Chapter 7

SIMULATED ANNEALING

Emile Aarts[1,2], Jan Korst[1], Wil Michiels[1,2]

[1] *Philips Research Laboratories, Eindhoven, the Netherlands*

[2] *Eindhoven University of Technology, Eindhoven, the Netherlands*

7.1 INTRODUCTION

Many problems in engineering, planning and manufacturing can be modeled as that of minimizing or maximizing a cost function over a finite set of discrete variables. This class of so-called combinatorial optimization problems has received much attention over the last two decades and major achievements have been made in its analysis (Papadimitriou and Steiglitz, 1982). One of these achievements is the separation of this class into two subclasses. The first one contains the problems that can be solved efficiently, i.e. problems for which algorithms are known that solve each instance to optimality in polynomial time. Examples are linear programming, matching and network problems. The second subclass contains the problems that are notoriously hard, formally referred to as NP-hard (see Chapter 11 for more details). For an NP-hard algorithm it is generally believed that no algorithm exists that solves each instance in polynomial time. Consequently, there are instances that require superpolynomial or exponential time to be solved to optimality. Many known problems belong to this class and probably the best known example is the traveling salesman problem. The above-mentioned distinction is supported by a general framework in computer science called complexity theory; for a detailed introduction and an extensive listing of provably hard problems see Garey and Johnson (1979) and Ausiello et al. (1999).

Clearly, hard problems must be handled in practice. Roughly speaking this can be done by two types of algorithms of inherently different nature: either one may use *optimization algorithms* that find optimal solutions possibly using large amounts of computation time or one may use *heuristic algorithms* that find approximate solutions in small amounts of computation time. Local search algorithms are of the latter type (Aarts and Lenstra, 2003). Simulated annealing, the subject of this chapter, is among the best known local search algorithms, since it performs quite well and is widely applicable. In this chapter we present the basics of simulated annealing. First, we introduce some elementary local search concepts. We introduce basic simulated annealing as an approach following directly from the strong analogy with the physical process of the annealing of solids. We analyze the asymptotic performance of basic simulated annealing. Next, we present some cooling schedules that allow for a finite-time implementation. Finally, we discuss some issues related to the practical use of simulated annealing and conclude with some suggestions for further reading.

7.2 LOCAL SEARCH

Local search algorithms constitute a widely used, general approach to hard combinatorial optimization problems. They are typically instantiations of various general search schemes, but all have the same feature of an underlying neighborhood function, which is used to guide the search for a good solution.

An instance of a combinatorial optimization problem consists of a set S of feasible solutions and a non-negative cost function f. The problem is to find a *globally optimal solution* $i^* \in S$, i.e. a solution with optimal cost f^*. A *neighborhood function* is a mapping $N : S \to 2^S$, which defines for each solution $i \in S$ a set $N(i) \subseteq S$ of solutions that are in some sense close to i. The set $N(i)$ is called the *neighborhood* of solution i, and each $j \in N(i)$ is called a *neighbor* of i. The simplest form of local search is called *iterative improvement*. An iterative improvement algorithm starts with an initial solution and then continuously explores neighborhoods for a solution with lower cost. If such a solution is found, then the current solution is replaced by this better solution. The procedure is continued until no better solutions can be found in the neighborhood of the current solution. By definition, iterative improvement terminates in a *local optimum*, i.e. a solution $\hat{i} \in S$ that is at least as good as all its neighbors with regard to the cost. Note that the concept of local optimality depends on the neighborhood function that is used.

For many combinatorial optimization problems one can represent solutions as sequences or collections of subsets of elements that constitute the solutions; examples are tours in the traveling salesman problem (TSP), partitions in the graph partitioning problem (GPP), and schedules in the job shop scheduling problem (JSSP). These solution representations enable the use of k-change neighborhoods, i.e. neighborhoods that are obtained by defining k exchanges of elements in a given sequence or collection of subsets. These so-called k-change neighborhoods constitute a class of basic local search algorithms that is widely applicable; see Lin (1965) and Lin and Kernighan (1973) for the TSP, Kernighan and Lin (1970) for the GPP, and Van Laarhoven et al. (1992) for the JSSP.

As an example we discuss the TSP. In an instance of the TSP we are given n cities and an $n \times n$-matrix $[d_{pq}]$, whose elements denote the distance from city p to city q for each pair p, q of cities. A tour is defined as a closed path visiting each city exactly once. The problem is to find a tour of minimal length. For this problem, a solution is given by a permutation $\pi = (\pi(1), \ldots, \pi(n))$. Each solution then corresponds uniquely to a tour. The solution space is given by

$$S = \{\text{all permutations } \pi \text{ on } n \text{ cities}\}$$

The cost function is defined as

$$f(\pi) = \sum_{i=1}^{n-1} d_{\pi(i),\pi(i+1)} + d_{\pi(n),\pi(1)}$$

i.e. $f(\pi)$ gives the length of the tour corresponding to π. Furthermore, we have $|S| = (n-1)!$

In the TSP, a neighborhood function N_k, called the k-change neighborhood, defines, for each solution i, a neighborhood S_i consisting of the set of solutions that can be obtained from the given solution i by removing $k' \leq k$ edges from the tour corresponding to solution i, replacing them with k' other edges such that again a tour is obtained, and choosing the direction of the tour arbitrarily (Lin, 1965; Lin and Kernighan, 1973). The simplest non-trivial version of this is the 2-change neighborhood. In that case we have

$$N = \{\pi' \in S \mid \pi' \text{ is obtained from } \pi \text{ by a 2-change}\}$$

and

$$|N| = 2 + n(n-3), \quad \text{for all } \pi \in S$$

Furthermore, it can be shown that each solution j can be obtained from any other solution i by at most $n-2$ successive 2-changes, which implies that the N_2 neighborhood graph is strongly connected.

In general, local search can be viewed as a walk in a *neighborhood graph*. The node set of the neighborhood graph is given by the set of solutions and there is an arc from node i to node j if j is a neighbor of i. The sequence of nodes visited by the search process defines a walk. Roughly speaking, the two main issues of a local search algorithm are the choice of the neighborhood function and the search strategy that is used. Good neighborhoods often take advantage of the combinatorial structure of the problem at hand, and are therefore typically problem dependent. A disadvantage of using iterative improvement as search strategy is that it easily gets trapped in poor local optima. To avoid this disadvantage—while maintaining the basic principle of local search algorithms, i.e. iteration among neighboring solutions—one can consider the extension of accepting, in a limited way, neighboring solutions that yield a deterioration in the value of the cost function. This in fact is the basic idea underlying simulated annealing.

