Mathematical Programming 34 (1986) 111-124
North-Holland

CONVERGENCE OF AN ANNEALING ALGORITHM

M. LUNDY

Department of Pure Mathematics and Mathematical Statistics, Cambridge University, 16 Mill Lane, Cambridge CB2 1SB, UK

A. MEES

Department of Mathematics, University of Western Australia, Nedlands, Western Australia 6009

Received 14 November 1983
Revised manuscript received 7 October 1984

The annealing algorithm is a stochastic optimization method which has attracted attention because of its success with certain difficult problems, including NP-hard combinatorial problems such as the travelling salesman, Steiner trees and others. There is an appealing physical analogy for its operation, but a more formal model seems desirable. In this paper we present such a model and prove that the algorithm converges with probability arbitrarily close to 1. We also show that there are cases where convergence takes exponentially long—that is, it is no better than a deterministic method. We study how the convergence rate is affected by the form of the problem. Finally we describe a version of the algorithm that terminates in polynomial time and allows a good deal of 'practical' confidence in the solution.

Key words: Global Optimization, Stochastic Optimization, Annealing, Metropolis Method, NP, Hill Climbing, Local Improvement.

1. Introduction

The main idea used in the annealing algorithm was first described by Metropolis et al. (1953), though not in an optimization context. It was recently developed and applied to optimization problems independently by Kirkpatrick et al. (1982) and Cerny (1984). Other applications have also been reported by Schwarzschild (1982) and Giman and Giman (1984). These authors attempt to solve difficult optimization problems by what amounts to a combination of a Monte Carlo method and a deterministic descent process. This may be thought of as an attempt to get some of the speed and reliability of descent algorithms while avoiding their propensity to stick at local minima.

The idea came from an analogy with the annealing process used in making spin glasses. The physical system operates so as to minimize potential energy, which is related to the extent to which atomic spins fail to line up with one another. If the glass is held at any nonzero temperature it eventually reaches thermodynamic equilibrium in which there are typically a number of domains within which the spins are (approximately) aligned, but between which spins differ. Random kinetic fluctuations prevent a global minimum of potential energy from being attained over

any reasonable time, giving instead an average of a (local) minimum state and nearby states. In addition, the system is prevented from sticking in local minima whose basins are not very deep compared with the size of typical fluctuations. More physically, we never get exact alignment of spins, but we avoid a huge number of small domains. The global minimum—in which the whole piece of glass is a single domain—will probably never be attained, but by gradually lowering the temperature one hopes to end up with relatively few domains. The point is that at low temperatures the macroscopic form of the domains is fixed over very long times. Low temperatures are needed to get good alignment within single domains but must be approached cautiously to avoid being stuck in local minima corresponding to many domains.

This analogy was first pointed out by Kirkpatrick et al. (1982). The purpose of this paper is to formalize 'annealing' so as to allow the application of other powerful metaphors (Markov chains) that might give additional insights. Although optimization problems of almost any sort may be solved, we shall concentrate on combinatorial problems.

In our formulation, the temperature becomes a control parameter that changes as the optimization proceeds, and thermodynamic equilibrium is replaced by equilibrium of a Markov process. We will prove that the algorithm converges in that it can be made to terminate arbitrarily close to the global optimum with probability arbitrarily near one. Unfortunately the time to termination cannot be shown to be polynomially bounded for a general NP-hard problem. In fact, we give examples where the time is exponentially long, and the annealing method may actually take longer than a deterministic algorithm that simply examines all the solutions. It may be that for certain types of problem the time for stochastic convergence is polynomial, and we discuss conditions that would ensure this.

These conditions are not generally practical to check, so we describe a cautious heuristic method of adjusting the control parameter which leads to termination in polynomial time and which has behaved well in tests we have made on the travelling salesman, cluster analysis, subset selection, evolutionary trees and some problems in wood stocking (Lundy, 1984, 1985).

2. The algorithm

The problem is to minimize a function $f: X \to R$ where X is a space with a finite number $|X|$ of points. $|X|$ is exponentially large in terms of a natural measure of the size of the problem: for example, a travelling salesman problem with n cities has $|X| = (n-1)!$

So far, X has no structure at all. Now we define neighbours of the points of X: that is, we define a topology on X. In the travelling salesman case, if we regard a permutation of the integers $\{2, \ldots, n\}$ as a point in X then its neighbours might be all routes generated from this one by interchanging two city indices. The essential feature of the topology is that every point must be eventually reachable from every

other point. We shall prove that in equilibrium, the relative amounts of time spent in different states are *independent of the topology.*

At the ith step, call the current solution x_i, call the control parameter (a nonnegative number) c_i, and write $f_i = f(x_i)$. Start with an initial solution x_0, possibly selected randomly, and assume we are given values c_0, dc_i and c_f of the control parameter, obtained as discussed later.

