Optimizing Simulated Annealing

Patrick BANGERT
School of Engineering and Science, International University Bremen
P.O. Box 750 561, 28725 Bremen, Germany.

March 21, 2005

ABSTRACT

The method of simulated annealing was used to get a heuristic solution for the minimum length word equivalent to a given word in the braid groups (a known NP-complete problem). The simulated annealing paradigm with a simple cooling schedule leaves five parameters up to the user to choose that were chosen empirically based on performance experiments as is the usual practise. After this, a downhill simplex method was developed to further optimize these critical parameters and a quality improvement of up to 26.1% was observed. This additional improvement made the algorithm competitive, on average, with custom designed heuristics for this problem.

The conclusions going beyond the present combinatorial problem are: (1) Fine-tuning of cooling schedule parameters is critical for the solution quality in simulated annealing, (2) downhill simplex methods (as opposed to Newton's method, for example) are well-suited for this task and (3) significant quality improvement is possible even for a simple cooling schedule.

INTRODUCTION

Simulated Annealing

Simulated annealing is a paradigm method for optimization problems of many types [8]. It was first presented in 1953 as an idea from industrial physics together with an interpretation of this industrial process from statistical mechanics [4]. To form a strong alloy, one heats up the components, mixes them well and lets them cool slowly according to a strict (empirically determined) cooling schedule in order to minimize the number of defects in the solidification process of the material. This physical annealing process is interpreted in the setting of combinatorial optimization (by which we mean finding the minimum of a cost function $C(\mathbf{x})$ for the vector of variables $\mathbf{x}$ in many dimensions) by the following meta-algorithm:

Data : A candidate solution S and a cost function $C(\mathbf{x})$.
Result : A solution S' that minimizes the cost function $C(\mathbf{x})$.

$T \leftarrow$ Starting Temperature
while *not frozen* **do**
  while *not at equilibrium* **do**
    $S' \leftarrow$ perturbation of S.
    if $C(S') < C(S)$ *or selection criterion* **then** $S \leftarrow S'$
  end
end

Algorithm 1: General Simulated Annealing

In words, we begin with a guessed solution S and heat it up using some starting temperature. While at one temperature, the system is allowed to transit to other states until it reaches equilibrium at that temperature. Transitions that lower cost are always accepted and transitions that increase cost are accepted relative to some selection criterion that usually is a function of both the current temperature and the cost increase of the proposed change. When equilibrium has been reached the temperature is changed and this is continued until the system is at equilibrium at a very low temperature at which time we call it frozen. Due to the equilibration and the guaranteed acceptance of a downhill transition, the final state is certain to be a minimum in the cost function. It may however be a local minimum and not the global one. In order to increase the chances of getting the global minimum, the algorithm allows uphill transitions preferentially early in the execution of the algorithm, i.e. at high temperatures.

Simple as this idea may seem, it is very powerful and this general algorithm has been used to solve a variety of problems. It is regarded to be *the* solution to the most studied problem of optimization, the travelling salesman problem [6]. Research about simulated annealing reached its heyday in the mid 1980's and has been employed in a variety of settings since then; a search reveals 986 Ph.D. theses in the last 18 years that focussed on using the method to solve some problem of practical significance [3, 7]. The later advent

of genetic algorithms stole the limelight from simulated annealing for some time [8]. However, from direct comparisons between these two approaches it appears that simulated annealing nearly always wins in all three important categories: implementation time, use of computing resources (memory and time) and solution quality [3, 8].

In our general presentation in algorithm 1, we have not specified five crucial components in the method: (1) starting temperature, (2) freezing condition, (3) equilibrium condition, (4) perturbation mechanism and (5) selection criterion. The perturbation mechanism is problem dependent. It defines a neighborhood structure in the space of possible solutions (which solutions can be reached by means of a single transition from any given solution) and thus the mechanism has considerable influence over the convergence speed of simulated annealing. If the neighborhood is very large, then we are able to explore the solution space quickly but will have trouble settling down later. Conversely, if the neighborhood is very small, convergence might be prohibitively slow. A healthy medium must be chosen but there is no general theory available at present to discuss what a good medium might be; we are firmly in the realm of empirical testing where this issue is concerned. Below, we shall discuss an optimization problem coming from group theory. Here, the transition mechanism will be given naturally by the presentation of the group but which presentation to choose is still up to the user. In our example, there is one presentation that is special for physical reasons and so we do not, in fact, have a choice of transition mechanism. The transition mechanism is certainly more complex than the other elements of simulated annealing and cannot be treated using the same methods as we shall treat these, i.e. it cannot be continuously parameterized in general.

