Simulated Annealing 2

This chapter is dedicated to simulated annealing (SA) metaheuristic for optimization. SA is a probabilistic single-solution-based search method inspired by the annealing process in metallurgy. Annealing is a physical process where a solid is slowly cooled until its structure is eventually frozen at a minimum energy configuration. Various SA variants are also introduced.

2.1 Introduction

Annealing is referred to as tempering certain alloys of metal, glass, or crystal by heating above its melting point, holding its temperature, and then cooling it very slowly until it solidifies into a perfect crystalline structure. This physical/chemical process produces high-quality materials. The simulation of this process is known as *simulated annealing (SA)* [4,10]. The defect-free crystal state corresponds to the global minimum energy configuration. There is an analogy of SA with an optimization procedure. The physical material states correspond to problem solutions, the energy of a state to cost of a solution, and the temperature to a control parameter.

The Metropolis algorithm is a simple method for simulating the evolution to the thermal equilibrium of a solid for a given temperature [14]. SA [10] is a variant of the Metropolis algorithm, where the temperature is changing from high to low. SA is basically composed of two stochastic processes: one process for the generation of solutions and the other for the acceptance of solutions. The generation temperature is responsible for the correlation between generated probing solutions and the original solution.

SA is a descent algorithm modified by random ascent moves in order to escape local minima which are not global minima. The annealing algorithm simulates a nonstationary finite state Markov chain whose state space is the domain of the cost function to be minimized. Importance sampling is the main principle that underlies

K.-L. Du and M.N.S. Swamy, *Search and Optimization by Metaheuristics*,
DOI 10.1007/978-3-319-41192-7_2

SA. It has been used in statistical physics to choose sample states of a particle system model to efficiently estimate some physical quantities. Importance sampling probabilistically favors states with lower energies.

SA is a general-purpose, serial algorithm for finding a global minimum for a continuous function. It is also a popular Monte Carlo algorithm for any optimization problem including COPs. The solutions by this technique are close to the global minimum within a polynomial upper bound for the computational time and are independent of the initial conditions. Some parallel algorithms for SA have been proposed aiming to improve the accuracy of the solutions by applying parallelism [5].

2.2 Basic Simulated Annealing

According to statistical thermodynamics, P_α, the probability of a physical system being in state α with energy E_α at temperature T satisfies the Boltzmann distribution[1]

$$P_\alpha = \frac{1}{Z} \mathrm{e}^{\frac{-E_\alpha}{k_B} T}, \tag{2.1}$$

where k_B is the Boltzmann's constant, T is the absolute temperature, and Z is the partition function, defined by

$$Z = \sum_\beta e^{-\frac{E_\beta}{k_B} T}, \tag{2.2}$$

the summation being taken over all states β with energy E_β at temperature T. At high T, the Boltzmann distribution exhibits uniform preference for all the states, regardless of the energy. When T approaches zero, only the states with minimum energy have nonzero probability of occurrence.

In SA, the constant k_B is omitted. At high T, the system ignores small changes in the energy and approaches thermal equilibrium rapidly, that is, it performs a coarse search of the space of global states and finds a good minimum. As T is lowered, the system responds to small changes in the energy, and performs a fine search in the neighborhood of the already determined minimum and finds a better minimum. At $T = 0$, any change in the system states does not lead to an increase in the energy, and thus, the system must reach equilibrium if $T = 0$.

When performing SA, theoretically a global minimum is guaranteed to be reached with high probability. The artificial thermal noise is gradually decreased in time. T is a control parameter called *computational temperature*, which controls the magnitude of the perturbations of the energy function $E(\boldsymbol{x})$. The probability of a state change is determined by the Boltzmann distribution of the energy difference of the two states:

$$P = e^{-\frac{\Delta E}{T}}. \tag{2.3}$$

[1]Also known as the Boltzmann–Gibbs distribution.