The algorithm repeats the following steps until $c_i \leqslant c_f$:

1. Generate a neighbour x_p of the current solution x_i. This is a potential candidate for x_{i+1}.

2. Set x_{i+1} to x_p with probability $\min\{1, \exp((f_i - f_p)/c_i)\}$, and to x_i with the complementary probability.

3. Set c_{i+1} to $c_i - dc_i$ and replace i with $i+1$.

Note that we may set $dc_i = 0$; that is, we may hold the system at a fixed control value for many iterations. Also, the above choice of acceptance probability is natural but is not the only possibility, as we shall see later. The effect is that downhill steps are always accepted while uphill steps are more likely to be accepted if they are small than if they are large.

As an illustration of this effect, Fig. 2.1 shows a series of steps taken for each of two separate fixed values of c, in a two dimensional problem with two local minima. (The space has been discretised in a simple way.) A deterministic descent method will always go to the minimum in whose basin it finds itself, whereas the annealing method can climb out of one basin into the next. In the long run, the annealing method will spend a proportion of its time in each basin determined by the function values there, together with the current value of the control parameter. As the control parameter decreases, the proportion of time spent at the global optimum increases: if we run for long enough at a small enough value of c, the global optimum will consume almost all of the steps.

Because of its ability to make uphill steps, the annealing method avoids being trapped in local optima. However, it is far from apparent at first that one could not always do at least as well by running a deterministic algorithm many times, with randomly chosen starting points. The following example shows that cases exist where the annealing algorithm performs much better than such a method.

Let $X \subset R^2$ be the square centred at 0 and of side $2N$ (see Fig. 2.2a); discretise X on a grid of mesh size δ, centre 0. Let $r = \max\{|x_1|, |x_2|\}$ be the l_∞ distance of any point $x \in X$ from 0; and let $f: X \to R$, a function of r only, be as shown in Fig. 2.2b. In fact, $f(x) = r$ except when $r = n + \delta$ where n is any positive integer; in that case, $f = n - \varepsilon$ for some $\varepsilon \ll \delta$. The problem is to find the minimum of f on X, which of course lies at 0.

We describe two search techniques to do the minimization. They are a version of downhill search with random starting point, and the annealing method. To provide a realistic comparison with combinatorial problems, we assume that the downhill search takes steps of size δ and does not have access to gradient information, so it locates a local optimum in a time which is much the same as the annealing method

