Function Optimization based on Advanced Simulated Annealing[1]

Bruce Rosen

Division of Mathematics, Computer Science and Statistics
The University of Texas at San Antonio, San Antonio, Texas, 78249

Abstract

Solutions to numerical problems often involve finding (or fitting) a set of parameters to optimize a function. A novel extension of the Simulated Annealing method, Very Fast Simulated Reannealing (VFSR) [6, 14, 9], has been proposed for optimizing difficult functions. VFSR has an exponentially decreasing temperature reduction schedule which is faster than both Boltzmann Annealing and Fast (Cauchy) Annealing. VFSR is shown to be is superior to these two methods on optimizing a difficult multimodal function.

1. Introduction

Solutions to numerical problems often involve finding (or fitting) a set of parameters to minimize a function. In the design of computer chips, for example, it is often the designer's goal to find a chip layout that is optimal with respect to one or more cost measures (e.g. chip area and maximum wire length [12]) and that meets certain layout requirements (such as sufficient distance between different wires, restricted number of wiring layers, etc). These types of practical problems are of high complexity and are often symbolically difficult to solve. Thus, direct numerical algorithmic methods are often employed to find a set of optimal task parameters. The optimal parameter set is the one with the minimal function value. It is often the function optimization task to find the parameters yielding the lowest function value in the solution space.

Simulated Annealing (derived from the physical characteristics of spin glasses [13, 12, 1, 2]), allows the system to probabilistically sample different locations (points) of the function landscape, both in and out of different local minima. As the temperature is reduced the likelihood that the system samples lower local minima than higher ones increases. Finally, when the temperature is at zero, the system finds the lowest minima, the global minimum. This is mathematically guaranteed given time and a proper temperature annealing schedule [13]. The major problem with simulated annealing methods is that they are slow to converge to the lowest functional minima, the global minimum.

Although deterministic algorithms may have efficient implementations and quickly converge to solutions, they are not guaranteed to find the *optimal* global minimum of a given optimization function. This is especially true if the function has may distinct local minima.

1.1. Annealing optimization algorithms

Annealing based optimization algorithms are relatively new optimization method and employs *noise* to choose new parameter values. General simulated annealing optimization methods [7] choose new points at various distances from their current point x. Each new point x' is generated probabilistically according to a given generating distribution $g()$. These algorithms calculates the function value $E = f(x)$, and then probabilistically decides to *accept* or reject it. If accepted, the new point becomes the current point. The new point may be accepted even if it is worse and has a larger function value than the current point. The criteria for acceptance is determined by an acceptance function $h()$, the temperature parameter T, and the difference in the function values of the two points. Initially, the temperature T is large, and a new point is accepted roughly half the time. As the algorithm progresses, T is reduced, thus lowering the probability that the acceptance function will accept a new point if it's functional value is greater than that of the current point. The general simulated annealing procedure is defined as:

1) Choose a high starting temperature T and a random starting point x.
$T \leftarrow T_0,\ x \leftarrow x_0$

2) Calculate the function value of the starting point.

[1] This research was partially supported by a grant from The University of Texas at San Antonio Faculty Research Awards program.

$E \leftarrow f(x)$

3) For each iteration k, $k = 1 \ldots k_f$
and while T is sufficiently large,
do the following:

a) Choose a new point x', using a generating function.
$x' \leftarrow g(x)$

b) Calculate the function value of x'.
$E' \leftarrow f(x')$

c) Set $x \leftarrow x'$ and $E \leftarrow E'$ with probability determined by the acceptance function $h()$.

d) Reduce the temperature T by annealing, e.g.
$T(k+1) \leftarrow \mathrm{c}^* \, T(k)$, $0 < \mathrm{c} < 1$.

4) Return x and E as the optimal point and the optimal function value.

Because the algorithm occasionally chooses points *uphill* from its current point, it can escape from local minima and more effectively search the function space to find the global minimum. Thus, simulated annealing algorithms are often well suited to solving constrained nonlinear optimization problems. The approach taken here to solving difficult optimization problems is to augment the general simulated annealing method to improve its performance.

