Journal of Statistical Physics, Vol. 34, Nos. 5/6, 1984

Optimization by Simulated Annealing: Quantitative Studies

Scott Kirkpatrick[1]

Received November 15, 1983

Simulated annealing is a stochastic optimization procedure which is widely applicable and has been found effective in several problems arising in computer-aided circuit design. This paper derives the method in the context of traditional optimization heuristics and presents experimental studies of its computational efficiency when applied to graph partitioning and traveling salesman problems.

KEY WORDS: Spin glasses; optimization; graph partitioning; algorithms.

1. INTRODUCTION

Dan Gelatt and I, with help from several of our colleagues, have explored a general framework for optimization which uses computer simulation methods from condensed matter physics and an equivalence (which can be made rigorous) between the many undetermined parameters of the system being optimized and the particles in an imaginary physical system. The energy of the physical system is given by the objective function of the optimization problem. States of low energy in the imaginary physical system are thus the near-global optimum configurations sought in the optimization problem. The trick we have used to find these is to model statistically the evolution of the physical system at a series of temperatures which allow it to "anneal" into a state of high order and very low energy.

Arguments for the validity of this approach, and some ideas which help in understanding how to use it effectively, are given in a paper which

[1] IBM Research, Yorktown Heights, New York 10598.

has appeared recently.[1] That paper also gives an introductory review of the optimization problems in computer design to which the method has been applied. Additional work on global wiring by Mario Vecchi and myself has been described elsewhere.[2]

One observation developed in Ref. 1 which must be repeated here is the importance of "frustration" in difficult optimization problems. Just as spin glass models are characterized by random interactions which cannot all be satisfied by any arrangement of the spins, the more challenging optimization problems are those in which conflicting constraints rule out any simple solution. Such constraints typically arise from tradeoffs between achieving high performance and assuring high reliability, and are ubiquitous. As in spin glasses, where frustration introduces metastability and degeneracy into the low-temperature states of the models, in optimization frustration will make the search for good solutions very difficult. However, the degeneracy induced by frustration implies that there should be many equivalent acceptable solutions to a given problem if there are any. Thus it should not be necessary to find the absolute optimal solution.

In this article, I discuss some of the questions of algorithmic efficiency and effectiveness which this work raises, by focusing attention on two problems for which there is extensive literature on good heuristic algorithms: min-cut partitioning of graphs, and finding an optimal tour for a traveling salesman.

RELATION OF ANNEALING TO ITERATIVE IMPROVEMENT

The most common framework used in heuristic methods of multivariate optimization is called iterative improvement. It can be seen as a special case of simulated annealing. In iterative improvement, one starts with the system in a legal arrangement, or configuration, C_i, then rearranges it until an improved configuration, C_j, is found. C_j then becomes the starting pont for further rearrangement. The process terminates when no further improvements can be found.

To apply iterative improvement to a problem, three things are necessary:

i. A concise representation of the configuration, C_i, of the system;

ii. A scalar objective function, $g(C_i)$, reducing the objectives of the optimization process to a single number, quantifying tradeoffs between conflicting objectives;

iii. A procedure for generating local rearrangements of the system.

If we take placement of circuits on a chip as an example, a configuration is a specific assignment of each circuit to one of the available positions on the chip and it can be represented by a list of the assigned circuit locations. The objective function should combine the amount of wire required to connect the circuits as placed (or the chip area needed to complete the wiring in certain styles of chip assembly), with other penalties for estimated circuit loading, heat buildup, timing requirements, or wiring congestion. Information about circuit function is contained in an appropriate data structure describing logical connections between circuits. This is used by whatever programs calculate $g(C_i)$ for a given configuration C_i. Rearrangements $C_i \rightarrow C_j$ are termed local if only a few elements of C_i are different in C_j and if calculation of the change

$$\Delta g = g(C_i) - g(C_j)$$

requires much less computation than does calcuation of $g(C_i)$ itself. For circuit placement, interchange of positions of two circuits is a local move. Interchanges are also sufficient, in the sense that the system can evolve by a sufficient number of interchanges from an arbitrary initial configuration to any desired final configuration.

The inherent limitation in iterative improvement is that the process finds only local minima. Some tricks must be found in each application to increase the likelihood that the solutions are reasonably close in quality to the unknown global minimum.

