Chapter 10
Simulated Annealing

Emile Aarts, Jan Korst and Wil Michiels

10.1 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 years and major achievements have been made in its analysis (Ausiello et al. 1999). One of these achievements is the separation of this class into two subclasses. The first one contains the problems that can be efficiently solved, 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—and for which it is generally believed that no algorithms exist that solve 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 (TSP). The above-mentioned distinction is supported by a general discipline in computer science called complexity theory; for a detailed introduction and an extensive listing of provably hard problems see Garey and Johnson (1979) and Arora and Barak (2009).

E. Aarts
Eindhoven University of Technology, Eindhoven, The Netherlands
e-mail: e.h.l.aarts@tue.nl

J. Korst
Philips Research Laboratories, Eindhoven, The Netherlands
e-mail: jan.korst@philips.com

W. Michiels (✉)
NXP, Eindhoven, The Netherlands
e-mail: wil.michiels@philips.com

Clearly, also 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 relatively small amounts of computation time. Local search algorithms are of the latter type (Aarts and Lenstra 2003; Michiels et al. 2007). 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. The chapter summarizes the treatment of simulated annealing contained in Michiels et al. (2007). 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.

10.2 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. To make this more precise, we introduce in this section some notation and definitions.

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 $N : S \rightarrow 2^S$, 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 $\hat{\imath} \in S$ 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; examples are tours in the 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, where the k-change neighborhood $N(i)$ of a solution i is defined as the set of solutions that can be obtained from i by exchanging at most k elements. These k-change neighborhoods are widely applied; 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 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 can be written as a permutation $\pi = (\pi(1), \dots, \pi(n))$ as each permutation 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

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

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

For a TSP instance, the k-change neighborhood function N_k defines for each solution i a neighborhood N_k consisting of the set of solutions that can be obtained from the given solution i by removing $k' \le k$ 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

$$N_2(\pi) = \{\pi' \in S \mid \pi' \text{ is obtained from } \pi \text{ by a 2-exchange}\}$$

and

$$|N_2(\pi)| = 2 + n(n-3), \quad \text{for all } \pi \in S.$$

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 and only if j is a neighbor of i. The sequence of nodes visited by the search process defines the walk. Note that, as for the TSP, each solution j can be obtained from any other solution i by at most $n-2$ successive 2-changes, so the 2-change neighborhood graph is strongly connected. 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 a search strategy is that it easily gets trapped in poor local minima. 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 corresponding to a deterioration in the value of the cost function. This in fact is the basic idea underlying simulated annealing.

10.3 Basic Simulated Annealing

In the early 1980s Kirkpatrick et al. (1983) and independently Černý (1985) introduced the concept of annealing in combinatorial optimization. Originally this concept was 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 sufficiently slow. Otherwise the solid will be frozen into a meta-stable state rather than into the true ground state.

As far back as 1953, 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, 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 0, the state j is accepted as the current state. If the energy difference is greater than 0, 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 a physical constant known as 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 sufficiently slow, 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 temperature value. 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

$$\mathbb{P}_T\{\mathbf{X}=i\} = \frac{\exp(-E_i/k_BT)}{\sum\limits_j \exp(-E_j/k_BT)}, \tag{10.1}$$

where $\mathbf{X}$ is a random variable denoting the current state of the solid and the summation extends over all possible states. As we show 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 a combinatorial optimization problem are equivalent to states of a physical system.
- The cost of a solution is equivalent to the energy of a state.

Furthermore, we introduce a *control parameter* c which plays the role of the temperature. In this way simulated annealing can be viewed as an iteration of Metropolis algorithms, evaluated at decreasing values of the control parameter.

We 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 paper that we are dealing with a minimization problem. The discussion easily translates to maximization problems. Figure 10.1 describes simulated annealing in pseudo-code for an instance (S, f) of a combinatorial optimization problem and a neighborhood function N.

