Although the analogy is not perfect, there is a sense in which all of the minimization algorithms thus far in this chapter correspond to rapid cooling or quenching. In all cases, we have gone greedily for the quick, nearby solution: From the starting point, go immediately downhill as far as you can go. This, as often remarked above, leads to a local, but not necessarily a global, minimum. Nature's own minimization algorithm is based on quite a different procedure. The so-called Boltzmann probability distribution,

$$\text{Prob}\,(E) \sim \exp(-E/kT) \tag{10.9.1}$$

expresses the idea that a system in thermal equilibrium at temperature $T$ has its energy probabilistically distributed among all different energy states $E$. Even at low temperature, there is a chance, albeit very small, of a system being in a high energy state. Therefore, there is a corresponding chance for the system to get out of a local energy minimum in favor of finding a better, more global, one. The quantity $k$ (Boltzmann's constant) is a constant of nature that relates temperature to energy. In other words, the system sometimes goes *uphill* as well as downhill; but the lower the temperature, the less likely is any significant uphill excursion.

In 1953, Metropolis and coworkers [5] first incorporated these kinds of principles into numerical calculations. Offered a succession of options, a simulated thermodynamic system was assumed to change its configuration from energy $E_1$ to energy $E_2$ with probability $p = \exp[-(E_2 - E_1)/kT]$. Notice that if $E_2 < E_1$, this probability is greater than unity; in such cases the change is arbitrarily assigned a probability $p = 1$, i.e., the system *always* took such an option. This general scheme, of always taking a downhill step while *sometimes* taking an uphill step, has come to be known as the Metropolis algorithm.

To make use of the Metropolis algorithm for other than thermodynamic systems, one must provide the following elements:

1. A description of possible system configurations.
2. A generator of random changes in the configuration; these changes are the "options" presented to the system.
3. An objective function $E$ (analog of energy) whose minimization is the goal of the procedure.
4. A control parameter $T$ (analog of temperature) and an *annealing schedule* which tells how it is lowered from high to low values, e.g., after how many random changes in configuration is each downward step in $T$ taken, and how large is that step. The meaning of "high" and "low" in this context, and the assignment of a schedule, may require physical insight and/or trial-and-error experiments.

## Combinatorial Minimization: The Traveling Salesman

A concrete illustration is provided by the traveling salesman problem. The proverbial seller visits $N$ cities with given positions $(x_i, y_i)$, returning finally to his or her city of origin. Each city is to be visited only once, and the route is to be made as short as possible. This problem belongs to a class known as *NP-complete* problems, whose computation time for an *exact* solution increases with $N$ as $\exp(\text{const.} \times N)$, becoming rapidly prohibitive in cost as $N$ increases. The traveling salesman problem also belongs to a class of minimization problems for which the objective function $E$

Sample page from NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (ISBN 0-521-43108-5)
Copyright (C) 1988-1992 by Cambridge University Press. Programs Copyright (C) 1988-1992 by Numerical Recipes Software.
Permission is granted for internet users to make one paper copy for their own personal use. Further reproduction, or any copying of machine-readable files (including this one) to any server computer, is strictly prohibited. To order Numerical Recipes books or CDROMs, visit website
http://www.nr.com or call 1-800-872-7423 (North America only), or send email to directcustserv@cambridge.org (outside North America).

has many local minima. In practical cases, it is often enough to be able to choose from these a minimum which, even if not absolute, cannot be significantly improved upon. The annealing method manages to achieve this, while limiting its calculations to scale as a small power of $N$.

As a problem in simulated annealing, the traveling salesman problem is handled as follows:

1. *Configuration.* The cities are numbered $i = 1 \ldots N$ and each has coordinates $(x_i, y_i)$. A configuration is a permutation of the number $1 \ldots N$, interpreted as the order in which the cities are visited.

2. *Rearrangements.* An efficient set of moves has been suggested by Lin [6]. The moves consist of two types: (a) A section of path is removed and then replaced with the same cities running in the opposite order; or (b) a section of path is removed and then replaced in between two cities on another, randomly chosen, part of the path.

3. *Objective Function.* In the simplest form of the problem, $E$ is taken just as the total length of journey,

$$E = L \equiv \sum_{i=1}^{N} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \tag{10.9.2}$$

with the convention that point $N + 1$ is identified with point 1. To illustrate the flexibility of the method, however, we can add the following additional wrinkle: Suppose that the salesman has an irrational fear of flying over the Mississippi River. In that case, we would assign each city a parameter $\mu_i$, equal to $+1$ if it is east of the Mississippi, $-1$ if it is west, and take the objective function to be

$$E = \sum_{i=1}^{N} \left[ \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} + \lambda(\mu_i - \mu_{i+1})^2 \right] \tag{10.9.3}$$

A penalty $4\lambda$ is thereby assigned to any river crossing. The algorithm now finds the shortest path that avoids crossings. The relative importance that it assigns to length of path versus river crossings is determined by our choice of $\lambda$. Figure 10.9.1 shows the results obtained. Clearly, this technique can be generalized to include many conflicting goals in the minimization.

4. *Annealing schedule.* This requires experimentation. We first generate some random rearrangements, and use them to determine the range of values of $\Delta E$ that will be encountered from move to move. Choosing a starting value for the parameter $T$ which is considerably larger than the largest $\Delta E$ normally encountered, we proceed downward in multiplicative steps each amounting to a 10 percent decrease in $T$. We hold each new value of $T$ constant for, say, $100N$ reconfigurations, or for $10N$ successful reconfigurations, whichever comes first. When efforts to reduce $E$ further become sufficiently discouraging, we stop.

The following traveling salesman program, using the Metropolis algorithm, illustrates the main aspects of the simulated annealing technique for combinatorial problems.

Sample page from NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (ISBN 0-521-43108-5)
Copyright (C) 1988-1992 by Cambridge University Press. Programs Copyright (C) 1988-1992 by Numerical Recipes Software.
Permission is granted for internet users to make one paper copy for their own personal use. Further reproduction, or any copying of machine-readable files (including this one) to any server computer, is strictly prohibited. To order Numerical Recipes books or CDROMs, visit website
http://www.nr.com or call 1-800-872-7423 (North America only), or send email to directcustserv@cambridge.org (outside North America).