The probability of uphill moves in the energy function ($\Delta E > 0$) is large at high T, and is low at low T. SA allows uphill moves in a controlled fashion: It attempts to improve on greedy local search by occasionally taking a risk and accepting a worse solution. SA can be performed as Algorithm 2.1 [10].

Algorithm 2.1 (SA).

```
1. Initialize the system configuration.
   Randomize $\boldsymbol{x}(0)$.
2. Initialize $T$ with a large value.
3. **Repeat**:
   a. **Repeat**:
      i. Apply random perturbations to the state $\boldsymbol{x} = \boldsymbol{x} + \Delta\boldsymbol{x}$.
      ii. Evaluate $\Delta E(\boldsymbol{x}) = E(\boldsymbol{x} + \Delta\boldsymbol{x}) - E(\boldsymbol{x})$:
          **if** $\Delta E(\boldsymbol{x}) < 0$, keep the new state;
          **otherwise**, accept the new state with probability $P = e^{-\frac{\Delta E}{T}}$.
      **until** the number of accepted transitions is below a threshold level.
   b. Set $T = T - \Delta T$.
   **until** $T$ is small enough.
```

The basic SA procedure is known as *Boltzmann annealing*. The cooling schedule for T is critical to the efficiency of SA. If T is reduced too rapidly, a premature convergence to a local minimum may occur. In contrast, if it is too slow, the algorithm is very slow to converge. Based on a Markov-chain analysis on the SA process, Geman and Geman [6] have proved that a simple necessary and sufficient condition on the cooling schedule for the algorithm state to converge in probability to the set of globally minimum cost states is that T must be decreased according to

$$T(t) \geq \frac{T_0}{\ln(1+t)}, \qquad t = 1, 2, \ldots \tag{2.4}$$

to ensure convergence to the global minimum with probability one, where T_0 is a sufficiently large initial temperature.

Given a sufficiently large number of iterations at each temperature, SA is proved to converge almost surely to the global optimum [8]. In [8], T_0 is proved to be greater than or equal to the depth of the deepest local minimum which is not a global minimum state. In order to guarantee Boltzmann annealing to converge to the global minimum with probability one, $T(t)$ needs to decrease logarithmically with time. This is practically too slow. In practice, one usually applies, in Step 3b, a fast schedule $T(t) = \alpha T(t-1)$ with $0.85 \leq \alpha \leq 0.96$, to achieve a suboptimal solution.

However, due to its Monte Carlo nature, SA would require for some problems even more iterations than complete enumeration in order to guarantee convergence to an exact solution. For example, for an n-city TSP, SA using the logarithmic cooling schedule needs a computational complexity of $O\left(n^{n^{2n-1}}\right)$, which is far more than $O((n-1)!)$ for complete enumeration and $O\left(n^2 2^n\right)$ for dynamic programming [1]. Thus, one has to apply heuristic fast cooling schedules to improve the convergence speed.

Example 2.1: We want to minimize the Easom function of two variables:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) = -\cos x_1 \cos x_2 \exp\left(-(x_1-\pi)^2 - (x_2-\pi)^2\right), \quad \boldsymbol{x} \in [-100, 100]^2.$$

The Easom function is plotted in Figure 2.1. The global minimum value is -1 at $\boldsymbol{x} = (\pi, \pi)^T$. This problem is hard since it has wide search space and the function rapidly decays to values very close to zero, and the function has numerous local minima with function value close to zero. This function is similar to a needle-in-a-hay function. The global optimum is restricted in a very small region.

MATLAB Global Optimization Toolbox provides a SA solver `simulannealbnd`, which assumes the objective function will take one input $\boldsymbol{x}$.

We implement `simulannealbnd` with the default settings: initial temperature of 100 for each dimension, temperature function as `temperatureexp` with a factor of 0.95. The SA solver always fails to find the global optimum after ten runs, when intial point $\boldsymbol{x}_0$ is randomly selected within the range $[-100, 100]^2$. Even if we set $\boldsymbol{x}_0 = [3, 3]$, which is very close to the global optimum, the algorithm still cannot find the global minimum.