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, where L denotes the number of iterations at a given value of the control parameter c; 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 to a limited extent accepts deteriorations in cost. 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 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

```
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;
```

Fig. 10.1 The simulated annealing algorithm in pseudo-code

c 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 a globally optimal solution, 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 such that both algorithms have used the same computation time.

Figure 10.2 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 far from optimal evidently. It looks very chaotic; the corresponding value of the tour length is large. In the course of the optimization process the solutions become less

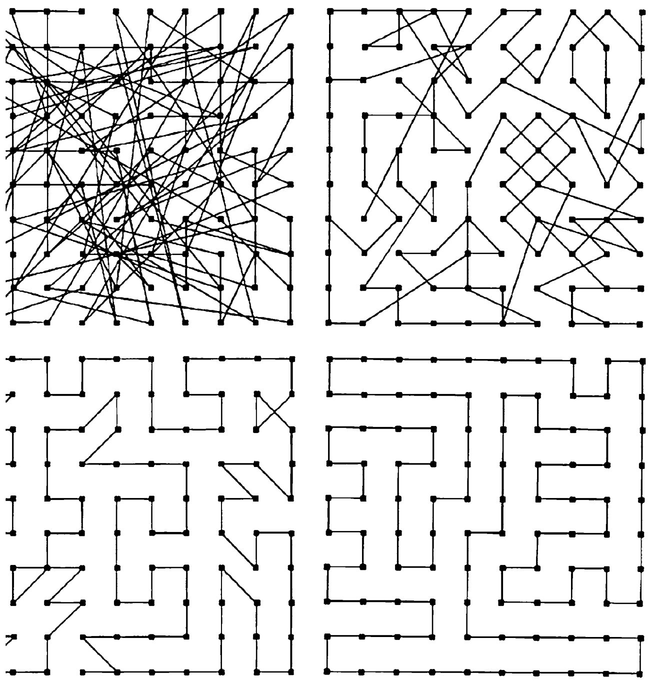
Fig. 10.2 Evolution of simulated annealing for an instance with 100 cities on a regular grid

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.

10.4 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. For each Markov chain the value of the control parameter is constant, but it decreases for successive Markov chains. Each chain consists of a sequence of trials, where the outcomes of the *i*th trial corresponds to the solution generated in the *i*th iteration at the considered value of the control parameter.

Let (S, f) be a problem instance, N be a neighborhood function, and $\mathbf{X}(k)$ be a stochastic variable denoting the outcome of the kth trial. Then the *transition probability* $P_{i,j}(k)$ is the probability to make a transition from solution i to solution j at the kth trial and it is given by

$$P_{ij}(k) = \mathbb{P}\{\mathbf{X}(k) = j | \mathbf{X}(k-1) = i\}$$

$$= \begin{cases} G_{ij}(c_k)A_{ij}(c_k) & \text{if } i \neq j \\ 1 - \sum\limits_{l \in S, l \neq i} G_{il}(c_k)A_{il}(c_k) & \text{if } i = j, \end{cases} \tag{10.2}$$

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

$$G_{ij}(c_k) = \begin{cases} |N(i)|^{-1} & \text{if } j \in S_i \\ 0 & \text{if } j \notin S_i \end{cases} \tag{10.3}$$

and

$$A_{ij}(c_k) = \begin{cases} 1 & \text{if } f(j) \leq f(i) \\ \exp((f(i) - f(j))/c) & \text{if } f(j) > f(i). \end{cases} \tag{10.4}$$

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 and not all solutions have the same cost—in which case the Markov chain is *irreducible* and *aperiodic*—there exists a unique stationary distribution of the outcomes. This distribution is the probability distribution of the solutions after an infinite number of trials. If $G_{i,j}(c) = G_{j,i}(c)$, then the distribution takes the following form (Aarts and Korst 1989).