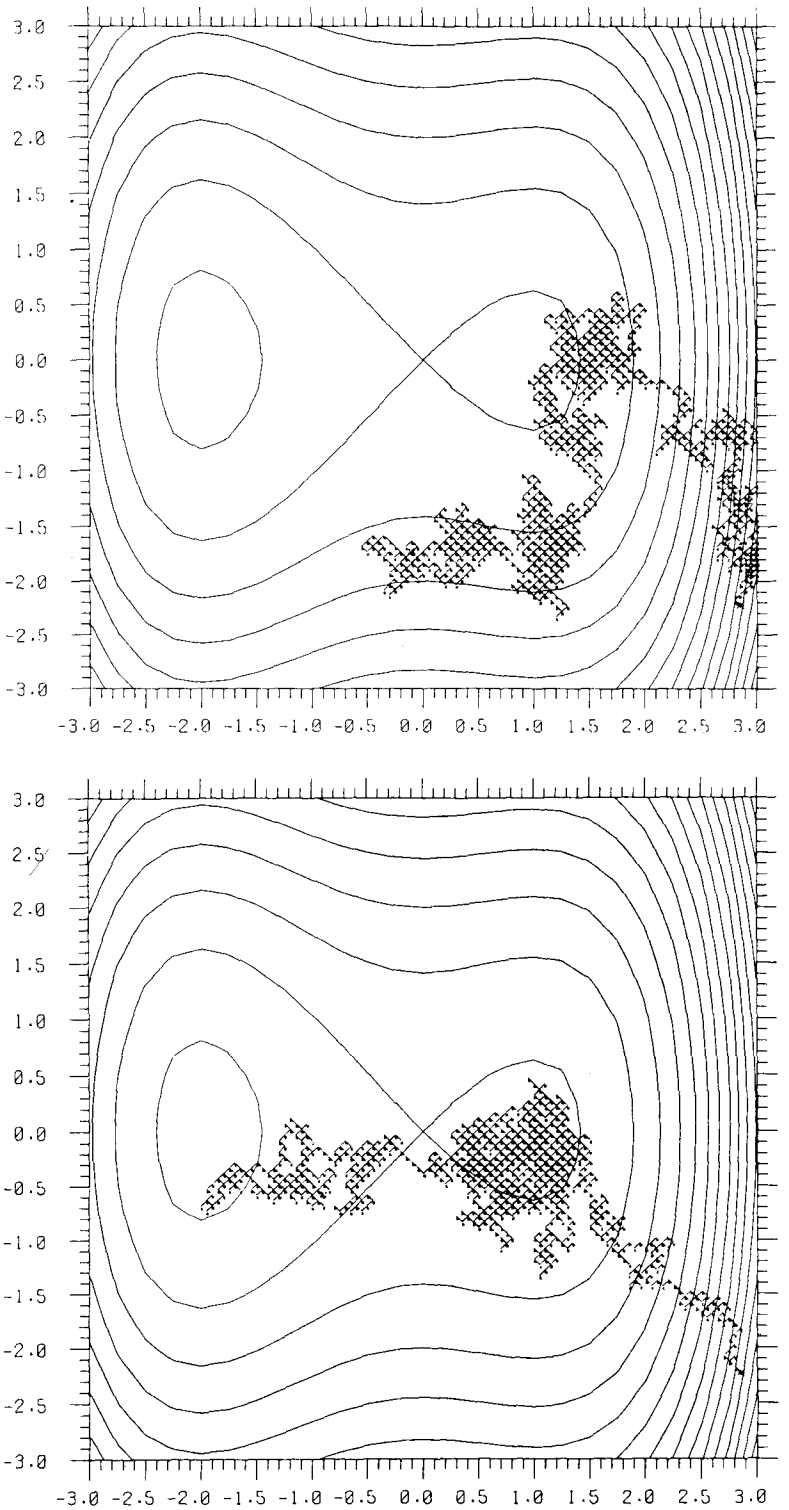
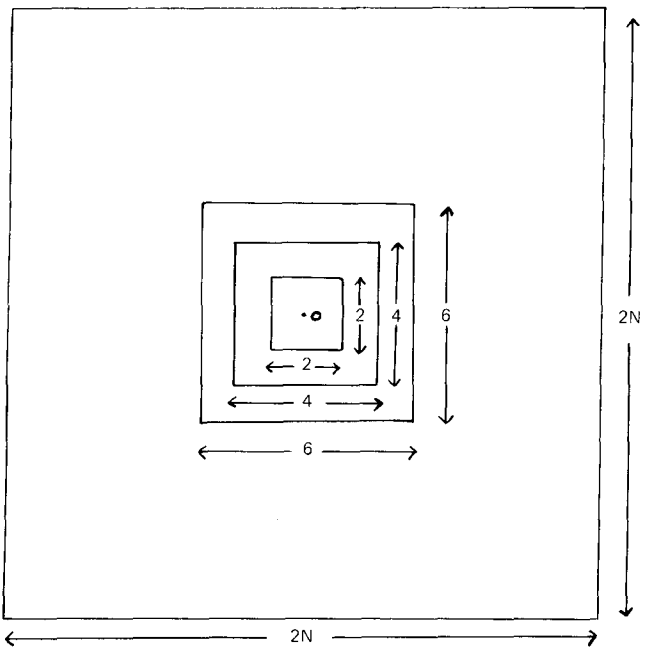


Fig. 2.1. Random walk on a contour map. (a) Larce c. (b) Small c.



