing simulated annealing.

```
PSEUDOCODE                               NOTES

Basic Program Algorithm
{
    Setup()                              Create initial list and set needed variables

    for (i = 1; i <= 100; i = i + 1)     Try up to 100 different temperatures
    {   numSuc = Anneal ()               Anneal list at this temperature; return the
                                             number of successful alterations made
        if(numSuc == 0) break            If there were no improvements found,
                                             we're done
        temperature = temperature *0.9   Reduce temperature by 10 percent
    }

    PrintResults()
}

Setup Algorithm (creates the initial list and sets the initial temperature)
{
    CreateInitialList (list)
    energy = GetEnergy (list)
    temperature = energy/number of items in the list
    runLimit = 100 * number of items in the list
    sucLimit = 10 * number of items in the list
}

Anneal Algorithm (anneals the list at one temperature to find the best solution)
{
    numSuc = 0                           Number of successful changes to the list

    for (I = 0; i <= runLimit; i = i + 1){   Do up to "runLimit" trials at current temperature
        GetSegment (start, end, insertPoint) Randomly select a section of the list to alter
        alteration = PickAlteration()    Randomly decide to reverse the segment or move it
        Edif = EnergyDif (alteration)    Calculate energy change if this alteration is made

        answer = Oracle (Edif, temperature)  Decide if current list will be replaced by the new list
            if (answer == YES){          If the Oracle algorithm said to alter the list ...
            AlterList ()                 ... alter the list permanently
            numSuc = numSuc + 1          Another success!
        }

        if (numSuc >= sucLimit) break    If there have been enough successes, get out of
                                             this loop so you can lower the temperature
    }
    return numSuc
}

Oracle Algorithm (decides whether to accept a proposed new list)
{
    if (Edif < 0) return YES             Always keep a new list if it has
                                             lower energy than the old list

    if (random () < exp(-Edif/temperature))  Next, use the Boltzmann factor to decide if a
        return YES                           higher energy list should be kept
                                         Note, random() returns a random number
                                             between zero and one

    return NO                            If all else fails, reject the list
}
```

er, can "knock" these trapped molecules out of these overly energetic bound states and into states of lower energy. This energy transference from molecule to molecule allows the molecules to redistribute themselves as the fluid cools, thereby nudging the growing edges of the crystal in such a way that the molecules settling into them can find the minimum energy state. This orderly crystal growth occurs only during a slow cooling because a lot of time is needed for the molecules to find the lowest-energy state.

So what does all this have to do with the traveling salesman problem? To use Metropolis's procedure, you must be able to describe your problem as a search for a sequence. In the traveling salesman problem, for example, the problem is to determine the best order in which to visit the cities on a list. With a little cleverness, a great many problems can be formulated in this way. It is also necessary for you to be able to generate a number that tells how well any given sequence works; the better the solution, the smaller this number must be. This number is analogous to the energy of the crystal in the crystal-growth example. For my supernovae search problem, the sequence was the order of galaxies searched; the quantity analogous to energy was the total angle through which the telescope had to move.

The box at the left outlines the algorithm. It works by calculating the energy levels represented by different sequences, or "paths." If a new sequence has a lower energy than the previous one, the program always adopts it. Returning to the crystal-growth example, a lower-energy path corresponds to one or more molecules finding their way into a position of lower energy in the lattice. But what about a higher-energy path? Here is where it gets interesting. The program does not immediately reject higher-energy paths; after all, the dislodging of a molecule trapped in a higher-energy state in a lattice is an example of a higher-energy path, and such occurrences are the heart of the procedure.

Yet not all higher-energy paths are likely to nudge the system into a lower-energy state. To determine which ones do, the procedure uses the so-called Boltzmann probability distribution. This factor determines the probability of finding a system, such as a molecule or a group of molecules, with a certain energy E at a given temperature T. When applied to simulated annealing, this probability turns out to be the number e (a constant, equal to about 2.7183, that comes up often in physics and other fields) raised to the power ($-\Delta E/kT$), where k is Boltzmann's constant, which relates temperature to energy, and ΔE is the energy difference between the two paths.

A clever trick allows the list to be altered without requiring the energy to be recalculated from scratch every time. The alteration begins by randomly selecting some subsection of the list. Then, either the order of the subsection's elements is reversed, or the subsection is moved and reinserted somewhere else in the list at random. In either case, the energy associated with the subsection does not change. The energy changes only where the subsection joins the rest of the list.

The box on the opposite page contains pseudocode that outlines the necessary procedures. The equivalent C code can be downloaded from the Society for Amateur Scientists's World Wide Web site.

Simulated annealing isn't the only algorithm that mimics nature to solve complex problems. So-called genetic algorithms simulate evolution at the genetic level to create cyberorganisms that are themselves solutions to research problems [see "The Amateur Scientist," July 1992]. Neural networks—which simulate the activity of neurons in the brain to store information, learn from experience and do calculations—are another field of intense research.