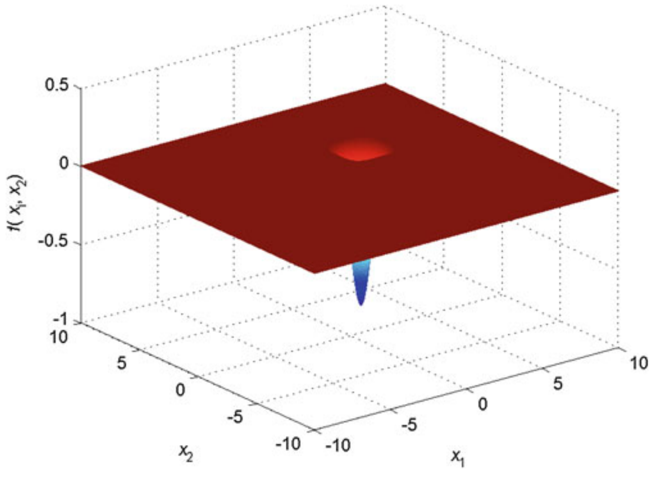

Figure 2.1 The Easom function when $\boldsymbol{x} \in [-10, 10]^2$.

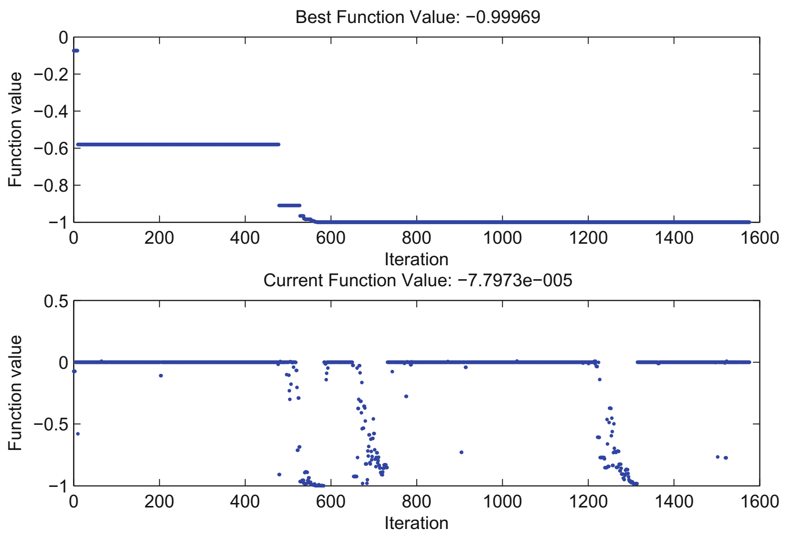

Figure 2.2 The evolution of a random run of simple GA: the minimum and average objectives.

After restricting the search space to $[-10, 10]^2$, and selecting a random intial point $\boldsymbol{x}_0 \in [-0.5, 0.5]^2$, we have the results of a random run as $f(\boldsymbol{x}) = -0.9997$ at $(3.1347, 3.1542)$ with 1597 function evaluations. The evolution of the `simulannealbnd` solver is given in Figure 2.2.

These results are very close to the global minimum.

2.3 Variants of Simulated Annealing

Standard SA is a stochastic search method, and the convergence to the global optimum is too slow for a reliable cooling schedule. Many methods, such as Cauchy annealing [18], simulated reannealing [9], generalized SA [19], and SA with known global value [13] have been proposed to accelerate SA search. There are also global optimization methods that make use of the idea of annealing [15,17].

Cauchy annealing [18] replaces the Boltzmann distribution with the Cauchy distribution, also known as the Cauchy–Lorentz distribution. The infinite variance provides a better ability to escape from local minima and allows for the use of faster schedules, such as T decreasing according to $T(t) = \frac{T_0}{t}$.