7.3 BASIC SIMULATED ANNEALING

In the early 1980s Kirkpatrick et al. (1983) and independently Černý (1985) introduced the concepts of annealing in combinatorial optimization. Originally these concepts were heavily inspired by an analogy between the physical annealing process of solids and the problem of solving large combinatorial optimization problems. Since this analogy is quite appealing we use it here as a background for introducing simulated annealing.

In condensed matter physics, annealing is known as a thermal process for obtaining low energy states of a solid in a heat bath. The process consists of the following two steps (Kirkpatrick et al., 1983):

- increase the temperature of the heat bath to a maximum value at which the solid melts;
- decrease *carefully* the temperature of the heat bath until the particles arrange themselves in the ground state of the solid.

In the liquid phase, all particles arrange themselves randomly, whereas in the ground state of the solid, the particles are arranged in a highly structured lattice, for which the corresponding energy is minimal. The ground state of the solid is obtained only if the maximum value of the temperature is sufficiently high and the cooling is performed sufficiently slowly. Otherwise, the solid will be frozen into a meta-stable state rather than into the true ground state.

Metropolis et al. (1953) introduced a simple algorithm for simulating the evolution of a solid in a heat bath to thermal equilibrium. Their algorithm is based on Monte Carlo techniques (Binder, 1978) and generates a sequence of states of the solid in the following way. Given a current state i of the solid with energy E_i, then a subsequent state j is generated by applying a perturbation mechanism which transforms the current state into a next state by a small distortion, for instance by displacement of a particle. The energy of the next state is E_j. If the energy difference, $E_j - E_i$, is less than or equal to zero, the state j is accepted as the current state. If the energy difference is greater than zero, then the state j is accepted with a probability given by

$$\exp\left(\frac{E_i - E_j}{k_B T}\right)$$

where T denotes the temperature of the heat bath and k_B is a physical constant called the Boltzmann constant. The acceptance rule described above is known as the Metropolis criterion and the algorithm that goes with it is known as the Metropolis algorithm. It is known that, if the lowering of the temperature is done sufficiently slowly, the solid can reach thermal equilibrium at each temperature. In the Metropolis algorithm this is achieved by generating a large number of transitions at a given value of the temperature. Thermal equilibrium is characterized by the Boltzmann distribution, which gives the probability of the solid of being in a state i with energy E_i at temperature T, and which is given by

$$\mathbb{P}_T\{\mathbf{X} = i\} = \frac{\exp(-E_i/k_B T)}{\sum_j \exp(-E_j/k_B T)} \tag{7.1}$$

where $\mathbf{X}$ is a random variable denoting the current state of the solid and the summation extends over all possible states. As we indicate below, the Boltzmann distribution plays an essential role in the analysis of the convergence of simulated annealing.

Returning to simulated annealing, the Metropolis algorithm can be used to generate a sequence of solutions of a combinatorial optimization problem by assuming the following equivalences between a physical many-particle system and a combinatorial optimization problem:

- solutions in the combinatorial optimization problem are equivalent to states of the physical system;
- the cost of a solution is equivalent to the energy of a state.

Furthermore, we introduce a *control parameter* which plays the role of the temperature. Simulated annealing can thus be viewed as an iteration of Metropolis algorithms, executed at decreasing values of the control parameter.

We can now let go of the physical analogy and formulate simulated annealing in terms of a local search algorithm. To simplify the presentation, we assume, in the remainder of this chapter, that we are dealing with a minimization problem. The discussion easily translates to maximization problems. For an instance (S, f) of a combinatorial optimization problem and a neighborhood function N, Figure 7.1 describes simulated annealing in pseudo-code.

The meaning of the four functions in the procedure SIMULATED_ANNEALING is obvious: INITIALIZE computes a start solution and initial values of the parameters c and L; GENERATE selects a solution from the neighborhood of the current solution; CALCULATE_LENGTH and CALCULATE_CONTROL compute new values for the parameters L and c, respectively.

As already mentioned, a typical feature of simulated annealing is that, besides accepting improvements in cost, it also accepts deteriorations to a limited extent. Initially, at large values of c, large deteriorations will be accepted; as c decreases, only smaller deteriorations will be accepted and, finally, as the value

```
procedure SIMULATED_ANNEALING;
begin
    INITIALIZE ($i_{start}$, $c_0$, $L_0$);
    $k := 0$;
    $i := i_{start}$;
    repeat
        for $l := 1$ to $L_k$ do
        begin
            GENERATE ($j$ from $S_i$);
            if $f(j) \leq f(i)$ then $i := j$
            else
            if $\exp\left(\frac{f(i)-f(j)}{c_k}\right) > \text{random}[0, 1)$ then $i := j$
        end;
        $k := k + 1$;
        CALCULATE_LENGTH ($L_k$);
        CALCULATE_CONTROL ($c_k$);
    until stopcriterion
end;
```

Figure 7.1. The simulated annealing algorithm in pseudo-code.

of c approaches 0, no deteriorations will be accepted at all. Furthermore, there is no limitation on the size of a deterioration with respect to its acceptance. In simulated annealing, arbitrarily large deteriorations are accepted with positive probability; for these deteriorations the acceptance probability is small, however. This feature means that simulated annealing, in contrast to iterative improvement, can escape from local minima while it still exhibits the favorable features of iterative improvement, i.e. simplicity and general applicability.

Note that the probability of accepting deteriorations is implemented by comparing the value of $\exp((f(i) - f(j))/c)$ with a random number generated from a uniform distribution on the interval [0,1). Furthermore, it should be obvious that the speed of convergence of the algorithm is determined by the choice of the parameters L_k and c_k with $k = 0, 1, \ldots$, where L_k and c_k denote the values of L and k in iteration k of the algorithm. In the next section we will argue that under certain mild conditions on the choice of the parameters simulated annealing converges asymptotically to globally optimal solutions, and that it exhibits an equilibrium behavior from which some performance characteristics can be derived. In the subsequent section we present more practical, implementation-oriented choices of the parameter values that lead to a finite-time execution of the algorithm.

Comparing simulated annealing to iterative improvement, it is evident that simulated annealing can be viewed as a generalization. Simulated annealing becomes identical to iterative improvement in the case where the value of the control parameter is taken equal to zero. With respect to a comparison between the performance of both algorithms we mention that for most problems simulated annealing performs better than iterative improvement, repeated for a number of different initial solutions. We return to this subject in the concluding sections.