Theorem 10.1. *Given an instance (S, f) of a combinatorial optimization problem in which not all solutions are optimal and a neighborhood function that induces a strongly connected neighborhood graph, then, after a sufficiently large number of transitions at a fixed value of c, applying the transition probabilities of (10.2)–(10.4), simulated annealing will find a solution $i \in S$ with a probability equal to*

$$\mathbb{P}_c\{\mathbf{X} = i\} \overset{def}{=} q_i(c) = \frac{1}{N_0(c)} \exp\left(-\frac{f(i)}{c}\right), \tag{10.5}$$

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

$$N_0(c) = \sum_{j \in S} \exp\left(-\frac{f(j)}{c}\right) \tag{10.6}$$

denotes a normalization constant.

A proof of this theorem is considered beyond the scope if this chapter. For those interested we refer to Michiels et al. (2007). The probability distribution of Eq. (10.5) is called the stationary or equilibrium distribution and it is the equivalent of the Boltzmann distribution of Eq. (10.1). Next we can formulate the following important result.

Corollary 10.1. *Given an instance (S, f) of a combinatorial optimization problem and a suitable neighborhood function, and furthermore let the stationary distribution be given by Eq. (10.5), then*

$$\lim_{c \downarrow 0} q_i(c) \overset{def}{=} q_i^* = \frac{1}{|S^*|} \chi_{(S^*)}(i), \tag{10.7}$$

where S^* denotes the set of globally optimal solutions.[1]

Proof. Using the fact that for all $a \leq 0, \lim_{x \downarrow 0} \mathrm{e}^{\frac{a}{x}} = 1$ 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 Eq. (10.5) is attained at each value of c. More specifically, it implies that asymptotically optimal solutions are obtained which can be expressed as

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

[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.

We end this section with some remarks:

- We can also prove asymptotic convergence to optimality in the case that the constraints on the generation probabilities are weakened to the extent that they only need to induce a symmetric neighborhood graph.
- 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 between subsequent trials. In this case, asymptotic convergence can again be proved. However, an additional condition on the sequence $\{c_k\}$ of values of the control parameter is needed, namely

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

  for some constant Γ that can be related to the neighborhood function that is applied.
- Asymptotic 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 one if it is allowed an infinite number of transitions and it can get arbitrarily close to an optimal solution if at least an exponential amount of transitions is allowed. In Sect. 10.6 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.

10.5 Equilibrium Statistics

In order to enhance our understanding of the algorithm, we discuss some characteristic features of simulated annealing under the assumption that we are at equilibrium, i.e. at the stationary distribution given by Eq. (10.5). The expected cost $\mathbb{E}_c(f)$ at equilibrium is defined as

$$\begin{aligned} \mathbb{E}_c(f) &\overset{\text{def}}{=} \langle f \rangle_c \\ &= \sum_{i \in S} f(i) \mathbb{P}_c\{\mathbf{X} = i\} \\ &= \sum_{i \in S} f(i) q_i(c). \end{aligned} \tag{10.8}$$

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

$$\begin{aligned}\mathbb{E}_c(f^2) &\stackrel{\text{def}}{=} \langle f^2\rangle_c \\ &= \sum_{i\in S} f^2(i)\mathbb{P}_c\{\mathbf{X}=i\} \\ &= \sum_{i\in S} f^2(i) q_i(c). \end{aligned} \tag{10.9}$$

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

$$\begin{aligned}\mathrm{Var}_c(f) &\stackrel{\text{def}}{=} \sigma_c^2 \\ &= \sum_{i\in S}(f(i)-\mathbb{E}_c(f))^2\mathbb{P}_c\{\mathbf{X}=i\} \\ &= \sum_{i\in S}(f(i)-\langle f\rangle_c)^2 q_i(c) \\ &= \langle f^2\rangle_c - \langle f\rangle_c^2. \end{aligned} \tag{10.10}$$

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

Corollary 10.2. *Let the stationary distribution be given by Eq. (10.5), then the following relation holds:*

$$\frac{\partial}{\partial c}\langle f\rangle_c = \frac{\sigma_c^2}{c^2}. \tag{10.11}$$