Taking the transition mechanism as discussed, we may simplify the other steps in a general way. The simplest version of simulated annealing sets five constants A, B, C, D and E to some initial values and looks like this:

Data : A candidate solution S and a cost function $C(\mathbf{x})$.
Result : A solution S' that minimizes the cost function $C(\mathbf{x})$.

$T \leftarrow A$
while $T > B$ **do**
  for $i = 1$ *to* C **do**
    $S' \leftarrow$ perturbation of S.
    if $C(S') < C(S)$ *or Random* $< \exp[(C(S') - C(S))/DT]$ **then** $S \leftarrow S'$
  end
  $T \leftarrow ET$
end

Algorithm 2: Simple Simulated Annealing

In words, we start with a constant temperature A and define a constant temperature B to be the freezing point. Equilibration is assumed to occur after or within C steps of the proposal-acceptance loop where the selection criterion is the thermodynamic Maxwell-Boltzmann distribution after which the temperature is decremented by a constant factor. The standard choices for these constants are $A = \mathrm{C}(S)$, B is 100 times smaller than the best lower bound on cost, $C = 1000$, $D = 1$ and $E = 0.9$. After successful implementation of this algorithm, one usually plays with these parameters until the program behaves satisfactorily. It is clear that implementing this method is very fast and we observe from the literature that the vast majority of applications are computed using the version of simulated annealing given in algorithm 2 where the five parameters are determined manually [3].

The entire literature on simulated annealing makes two marked omissions: No direct comparison of a variety of cooling schedules nor a systematic investigation of the dependence of simulated annealing on the above parameters is made (only partial and scattered data are available with regard to both issues). In this paper, we wish to rectify the second omission. We find that small changes in the parameters of the simple cooling schedule presented above can have significant effects on the solution quality in the average case of the resultant simulated annealing algorithm. We do this by means of a representative example: Finding the minimum length word in a braid group.

The Minimum Length Braid Word Problem

The braid groups are given by their Artin presentation

$$B_n = \left\langle \begin{array}{l} \{\sigma_i\} : \ \sigma_i\sigma_j \approx \sigma_j\sigma_i, \\ \qquad\quad \sigma_i\sigma_{i+1}\sigma_i \approx \sigma_{i+1}\sigma_i\sigma_{i+1}, \\ \qquad\quad 1 \le i,j < n, \ \ |i-j| > 1 \end{array} \right\rangle \qquad (1)$$

Suppose that $w \in B_n$ and let $L(w)$ give the letter length of the word w. The minimum word problem asks us to find a word $u \in B_n$ such that $u \approx w$ and that $L(u) \le L(v)$ for any word $v \in B_n$ and $v \approx w$; that is for an equivalent word of minimum length. In B_1 and B_2, the problem is trivial and in B_3, a polynomial-time algorithm exists [2]. For B_n with $n > 3$, the problem is known to be NP-complete [5]. A number of heuristic methods have been designed for this problem and seem to work very well [1].

The cost function is clearly $L(w)$ and the transitions are the group relations with the obvious possibility of either cancelling or including the pair $\sigma_i\sigma_i^{-1}$ anywhere in the word. As such the cost function is always positive or zero.

This problem is representative of many combinatorial problems. It is a symbol sequence with a certain number of allowed moves on it together with a cost function. The travelling salesman problem and protein folding are just two industrial examples that are very similar in nature to minimizing braids [8]. The issue of braid minimization is important when one models the magnetic field lines on the solar corona [1]. As one cannot see inside the sun, the

lines appear to have endpoints and due to magnetohydrodynamics the topology must be conserved, i.e. the field lines are braided. The amount of energy stored in a braid is proportional to the number of essential crossings in it - the crossings that are needed to achieve the topology defined by this braid. However most braids are randomly generated due to the erratic motion of the endpoints on the photospheric surface. Hence the attempt to find the shortest braid word topologically equivalent to a given one. The modelling effort for magnetic field lines is directed towards predicting solar flares and coronal mass ejections that have significant effects on the Earth and some of its important systems (electrical power grids, satellites, communication networks etc.). For these reasons, we believe braid minimization to be a representative and important problem to be studied. Heuristics must be used as the problem is NP-complete for $n > 3$. Past efforts at designing a custom heuristic have been successful but the problem lends itself well to simulated annealing which we compare with the custom methods here.