When stuck, one might resort to more complicated moves, for example interchanging larger numbers of circuits. Typically, there are orders of magnitude more of these, and some care must be used to search only the ones likely to help get the system unstuck. The process of choosing complex moves effectively often embodies methods used in hand solution by experts familiar with the problem, and is a common feature of "expert systems" programming approaches. The drawback to this approach is that consulting experts is difficult and time-consuming, and the strategies gained in this way may become obsolete as the problems change. To a physicist, it is natural to view the expert's moves as a way of getting unstuck by "tunnelling" in the directions in which the barriers between locally stable states are expected to be thin.

One can also improve the results of iterative improvement by finding many such local minima, and keeping the best of them as the final result. But the physical analog is suggestive here. Iterative improvement is like splat cooling in metallurgy, in which energy is rapidly extracted from the system by contact with a massive cold substrate. The result in metallurgy is usually a glassy substance, or at best a polycrystalline material with some fixed small grain size. The structures obtained are characteristic of the fluid at the temperatures where its viscosity becomes too large for further rearrangements. The fixed density of defects quenched into such a glass will cost some extensive amount of energy with respect to the energy of the substance in its ground state.

By taking many quenches, one will find variations in this energy, but those variations will be intensive, and for sufficiently large systems will be much smaller than the energy difference between quenched and ground states. For finite systems, like the models studied in engineering applications of simulated annealing, the difference between extensive and intensive quantities may not be so great, so experiments are needed to determine the variation observed in repeated iterative improvement and its size dependence. Some experiments of this sort are discussed below.

It should be apparent at this point that simulated annealing is just iterative improvement done at a sequence of finite temperatures, with the Metropolis criterion(3) for accepting or rejecting a randomly generated trial move replacing the "improvement-only" rule used in the discussion above.

The annealing schedule is a less well-defined concept. Loosely speaking, one wants to warm the system under study until it is fluid, cool slowly through the range of temperatures in which large decreases in the objective function are observed (indicating that freezing is occurring), then cool more rapidly through the lower temperatures until no further improvements are observed. It is not difficult to do this interactively for a particular problem, recording the number of iterations taken at each temperature for use on subsequent problems of the same type and size.

An annealing schedule can be generated automatically by a slight generalization of the procedure described in Ref. 1. One first finds the "melting temperature" by starting at an arbitrary temperature, attempting a few hundred moves, and determining the fraction of the moves which are accepted. If that fraction is less than, say, 80%, the temperature is doubled. When the fraction of moves exceeds this threshold, a sufficient number of moves is taken to completely "melt" the system, and the cooling process can begin. In cooling the system, one decreases the temperature by a constant ratio, running at each temperature until every movable object has been moved a fixed number of times, or some allotted number of attempts at that temperature have been taken. If by taking running averages of the current value of the objective function it is determined that $g(T)$ is decreasing rapidly, it may be necessary to repeat a temperature, or decrease T by a smaller ratio to ensure that the system stays close to equilibrium.

In systems with relatively few degrees of freedom we have sometimes found that the objective function is not very smooth, so that it is difficult to approach a low-lying local optimum gradually at low temperatures. In such cases, it may be more effective to search at a moderate temperature, with the program recording the best few solutions obtained so far.

APPLICATION TO MIN-CUT PARTITIONING

Partitioning of a system, usually represented by some sort of sparse graph, into two or more parts in such a way that the number of bonds of the graph which cross interpartition boundaries is minimized, is a very common problem in optimization. It occurs, for example, in many forms of computer-aided design of electronic hardware. The frustration in this problem arises from the competition between minimizing cuts and the requirement that the number of graph vertices in each partition be balanced. In Ref 1, we showed by explicit transformation that the objective function for a simple two-way partition problem is identical to an Ising spin glass Hamiltonian. The balance requirement appears as an infinite-ranged antiferromagnetic interaction and the cut minimization gives a random ferromagnetic interaction between vertices which are directly connected.