Proof. The relation can be straightforwardly verified by using the definition of Eq. (10.8) and substituting the expression for the stationary distribution given by Eq. (10.5). □

Corollary 10.3. *Let the stationary distribution be given by (10.5). Then we have*

$$\lim_{c\to\infty}\langle f\rangle_c \stackrel{def}{=} \langle f\rangle_\infty = \frac{1}{|S|}\sum_{i\in S} f(i) \tag{10.12}$$

$$\lim_{c\downarrow 0}\langle f\rangle_c = f^* \tag{10.13}$$

$$\lim_{c\to\infty}\sigma_c^2 \stackrel{def}{=} \sigma_\infty^2 = \frac{1}{|S|}\sum_{i\in S}(f(i)-\langle f\rangle_\infty)^2 \tag{10.14}$$

and

$$\lim_{c\downarrow 0}\sigma_c^2 = 0. \tag{10.15}$$

Proof. The relations can be easily verified by using the definitions of the expected cost (10.8) and the variance (10.10), and by substituting the stationary distribution of Eq. (10.5). □

Since $\frac{\partial}{\partial c}\langle f\rangle_c$ is strictly positive, as follows from Eq. (10.11), we get 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 Eq. (10.5) on the control parameter c is the subject of the following corollary.

Corollary 10.4. *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 (10.5). 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) $\forall i \in S \backslash S^*, f(i) < \langle f\rangle_\infty, \exists \tilde{c}_i > 0$

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

Proof. From (10.6) we can derive the following expression:

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

Hence, we obtain

$$\begin{aligned}\frac{\partial}{\partial c} q_i(c) &= \frac{\partial}{\partial c} \frac{\exp\left(\frac{-f(i)}{c}\right)}{N_0(c)} \\ &= \left\{ \frac{f(i)}{c^2} \frac{\exp\left(\frac{-f(i)}{c}\right)}{N_0(c)} - \frac{\exp\left(\frac{-f(i)}{c}\right)}{N_0^2(c)} \frac{\partial}{\partial c} N_0(c) \right\} \\ &= \frac{q_i(c)}{c^2} f(i) - \frac{q_i(c)}{c^2} \frac{\sum_{j \in S} f(j) \exp\left(\frac{-f(j)}{c}\right)}{N_0(c)} \\ &= \frac{q_i(c)}{c^2} (f(i) - \langle f\rangle_c). \end{aligned} \tag{10.16}$$

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 Eqs. (10.11) to (10.13) 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, $\frac{\partial}{\partial c}q_i(c) < 0$ (see Eq. (10.16)), 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 (10.16), 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. Consequently, we have

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

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

From Corollary 10.4 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 conclude 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

$$\omega(c) = \left. \frac{\text{number of accepted transitions}}{\text{number of proposed transitions}} \right|_c . \tag{10.17}$$

Figure 10.3 shows the behavior of the acceptance ratio as a function of the value of the control parameter for typical implementations of simulated annealing. The figure illustrates the behavior as it would be expected from the acceptance criterion given in Eq. (10.4). 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 10.4 shows the typical behavior of (a) the normalized average cost and (b) the normalized spread 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 spread of the cost are about constant and equal to $\langle f \rangle_\infty$ and σ_∞,