2. The VFSR algorithm

The Very Fast Simulated Reannealing (VFSR) algorithm [6, 9] is a new simulated annealing optimization method that stochastically and ergodically searches a solution space for the global minima. Preliminary research [14, 9] has been shown it to be superior to some evolutionary optimization methods such as Genetic Algorithms [5, 10]. VFSR has an exponential annealing schedule that causes the algorithm to run faster than Boltzmann Annealing (traditional Simulated Annealing) [4], and Cauchy Annealing (Fast Annealing) [15]. Unlike Boltzmann Annealing and Cauchy Annealing, VFSR was primarily designed to searching bounded and constrained search spaces.

This paper explores the effectiveness of the VFSR algorithm in solving difficult optimization problems and compares its performance with more established simulated annealing algorithms such as Boltzmann Annealing and Fast (Cauchy) Annealing. The VFSR approach presented here is quite new, and has not yet been thoroughly tested, analyzed and explored. It remains to be seen what future difficulties exist, and how well these difficulties will be overcome.

The Very Fast Simulated Reannealing algorithm was originally developed to fit empirical data to a D-dimensional function It was derived from the Boltzmann Annealing (BA) method, but was then generalized to apply to fitting multimodal and difficult to optimize more complex functions. The description of the VFSR method is similar to that of Ingber and Rosen [9].

In general, simulated Annealing consists of four functional relationships.

1. $g(x)$: The probability density function of the state space $x = \{x_i; i = 1 \ldots D\}$.

2. $h(x)$: The probability density function of accepting a new value given the just previous value.

3. $T(k)$: An annealing temperature (T) schedule in annealing-time steps k.

4. $f(x)$: The function to optimize.

The Boltzmann algorithm chooses a Gaussian probability density function,

$$g(\Delta x) = (2\pi T)^{-D/2} \exp[-\Delta x^2/(2T)] \;, \quad (1)$$

where Δx ($x_{new} - x$) is the deviation of x_{new} from the currently accepted point x. T represents the temperature in the system.

The acceptance probability $h(x)$ is based on the chances of obtaining a new point x_{new} relative to a previously accepted point x, based on the difference of their functional values. Typically, the acceptance function is defined as

$$h(x) = \frac{1}{1 + \exp(\Delta E/T)} \;, \quad (2)$$

where $E = f(x)$, and ΔE (i.e., $\Delta E = E_{new} - E$) represents the energy difference between the present and previous function values. Geman and Geman [3] have shown that for the Boltzmann Annealing generating function $g(x)$, that the system can find a global minimum of $f(x)$ if the annealing temperature is lowered at a rate of $T(k) = T_0/\ln k$ or slower.

A faster annealing schedule is obtained using a Cauchy distribution. Fast Annealing (FA) uses the Cauchy distribution, and is often superior to that of Boltzmann Annealing. The fatter tail of the Cauchy distribution allows it to test states further from it current local minima during the search process. To guarantee that the system will statistically find the global minimum, the annealing schedule for the Cauchy distribution is $T(k) = T_0/k$.

Fast Annealing has an annealing schedule exponentially faster than the method of Boltzmann Annealing. For a D dimensional search space, a D dimensional Cauchy distribution (created as D products of a one dimensional

Cauchy distributions) has a very slow annealing schedule, $T_i(k) = T_0/k^{1/D}$, which is still faster than BA.

The VFSR algorithm was designed to search for optimal function values in a constrained search space. New parameters (states or points) $x_i(k)$ in dimension i are generated at annealing-time k and are calculated using the random variable y_i, i.e., y_i's are repeatedly generated until a valid set of Dx_i parameters are found

$$\Delta x_i = y_i(B_i - A_i) \ . \tag{3}$$

$$y_i \in [-1, 1] \ ,$$

A_i and B_i are the minimum and maximum of the ith dimensional range and newly generated points x_i are constrained by

$$x \in [A_i, B_i] \ . \tag{4}$$

The VFSR generating distribution [6] is defined as

$$g_T(y) = \prod_{i=1}^{D} \frac{1}{2(|y_i| + T_i)\ln(1 + 1/T_i)}$$

To generate new points according to this distribution, new values of y_i are generated from a u_i from the uniform distribution $u_i \in U[0, 1]$, by

$$y_i = \operatorname{sgn}(u_i - \frac{1}{2})T_i[(1 + 1/T_i)^{|2u_i - 1|} - 1] \tag{5}$$

The associated annealing schedule for T_i

$$T_i(k) = T_i(0)\exp(-c_i k^{1/D}) \ . \tag{6}$$

ensures that a global minimum statistically can be obtained. Above, c_i is a user defined parameter. The same type of annealing schedule is used for the acceptance function h as is used for the generating function g.