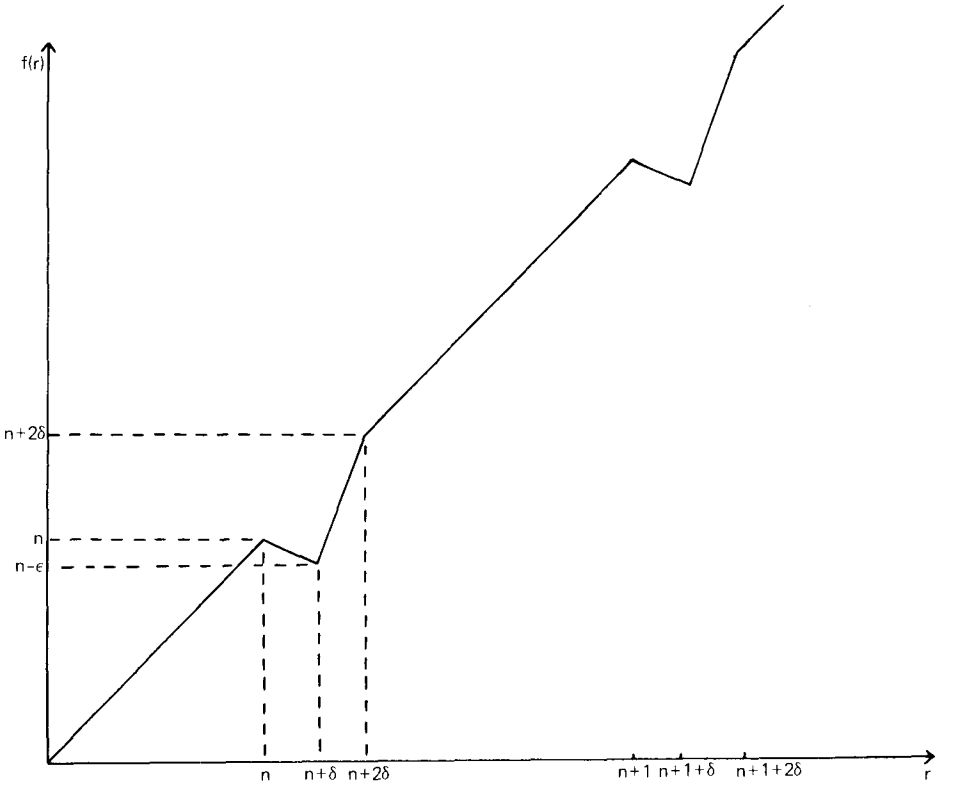


Fig. 2.2. Descent vs. annealing method. (a) Square solution space for example. (b) Radial behaviour of objective function.

would if it were run with $c=0$ (when downhill steps are always accepted while uphill steps never are). Let S be the number of steps taken to reach the global minimum. We use the expected value $E(S)$ to compare the two methods.

Method 1. *Downhill search.* Randomly select a starting point x_0, so $E(f(x_0))=O(N)$. Carry out a strict downhill search to locate the local minimum value $m(x_0)$ which is $[r]-\varepsilon$ where $[r]$ is the integer part of $r(x_0)$, unless $[r]=0$ in which case $m(x_0)=0$. With this method we reach $f(0)$ when and only when we generate a starting point in $[-1,1]\times[-1,1]$ (ignoring uncertainties which occur when $[r]=r$). Thus the number of starts $S\delta/4$ is geometrically distributed with parameter $1/N^2$ and $E(S)=4N^2/\delta$.

Method 2. *Annealing method.* Randomly select a starting point x_0, so $E(f(x_0))=O(N)$. Assume $\varepsilon \ll c \ll \delta$. At any point x_i, choose (randomly) one of the four neighbours $(x_i \pm \delta, y_i)$ and $(x_i, y_i \pm \delta)$. Let df be the change in objective function value on moving to the chosen neighbour. If $df<0$ then accept the chosen neighbour, while if $df>0$ accept it with probability $e^{-df/c}$. Since $df>0$ implies $df=\varepsilon+2\delta$ or $df=\varepsilon$ or $df=\delta$, the choice of c ensures that uphill steps are taken with probability essentially zero except for the step of ε, which is taken with probability essentially 1.

Consequently the method proceeds in a deterministic manner between $r=n+1$ and $r=n+\delta$, for any integer $n>0$, taking a time $O(1/\delta)$ to do so. At a local minimum $r=n+\delta$ it performs a random walk starting at $r=n+\delta$, with an absorbing barrier to the left ($r=n-\delta$) and a reflecting barrier to the right ($r=n+2\delta$). The time for absorption is therefore $O(1)$; for example, except near a corner (i.e. away from the lines $x_1=\pm x_2$) the probability of reaching $n-\delta$ from $n+\delta$ in two steps is arbitrarily close to $P_L=1/16$, while the probability of reaching $n+2\delta$ is arbitrarily small. At a corner there are minor complications (absorption takes at least 3 steps) but the time is still $O(1)$.

Denote by E_1 the expected passage time from $r=n+\delta$ to $r=n-\delta$. The expected time $E(S)$ to reach the global minimum from $r=r(x_0)$ is (if $r\neq[r]$) close to $4[r]/\delta+[r]E_1$ steps, which is $O(N)$. So the orders of magnitude of the times to solve the problem are $O(N)$ for the annealing method as against $O(N^2)$ for the descent method. In higher dimensions, the effect will be even more pronounced: E_1 will change, as will the factor 4 above, but the time will still be $O(N)$ as against $O(N^d)$ for the descent method, where d is the problem dimension.

3. Equilibrium theory

We are given a function $f: X \to R$ and asked to find its minimum. Assume the states $i \in X$ are labelled $1, \ldots, |X|$ such that $i<j$ if $f_i<f_j$. The problem is therefore

to find f_1 in a reasonable time without having access to the labelling which orders objective function values.