SIMULATED ANNEALING AS A FUNCTION

Suppose we start simulated annealing with configuration S and obtain configuration S' as a result, then we define the *reduction ratio* α by

$$\alpha = \frac{C(S')}{C(S)} \tag{2}$$

As $C(S') \leq C(S)$ and cost is always non-negative, we have $0 \leq \alpha \leq 1$. The efficacy of the optimization method may be measured by α at least on average when the initial solution S is generated randomly. The quality is high when α is low (if $\alpha = 0.2$, then 80% of initial cost has been removed). Clearly the α obtained varies between different starting configurations and even between runs of simulated annealing with the same starting configuration because of the random element in the method. Therefore, we agree to run simulated annealing N times using randomly generated initial configurations of the same cost. The configurations are all different but the cost must be the same in order to get an accurate measurement of the reduction for a braid of a particular length. Then we define,

$$\overline{\alpha} = \frac{C(S_1) + C(S_2) + \cdots + C(S_N)}{N \cdot C(S)} \tag{3}$$

which is the average reduction ratio over the N runs. As long as N is large enough, the random variations of the method should cancel out and the result should become predictable to within a given accuracy. Using this interpretation, we may regard the simulated annealing method as defining a function $\overline{\alpha} = \overline{\alpha}(A, B, C, D, E)$ depending on five parameters (for the simple schedule). We would like the average reduction ratio to be as large as possible.

This is yet another optimization problem with a function instead of a combinatorial problem. We are able to evaluate the function only at considerable computational cost (N runs of simulated annealing for N randomly generated initial configurations) and we do not know its derivative accurately. Even approximating the derivative comes only at heavy computational cost. There are many optimization methods such as Newton's method or more generally a family of methods known under the names Quasi-Newton or also Newton-Kantorovich methods that rely on computing the derivative of the objective function. Some of them require high computational complexity due to the computation of the Hessian matrix but complexity considerations are secondary here. The most important reason against all these methods is that the derivative computation is not very accurate for the function constructed here and this loss of accuracy in an iterative method would yield meaningless answers. Indeed, such methods were tried and the results found to be unpredictable because of error accumulation and much worse than the results obtained by methods not requiring the computation of derivatives. The method of choice for optimizing a function over several dimensions without computing its derivative is the downhill simplex method (alternatively one may use direction set methods). Thus, we use the downhill simplex method to minimize $\overline{\alpha}(A, B, C, D, E)$.

The starting point for the simplex method will be given by those values of the five parameters that we obtain after some manual experiments. This is done for the reason that most practitioners of the simulated annealing paradigm choose their parameters based on manual experiments [3]. The other points on the simplex are set by manually estimating the length scale for each parameter [6].

RESULTS AND DISCUSSION

We implemented the downhill simplex method to minimize $\overline{\alpha}(A, B, C, D, E)$ which was implemented according to algorithm 2 and evaluated using equation 3. Manual experiments were started looking at the reduction ratio and attempting to find optimal parameters manually. This is the approach taken by the vast majority of practitioners of the simulated annealing methodology thinking that the algorithm is robust enough to find the global minimum in any case. This point was used as the starting point for downhill simplex method that found different values from the manual search.

In table 1, we give the computational results representing about four months of computation time. The first column gives the index of the braid group studied. The second column is a custom heuristic that represents the state-of-the-art solution for this problem [1]. The third column is the result of the manual simulated annealing search. The fourth column gives the result of the downhill simplex optimized average reduction ratio (we took $N = 1000$ and

Table 1: Reduction ratios $\overline{\alpha}$ (average final length divided by initial length) as a function of the number of strings n. The quality of the solution is high when the reduction ratio is low. For $n < 6$ the optimized simulated annealing algorithm is better than the best hitherto known custom heuristic algorithm for the particular problem of braid minimization. Moreover, the downhill simplex method significantly improves (by up to 26% in this experiment) the quality of simulated annealing over the common manual tuning.

