An Efficient Simulated Annealing Schedule

Jimmy Lam

Report 8818

September 1988

Department of Computer Science, Yale University.

Abstract

The popularity of simulated annealing comes from its ability to find close to optimal solutions for NP- hard combinatorial optimization problems. Unfortunately, the method has a major drawback: its massive requirement of computation time. In this dissertation, we present an efficient annealing schedule which speeds up simulated annealing by a factor of up to twenty-four when compared with general schedules currently available in the literature. The efficient annealing schedule, which lowers the temperature at every step and keeps the system in quasi-equilibrium at all times, is derived from a new quasi-equilibrium criterion. For a given move generation strategy and a given number of steps, this schedule is shown to give the minimum final average cost among all schedules that maintain the system in quasi-equilibrium. Furthermore, with the introduction of two models, we derive an alternate form of this schedule that relates move generation to temperature decrement. At every step, the move generation is controlled to minimize the response time of the system to a change of temperature, leading to the largest decrement in average cost while satisfying the quasi-equilibrium criterion. Most of the practical applications of simulated annealing have been in complicated problem domains, where algorithms either did not exist or performed poorly. To assess the performance of simulated annealing as a general method for solving combinatorial optimization problems, we also compare the method with efficient heuristics on well-studied problems: the traveling salesman problem and the graph partition problem. For high quality solutions and for problems with a small number of close to optimal solutions, our test results indicate that simulated annealing out-performs the heuristics by Lin and Kernighan and by Karp for the traveling salesman problem, and multiple executions of the heuristic by Fiduccia and Mattheyses for the graph partition problem.

This research was supported by the Army Research Office under contract DAAL03-86-K-0158, by the National Science Foundation under grant ECS-8314750, and by the Office of Naval Research under contracts N00014-84-K-0092 and N00014-85-K-0461.

An Efficient Simulated Annealing Schedule

A Dissertation

Presented to the Faculty of the Graduate School

of

Yale University

in Candidacy for the Degree of

Doctor of Philosophy

by

Jimmy Kwok-Ching Lam

December 1988

ACKNOWLEDGMENTS

I would like to thank Jean-Marc Delosme, my advisor, for his guidance, his constant support and his insights. He guided me through the process of establishing the research ideas, conducting this research, writing this dissertation, and polishing it over and over again till its final form. I also owe special thanks to Carl Sechen, Kai-Win Lee, Bill Swartz and Dahe Chen for their helpful discussions on TimberWolf.

I thank all the members of my committee, Jean-Marc Delosme, Sandeep Bhatt, William Gropp and Joel Saltz, for their helpful comments and constructive criticism.

I am indebted to my colleagues and friends especially Yiwan Wong, Arupjyoti Bhuyan, Ataullah and Hedyeh Arjomand for many lively discussions, and to Alfred Ganz for his technical assistance.

Most of all, I wish to thank my wife Rosanna who shares sorrow and happiness with me, my new-born son Hailey who gives my wife and me many sleepless nights and innumerable joyful moments, and my parents and grandmother for their support and love.

Table of Contents

CHAPTER 1

Method of Simulated Annealing

1.1. Introduction

The ground states of a complex physical system such as a solid can be reached by heating the system up to some high temperature and then cooling it down slowly. This process, called annealing, lets the system settle into a low energy state without getting trapped in a local minimum. The *simulated annealing method*, first proposed by Kirkpatrick *et al.* [Kirkp82] and, independently, by Cerny [Cerny85], exploits this analogy with physical systems in order to solve combinatorial optimization problems. (A combinatorial optimization problem is a minimization (maximization) problem consisting of three parts: a set of instances; a finite set of candidate solutions for each instance; and a cost function that assigns to each candidate solution for each instance a positive number called cost. The optimal solution to an instance of a minimization problem is the candidate solution having the minimum cost.) In the simulated annealing method, the cost function to be minimized is identified with the energy of a physical system, and the solution space is identified with the state space. The solution space of the optimization problem is explored by a probabilistic hill climbing search, whose step size is controlled by a parameter T that plays the role of the temperature in a physical system. By slowly lowering the temperature towards zero according to a properly chosen schedule, one can show that the globally optimal solutions are approached asymptotically.