With each state i we associate a neighbourhood N_i of all states which can be reached from i in a single transition. Neighbourhoods are implicitly defined by way of a perturbation matrix $P=\{p_{ij}\}$. Here p_{ij} is the probability of generating state j as a candidate next state, when we are in state i. Of course, $N_i=\{j: p_{ij}>0\}$. We assume that for any two states i_0 and i_f, there exists a finite sequence of K states $j_1, j_2, \ldots, j_K$ such that $i_0=j_1$, $i_f=j_K$, and $p_{j_m j_{m+1}}>0$ for $m=1, 2, \ldots, K-1$. This ensures X is connected and P is irreducible (Seneta, 1981).

The algorithm accepts j with probability $a_{ij}(c)$. We require a_{ij} to be a continuous function of c with the following properties:

$$a_{ij}(c)=1 \qquad \text{for all } i>j, \text{ for all } c\geq 0,$$

$$0<a_{ij}(c)<1 \qquad \text{for all } i<j, \text{ for all } c>0,$$

$$a_{ij}(0)=0 \qquad \text{for all } i<j,$$

$$a_{ij}(c)\to 1 \qquad \text{for all } i<j \text{ as } c\to\infty.$$

If state j is not accepted the algorithm stays in state i and tries again to generate a neighbour.

The sequence of solutions is generated by a Markov process with transition matrix $T=\{t_{ij}\}$ where

$$t_{ij}(c)=\begin{cases} a_{ij}(c)p_{ij} & (i\neq j), \\ 1-\sum\limits_{k\neq i} a_{ik}(c)p_{ik} & (i=j). \end{cases}$$

For $c>0$, irreducibility of P together with $a_{ij}(c)>0$ implies T is irreducible. As $c\to 0$, T approaches a lower triangular matrix. In particular, $t_{ii}(0)=1$ if state i is a local minimum: that is, if $j\in N_i$ implies $j>i$, so $p_{ij}=0$ for $j<i$. The effect of accepting uphill steps for positive c is that there is a path from every local minimum to the global minimum. (This is just a restatement of irreducibility.)

Let e_i be the i'th unit vector in $R^{|X|}$. Then if state i_0 is a local minimum,

$$e_{i_0}T(0)=e_{i_0}.$$

In particular, we always have $e_1T(0)=e_1$. Now for fixed $c>0$, T has a unique equilibrium vector π with

$$\pi(c)T(c)=\pi(c).$$

The annealing algorithm works because

$$\pi(c)\to e_1$$

as $c\to 0$, rather than $\pi(c)\to e_{i_0}$ for some local minimum i_0. To prove this, we calculate $\pi(c)$.

We make two additional assumptions:

Assumption I. P is symmetric.

Assumption II. For all $c>0$, $a_{ij}(c)a_{jk}(c)=a_{ik}(c)$ if $i<j<k$.

The first of these will hold in any straightforward neighbour generating scheme. It says that if j is a neighbour of i then i is a neighbour of j, and they are chosen with the same probability. Usually there will be some fixed size Q for all N_i, and neighbours will be selected with uniform probability $1/Q$. We will show later how this assumption may be altered to produce a slightly different convergence result.

The second assumption holds for the standard acceptance criterion having

$$a_{ij}(c)=\begin{cases}\exp(-(f_j-f_i)/c) & \text{if } f_j>f_i,\\ 1 & \text{if } f_j\leq f_i.\end{cases}$$

It obviously also holds for many other functional forms of a_{ij}: anything of the form $a_{ij}(c)=\phi(f_i,c)/\phi(f_j,c)$ will do.

We now come to the main result.

Theorem. *If P is irreducible and Assumptions I and II hold, then for $c>0$ the unique equilibrium vector corresponding to $T(c)$ is*

$$\pi(c)=\pi_1(c)(1, a_{12}(c), a_{13}(c), \ldots, a_{1,|X|}(c)).$$

Proof. We show that $\pi(c)$ satisfies the detailed balance equations

$$\pi_i t_{ij}=\pi_j t_{ji}$$

for all i, j. (See Kelly, 1979.)

Suppose $1<j<i$. Then

$$\begin{aligned}\pi_i t_{ij}&=(\pi_1 a_{1i})(a_{ij}p_{ij})=\pi_1 a_{1i}p_{ij}\\ &=\pi_1 a_{1j}a_{ji}p_{ij} \qquad \text{by assumption II}\\ &=(\pi_1 a_{1j})(a_{ji}p_{ji}) \quad \text{by assumption I}\\ &=\pi_j t_{ji}.\end{aligned}$$

The case $1<i<j$ is similar. When $i=j$ the result is trivial.

This explicit formula for the equilibrium distribution agrees with results derived from physical reasoning and quoted by Metropolis (1953) and Cerny (1984). The same result has been obtained by Sangiovani et al. (1984). The consequences of the theorem are as follows.