A natural ensemble of partitioning problems is provided by the set of random graphs with N vertices and zN edges. The minimum number of edges which need be cut in bisecting such a graph can be extracted from the ground-state energy of the associated spin glass Hamiltonian. From extensive studies of the infinite-ranged spin glass we know that the ground-state energy has two terms, one proportional to the mean interaction strength and the other to the variance of the random interactions. For the min-cut graph partitioning problem this mean field theory result translates into the prediction that the optimal partition will cut $zN/2$ of the edges, minus a correction which is also proportional to N and scales as $\sqrt{z}$. This correction term is the ordering energy of the spin glass.

Bui (in an unpublished masters' thesis) has recently surveyed the literature of this problem and derived some exact bounds on the minimum bisections of random graphs.[4] In the general case his upper and lower bounds for the correction term scale as different powers of N, although for sufficiently dense graphs ($z > 9$), he obtains a result similar to the mean field theory prediction.

To check the accuracy of the spin glass mean field theory in predicting optimal partitioning, I generated samples of random graphs with z ranging from 2 to 8, and N from 128 to 2048, and used a simulated annealing program to obtain good partitions. The resulting ordering energies agreed with mean field theory in form, with a coefficient about 20% smaller than the mean field theory expression.

These samples were also used to determine whether iterative improvement with many random trials was more computationally efficient than simulated annealing. The distribution of results obtained in many "splat quenches" was Gaussian out to at least two standard deviations. The difference in number of bonds cut between the average result of iterative improvement and the result of annealing was five standard deviations for $N = 512$, $z = 4$ or 8, and 10 standard deviations for $N = 512$, $z = 2$. In all three situations, the running time of the annealing program was about 12 times the cpu time of a single "splat quench." Since the best result one would expect to find by sampling 12 times from a normal distribution is about two standard deviations below the mean, simulated annealing is more efficient.

As expected, the difference between splat quenched and annealed partitions sharpened as N increased. For $N = 2048$, the annealed result was 18, 14, and 10 standard deviations below the mean splat quenched results, for $z = 2$, 4, and 8, respectively. The computing time for annealing, normalized to the time for a single splat quench, also increased with N. For $N = 2048$, $z = 2$, 4, and 8, this ratio was 24, 18, and 14, respectively.

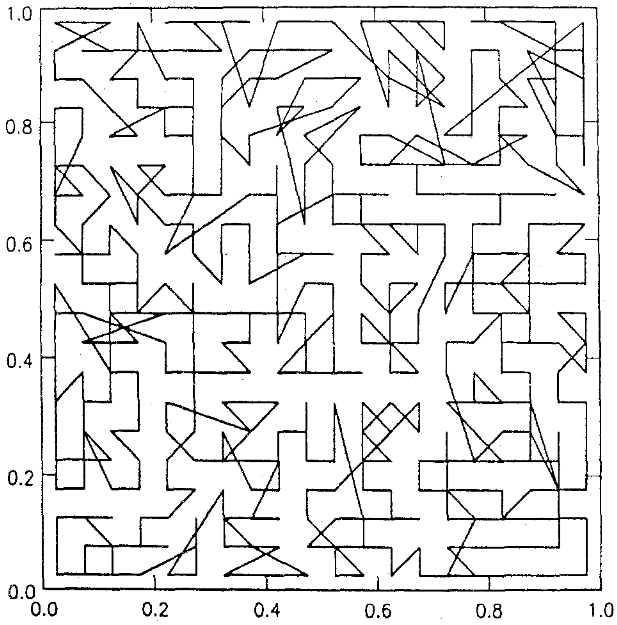
In one partitioning study using real logic to define the connection graph,[1] the separation between the splat quenched result and the result of annealing is even greater. Apparently the presence of hierarchical structure in the logic causes freezing to begin at higher temperatures than would be the case in the artificially generated examples. D. S. Johnson (private communication) has also observed this difference in studies of annealing on other optimization problems.

Apparently iterative improvement based on moving randomly selected sites from one side of the partition to the other is not very effective, but better algorithms are known. The best known scheme is due to Kernighan and Lin,[5] and has been implemented most efficiently by Fiduccia and Mattheyses.[6] Burstein and Goldberg have recently shown[7] that blocking transformations (not unlike a renormalization group transformation) can improve the results of this algorithm. I have compared my results with those of Burstein's program by generating random graphs and using both programs to look for optimal bisections. The best solutions found by one program can also be found by the other, but some interesting differences in the computational efficiency of the two algorithms appeared. The Kernighan–Lin procedure appears to be very inefficient for low z. Burstein's program required longer running times than my program for $z = 2$, while for large values of z, annealing required much longer times. Since the two programs were written in different languages, and used different data representations, no quantitative comparison was attempted. It should be noted, however, that the traditional algorithms consider only exact bisections. If the problem permits some leeway in balancing the numbers of vertices assigned to the two sides (most applications do), then the extra freedom in the annealing Hamiltonian permits better solutions to be found.