7.4 MATHEMATICAL MODELING

Simulated annealing can be mathematically modeled by means of Markov chains (Feller, 1950; Isaacson and Madsen, 1976; Seneta, 1981). In this model, we view simulated annealing as a process in which a sequence of Markov chains is generated, one for each value of the control parameter. Each chain consists of a sequence of trials, where the outcomes of the trials correspond to solutions of the problem instance.

Let (S, f) be a problem instance, N a neighborhood function, and $\mathbf{X}(k)$ a stochastic variable denoting the outcome of the kth trial. Then the *transition probability* at the kth trial for each pair $i, j \in S$ of outcomes is defined as

$$\begin{aligned} P_{ij}(k) &= \mathbb{P}\{\mathbf{X}(k) = j \mid \mathbf{X}(k-1) = i\} \\ &= \begin{cases} G_{ij}(c_k)A_{ij}(c_k) & \text{if } i \neq j \\ 1 - \sum\limits_{l \in S, l \neq i} G_{il}(c_k)A_{il}(c_k) & \text{if } i = j \end{cases} \end{aligned} \tag{7.2}$$

where $G_{ij}(c_k)$ denotes the *generation probability*, i.e. the probability of generating a solution j when being at solution i, and $A_{ij}(c_k)$ denotes the *acceptance probability*, i.e. the probability of accepting solution j, once it is generated from solution i. The most frequently used choice for these probabilities is the following (Aarts and Korst, 1989):

$$G_{ij}(c_k) = \begin{cases} |N(i)|^{-1} & \text{if } j \in S_i \\ 0 & \text{if } j \notin S_i \end{cases} \tag{7.3}$$

and

$$A_{ij}(c_k) = \begin{cases} 1 & \text{if } f(j) \leq f(i) \\ \exp((f(i) - f(j))/c) & \text{if } f(j) > f(i) \end{cases} \tag{7.4}$$

For fixed values of c, the probabilities do not depend on k, in which case the resulting Markov chain is *time-independent* or *homogeneous*. Using the theory of Markov chains it is fairly straightforward to show that, under the condition that the neighborhoods are strongly connected—in which case the Markov chain is *irreducible* and *aperiodic*—there exist a unique stationary distribution of the outcomes. This distribution is the probability distribution of the solutions after an infinite number of trials and assumes the following form (Aarts and Korst, 1989).

THEOREM 7.1 *Given an instance (S, f) of a combinatorial optimization problem and a suitable neighborhood function, then, after a sufficiently large number of transitions at a fixed value c of the control parameter, applying the transition probabilities of (7.2)–(7.4), simulated annealing will find a solution $i \in S$ with a probability given by*

$$\mathbb{P}_c\{\mathbf{X} = i\} \overset{\text{def}}{=} q_i(c) = \frac{1}{N_0(c)} \exp\left(-\frac{f(i)}{c}\right) \tag{7.5}$$

where $\mathbf{X}$ is a stochastic variable denoting the current solution obtained by simulated annealing and

$$N_0(c) = \sum_{j \in S} \exp\left(-\frac{f(j)}{c}\right) \tag{7.6}$$

denotes a normalization constant.

A proof of this theorem is beyond the scope of this chapter. For those interested, we refer to Aarts and Korst (1989). The probability distribution of (7.5) is called the stationary or equilibrium distribution and it is the equivalent of the Boltzmann distribution of (7.1). We can now formulate the following important result.

COROLLARY 7.2 *Given an instance (S, f) of a combinatorial optimization problem and a suitable neighborhood function, and, furthermore, let the probability $q_i(c)$ that simulated annealing finds solution i after an infinite number of trials at value c of the control parameter be given by (7.5), then*[1]

$$\lim_{c \downarrow 0} q_i(c) \overset{\text{def}}{=} q_i^* = \frac{1}{|S^*|} \chi_{(S^*)}(i) \tag{7.7}$$

where S^* denotes the set of globally optimal solutions.

[1]Let A and $A' \subset A$ be two sets. Then the characteristic function $\chi_{(A')} : A \to \{0, 1\}$ of the set A' is defined as $\chi_{(A')}(a) = 1$ if $a \in A'$, and $\chi_{(A')}(a) = 0$ otherwise.

Proof. Using the fact that for all $a \leq 0$, $\lim_{x \downarrow 0} e^{\frac{a}{x}} = 1$ if $a = 0$, and 0 otherwise, we obtain

$$\begin{aligned}
\lim_{c \downarrow 0} q_i(c) &= \lim_{c \downarrow 0} \frac{\exp\left(-\frac{f(i)}{c}\right)}{\sum_{j \in S} \exp\left(-\frac{f(j)}{c}\right)} \\
&= \lim_{c \downarrow 0} \frac{\exp\left(\frac{f^* - f(i)}{c}\right)}{\sum_{j \in S} \exp\left(\frac{f^* - f(j)}{c}\right)} \\
&= \lim_{c \downarrow 0} \frac{1}{\sum_{j \in S} \exp\left(\frac{f^* - f(j)}{c}\right)} \chi_{(S^*)}(i) \\
&\quad + \lim_{c \downarrow 0} \frac{\exp\left(\frac{f^* - f(i)}{c}\right)}{\sum_{j \in S} \exp\left(\frac{f^* - f(j)}{c}\right)} \chi_{(S \setminus S^*)}(i) \\
&= \frac{1}{|S^*|} \chi_{(S^*)}(i) + \frac{0}{|S^*|} \chi_{(S \setminus S^*)}(i)
\end{aligned}$$

which completes the proof. □

As already mentioned, the result of this corollary is important since it guarantees asymptotic convergence of the simulated annealing algorithm to the set of globally optimal solutions under the condition that the stationary distribution of (7.5) is attained at each value of c. More specifically, it implies that asymptotically optimal solutions are obtained which can be expressed as

$$\lim_{c \downarrow 0} \lim_{k \to \infty} \mathbb{P}_c\{\mathbf{X}(k) \in S^*\} = 1$$

We end this section with some remarks.

- It is possible to formulate a more general class of acceptance and generation probabilities than the ones we considered above, and prove asymptotic convergence to optimality in that case. The probabilities we used above are imposed by this more general class in a natural way and used in practically all applications reported in the literature.

- The simulated annealing algorithm can also be formulated as an *inhomogeneous algorithm*, namely as a single inhomogeneous Markov chain, where the value of the control parameter c is decreased in between subsequent trials. In this case, asymptotic convergence again can be proved.