In a typical execution of simulated annealing, the initial temperature is set sufficiently high. A new state (or solution) is generated incrementally from the current state by randomly selecting and proposing a move from a set of predefined moves. A move is a perturbation whose application to the current state leads to a new state. We

1. get an initial state with energy x.
2. make that initial state the current state.
3. select an initial "*high temperature*" T.
4. while the system is "*not yet frozen*" do
   begin
   while the system is "*not yet in thermal equilibrium*" do
      begin
      pick a random "*nearby*" state with energy x_p.
      let $\Delta x = x_p - x$.
      if $\Delta x \leq 0$
         the newly proposed state becomes the current state,
      else
         the newly proposed state becomes the current state
            with probability $= e^{-\Delta x / T}$
         else no change in state (i.e., reject state).
      end
   "*reduce the temperature* T *by* ΔT".
   end
5. output the current state.

Figure 1.1: The simulated annealing method.

shall call the move selection process the *move generation strategy.* Let the energy (or cost) of the current state be x and the energy of the new state be x_p. The probability that a proposed move is accepted or rejected in simulated annealing is determined by the Metropolis criterion [Metro53]:

$$p(\Delta x) = \min(1, e^{-\Delta x / T}), \tag{1.1}$$

where $\Delta x = x_p - x$ is the proposed energy change and $p(\Delta x)$ denotes the probability of accepting that move. If a proposed move is accepted, the new state becomes the current state; if a proposed move is rejected, the current state remains unchanged. By controlling the temperature T, we control the probability of accepting a hill climbing move (a move that results in a positive Δx) and, therefore, the exploration of the state space. This process of selecting and proposing a move is repeated until the system is considered in thermal equilibrium. Then, the temperature is reduced according to a temperature schedule called *simulated annealing schedule* (or annealing schedule in short), and the system is allowed to reach thermal equilibrium again. The system is considered frozen and the process is terminated when no significant improvement is

expected by further lowering the temperature. At this point, the current state of the system is the solution to the optimization problem. The simulated annealing method is summarized in Fig. 1.1. Note that when the temperature is lowered from infinity to zero, the exploration of the state space is changed gradually from random—all proposed moves are accepted—to greedy—only downhill moves are accepted.

Since its introduction, simulated annealing has been applied to optimization problems in diverse areas from code design [Beenk85], computer-aided IC design [Seche85], image restoration [Sonta85] to seismology [Jakob87]. (Refer to [Laarh87, Chapter 7] for an extensive list of applications.) The popularity of simulated annealing comes from its ability to find close to optimal solutions for NP-hard combinatorial optimization problems. Unfortunately, the method also has its drawbacks; one of the major obstacles towards successful application of simulated annealing to combinatorial optimization problems in general is its massive requirement of computation time, which arises from the probabilistic hill climbing nature of the method. In order to tackle this problem, current research in simulated annealing is concentrated in two areas: parallel implementation of simulated annealing [Aarts86], [Baner86], [Casot86], [Kravi86], and optimization of the simulated annealing schedule [Aarts85], [Mitra85], [Huang86]. This dissertation belongs to the latter category. For a review of current research in parallel implementation of simulated annealing, the interested reader is referred to [Laarh87, Chapter 8].

This dissertation is organized into six chapters. In the remainder of this chapter, we shall discuss some of the approaches used by researchers to obtain different annealing schedules. Representative annealing schedules from various approaches will be covered. In Chapter 2 we shall derive an intermediate form of our annealing schedule starting from an approximate thermal equilibrium criterion. Two models, one for the move generation strategy and one for the energy distribution of the states, will be introduced in Chapter 3. Using these models, we shall derive the final form of our annealing schedule. In Chapter 4 a procedure for the estimation and control of the parameters in our annealing schedule will be described together with other practical issues. The performance of our annealing schedule will be assessed in Chapter 5. Moreover, the

performance of simulated annealing as a general combinatorial optimization method will also be discussed. Finally, in Chapter 6 we shall give our concluding remarks.

1.2. Simulated annealing schedules

The simulated annealing method still raises many open questions: what is a sufficiently high initial temperature; how fast should the temperature be lowered; how is thermal equilibrium detected; what is the freezing temperature. Among these questions the most important and fundamental one is how to minimize the computation time while getting a good quality solution. In this section we discuss two approaches that lead to different annealing schedules. The first approach is the study of simulated annealing based on time-homogeneous and time-inhomogeneous Markov chains. The second approach is the study of simulated annealing based on an approximate equilibrium (or quasi-equilibrium) criterion.