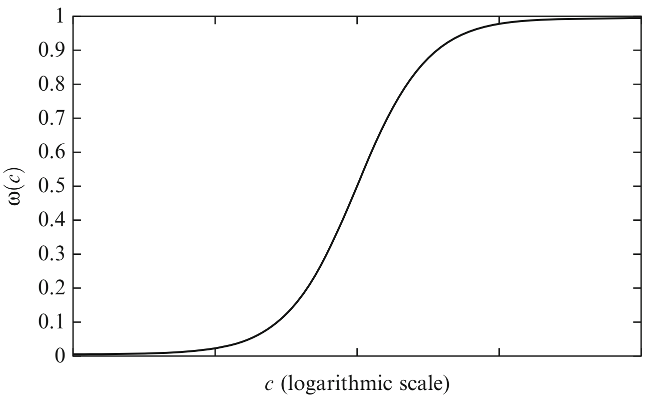

Fig. 10.3 Acceptance ratio as function of the control parameter

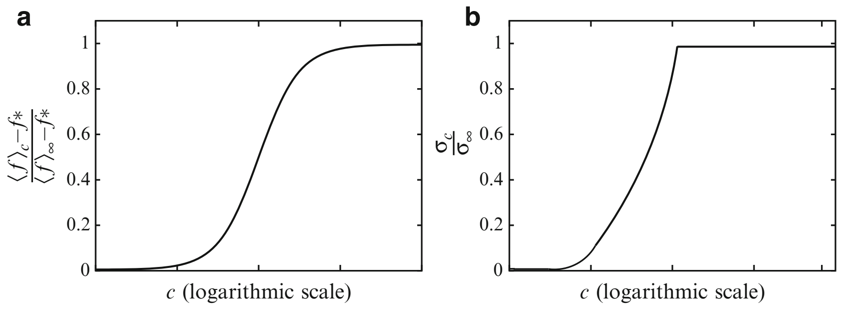

Fig. 10.4 (**a**) Normalized average value $\frac{\langle f\rangle_c - f^*}{\langle f\rangle_\infty - f^*}$, and (**b**) normalized spread $\frac{\sigma_c}{\sigma_\infty}$ of the cost function, as a function of the control parameter

respectively. This behavior is directly explained by Eqs. (10.12) and (10.14), 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

$$\langle f\rangle_{c_t} \approx \frac{1}{2}(\langle f\rangle_\infty + f^*) \tag{10.18}$$

and

$$\begin{aligned}\sigma_c^2 &\approx \sigma_\infty^2 \quad \text{if } c \geq c_t\\ &< \sigma_\infty^2 \quad \text{if } c < c_t.\end{aligned} \tag{10.19}$$

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

10.6 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. 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 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.

10.6.1 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.

10.6.1.1 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.

10.6.1.2 Lowering the Control Parameter Value

A frequently used decrement function is given by

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

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

10.6.1.3 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.

10.6.1.4 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.

10.6.2 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.

10.7 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 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 and in most cases existing programs, written for another problem, can be efficiently used.

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 30 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 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 (Michiels et al. 2007).

With respect to performance, one typically trades solution quality against running time. 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; results are often not compared to the results obtained with other (tailored) algorithms.

We conclude this section with two remarks.

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).

Finally, 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 allowing 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.

10.8 Conclusions

Since its introduction in 1983, simulated annealing has been applied to many different problems in many different areas. Thirty years of experience has 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 every practical mathematician and computer scientist should have in his toolbox.

Sources of Additional Information

Introductory textbooks describing both theoretical and practical issues of simulated annealing are given by Aarts and Korst (1989), van Laarhoven and Aarts (1987), and Michiels et al. (2007). Salamon et al. (2002) present a basic textbook on simulated annealing with improvements for practical implementations and references to software tools. Azencott (1992) presents a theoretical textbook 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 asymptotic convergence of the inhomogeneous model. Anily and Federgruen (1987) present theoretical results on the convergence of simulated annealing for a set of acceptance probabilities that are much more general than the classical Metropolis acceptance probabilities. Villalobos-Arias et al. (2006) prove asymptotic convergence of simulated annealing when applied to multi-objective optimization problems. 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 values of the control parameters are 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 theory, and the other on applications. The applications range from graph-theoretic problems 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 EHL, Korst JHM (1989) Simulated annealing and Boltzmann machines. Wiley, Chichester

Aarts EHL, van Laarhoven PJM (1985) Statistical cooling: a general approach to combinatorial optimization problems. Philips J Res 40:193–226