However, an additional condition on the sequence $\{c_k\}$ of values of the control parameter is needed, namely

$$c_k \geq \frac{\Gamma}{\log(k+2)} \qquad k = 0, 1, \ldots$$

for some constant Γ that can be related to the neighborhood function that is applied.

- Asymptoticity estimates of the rate of convergence show that the stationary distribution of simulated annealing can only be approximated arbitrarily closely if the number of transitions is proportional to $|S|^2$. For hard problems, $|S|$ is necessarily exponential in the size of the problem instance, thus, implying that approximating the asymptotic behavior arbitrarily close results in an exponential-time execution of simulated annealing. Similar results have been derived for the asymptotic convergence of the inhomogeneous algorithm.

Summarizing, simulated annealing can find optimal solutions with probability 1 if it is allowed an infinite number of transitions. In Section 7.6 we show how a more efficient finite-time implementation of simulated annealing can be obtained. Evidently, this will be at the cost of the guarantee of obtaining optimal solutions. Nevertheless, practice shows that high-quality solutions can be obtained in this way.

7.5 EQUILIBRIUM STATISTICS

In this section, we discuss some characteristic features of simulated annealing under the assumption that we are at equilibrium, i.e. at the stationary distribution $q(c)$ given by (7.5). The expected cost $\mathbb{E}_c(f)$ at equilibrium is defined as

$$\begin{aligned} \mathbb{E}_c(f) &\overset{\text{def}}{=} \langle f \rangle_c \\ &= \sum_{i \in S} f(i) \mathbb{P}_c\{\mathbf{X} = i\} \\ &= \sum_{i \in S} f(i) q_i(c) \qquad (7.8) \end{aligned}$$

Similarly, the expected squared cost $\mathbb{E}_c(f^2)$ is defined as

$$\begin{aligned}\mathbb{E}_c(f^2) &\overset{\text{def}}{=} \langle f^2\rangle_c \\ &= \sum_{i\in S} f^2(i)\mathbb{P}_c\{\mathbf{X}=i\} \\ &= \sum_{i\in S} f^2(i)q_i(c) \end{aligned} \tag{7.9}$$

Using the above definitions, the variance $\text{Var}_c(f)$ of the cost is given by

$$\begin{aligned}\text{Var}_c(f) &\overset{\text{def}}{=} \sigma_c^2 \\ &= \sum_{i\in S}(f(i) - \mathbb{E}_c(f))^2\mathbb{P}_c\{\mathbf{X}=i\} \\ &= \sum_{i\in S}(f(i) - \langle f\rangle_c)^2 q_i(c) \\ &= \langle f^2\rangle_c - \langle f\rangle_c^2 \end{aligned} \tag{7.10}$$

The notation $\langle f\rangle_c$, $\langle f^2\rangle_c$, and σ_c^2 is introduced as shorthand notation, and will be used in the remainder of this paper.

COROLLARY 7.3 *Let the stationary distribution be given by (7.5), then the following relation holds:*

$$\frac{\partial}{\partial c}\langle f\rangle_c = \frac{\sigma_c^2}{c^2} \tag{7.11}$$

Proof. The relation can be straightforwardly verified by using (7.8) and by substituting the expression for the stationary distribution given by (7.5). □

COROLLARY 7.4 *Let the stationary distribution be given by (7.5). Then we have*

$$\lim_{c\to\infty}\langle f\rangle_c \overset{\text{def}}{=} \langle f\rangle_\infty = \frac{1}{|S|}\sum_{i\in S} f(i) \tag{7.12}$$

$$\lim_{c\downarrow 0}\langle f\rangle_c = f^* \tag{7.13}$$

$$\lim_{c\to\infty}\sigma_c^2 \overset{\text{def}}{=} \sigma_\infty^2 = \frac{1}{|S|}\sum_{i\in S}(f(i) - \langle f\rangle_\infty)^2 \tag{7.14}$$

and

$$\lim_{c \downarrow 0} \sigma_c^2 = 0 \tag{7.15}$$

Proof. The relations can be easily verified by using the definitions of the expected cost (7.8) and the variance (7.10), and by substituting the stationary distribution of (7.5) and applying similar arguments as in the proof of Corollary 7.11. □

Using (7.11), it follows that during execution of simulated annealing the expected cost decreases monotonically—provided equilibrium is reached at each value of the control parameter—to its final value, i.e. f^*. The dependence of the stationary distribution of (7.5) on the control parameter c is the subject of the following corollary.

COROLLARY 7.5 *Let (S, f) denote an instance of a combinatorial optimization problem with $S^* \neq S$, and let $q_i(c)$ denote the stationary distribution associated with simulated annealing and given by (7.5). Then we have*

(i) $\forall i \in S^*$

$$\frac{\partial}{\partial c} q_i(c) < 0$$

(ii) $\forall i \in S \backslash S^*, f(i) \geq \langle f \rangle_\infty$

$$\frac{\partial}{\partial c} q_i(c) > 0$$

(iii) $\forall i \in S \backslash S^*, f(i) < \langle f \rangle_\infty, \exists \tilde{c}_i > 0$

$$\begin{aligned} \frac{\partial}{\partial c} q_i(c) &< 0 \text{ if } c > \tilde{c}_i \\ &= 0 \text{ if } c = \tilde{c}_i \\ &> 0 \text{ if } c < \tilde{c}_i \end{aligned}$$

Proof. From (7.6) we can derive the following expression:

$$\frac{\partial}{\partial c} N_0(c) = \sum_{j \in S} \frac{f(j)}{c^2} \exp\left(\frac{-f(j)}{c}\right)$$

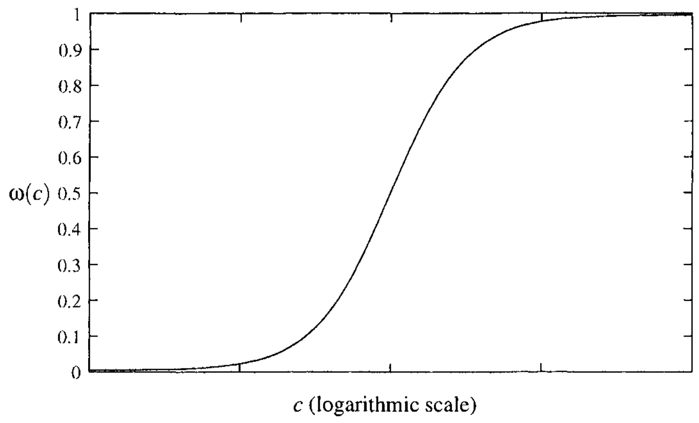

Figure 7.2. Acceptance ratio as function of the control parameter.