n	Crossing Force	Manual Annealing	Downhill Annealing	Change (%)
3	0.426(6)	0.327(3)	0.293(3)	10.4
4	0.430(5)	0.451(4)	0.384(4)	14.9
5	0.447(4)	0.537(6)	0.438(6)	18.4
6	0.455(4)	0.596(5)	0.474(5)	20.4
7	0.469(4)	0.734(3)	0.550(3)	25.1
8	0.471(4)	0.681(4)	0.518(4)	23.9
9	0.476(4)	0.541(5)	0.534(5)	1.3
10	0.481(4)	0.736(3)	0.544(3)	26.1

$L(w) = 100$). The fifth column compares the improvement of the downhill simplex values to the manual values. The numbers in brackets express the computed error in the last digit. We easily see that the (optimized) simulated annealing algorithm is competitive with the custom heuristic (it is actually better for $n < 6$).

Many simulated annealing papers have been published that center around the topic of performance of the algorithm in terms of getting to an acceptable minimum *quickly* [8]. A variety of cooling schedules have been designed that can reduce the computation time at the expense of solution quality. While the author experimented with a number of open-source implementations of simulated annealing for a variety of optimization problems with tools such as a profiler, speed-ups of up to three orders of magnitude were achieved. This is in contrast to claimed speed-up factors of between 1.2 and 2.0 that come from changing the cooling schedule *at the expense of solution quality* [8]. Thus the author believes the speed of the simulated annealing method to be so dominated by programming care that he has not attempted to simultaneously optimize solution quality *and* execution speed. This simultaneous optimization would, however, be no problem in principle after one made the, completely random, decision how relatively important speed is in relation to quality.

While the results shown in columns three and four of table 1 where computed for initial braid lengths of 100 generators, a large number of experiments were made with other initial lengths. As in previous work, we find an approximately logarithmic growth of $\overline{\alpha}$ with respect to *both* group index n and initial length $L(w)$ [1]. The average reduction ratio seems to become relatively constant after $n = 10$ and $L(w) = 100$. However, the final parameter list found for a particular n and $L(w)$ depends *very strongly* on n and $L(w)$. This means that the five parameters for the simple simulated annealing schedule must be viewed as being functions of the input size and not as being constants.

CONCLUSIONS

Thus we may draw a number of conclusions that would appear to hold in general: (1) The solution quality obtained using simulated annealing depends strongly on the numerical values of the parameters of the cooling schedule, (2) the downhill simplex method is effective in locating the optimal parameter values for a specific input size, (3) the parameters depend strongly on input size and should therefore not be global constants for an optimization problem, (4) the improvement in solution quality can be significant for theoretical and practical problems (up to 26.1% improvement was measured in these experiments which is large enough to have significant industrial impact).

Furthermore, the reason that the usual manual search is so much worse than an automated search seems to be that the solution quality (as measured by the average reduction ratio) depends strongly on the cooling schedule parameters, i.e. the landscape is a complex mountain range with narrow valleys that are hard to find manually. Finally, the improved schedule parameters, in general, lead to slightly greater execution time but in view of the dramatic improvement of quality (as well as the fact that execution time seems to be dominated by programming care) this is well worth it.

ACKNOWLEDGEMENTS

I am very grateful to Milko Krastev for helping with the software and the initial experiments and to the CLAMV computing facility at the International University Bremen for supplying the computing power.

REFERENCES

[1] P. D. Bangert, M. A. Berger and R. Prandi, **In Search of Minimal Random Braid Configurations**, J. Phys. A, 35 (2002), pp. 43–59.

[2] M. A. Berger, **Minimum Crossing Numbers for Three-Braids**, J. Phys. A, 27 (1994), pp. 6205–6213.

[3] N. E. Collins, R. W. Eglese, B. L. Golden, **Simulated Annealing - An Annotated Bibliography**, Am. J. Math. Manag. Sci., 8 (1988), pp. 209–307.

[4] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, **Equation of State Calculations by Fast Computing Machines**, J. Chem. Phys., 21 (1953), pp. 1087–1092

[5] M. S. Paterson and A. A. Razborov, **The set of minimal braids in co-NP-complete**, J. Algorithms, 12 (1991), pp. 393–408.

[6] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, **Numerical Recipes in C**, Cambridge University Press, Cambridge, 1992.

[7] ProQuest, http://www.umi.com/proquest/

[8] P. Salamon, P. Sibani and R. Frost, **Facts, conjectures, and improvements for simulated annealing**, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2002.