1. The equilibrium distribution is independent of the topology.
2. $\pi(c)\to e_1$ as $c\to 0$.
3. $\pi(c)\to(1/|X|, 1/|X|, \ldots, 1/|X|)$ as $c\to\infty$
4. With the standard exponential form of $a_{ij}(c)$, the equilibrium probability that we are no more than $\varepsilon>0$ above the optimum (i.e. that we are in a state i with $f_i-f_1\leq\varepsilon$) is at least $1-(|X|-1)\,e^{-\varepsilon/c}$.

Conclusion (1) tells us that, subject to the space being connected, the choice of topology can only have an effect through the convergence rate, not on the final result. Conclusion (2) gives us convergence with probability as close as we like to 1 but says nothing about how long the convergence takes. Conclusion (3) confirms the intuitive idea that at high enough control values the function values should be irrelevant. Conclusion (4) allows us to determine a value of c to terminate the process: for error probability α we require $\alpha/(1-\alpha) > (|X|-1)\,e^{-\varepsilon/c}$.

Note that we can incorporate complex constraints into the problem by disallowing certain neighbours. If we do so, P is no longer symmetric and the size of the neighbourhood N_i varies with i, so the above Theorem no longer applies. However, it is easy to check that if Assumption I is replaced by

Assumption I'. $p_{ij} = 1/|N_i|$ for $j \in N_i$ and $p_{ij} = 0$ if $j \notin N_i$.

then $\pi_j(c)$ is proportional to $|N_j| a_{ij}(c)$. There is an obvious change in the above inequality for α.

It is also true that constraints can sometimes be usefully handled using penalty functions; in such cases (which are perhaps more relevant to continuous problems than to combinatorial ones) the annealing algorithm is still easy to apply.

4. Reaching equilibrium

We now understand the equilibrium behaviour of the annealing algorithm, but we have not yet discussed the problem of how to get to equilibrium. Indeed, it is not apparent so far why we should vary c at all: why not just choose $c = c_f$ in the first place?

For any fixed value of c, the rate of convergence to the steady state is geometric since the state space is finite, and the convergence factor is given by the second largest eigenvalue of the transition matrix. The latter is however impossible to compute for matrices of size $|X|$. It is to be expected that when c is small it will be very close to 1 for problems with local optima; in that case, convergence will be impossibly slow if we simply start at a small value of c.

4.1. The effect of the topology

Rather than study the optimization problem directly, we work with the related recognition problem

D: Given F, is there a state j with $f_j < F$?

We try to estimate how long it would take the annealing algorithm to confirm that the answer to D is 'yes' when the answer is indeed 'yes'. This is a common simplification in dealing with performance of algorithms. For a full discussion of the relationship between recognition problems and optimization problems, see Garey and Johnson (1979).

Observe first that for arbitrary topologies, $E(S) = O(|X|)$ may easily occur. For example, let $p_{i,j-m} = 1$ for some $m > 0$; then $E(S) = O(|X|/m)$.

We now analyze the problem more formally. In yes-instances of D the annealing algorithm stops as soon as it finds a suitable f_j. Let this happen at step $S(i)$ when the initial state is i. Let df_j be the change in the value of f after one step starting in state j. Then

$$E(df_j) = \sum_k a_{jk} p_{jk} (f_j - f_k).$$

Suppose that $df_j > -M$ and $E(df_j) \leq -\alpha < 0$ when $f_j > F$.

Write $E(S \mid i)$ for the expected number of steps to give a yes answer starting from state i. A trivial extension of Wald's equation (Heyman and Sobel, 1982) would give

$$E(S \mid i) \leq (f_i - F + M)/\alpha. \tag{4.1.1}$$

If there are any local optima, the assumption $E(df_j) \leq -\alpha$ is too strong and instead we should assume

$$E(d^r f_j) \leq -r\alpha_r < 0 \tag{4.1.2}$$

where $d^r f_j$ is the change in f after r steps from state j and α_r is then a bound on the improvement per step, averaged over the r steps.

Proposition. *If* (4.1.2) *holds for some* r,

$$E(S \mid i) \leq (f_i - F + rM)/\alpha_r.$$

For each $j \in X$, define its depth

$$d(j) = \min\{r - 1 : (T^r)_{jk} > 0 \text{ for some } k < j\}.$$

and define $d(F) = \max\{d(j) : f_j > F\}$. Note that the $d(j)$'s and $d(F)$ are independent of c. The integer r in (4.1.2) is to be chosen so that $r > d(F)$. For the travelling salesman problem $d(F) \leq n - 3$, but we conjecture that in typical instances of it, $d(F)$ is less than 10.