In simulated reannealing [9], T decreases exponentially with t:

$$T = T_0 e^{-\frac{c_1 t}{J}}, \tag{2.5}$$

where the constant $c_1 > 0$, and J is the dimension of the input space. The introduction of reannealing also permits adaptation to changing insensitivities in the multidimensional parameter space.

Generalized SA [19] generalizes both Cauchy annealing [18] and Boltzmann annealing [10] within a unified framework inspired by the generalized thermostatistics. Opposition-based SA [20] improves SA in accuracy and convergence rate using opposite neighbors.

An SA algorithm under the simplifying assumption of known global value [13] is the same as Algorithm 2.1 except that at each iteration a uniform random point is generated over a sphere whose radius depends on the difference between the current function value $E(\boldsymbol{x}(t))$ and the optimal value E^*, and T is also decided by this difference. The algorithm has guaranteed convergence and an upper bound for the expected first hitting time, namely, the expected number of iterations before reaching the global optimum value within a given accuracy [13].

The idea of annealing is a general optimization principle, which can be extended using fuzzy logic. In the fuzzy annealing scheme [15], fuzzification is performed by adding an entropy term. The fuzziness at the beginning of the entire procedure is used to prevent the optimization process getting stuck at an inferior local optimum. Fuzziness is reduced step by step. The fuzzy annealing scheme results in an increase in the computation speed by a factor of one hundred or more compared to SA [15].

Since SA works by simulating from a sequence of distributions scaled with different temperatures, it can be regarded as Markov chain Monte Carlo (MCMC) with a varying temperature. The stochastic approximation Monte Carlo (SAMC) algorithm [12] has a remarkable feature of its self-adjusting mechanism. If a proposal is rejected, the weight of the subregion that the current sample belongs to will be adjusted to a larger value, and thus the proposal of jumping out from the current subregion will be less likely rejected in the next iteration. Annealing SAMC [11] is a space annealing version of SAMC. Under mild conditions, it can converge weakly at a rate of $\Omega(1/\sqrt{t})$ toward a neighboring set (in the space of energy) of the global minimizers.

Reversible jump MCMC [7] is a framework for the construction of reversible Markov chain samplers that jump between parameter subspaces of differing dimensionality. The measure of interest occurs as the stationary measure of the chain. This iterative algorithm does not depend on the initial state. At each step, a transition from the current state to a new state is accepted with a probability. This acceptance ratio is computed so that the detailed balance condition is satisfied, under which the algorithm converges to the measure of interest. The proposition kernel can be decomposed into several kernels, each corresponding to a reversible move. In order for the underlying sampler to ensure the jump between different dimensions, the various moves used are the birth move, death move, split move, merge move, and perturb move, each with a probability of 0.2 [2]. SA with reversible-jump MCMC method [2] has proved convergence.

SA makes a random search on the energy surface. Deterministic annealing [16,17] is a deterministic method that replaces stochastic simulations by the use of expectation. It is a method where randomness is incorporated into the energy or cost function, which is then deterministically optimized at a sequence of decreasing temperature. The iterative procedure of deterministic annealing is monotone nonincreasing in the cost function. Deterministic annealing is able to escape local minima and reach a global solution quickly. The approach is derived within a probabilistic framework from basic information-theoretic principles (e.g., maximum entropy and random coding). The application-specific cost is minimized subject to a constraint on the randomness (Shannon entropy) of the solution, which is gradually lowered [17]. The annealing process is equivalent to computation of Shannon's rate-distortion function, and the annealing temperature is inversely proportional to the slope of the curve.