A reannealing, or temperature rescaling, can often improve the search characteristics and overall performance of the system. Since different parameter dimensions often have different sensitivities in different areas of the state space, it is often advantageous to expand or contract the distribution of the coordinates/parameters of the newly generated states. The generating distribution used when searching a relatively flat and insensitive dimension should be wider than that of the distribution produced in a dimension more sensitive to change. The reannealing component of the VFSR algorithm periodically rescales the generating temperature by sampling the pseudo gradient of the deepest functional minimum found ($E^* = f(x^*)$) and determining the sensitivities $s_i = |\partial E^*/\partial x_i|$ of each parameter. The acceptance temperature is similarly rescaled. All temperatures are initialized to one at the beginning of the search process and periodically reannealed. A detailed description of the reannealing process can be found in [9]. The VFSR source code is also publically available [8]

3. A sample test problem

The performances of Boltzmann Annealing, Fast Annealing, and Very Fast Simulated Annealing algorithms were tested on an extremely difficult to minimize function. Figure 1 shows the one dimensional version of this function.

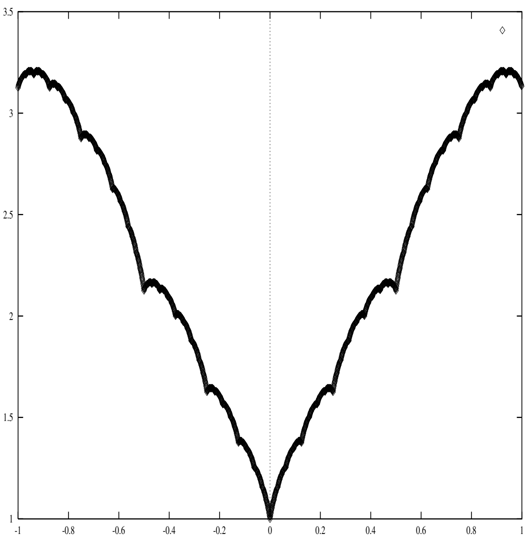

Figure 1: The Nowhere-differentiable test function (D=1).

The function

$$f(x) = \prod_{k=1}^{D}(1 + k \sum_{n=0}^{\beta} \frac{\left| 2^n x_i - \left\lfloor 2^i x_i \right\rfloor \right|}{2^n}) \tag{7}$$

(where $\beta = \infty$) is continuous but nowhere differentiable. It is a D dimensional variation of a contraction mapping defined in [11]. The test function has an infinite number of local minima. Figure 2 shows the two dimensional version of the test function.

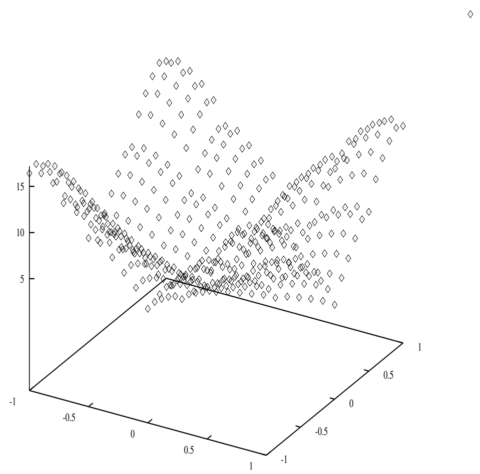

Figure 2: The Nowhere-differentiable test function (D=2).

The value of β was set to 35 to illustrate the behavior of the function in figure 1, and β was set to 30 for figure 2. For the actual optimization tests, β was set to 60. As β approaches ∞, the function resembles $f(x) = |x|$, but it is nondifferentiable. The actual test function was four dimensional, and the search space was bound by $-1000 \leq x_i \leq 1000$. The function minimum is located at the origin and has value unity. Unlike the VFSR runs, the Boltzmann and Fast Annealing runs had no reannealing component.

An exponential annealing schedule was used to determine acceptance temperature and to more accurately compare the VFSR runs with the FA and BA runs. For the VFSR runs, no parameter searches was necessary or performed, but for the Boltzmann and Fast Annealing tests, several runs were performed with different σ of the generating functions, and significant performance differences were found. The runs listed below used the best σ found.