TRAVELING SALESMAN PROBLEMS

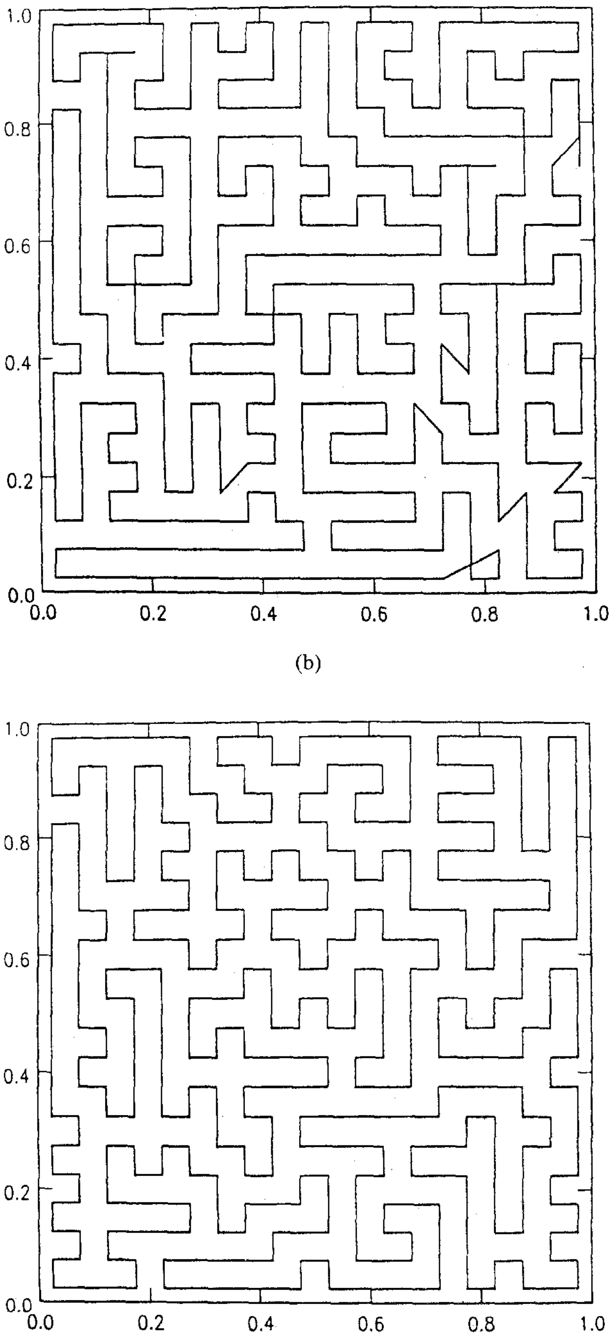
Another easily formalized optimization problem is the well-known "traveling salesman" problem, in which we are required to construct the shortest tour of a prescribed list of N cities. The frustration arises between the requirement that the path be as short as possible and the fact that the path must be a tour. An example of the application of simulated annealing to a traveling salesman problem is shown in Figs. 1a–1c.

To have a problem with known solution, we have put the cities in this example on the points of a regular 20×20 square grid, in a square of unit linear dimension. The optimal path can be no shorter than one grid spacing per step, and one can easily convince oneself that such a path is achievable for a grid with an even number of points on a side. A normalized path



(a)

Fig. 1. (a)–(c). Traveling salesman tours obtained by simulated annealing at temperatures (a) $T = 1.0$; (b) $T = 0.3$; and (c) $T = 0.0$.



(b)

(c)

Fig. 1. Continued.

length, α, can be defined for each tour by dividing the tour length by the minimum path length, $\sqrt{N}$, which is a natural scale for less regular problems as well.