Parallel SA algorithms take advantage of parallel processing. In [3], a fixed set of samplers operates each at a different temperature. Each sampler performs the generate, evaluate, and decide cycle at a different temperature. A solution that costs less is propagated from the higher temperature sampler to the neighboring sampler operating at a lower temperature. Therefore, the best solution at a given time is propagated to all the samplers operating at a lower temperature. Coupled SA [21] is characterized by a set of parallel SA processes coupled by their acceptance probabilities. Coupling is performed by a term in the acceptance probability function, which is a function of the energies of the current states of all SA processes. The addition of the coupling and the variance control leads to considerable improvements with respect to the uncoupled case.

Problems

2.1 Implement SA to minimize the 5-dimensional Ackley function. The parameters are inverse cooling $\beta = 0.01$, initial temperature 100, iteration number 1000. Keep track of the best-so-far solution $\boldsymbol{x}_k^*$ as a function of the iteration number k for 10 runs. Plot the average value of x_k^* for the 10 runs.

2.2 Implement the `simulannealbnd` solver of MATLAB Global Optimization Toolbox for solving a benchmark function. Test the influence of different parameter settings.

2.3 Run the accompanying MATLAB code of SA to find the global minimum of six-hump-camelback function in the Appendix. Investigate how the parameters influence the performance.

References

1. Aarts E, Korst J. Simulated annealing and Boltzmann machines. Chichester: Wiley; 1989.
2. Andrieu A, de Freitas JFG, Doucet A. Robust full Bayesian learning for radial basis networks. Neural Comput. 2001;13:2359–407.
3. Azencott R. Simulated annealing: parallelization techniques. New York: Wiley; 1992.
4. Cerny V. Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl. 1985;45:41–51.

5. Czech ZJ. Three parallel algorithms for simulated annealing. In: Proceedings of the 4th international conference on parallel processing and applied mathematics, Naczow, Poland. London: Springer; 2001. p. 210–217.
6. Geman S, Geman D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans Pattern Anal Mach Intell. 1984;6:721–41.
7. Green PJ. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Biometrika. 1995;82:711–32.
8. Hajek B. Cooling schedules for optimal annealing. Math Oper Res. 1988;13(2):311–29.
9. Ingber L. Very fast simulated re-annealing. Math Comput Model. 1989;12(8):967–73.
10. Kirkpatrick S, Gelatt CD Jr, Vecchi MP. Optimization by simulated annealing. Science. 1983;220:671–80.
11. Liang F. Annealing stochastic approximation Monte Carlo algorithm for neural network training. Mach Learn. 2007;68:201–33.
12. Liang F, Liu C, Carroll RJ. Stochastic approximation in Monte Carlo computation. J Am Stat Assoc. 2007;102:305–20.
13. Locatelli M. Convergence and first hitting time of simulated annealing algorithms for continuous global optimization. Math Methods Oper Res. 2001;54:171–99.
14. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E. Equations of state calculations by fast computing machines. J Chem Phys. 1953;21(6):1087–92.
15. Richardt J, Karl F, Muller C. Connections between fuzzy theory, simulated annealing, and convex duality. Fuzzy Sets Syst. 1998;96:307–34.
16. Rose K, Gurewitz E, Fox GC. A deterministic annealing approach to clustering. Pattern Recognit Lett. 1990;11(9):589–94.
17. Rose K. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. Proc IEEE. 1998;86(11):2210–39.
18. Szu HH, Hartley RL. Nonconvex optimization by fast simulated annealing. Proc IEEE. 1987;75:1538–40.
19. Tsallis C, Stariolo DA. Generalized simulated annealing. Phys A. 1996;233:395–406.
20. Ventresca M, Tizhoosh HR. Simulated annealing with opposite neighbors. In: Proceedings of the IEEE symposium on foundations of computational intelligence (SIS 2007), Honolulu, Hawaii, 2007. p. 186–192.
21. Xavier-de-Souza S, Suykens JAK, Vandewalle J, Bolle D. Coupled simulated annealing. IEEE Trans Syst Man Cybern Part B. 2010;40(2):320–35.