For ease of presentation, we define the *inverse temperature* s as

$$s \equiv \frac{1}{T}.$$

Thus, a decrease in the temperature T corresponds to an increase in the inverse temperature s. We also assume time (or step) is discrete and is incremented whenever a proposed move is either accepted or rejected. Furthermore, we assume that the inverse temperature is a non-decreasing function of time, that is,

$$s_{n+k} \geq s_n, \quad k \geq 1, \tag{1.2}$$

where s_n is the inverse temperature at step n.

1.2.1. Markov chain approach

The analysis of simulated annealing based on time-homogeneous and time-inhomogeneous Markov chains [Isaac76] has been carried out independently in [Aarts85], [Hajek85], [Mitra85], and [Romeo85] among others. In this section we summarize some of their main results. For proofs of these results and a list of contributing authors, the reader is referred to [Laarh87, Chapter 3].

Let x_i be the energy of state i, $w_i(n)$ be the probability that the system is at state i at step n, and $g_{i,j}$ be the probability of proposing a move from state i to state j. Since the probability that a move from state i to state j is accepted is determined by the Metropolis criterion,

$$p(x_j - x_i) = \min(1, e^{-s(x_j - x_i)}),$$

the one-step transition probability from state i to state j at step n is

$$p_{i,j}(n) = g_{i,j} \min(1, e^{-s_n(x_j - x_i)}), \quad i \neq j. \tag{1.3}$$

If a proposed move is rejected, the system remains in the same state. Therefore, the probability of staying at state i is

$$p_{i,i}(n) = 1 - \sum_{j \neq i} p_{i,j}(n). \tag{1.4}$$

In matrix notation the probability vector of the system at step n is

$$\mathbf{W}(n) = \mathbf{P}(n)\mathbf{P}(n-1) \cdots \mathbf{P}(1)\mathbf{W}(0), \tag{1.5}$$

where $\mathbf{W}(n)$ is a column matrix with elements $w_i(n)$ and $\mathbf{P}(n)$ is a square matrix with elements $p_{i,j}(n)$. The preceding relation for a *time-inhomogeneous* Markov chain ($\mathbf{P}(n)$ is a function of time) is a precise mathematical model for simulated annealing. However, it is also of interest to study simulated annealing based on a *time-homogeneous* Markov chain in which the inverse temperature s_n is fixed at a particular value. The analysis of simulated annealing based on both Markov chains leads to a number of interesting theoretical results.

Let $\mathbf{P}^{[k]}(n) = \mathbf{P}(n+k-1)\mathbf{P}(n+k-2) \cdots \mathbf{P}(n)$ be the k-step transition probability matrix. The first result is that if the *irreducibility condition*

$$\forall i,j: p^{[k]}_{i,j}(n) > 0, \quad \text{for some finite } k, \tag{1.6}$$

and the *reversibility condition*

$$\forall i,j: g_{i,j} = g_{j,i}, \tag{1.7}$$

are satisfied, then it can be shown that the stationary (equilibrium) probability

distribution for a time-homogeneous Markov chain at a fixed inverse temperature s, $\Pi(s)$, is given by

$$\pi_i(s) = \frac{e^{-sx_i}}{z(s)}, \tag{1.8}$$

where $z(s) = \sum_{\forall i} e^{-sx_i}$ is known as the *partition function* for the system. In other words, the preceding expression gives the probability that a system is at state i with energy x_i when the system is in thermal equilibrium at inverse temperature s. Note that the stationary probability distribution is independent of $g_{i,j}$ and, hence, the move generation strategy.

The partition function $z(s)$ is a very useful quantity. From the partition function we can compute the equilibrium average energy of the system,

$$\mu(s) = \frac{\sum_{\forall i} x_i e^{-sx_i}}{z(s)} = -\frac{1}{z(s)}\frac{\partial z(s)}{\partial s}. \tag{1.9}$$

This expression can be generalized to obtain the n^{th} order moment of the energy,

$$M_n(s) = \frac{(-1)^n}{z(s)}\frac{\partial^n z(s)}{\partial s^n}.$$

Since the equilibrium variance of the energy, $\sigma^2(s)$, is given by

$$\sigma^2(s) = M_2(s) - \mu^2(s),$$

it is easy to verify that

$$\sigma^2(s) = -\frac{\partial \mu(s)}{\partial s}, \tag{1.10}$$

a relation which will be used frequently in the subsequent chapters.