To understand how $E(S)$ varies with n we need a lower bound for α_r. In general we can merely state

$$-\alpha_r = \max_{j : f_j > F} \sum_k (f_j - f_k)(T^r)_{jk} \geq \Delta f T^r_{\min}$$

where $\Delta f = \min_k (f_{k+1} - f_k)$ and $T_{\min} = \min\{T_{jk} : T_{jk} > 0\}$. A polynomial bound is thus only guaranteed if $d(F) = O(\log n)$.

In simulations we have found, in agreement with Cerny (1984) that two topologies for the travelling salesman problem having very similarly structured P matrices behave quite differently with respect to convergence speed. This is plausibly explained in the light of the above discussion if one of the two topologies results in a small bound on the depth while the other does not.

Finally, let us see how all this would be seen in terms of the real annealing algorithm which varies c. Imagine a sequence of recognition problems $\{c_1, F_1\}$, $\{c_2, F_2\}, \ldots$, such that the algorithm runs with $c = c_1$ until it finds a value of f below F_1, then changes to $c = c_2$ and so on. When $F_i \gg f_1$, there will be many neighbours of points which result in $df < 0$ and so even when c is large it will still be the case that $E(df) < 0$ on states with $f > F$. Once we decrease F, however, we have to consider states with fewer neighbours giving negative df, and moreover the depth will be greater because of the monotonicity of $d(F)$. The control value will have to be smaller to ensure uphill steps are usually rejected and $E(df) < 0$ still. The result is that as we become more ambitious in our target value F, we are forced to decrease the control and wait longer.

4.2. *Varying the control parameter*

The criterion discussed above for varying c is not the only approach one could adopt. In fact, as we have stated it, it is not helpful as a practical bound since it is hard to estimate depth and α_r.

Ideally, for any function f and space X with given matrix P we would like to find a sequence $\{c_r\}$, possibly state-dependent, such that the error

$$E = \max_i |e_i T(c_1) T(c_2) \cdots T(c_R) - e_1|$$

is minimized. (Note that $c_r = c_{r+1} = \cdots = c_{r+q}$ is possible; also, we are allowed to have $c_{r+1} > c_r$, though this would only happen in a state-dependent scheme that decided it was necessary to increase the control to remove the system from a likely local minimum.)

Optimal $\{c_i\}$ sequences could doubtless be found for trivial problems, but it seems unlikely that this will be possible for real ones. To have a true algorithm, however, we need some way of specifying changes in c without operator intervention. The best way is probably to use feedback: the value of c depends on the history and present behaviour of the system. We do not yet know of a satisfactory feedback controller for the annealing method.

Instead, we adopt the following pragmatic approach which is designed to terminate after a number of steps that is essentially proportional to $\log|X|$, but without any guarantee of convergence. We have found over many simulations that this method performs well in terms of reaching good values of f, though rather conservatively in terms of number of steps needed. We emphasize that this method is heuristic, and is not in accord with the theory developed above.

Choice of c_0. Choose an initial value of c_0 large enough for $\pi(c_0)$ to be close to uniform: this means, if $\phi_j = f_j - f_1$, that $\exp(-\phi_j / c_0)$ must be close to 1 for all j, so c_0 must be large compared with every ϕ_j. We can take $c_0 \gg U$, where U is any upper bound on $\max_j \phi_j$. This is usually easy to find: in travelling salesman problems we could take $U = \sum_j d_j$ where d_j is the longest link leaving the j'th city.

Choice of dc. We make $\pi(c - dc)$ close to $\pi(c)$ so that if we were already in equilibrium, we would nearly be in the new equilibrium state after one step (because

$\pi(c)T(c-dc)$ is close to $\pi(c-dc)$). To first order in dc, the normalization constants in $\pi(c)$ and $\pi(c-dc)$ are equal and

$$\pi_j(c) = \exp(\phi_j dc/c(c-dc))\pi_j(c-dc).$$

Thus we need the exponential term to be nearly 1, so $\phi_j dc \ll c(c-dc)$ for all j. Thus $dc/c^2 = \delta/(U+c\delta)$ for some $\delta \ll 1$, and so, substituting for dc in $c_{i+1} = c_i - dc_i$,

$$c_{i+1} = c_i/(1+\beta c_i)$$

for some $\beta \ll 1/U$.

Note that the $c\delta$ term is only significant in the early stages when c is comparable with U. An almost equivalent method is therefore to choose dc to be $\min(\beta_1 c, \beta_2 c^2)$, for small constants β_1 and β_2. For small values of c this is slower than the exponential cooling scheme used by Kirkpatrick et al.

Choice of c_f. The choice of c_f depends on the acceptable error ε in the solution together with the error probability α. We need $\alpha/(1-\alpha) > (|X|-1)e^{-\varepsilon/c}$, and hence it is sufficient to take

$$c_f \leq \varepsilon/(\log(|X|-1) - \log \alpha).$$

Note that the $\log \alpha$ term will usually be swamped by the $\log(|X|-1)$ term: that is, if we really were at equilibrium, the error probability could be made very small indeed without significantly altering c_f.