To rearrange the salesman's path it suffices to select an arbitrary subsequence of points in the existing path and reverse the order in which they are traversed. This basic move is the simplest of a set introduced to the problem by Lin and Kernighan.(8,9) With the extra power of annealing, this is sufficient to obtain solutions of problems with a few thousand cities which are as good as those found using exhaustive search with more elaborate and thus more time-consuming moves.

At high temperatures (Fig. 1a), the salesman's path follows the underlying grid for only a few steps at a time, and $\alpha = 1.85$ in the example shown. At lower temperatures, the path is optimal for very long distances, with mistakes occurring in isolated local regions (Fig. 1b, where $\alpha = 1.04$). The mistakes cannot be removed by the basic subsequence reordering move, but they do diffuse about until two such defects meet and annihilate each other. Finally (Fig. 1c), the process concludes with one of the many possible exact minimum length tours, $\alpha = 1.0$.

The sequence of phenomena occurring from high to low temperature are quite like those occurring as liquids solidify, with a slowly growing correlation length at high temperature, and the excess energy at low temperatures associated with locally stable defects which can only be removed by diffusion to the surface or by recombination.

The regular arrangement of points studied in the figures is not representative of actual traveling salesman problems. A more realistic ensemble of problems is obtained by considering instances generated by placing points at random in a unit square. Distance will be defined as the sum of the horizontal and vertical separations of any two points (Manhattan metric). A succession of algorithms were considered: the "greedy" algorithm, in which the path always goes to the nearest remaining point; "greedy," followed by exhaustive two-bond optimization; "greedy," followed by exhaustive three-bond optimization; and simulated annealing using both two- and three-bond rearrangements as the elementary moves. "Two-bond" and "three-bond" refer to the rearrangements introduced by Lin and Kernighan(8,9) in which sequences are reversed, replacing two bonds, or displaced (possibly with reversal), replacing three bonds of the salesman's tour.

Searching by simulated annealing always found better solutions than exhaustive iterative improvement with the same set of rearrangements, and simulated annealing with two-bond moves gave better solutions than exhaustive search with three-bond moves for the sample sizes considered. The cost of simulated annealing with two-bond moves was kept proportional to

N^2 by fixing the annealing schedule. Thus for sufficiently large samples, simulated annealing with two-bond moves was less costly than exhaustive three-bond improvement, besides giving better answers. However, for this ensemble of problems the differences between the lengths of tours found by the different algorithms are not large, so in practice the simplest algorithms might be entirely adequate.

The normalized lengths obtained for 20 samples of 100 points apiece were: (greedy only) $\alpha = 1.185 \pm 0.062$; (greedy, followed by 2-opt) $\alpha = 1.024 \pm 0.042$; (greedy, followed by 3-opt) $\alpha = 0.981 \pm 0.041$; (greedy, followed by simulated 2-bond annealing, followed by 2-opt) $\alpha = 0.969 \pm 0.032$; and (greedy, followed by simulated annealing using 3-bond moves involving sequences of up to 10 sites as well as 2-bond moves) $\alpha = 0.957 \pm 0.030$. Computing times for each 20-case run were: greedy only = 0.3 sec; greedly and 2-opt = 2 sec; greedy and 3-opt = 2 minutes, greedy then 2-bond simulated annealing = 3.5 minutes; and for the final set of results, using restricted 3-bond moves and increasing all annealing times fourfold, elapsed time = 30 minutes. All times were on an IBM 3081K.

Since the sample-to-sample variations were large, we show in Table I some results for specific samples, in this case with 400 points in the unit square. Again simulated annealing with 2-bond moves finds consistently better solutions than does 3-bond exhaustive iterative improvement. For 400 points, the 3-bond exhaustive search took 544 sec to complete, while the simulated annealing program ran for 420 sec. The exact value of α in the limit of a large system is not known. For samples with 900 points in the unit square, we obtained $\alpha = 0.931$, averaged over four samples using restricted 3-bond moves and annealing.

Table I. Results, 8 2D Traveling Salesman Problems with 400 Cities Each, Randomly Distributed in a Unit Square. Lengths in Manhattan Metric, Normalized to $\sqrt{N}$.