The second result, which is based on the time-inhomogeneous Markov chain, is that the system converges to the globally optimal solutions asymptotically if the inverse temperature is changed according to the following annealing schedule:

$$s_n = \lambda \log(n+1), \quad n = 1, 2, \ldots, \tag{1.11}$$

where $\lambda \leq \lambda_{\max}$, and $\lambda_{\max}$ is a parameter that depends on the structure of the optimization problem. We shall call this annealing schedule the *logarithmic schedule.*

Although the logarithmic schedule can be shown to converge asymptotically to the globally optimal solutions, its computation time is higher for comparable results than other annealing schedules proposed in the literature. This is because the logarithmic schedule has to guarantee convergence even in the worst case of stepping into a deep local minimum, no matter how unlikely such an event is to occur. In order for the system to break away from this local minimum, the schedule has to raise the inverse temperature slowly and, therefore, has to settle for a slower rate of convergence. In Section 1.2.2 we discuss annealing schedules based on approximate equilibrium criteria. These annealing schedules no longer guarantee convergence to the globally optimal solutions, but give a better performance in practice.

1.2.2. Quasi-equilibrium approach

We present three annealing schedules based on different quasi-equilibrium criteria in this section. The first schedule is by Aarts and van Laarhoven [Aarts85]. Their analysis is also based on Markov chain theory. But instead of performing an exact asymptotic analysis of the behavior of simulated annealing, they introduce an approximate equilibrium criterion. They propose that the inverse temperature be raised after every k steps with the new inverse temperature, s_{n+k}, chosen such that the following quasi-equilibrium criterion is satisfied:

$$\|\Pi(s_n) - \Pi(s_{n+k})\| < \epsilon, \tag{1.12}$$

where ϵ is a small positive constant, $\Pi(s_n)$ is the stationary probability vector for the time-homogeneous Markov chain at inverse temperature s_n, and $\|\Pi\|$ is the 1-norm of vector Π. Based on this quasi-equilibrium criterion and under other assumptions, they derive an annealing schedule of the form

$$s_{n+k} = s_n + \frac{\lambda}{\sigma(s_n)}, \tag{1.13}$$

where λ is a user specified constant that is related to the ϵ in (1.12). Furthermore, they

argue that k, the number of steps before the temperature is changed, should be set to the maximum number of states that may be reached from any state in one step.

To determine the initial inverse temperature, s_0, they propose the following method. Assume that m_0 moves have been proposed and only $\rho_0 m_0$ of them are accepted. Of these m_0 moves, m_1 moves result in a positive proposed energy change, i.e., $\Delta x > 0$. From the Metropolis criterion it is easy to determine the equality

$$\rho_0 m_0 = (m_0 - m_1) + \sum_{\forall i:\, \Delta x_i > 0} e^{-s\,\Delta x_i},$$

where the summation is over the m_1 moves with positive proposed energy changes. The first term on the right hand side of the equality accounts for the number of accepted downhill moves while the second term accounts for the number of accepted uphill moves. In order to simplify the second term, they assume that

$$\sum_{\forall i:\, \Delta x_i > 0} e^{-s\,\Delta x_i} \approx m_1 e^{-s\,<\Delta x>^+},$$

where $<\Delta x>^+$ is the average value of those positive Δx_i's. With this assumption, an expression for the inverse temperature can be found:

$$s = \frac{\log[m_1 \,/\, (m_1 - (1-\rho_0)m_0)]}{<\Delta x>^+}. \tag{1.14}$$

Instead of specifying the initial inverse temperature, they suggest that the initial acceptance ratio, ρ_0, be specified. Then, equation (1.14) can be used to compute the inverse temperature after every step until its value is stable. This value will be taken as the initial inverse temperature, s_0.