Hence, we obtain

$$\begin{aligned}
\frac{\partial}{\partial c} q_i(c) &= \frac{\partial}{\partial c} \frac{\exp\left(\frac{-f(i)}{c}\right)}{N_0(c)} \\
&= \left\{ \frac{f(i)}{c^2} \frac{\exp\left(\frac{-f(i)}{c}\right)}{N_0(c)} - \frac{\exp\left(\frac{-f(i)}{c}\right)}{N_0^2(c)} \frac{\partial}{\partial c} N_0(c) \right\} \\
&= \frac{q_i(c)}{c^2} f(i) - \frac{q_i(c)}{c^2} \frac{\sum_{j \in S} f(j) \exp\left(\frac{-f(j)}{c}\right)}{N_0(c)} \\
&= \frac{q_i(c)}{c^2} (f(i) - \langle f \rangle_c) \qquad (7.16)
\end{aligned}$$

Thus, the sign of $\frac{\partial}{\partial c} q_i(c)$ is determined by the sign of $f(i) - \langle f \rangle_c$ since $\frac{q_i(c)}{c^2} > 0$, for all $i \in S$ and $c > 0$.

From (7.11)–(7.13) we have that $\langle f \rangle_c$ increases monotonically from f^* to $\langle f \rangle_\infty$ with increasing c, provided $S^* \neq S$. The remainder of the proof is now straightforward.

If $i \in S^*$ and $S \neq S^*$, then $f(i) < \langle f \rangle_c$. Hence, $\frac{\partial}{\partial c} q_i(c) < 0$ (cf. (7.16)), which completes the proof of part (i). If $i \notin S^*$, then the sign of $\frac{\partial}{\partial c} q_i(c)$ depends on the value of $\langle f \rangle_c$. Hence, using (7.16), we have that $\forall i \in S \backslash S^* : \frac{\partial}{\partial c} q_i(c) > 0$ if $f(i) \geq \langle f \rangle_\infty$, whereas $\forall i \in S \backslash S^*$, where $f(i) < \langle f \rangle_\infty$, there exists a $\tilde{c}_i > 0$ at which $f(i) - \langle f \rangle_c$ changes sign. Conse-

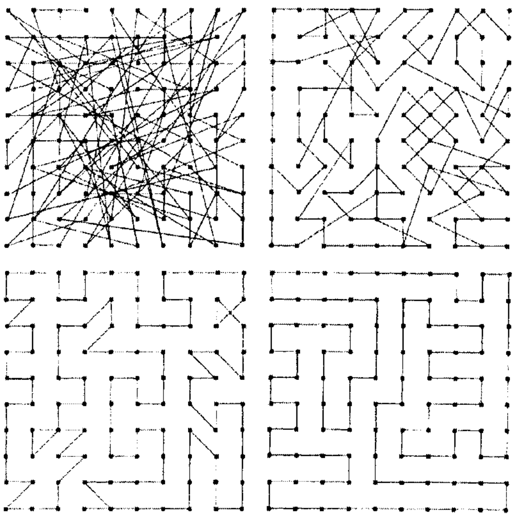

Figure 7.3. Evolution of simulated annealing for an instance with 100 cities on a regular grid.

quently, we have

$$\begin{aligned}\frac{\partial}{\partial c} q_i(c) &< 0 \text{ if } c > \tilde{c}_i \\ &= 0 \text{ if } c = \tilde{c}_i \\ &> 0 \text{ if } c < \tilde{c}_i\end{aligned}$$

which completes the proofs of parts (ii) and (iii). □

From Corollary 7.5 it follows that the probability of finding an optimal solution increases monotonically with decreasing c. Furthermore, for each solution, not being an optimal one, there exists a positive value of the control parameter $\tilde{c}_i$, such that for $c < \tilde{c}_i$, the probability of finding that solution decreases monotonically with decreasing c.

We complete this section with some results that illustrate some of the elements discussed in the analysis presented above. For this we need the definition of the *acceptance ratio* $\omega(c)$ which is defined as

$$\omega(c) = \left. \frac{\text{number of accepted transitions}}{\text{number of proposed transitions}} \right|_c \tag{7.17}$$

Figure 7.2 shows the behavior of the acceptance ratio as a function of the value of the control parameter for typical implementations of simulated annealing.

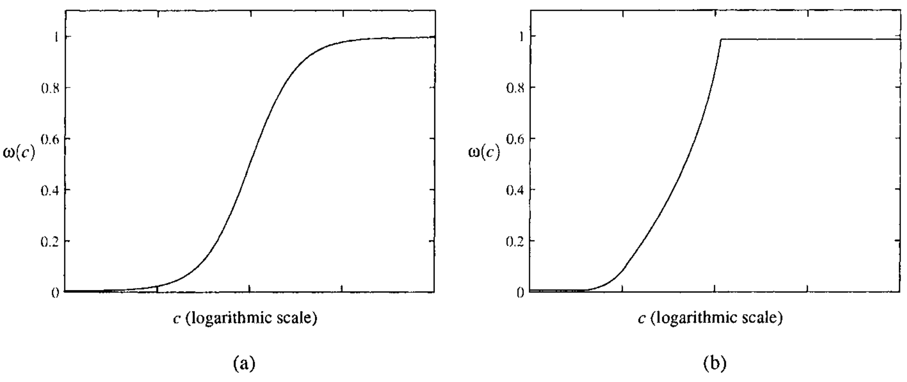

Figure 7.4. (a) Normalized average value $\frac{\langle f\rangle_c - f^*}{\langle f\rangle_\infty - f^*}$, and (b) normalized spreading $\frac{\sigma_c}{\sigma_\infty}$ of the cost function, both as a function of the control parameter.

The figure illustrates the behavior as it would be expected from the acceptance criterion given in (7.4). At large values of c, virtually all proposed transitions are accepted. As c decreases, ever fewer proposed transitions are accepted, and finally, at very small values of c, no proposed transitions are accepted at all.

Figure 7.3 shows four solutions in the evolution of simulated annealing running on a TSP instance with 100 cities on the positions of a 10×10 grid. The initial solution at the top left is given by a random sequence among 100 cities, which is evidently far from optimal. It looks very chaotic, and the corresponding value of the tour length is large. In the course of the optimization process the solutions become less and less chaotic (top right and bottom left), and the tour length decreases. Finally, the optimal solution shown at the bottom right is obtained. This solution has a highly regular pattern for which the tour length is minimal.