Ten optimization runs of each algorithm were performed from the same initial state. Different random number generater seeds were used on each run. The number of states generated and the average minimum function value found (of each algorithm) before finding termination is shown in table 1 below.

Optimizing Algorithm	Mean States Generated	Mean Minimum
BA	13484	9.28e10
FA	9269	8.77e8
VFSR	22336	1.05

Table 1: Number of states generated and mean minimum value found.

Since there was a large variance in the BA and FA runs, table 2 shows the best minimum found of each of the runs.

Algorithm	States Generated	Best Minimum
BA	22336	1.017e8
FA	19923	17.018
VFSR	26404	1.019

Table 2: Number of states generated and best minimum value found.

4. Conclusions

The superiority of the VFSR algorithm can largely be attributed to the distribution function's fatter tail and adaptive nature. The generating distribution progressively contracts and (sometimes) expands depending upon the relative sensitivities of each dimension. This can be thought of as a second order or adaptive stepsize algorithm.

The progressive narrowing of each dimensional distribution allows the algorithm to generate random values with varying σs during the search. Hence the σ parameter (used in FA and BA above) need not be specified *a priori.*

The important contributions of the VFSR algorithm are 1) the faster reduction of the generating function temperature, 2) the automatic rescaling of each parameter's generating function variance, and 3) the use of the very fast distribution function defined in equation (5). In our test comparisons, both the Boltzmann and Fast Annealing algorithms were repeatedly caught in high local minima, while the VFSR algorithm was more adept in its search. Thus, for at least this test function, the VFSR algorithm was highly superior.

Acknowledgments

I would like to thank Lester Ingber for discussions on this and many other topics and his helpful comments and reading of this paper. I would like to thank Zbignew Wojcik his helpful reading and comments on this paper.

References

1. K. Binder and D. Stauffer, ‘‘A Simple Introduction to Monte Carlo Simulations and some Specialized Topics,’’ in *Applications of the Monte Carlo Method in Statistical Physics*, ed. K. Binder, Springer Verlag, Berlin, 1985.

2. K. Binder and A. P. Young, ‘‘Spin Glasses,’’ *Rev. Mod. Phys.*, vol. 58, pp. 801-976, 1986.

3. S. Geman and D. Geman, ‘‘Stochastic Relaxation, Gibbs Distribution and the Bayesian Restoration in Images,’’ *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721-741, 1984.

4. G. E. Hinton, T. J. Sejnowski, and D. H. Ackley, ‘‘Boltzmann Machines: Constraint Satisfaction Networks That Learn,’’ Tech. Rept. CMU-CS-84-119, 1984.

5. J. Holland, *Adaptation in Natural and Artificial Systems,* University of Michigan Press, Ann Arbor, MI, 1975.

6. L. Ingber, ‘‘Very Fast Simulated re-annealing,’’ *Mathl. Comput. Modeling*, vol. 12, no. 8, pp. 967-973, 1989.

7. L. Ingber, ‘‘Simulated Annealing: Practice versus Theory,’’ *Statistics and Computing*, (to appear) 1993.

8. L. Ingber and B. Rosen, ‘‘Very Fast Simulated Reannealing (VFSR),’’ netlib@research.att.com:/netlib/opt/vfsr.Z, AT&T Bell Labs, Murray Hill, NJ, 1992.

9. L. Ingber and B. E. Rosen, ‘‘Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison,’’ *Mathematical and Computer Modelling*, vol. 16, no. 11, pp. 87-100, 1992.

10. K. A. De Jong, ‘‘An Analysis of the Behavior of a Class of Genetic Adaptive system,’’ Ph.D. Dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1981.

11. H. Katsuura, ‘‘Continuous Nowhere-Differentiable Function -- An Application of Contraction Mappings,’’ *The American Mathematical Monthly*, vol. 98, no. 5, 1991.

12. S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, ‘‘Optimization by Simulated Annealing,’’ *Science*, vol. 220, no. 4598, pp. 671-680, 1983.

13. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, ‘‘Equation of State Calculations by Fast Computing Machines,’’ *Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087-1092, 1953.

14. B. E. Rosen, ‘‘GA’s and Very Fast Simulated Reannealing,’’ *Genetic Algorithms Digest*, vol. 5, no. 36, 1991. An electronic Journal.

15. H. Szu and R. Hartley, ‘‘Fast Simulated Annealing,’’ *Phys. Lett. A*, vol. 122, pp. 157-162, 1987.