Case	Greedy	2-opt	3-opt	MC 2-opt
1	1.176	0.996	0.955	0.931
2	1.214	1.036	0.982	0.954
3	1.162	1.016	0.978	0.941
4	1.213	1.006	0.955	0.941
5	1.153	0.999	0.957	0.943
6	1.139	1.023	0.972	0.950
7	1.138	1.004	0.955	0.929
8	1.140	1.044	0.971	0.958
avg.	1.167	1.015	0.966	0.944

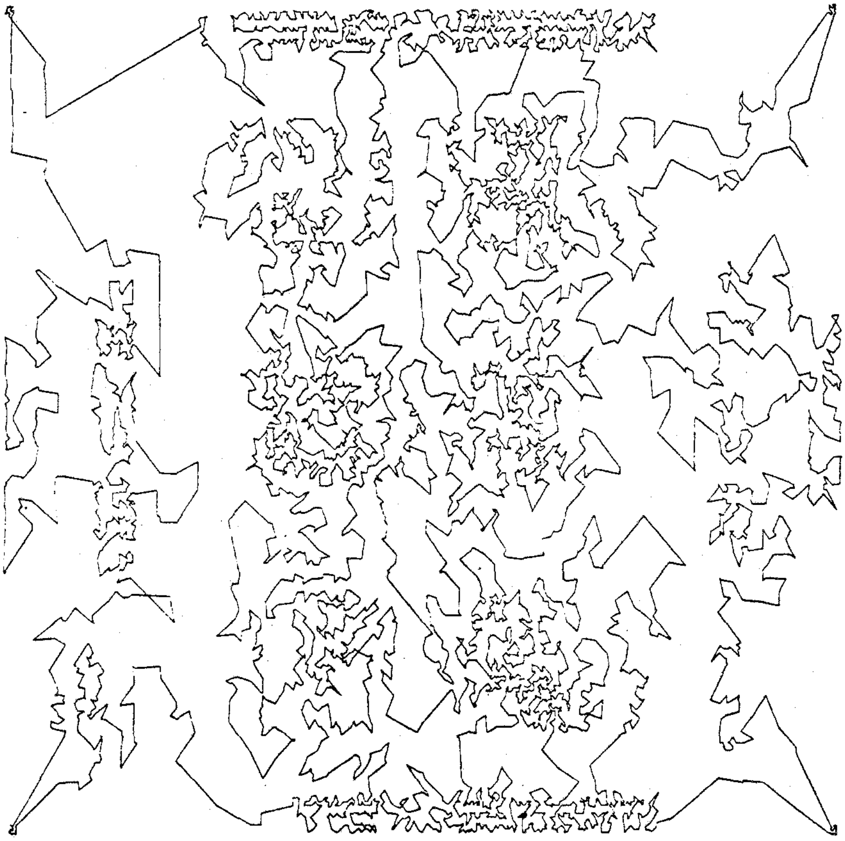


Fig. 2. A traveling salesman's tour taken in order to drill 6406 holes.

Simulated annealing is effective on still larger samples. Figure 2 shows a tour found for an actual problem, caused by the need to drill 6406 holes in a printed circuit board with a large automatically positioned laser. The tour shown was found by applying "greedy," then 2-bond simulated annealing, then exhaustive 2-opt, in less than 20 min of cpu time. It is about 25% shorter than the result of applying the greedy algorithm alone.

CONCLUSION

Experience with several optimization problems and the simulated annealing process for attacking them suggests that the metaphor connecting statistical physics in disordered matter and the sorts of hard optimization problems which arise in engineering of complex systems is a profound one, and can give useful insights for devising effective heuristics.

REFERENCES

1. S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, *Science* **220**:671–680 (1983).
2. M. P. Vecchi and S. Kirkpartrick, to appear in *IEEE Trans. Circuits Systems*.
3. N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, *J. Chem. Phys.* **21**:1087–1092 (1953).
4. T. N. Bui, private communication (MIT report).
5. B. W. Kernighan and S. Lin, *Bell Syst. Tech. J.* **49**:291–307 (1970).
6. C. M. Fiduccia and R. M. Mattheyses, Proceedings of the 19th DA Conference, Las Vegas (1982), pp. 175–181.
7. M. Burstein and M. K. Goldberg, Proc. ICCD, Port Chester (1983), pp. 122–125.
8. S. Lin, *Bell Syst. Tech. J.* **44**:2245 (1965).
9. S. Lin and B. W. Kernighan, *Oper. Res.* **21**:498 (1973).