Figure 7.4 shows the typical behavior of (a) the normalized average cost and (b) the normalized spreading of the cost for simulated annealing as a function of the control parameter c. The typical behavior shown in this figure is observed for many different problem instances and is reported in the literature by a number of authors (Aarts et al., 1988; Hajek, 1985; Kirkpatrick et al., 1983; Van Laarhoven and Aarts, 1987; White, 1984).

From the figures we can deduce some characteristic features of the expected cost $\langle f\rangle_c$ and the variance σ_c^2 of the cost. First, it is observed that for large values of c the average and the spreading of the cost are about constant and equal to $\langle f\rangle_\infty$ and σ_∞, respectively. This behavior is directly explained from (7.12) and (7.14), from which it follows that both the average value and the spreading of the cost function are constant at large c-values.

Secondly, we observe that there exists a threshold value c_t of the control parameter for which

$$\langle f \rangle_{c_t} \approx \frac{1}{2}(\langle f \rangle_\infty + f^*) \tag{7.18}$$

and

$$\begin{aligned} \sigma_c^2 &\approx \sigma_\infty^2 \quad \text{if } c \geq c_t \\ &< \sigma_\infty^2 \quad \text{if } c < c_t \end{aligned} \tag{7.19}$$

Moreover, we mention that c_t is roughly the value of c for which $\omega(c) \approx 0.5$.

7.6 PRACTICAL APPLICATION

A finite-time implementation of simulated annealing is obtained by generating a sequence of homogeneous Markov chains of finite length at descending values of the control parameter. For this, a set of parameters must be specified that governs the convergence of the algorithm. These parameters are combined in what is called a cooling schedule. A *cooling schedule* specifies a finite sequence of values of the control parameter, and a finite number of transitions at each value of the control parameter. More precisely, it is specified by

- an *initial value* of the control parameter c_0,
- a *decrement function* for lowering the value of the control parameter,
- a *final value* of the control parameter specified by a *stop criterion*, and
- a finite *length* of each homogeneous Markov chain.

The search for adequate cooling schedules has been the subject of many studies over the past years. Reviews are given by Van Laarhoven and Aarts (1987), Collins et al. (1988), and Romeo and Sangiovanni-Vincentelli (1991). Below we discuss some results.

Most of the existing work on cooling schedules presented in the literature deals with heuristic schedules. We distinguish between two broad classes: static and dynamic schedules. In a static cooling schedule the parameters are fixed; they cannot be changed during execution of the algorithm. In a dynamic cooling schedule the parameters are adaptively changed during execution of the algorithm. Below we present some examples.

7.6.1 Static Cooling Schedules

The following simple schedule is known as the geometric schedule. It originates from the early work on cooling schedules by Kirkpatrick et al. (1983), and is still used in many practical situations.

Initial value of the control parameter. To ensure a sufficiently large value of $\omega(c_0)$, one may choose $c_0 = \Delta f_{\max}$, where $\Delta f_{\max}$ is the maximal difference in cost between any two neighboring solutions. Exact calculation of $\Delta f_{\max}$ is quite time consuming in many cases. However, one often can give simple estimates of its value.

Lowering the control parameter value. A frequently used decrement function is given by

$$c_{k+1} = \alpha \cdot c_k \quad k = 0, 1, \ldots$$

where α is a positive constant smaller than but close to 1. Typical values lie between 0.8 and 0.99.

Final value of the control parameter. The final value is fixed at some small value, which may be related to the smallest possible difference in cost between two neighboring solutions.

Markov chain length. The length of Markov chains is fixed by some number that may be related to the size of the neighborhoods in the problem instance at hand.

7.6.2 Dynamic Cooling Schedules

There exist many extensions of the simple static schedule presented above that lead to a dynamic schedule. For instance, a sufficiently large value of c_0 may be obtained by requiring that the initial acceptance ratio $\omega(c_0)$ is close to 1. This can be achieved by starting off at a small positive value of c_0 and multiplying it with a constant factor, larger than 1, until the corresponding value of $\omega(c_0)$, which is calculated from a number of generated transitions, is close to 1. Typical values of $\omega(c_0)$ lie between 0.9 and 0.99. An adaptive calculation of the final value of the control parameter may be obtained by terminating the execution of the algorithm at a c_k-value for which the value of the cost function of the solution obtained in the last trial of a Markov chain remains unchanged for a number of consecutive chains. Clearly, such a value exists for each local minimum that is found. The length of Markov chains may be determined by requiring that at each value c_k, a minimum number of transitions is accepted. However, since transitions are accepted with decreasing probability, one would obtain $L_k \to \infty$ for $c_k \downarrow 0$. Therefore, L_k is usually bounded by some constant $L_{\max}$ to avoid extremely long Markov chains for small values of c_k.

In addition to this basic dynamic schedule the literature presents a number of more elaborate schedules. Most of these schedules are based on a statistical analysis of simulated annealing using the equilibrium statistics of the previous section.

7.7 TRICKS OF THE TRADE

To apply simulated annealing in practice three basic ingredients are needed: a concise problem representation, a neighborhood, and a cooling schedule. The algorithm is usually implemented as a sequence of homogeneous Markov chains of finite length, generated at descending values of the control parameter. This is specified by the cooling schedule. As for the choice of the cooling schedule, we have seen in the previous section that there exist some general guidelines. However, for the other ingredients no general rules are known that guide their choice. The way they are handled is still a matter of experience, taste, and skill left to the annealing practitioner, and we expect that this will not change in the near future.

Ever since its introduction in 1983, simulated annealing has been applied to a large number of different combinatorial optimization problems in areas as diverse as operations research, VLSI design, code design, image processing, and molecular physics. The success of simulated annealing can be characterized by the following elements:

- performance, i.e. running time and solution quality,
- ease of implementation, and
- applicability and flexibility.

With respect to the last two items we make the following remarks. It is apparent that simulated annealing is conceptually very simple and quite easy to implement. Implementation of the algorithm typically takes only a few hundred lines of computer code. Experience shows that implementations for new problems often take only a few days.

With respect to applicability and flexibility it has become obvious as a result of the overwhelming amount of practical experience that has been gathered over the past 20 years that simulated annealing can be considered as one of the most flexible and applicable algorithms that exist. However, one must bear in mind that it is not always trivial to apply the algorithm effectively to a given problem. Finding appropriate neighborhoods requires problem insight, and sometimes it is necessary to reformulate the problem or transform it into an equivalent or similar problem, before simulated annealing can be applied successfully. An example is graph coloring (Korst and Aarts, 1989).