Aarts EHL, Lenstra JK (eds) (2003) Local search in combinatorial optimization. Princeton University Press, Princeton

Aarts EHL, Korst JHM, van Laarhoven PJM (1988) A quantitative analysis of the simulated annealing algorithm: a case study for the traveling salesman problem. J Stat Phys 50:189–206

Andersen B (1996) Finite-time thermodynamics and simulated annealing. In: Shiner JS (ed) Entropy and entropy generation. Kluwer, Dordrecht, pp 111–127

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

Arora S, Barak B (2009) Computational complexity: a modern approach. Cambridge University Press, Cambridge/New York

Ausiello G, Crescenzi P, Gambosi G, Kann V, Marchetti-Spaccamela A, 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 NE, Eglese RW, Golden BL (1988) Simulated annealing: an annotated bibliography. Am J Math Manage Sci 8:209–307

Connors DP, Kumar PR (1987) Simulated annealing and balance of recurrence order in time-inhomogeneous Markov chains. In: Proceedings of the 26th IEEE conference on decision and control, Los Angeles, pp 2261–2263

Eglese RW (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, vol 1. Wiley, New York

Fox BL (1993) Integrating and accelerating tabu search, simulated annealing, and genetic algorithms. In: Glover F et al (eds) Tabu search. Annals of operations research, vol 41. Baltzer, Basel, pp 47–67

Garey MR, Johnson DS (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 Stat Phys 39:73–131

Hajek B (1985) A tutorial survey of the theory and application of simulated annealing. In: Proceedings of the 24th IEEE conference on decision and control, Fort Lauderdale, pp 755–760

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

Isaacson D, Madsen R (1976) Markov chains. Wiley, New York

Kernighan BW, Lin S (1970) An efficient heuristic procedure for partitioning graphs. Bell Syst Tech J 49:291–307

Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680

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

Lin S, Kernighan BW (1973) An effective heuristic algorithm for the traveling salesman problem. Oper Res 21:498–516

Lundy M, Mees A (1986) Convergence of an annealing algorithm. Math Program 34:111–124

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

Michiels W, Aarts E, Korst J (2007) Theoretical aspects of local search. Springer, Berlin

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

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

Orosz JE, Jacobson SH (2002) Finite-time performance analysis of static simulated annealing algorithms. Comput Optim Appl 21:21–53

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

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

Salamon P, Sibani P, Frost R (2002) Facts, conjectures, and improvements for simulated annealing. SIAM Monographs, Philadelphia

Seneta E (1981) Non-negative matrices and Markov chains, 2nd edn. Springer, New York

Steinhöfel K, Albrecht A, Wong CK (1998) On various cooling schedules for simulated annealing applied to the job shop problem. In: Luby M et al (eds) Randomization and approximation techniques in computer science. Lecture notes in computer science, vol 1518. Springer, Berlin, pp 260–279

Strenski PN, Kirkpatrick S (1991) Analysis of finite length annealing schedules. Algorithmica 6:346–366

van Laarhoven PJM (1988) Theoretical and computational aspects of simulated annealing. PhD thesis, Erasmus University Rotterdam

van Laarhoven PJM, Aarts EHL (1987) Simulated annealing: theory and applications. Reidel, Dordrecht

van Laarhoven PJM, Aarts EHL, Lenstra JK (1992) Job shop scheduling by simulated annealing. Oper Res 40:185–201

Vidal RVV (ed) (1993) Applied simulated annealing. Lecture notes in economics and mathematical systems, vol 396. Springer, Berlin

Villalobos-Arias M, Coello CA, Hernandez-Lerma O (2006) Asymptotic convergence of a simulated annealing algorithm for multiobjective optimization problems. Math Methods Oper Res 64:353–362

White SR (1984) Concepts of scale in simulated annealing. In: Proceedings of the IEEE international conference on computer design, New York, pp 646–651