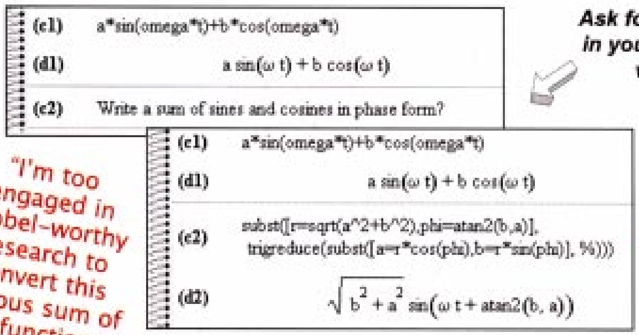
All this brings up a fascinating point. Nature is far more clever than humans will ever be. Simulating on a computer nature's methods of organizing and creating has already produced extraordinarily powerful tools to fell some of our own most vexing computing problems. Despite this, only a handful of nature's solutions have yet been duplicated on a computer. There simply have to be more out there to borrow from. The tricky part is in recognizing which systems to model and in realizing the kinds of problems that our simulations can solve.

So keep your eyes open! Perhaps, while contemplating ripples of turbulence in cigarette smoke, investigating how individual bees organize themselves into a hive or studying the simultaneous turns of a flock of birds, you will discover the next major innovation in computer problem solving. SA

To download all the C code you'll need to implement the simulated annealing algorithm, visit the Society for Amateur Scientists's World Wide Web site at http://www.thesphere.com/SAS/ For more information about other amateur scientist projects, visit the Web site or call (800) 873-8767 or (619) 239-8807.

Finally, Math Software That Understands English

Macsyma 2.2's new MathTips™ Natural Language Query revolutionizes your work. Plus, the new NumKit™ add-on speeds numerical matrix operations.


```
(c1)  a*sin(omega*t)+b*cos(omega*t)
(d1)          a sin(ω t) + b cos(ω t)
(c2)  Write a sum of sines and cosines in phase form?

(c1)  a*sin(omega*t)+b*cos(omega*t)
(d1)          a sin(ω t) + b cos(ω t)
(c2)  subst([r=sqrt(a^2+b^2),phi=atan2(b,a)],
       trigreduce(subst([a=r*cos(phi),b=r*sin(phi)], %)))
(d2)     sqrt(b^2 + a^2) sin(ω t + atan2(b, a))
```

Ask for help in your own words.

"I'm too engaged in Nobel-worthy research to convert this tedious sum of trig functions. Macsyma, take care of it."

"Like, I'm clueless about adding a bunch of functions for trig. Phasers, phases?? Whatever."

Call 1-800-macsyma for a free demo disk or visit us at http://www.macsyma.com

Macsyma Inc.
20 Academy Street
Arlington, MA 02174-6436

tel: 617-646-4550
fax: 617-646-3161
email: info@macsyma.com

Macsyma and PDEase are registered trademarks and MathTips, DataViewer, and NumKit are trademarks of Macsyma Inc. All other trademarks are the property of their respective owners.

Introducing Academic Press' Online Services

Located at www.apnet.com, APNet is the on-line home of Academic Press, where you can find information about the company, new and upcoming titles, catalogs, listings of upcoming meetings, plus IDEAL and inScight—all available in an accessible and easy-to-search format. IDEAL is Academic Press' on-line journal source—Tables of Contents and abstracts of the latest journals are available for browsing. IDEAL is located at: www.idealibrary.com/. inScight is AP's daily on-line news information service, produced in alliance with *Science*, and is devoted specifically to late-breaking science news. inScight is located at: http://www.apnet.com/inscight/

Academic Press
1-800-321-5068
Browse APNet at www.apnet.com
LK/KJ/RS 2/97 28037

Want to brush up on a foreign language?

With Audio-Forum's intermediate and advanced materials, it's easy to maintain and sharpen your foreign-language skills.

Besides intermediate and advanced audio-cassette courses–most developed for the U.S. State Department–we offer foreign-language mystery dramas, dialogs recorded in Paris, games, music, and many other helpful materials. And if you want to learn a new language, we have beginning courses for adults and for children.

We offer introductory and advanced materials in most of the world's languages: French, German, Spanish, Italian, Japanese, Mandarin, Greek, Russian, Arabic, Korean, Czech, and others.

Our 56-page *Whole World Language Catalog* offers courses in 96 languages. Call toll-free **1-888-773-2548**, fax (203) 453-9774, e-mail: 74537.550@compuserve.com, or write for your free copy. Our 25th year.

AUDIO-FORUM®
Room K310, 96 Broad Street,
Guilford, CT 06437