With respect to performance, one typically trades of solution quality against running times. Performance analyses of simulated annealing algorithms have been the subject of many studies. Despite numerous studies it is still difficult to judge simulated annealing on its true merits. This is predominantly due to the fact that many of these studies lack the depth required to draw reliable conclusions. For example, results are often limited to one single run of the algorithm, instead of taking the average over a number of runs, the applied cooling schedules are often too simple and do not get the best out of the algorithm, and results are often not compared to the results obtained with other (tailored) algorithms.

Lining up the most important and consistent results from the literature allows the following general observations.

- *Mathematical problems,* predominantly based on Johnson et al. (1989, 1991), and Van Laarhoven and Aarts (1987):
  - For graph partitioning problems, simulated annealing generally performs better, both with respect to error and running time, than the classical edge-interchange algorithms introduced by Kernighan and Lin (1970).
  - For a large class of basic problems, including the graph coloring, linear arrangement, matching, quadratic assignment, and scheduling problems, simulated annealing finds solutions with an error comparable to the error of tailored approximation algorithms but at the cost of much larger running times.
  - For some basic problems such as the number partitioning problem and the traveling salesman problem, simulated annealing is outperformed by tailored heuristics, both with respect to error and running time.
- *Engineering problems* (folklore). For many engineering problems, for example problems in the field of image processing, VLSI design and code design, no tailored approximation algorithms exist. For these problems simulated annealing seems to be a panacea. For instance, for the football pool problem (Van Laarhoven et al., 1989), it was able to derive solutions better than the best solutions found so far and for the VLSI placement problem (Sechen and Sangiovanni-Vincentelli, 1985) it outperforms the time-consuming manual process. For some problems, however, the running time can be very large.
- Comparing simulated annealing to time-equivalent iterative improvement using the same neighborhood function, i.e. repeating iterative improvement with different initial solutions for an equally long time as the running time of simulated annealing and keeping the best solution, reveals that simulated annealing performs substantially better (smaller error). This difference becomes even more pronounced for larger problem instances (Van Laarhoven et al., 1992; Van Laarhoven, 1988).
- Experience shows that the performance of simulated annealing depends as much on the skill and effort that is applied to the implementation as

on the algorithm itself. For instance, the choice of an appropriate neighborhood function, of an efficient cooling schedule, and of sophisticated data structures that allow fast manipulations can substantially reduce the error as well as the running time. Thus, in view of this and considering the simple nature of annealing, there lies a challenge in constructing efficient and effective implementations of simulated annealing.

7.8 CONCLUSIONS

Since its introduction in 1983, simulated annealing has been applied to a fairly large amount of different problems in many different areas. More than 20 years of experience had led to the following general observations.

- High-quality solutions can be obtained but sometimes at the cost of large amounts of computation time.
- In many practical situations, where no tailored algorithms are available, the algorithm is a real boon due to its general applicability and its ease of implementation.

So, simulated annealing is an algorithm that all practical mathematicians and computer scientists should have in their toolbox.

SOURCES OF ADDITIONAL INFORMATION

Introductory textbooks describing both theoretical and practical issues of simulated annealing are given by Aarts and Korst (1989) and Van Laarhoven and Aarts (1987). Salamon et al. (2002) present a basic text book on simulated annealing with recent improvements for practical implementations and references to software tools. Azencott (1992) presents a theoretical text book on parallelization techniques for simulated annealing for the purpose of speeding up the algorithm through effective parallel implementations.

Early proofs of the asymptotic convergence of the homogeneous Markov model for simulated annealing are presented by Aarts and Van Laarhoven (1985) and Lundy and Mees (1986). Proofs for the inhomogeneous algorithm have been published by Connors and Kumar (1987), Gidas (1985), and Mitra et al. (1986). Hajek (1988) was the first to present necessary and sufficient conditions for the asymptotic convergence of simulated annealing. Anily and Federgruen (1987) present theoretical results on the convergence of simulated annealing for a set of acceptance probabilities that are more general than the classical Metropolis acceptance probabilities. A comprehensive review of the theory of simulated annealing is given by Romeo and Sangiovanni-Vincentelli (1991).

Strenski and Kirkpatrick (1991) present an early analysis of the finite-time behavior of simulated annealing for various cooling schedules. Steinhöfel et al.

(1998) present a comparative study in which they investigate the performance of simulated annealing for different cooling schedules when applied to job shop scheduling. Nourani and Andersen (1998) present a comparative study in which they investigate the performance of simulated annealing with cooling schedules applying different types of decrement functions for lowering the value of the control parameter. Andersen (1996) elaborates on the thermodynamical analysis of finite-time implementations of simulated annealing. Orosz and Jacobson (2002) study the finite-time behavior of a special variant of simulated annealing in which the value of the control parameters is kept constant during the annealing process. Park and Kim (1998) present a systematic approach to the problem of choosing appropriate values for the parameters in a cooling schedule.

Vidal (1993) presents an edited collection of papers on practical aspects of simulated annealing, ranging from empirical studies of cooling schedules up to implementation issues of simulated annealing for problems in engineering and planning. Eglese (1990) presents a survey of the application of simulated annealing to problems in operations research. Collins et al. (1988) present an annotated bibliography with more than a thousand references to papers on simulated annealing. It is organized in two parts; one on the theory, and the other on applications. The applications range from graph-theoretic problems up to problems in engineering, biology, and chemistry. Fox (1993) discusses the integration of simulated annealing with other local search heuristics such as tabu search and genetic algorithms.

References

Aarts, E. H. L. and Korst, J. H. M., 1989, *Simulated Annealing and Boltzmann Machines,* Wiley, Chichester.

Aarts, E. H. L. and van Laarhoven, P. J. M., 1985, Statistical cooling: a general approach to combinatorial optimization problems, *Philips J. Res.* **40**:193–226.

Aarts, E. H. L., Korst, J. H. M. and van Laarhoven, P. J. M., 1988, A quantitative analysis of the simulated annealing algorithm: a case study for the traveling salesman problem, *J. Statist. Phys.* **50**:189–206.

Aarts, E. H. L. and Lenstra, J. K., eds, 2003, *Local Search in Combinatorial Optimization,* Princeton University Press, Princeton, NJ.

Andersen, B., 1996, Finite-time thermodynamics and simulated annealing, in: *Entropy and Entropy Generation,* J. S. Shiner, ed., Kluwer, Dordrecht, pp. 111–127.

Anily, S. and Federgruen, A., 1987, Simulated annealing methods with general acceptance probabilities, *J. Appl. Probab.* **24**:657–667.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A. and Protasi, M., 1999, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties,* Springer, Berlin.