It is trivial to show that the number R of steps before termination using the above control schedule satisfies

$$R < (\log|X| - \log \alpha)/(\beta\varepsilon).$$

Note that this means R is essentially proportional to $\log|X|$ because the $\log \alpha$ term is negligible as we have just seen.

In implementing the above scheme, a practical point arises. The value of c_f will nearly always be so small that on any real computer working to finite accuracy and having a pseudo-random number generator, long before we reach c_f we will have precisely zero probability of accepting nearly any uphill step that becomes a candidate. Consequently we might as well decide that as soon as c gets small enough for this to happen (say, $c < -1/\log p_{\min}$ where $p_{\min}$ is some small probability), the system is 'frozen'. In the physical analogy, the domains are now fixed and further cooling only improves the ordering within them. Rather than waste time attempting uphill steps which will always be rejected, we may as well perform an exhaustive search by selecting neighbours of the current point without replacement. If a neighbour is found that has a better objective function value we accept and start checking again; if there are no such neighbours we stop.

This may destroy the polynomial time termination property, since even although there is a polynomial bound on the number of neighbours of a point, there is no guarantee that there will only be a polynomial number of points to be visited before

a local optimum is reached. In practice this is unlikely to happen. (See e.g. Tovey, 1983.) If it did, one could decide to stop after R steps even though further improvement was possible.

5. Conclusions

The attraction of the annealing method is that it is general yet simple to apply. Solving a problem with it requires only that one provide an adequate way of generating neighbours of solution points.

We have shown that the method converges, but that it is not possible to ensure convergence in less than exponential time for general problems. The concept of depth of a state was introduced and was shown to have a major effect on convergence rate. We conjectured that for problems where the annealing algorithm has been found to perform well, the depth is bounded by a quantity which grows slowly with problem size. (Specifically, if the growth is logarithmic then the decision problem related to the optimization problem is solved, on average, in polynomial time.)

For practical implementation a sequence of control parameter changes was introduced and shown to terminate in polynomial time. Computational experience with this control schedule (Lundy 1984, 1985) is encouraging.

References

[1] A. Berman and R.J. Plemmons, *Non-negative matrices in the mathematical sciences* (Academic Press, New York, 1979).

[2] V. Cerny, "A thermodynamical approach to the travelling salesman problem: An efficient simulation algorithm", *Journal of Optimization Theory and Applications* (1984), to appear.

[3] A.W.F. Edwards, "Minimal evolution" (Unpublished, 1966).

[4] B. Everitt, *Cluster analysis* (Heineman Educational Books, London, 1977).

[5] M.R. Garey and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness* (W.H. Freeman, San Francisco, 1979).

[6] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images", *IEEE Transactions PAMI* 6(6) (1984) 721-741.

[7] D.P. Heyman and M.J. Sobel, *Stochastic models in operations research, Volume 1: Stochastic processes and operating characteristics* (McGraw-Hill Inc., New York, 1982).

[8] D.S. Johnson, private communication, 1984.

[9] F.P. Kelly, *Reversibility and stochastic networks* (Wiley, Chichester, 1979).

[10] S. Kirkpatrick, C.D. Gelatt, Jr. and M.P. Vecchi, "Optimization by simulated annealing", Research Report RC 9355, IBM (Yorktown Heights, NY, 1982).

[11] M. Lundy, "Applications of the annealing algorithm to combinatorial problems in statistics", *Biometrika* 72 (1) (1985) 191-198.

[12] M. Lundy, "The annealing algorithm", Ph.D. Thesis, University of Cambridge (Cambridge, 1984).

[13] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth and A.H. Teller, "Equation of state calculation by fast computing machines", *Journal of Chemical Physics* 21 (1953) 1087-1092.

[14] F. Romeo and A. Sangiovanni-Vincentelli, "Probabilistic hill-climbing algorithms: Properties and applications", Memorandum No. UCB/ERL M84/34, University of California (Berkeley, CA, March 1984).

[15] B.M. Schwarzschild, "*Statistical mechanics algorithm for Monte Carlo optimization*", *Physics Today* (May 1982) 17-19.
[16] E. Seneta, *Non-negative matrices and Markov chains* (Springer-Verlag, New York, 2nd Edition, 1981).
[17] E.A. Thompson, "The method of minimum evolution", *Annals of Human Genetics* 36 (1973) 333-340.
[18] C. Tovey, "On the number of iterations of local improvement algorithms", *Operations Research Letters* 2 (5) (1983) 231-238.