Azencott, R., 1992, *Simulated Annealing: Parallelization Techniques,* Wiley, Chichester.

Binder, K., 1978, *Monte Carlo Methods in Statistical Physics,* Springer, Berlin.

Černý, V., 1985, Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm, *J. Optim. Theory Appl.* **45**:41–51.

Collins, N. E., Eglese, R. W. and Golden, B. L., 1988, Simulated annealing: An annotated bibliography, *Am. J. Math. Manage. Sci.* **8**:209–307.

Connors, D. P. and Kumar, P. R., 1987, Simulated annealing and balance of recurrence order in time-inhomogeneous Markov chains, in: *Proc. 26th IEEE Conf. on Decision and Control,* pp. 2261–2263.

Eglese, R. W., 1990, Simulated annealing: a tool for operational research, *Eur. J. Oper. Res.* **46**:271–281.

Feller, W., 1950, *An Introduction to Probability Theory and Its Applications I,* Wiley, New York.

Fox, B. L., 1993, Integrating and accelerating tabu search, simulated annealing, and genetic algorithms, in: *Tabu Search,* Annals of Operations Research, Vol. 41, F. Glover, E. Taillard, M. Laguna, and D. de Werra, eds, Baltzer, Basel, pp. 47–67.

Garey, M. R. and Johnson, D. S., 1979, *Computers and Intractability: A Guide to the Theory of NP-Completeness,* Freeman, San Francisco.

Gidas, B., 1985, Nonstationary Markov chains and convergence of the annealing algorithm, *J. Statist. Phys.* **39**:73–131.

Hajek, B., 1985, A tutorial survey of the theory and application of simulated annealing, in: *Proc. 24th IEEE Conf. on Decision and Control,* pp. 755–760.

Hajek, B., 1988, Cooling schedules for optimal annealing, *Math. Oper. Res.* **13**:311–329.

Isaacson, D. and Madsen, R., 1976, *Markov Chains,* Wiley, New York.

Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C., 1989, Optimization by simulated annealing: an experimental evaluation. I: graph partitioning, *Oper. Res.* **37**:865–892.

Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C., 1991, Optimization by simulated annealing: an experimental evaluation. II: graph coloring and number partitioning, *Oper. Res.* **39**:378–406.

Kernighan, B. W. and Lin, S., 1970, An efficient heuristic procedure for partitioning graphs, *Bell Syst. Tech. J.* **49**:291–307.

Kirkpatrick, S., Gelatt Jr, C. D. and Vecchi, M. P., 1983, Optimization by simulated annealing, *Science* **220**:671–680.

Korst, J. H. M. and Aarts, E. H. L., 1989, Combinatorial optimization on a Boltzmann machine, *J. Parallel Distrib. Comput.* **6**:331–357.

van Laarhoven, P. J. M., 1988, Theoretical and Computational Aspects of Simulated Annealing, *Ph.D. Thesis,* Erasmus University, Rotterdam.

van Laarhoven, P. J. M. and Aarts, E. H. L., 1987, *Simulated Annealing: Theory and Applications,* Reidel, Dordrecht.

van Laarhoven, P. J. M., Aarts, E. H. L. and Lenstra, J. K., 1992, Job shop scheduling by simulated annealing, *Oper. Res.* **40**:185–201.

van Laarhoven, P. J. M., Aarts, E. H. L., van Lint, J. H. and Wille, L. T., 1989, New upper bounds for the football pool problem for 6, 7 and 8 matches, *J. Combinat. Theor.* A **52**:304–312.

Lam, J. and Delosme, J.-M., 1986, Logic minimization using simulated annealing, in: *Proc. IEEE Int. Conf. on Computer-Aided Design,* pp. 348–351.

Lin, S., 1965, Computer solutions of the traveling salesman problem, *Bell Syst. Tech. J.* **44**:2245–2269.

Lin, S. and Kernighan, B. W., 1973, An effective heuristic algorithm for the traveling salesman problem, *Oper. Res.* **21**:498–516.

Lundy, M. and Mees, A., 1986, Convergence of an annealing algorithm, *Math. Programm.* **34**:111–124.

Metropolis, M., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E., 1953, Equation of state calculations by fast computing machines, *J. Chem. Phys.* **21**:1087–1092.

Mitra, D., Romeo, F. and Sangiovanni-Vincentelli, A. L., 1986, Convergence and finite-time behavior of simulated annealing, *Adv. Appl. Probab.* **18**:747–771.

Nourani, Y. and Andersen, B., 1998, A comparison of simulated annealing cooling strategies, *J. Phys.* A **31**:8373–8385.

Orosz, J. E. and Jacobson, S. H., 2002, Finite-time performance analysis of static simulated annealing algorithms, *Comput. Optim. Appl.* **21**:21–53.

Papadimitriou, C. H. and Steiglitz, K., 1982, *Combinatorial Optimization: Algorithms and Complexity,* Prentice-Hall, New York.

Park, M.-W. and Kim, Y.-D., 1998, A systematic procedure for setting parameters in simulated annealing algorithms, *Comput. Oper. Res.* **25**:207–217.

Romeo, F. and Sangiovanni-Vincentelli, A., 1991, A theoretical framework for simulated annealing, *Algorithmica* **6**:302–345.

Salamon, P., Sibani, P. and Frost, R., 2002, *Facts, Conjectures, and Improvements for Simulated Annealing,* SIAM Monographs.

Sechen, C. and Sangiovanni-Vincentelli, A. L., 1985, The Timber–Wolf placement and routing package, *IEEE J. Solid-State Circuits* **30**:510–522.

Seneta, E., 1981, *Non-negative Matrices and Markov Chains,* 2nd edn, Springer, New York.

Steinhöfel, K., Albrecht, A. and Wong, C. K., 1998, On various cooling schedules for simulated annealing applied to the job shop problem, in: *Randomization and Approximation Techniques in Computer Science,* Lecture Notes in Computing Science, Vol. 1518, M. Luby, J. Rolim, and M. Serna, eds, pp. 260–279.

Strenski, P. N. and Kirkpatrick, S., 1991, Analysis of finite length annealing schedules, *Algorithmica* **6**:346–366.

Vidal, R. V. V., ed., 1993, *Applied Simulated Annealing,* Lecture Notes in Economics and Mathematical Systems, Vol. 396, Springer, Berlin.

White, S. R., 1984, Concepts of scale in simulated annealing, in: *Proc. IEEE Int. Conf. on Computer Design* (Port Chester), pp. 646–651.