To determine whether the system is frozen, they propose to use the slope of the *smoothed* average energy at different inverse temperatures. The idea is that as the system approaches its frozen state, the decrement of the smoothed average energy from one inverse temperature to the other is small. If this decrement is less than a given small value, the system is considered frozen and annealing is stopped. In order to minimize the statistical fluctuation of the measured average energy, they use the running average of the energy over a number of inverse temperatures as the smoothed average

energy.

Before proceeding to the next annealing schedule, we would like to comment on Aarts's quasi-equilibrium criterion and annealing schedule. We know from equation (1.5) that the probability vector of a system at step n, $\mathbf{W}(n)$, depends on the transition probability matrix $\mathbf{P}(r)$ for $r = 1, 2, \ldots, n$. Therefore, the rate of convergence of a time-homogeneous Markov chain towards its stationary probability vector also depends on the transition probability matrix. Now if we examine equation (1.3), we observe that the transition probability matrix depends on the inverse temperature and $g_{i,j}$. Since $g_{i,j}$ depends on the move generation strategy, the rate of convergence has to depend on the move generation strategy also. However, the annealing schedule in (1.13) and that the inverse temperature is changed after a *constant* number of steps do not account for either the dependency on the inverse temperature or the dependency on the move generation strategy.

Contrary to the two preceding annealing schedules which are based on Markov chain analysis on the state space, the next two annealing schedules are derived following White's approach [White84]. White has proposed to divide the energy values into small non-overlapping intervals of length δx and to group states according to their energy values. Let $P(x)$ be the *energy density function* defined as

$$P(x) \equiv \frac{\textit{number of states with energy in the interval containing } x}{\delta x \cdot \textit{total number of states}},$$

and let $P_s(x)$ be defined as

$$P_s(x) \equiv \frac{P(x)e^{-sx}}{Z(s)}, \tag{1.15}$$

with

$$Z(s) = \int_{-\infty}^{\infty} P(x)e^{-sx}\,dx. \tag{1.16}$$

Using these definitions, it is a simple matter to verify that the equilibrium average energy is given by

$$\mu(s) = -\frac{1}{Z(s)}\frac{\partial Z(s)}{\partial s}, \tag{1.17}$$

and, hence, the equilibrium variance of the energy is given by

$$\sigma^2(s) = -\frac{\partial \mu(s)}{\partial s}. \tag{1.18}$$

White has also proposed to model the energy density function as a normal density function:

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{1.19}$$

where $\mu = \mu(0)$ and $\sigma^2 = \sigma^2(0)$ are, respectively, the equilibrium average energy and the equilibrium variance of the energy at an inverse temperature of zero. With the normal density function, the probability that the energy of a system is in the interval $[\mu - \alpha\sigma, \mu + \alpha\sigma)$ is given by

$$p(\mu - \alpha\sigma \leq x < \mu + \alpha\sigma) = \mathrm{erf}(\alpha),$$

where $\mathrm{erf}(\alpha)$ is the error function [Felle70].

Huang *et al.* [Huang86] propose to detect thermal equilibrium by keeping track of two counters for the number of accepted moves that result in energy values within and without the interval $[\overline{X}(s) - \alpha\sigma, \overline{X}(s) + \alpha\sigma)$, where $\overline{X}(s)$ is the measured average energy at inverse temperature s. If the number of accepted moves within the interval reaches its target first, the system is considered in thermal equilibrium; if the number of accepted moves without the interval reaches its target first, both counters are reset to zero and the counting is initiated again. The target values for the within and without interval counters are set to $3\mathrm{erf}(0.5)N$ and $3(1 - \mathrm{erf}(0.5))N$, respectively, where N is a parameter measuring the size of the optimization problem. In order to guarantee the validity of $\overline{X}(s)$, Huang *et al.* invoke this detection mechanism only after a total of N moves has been accepted. It is possible that the system may never arrive at thermal equilibrium using this scheme. Therefore, the system is also considered in thermal equilibrium if the number of steps is greater than the *maximum generation limit*, M, where M is the number of states that may be reached in one move. This *dynamic*

equilibrium detection scheme is an improvement over Aarts's proposal of changing the inverse temperature after a constant number of steps, since it accounts for the dependency of the time to reach thermal equilibrium on the inverse temperature and on the move generation strategy.

Huang's annealing schedule is derived using the annealing curve—a curve of the equilibrium average energy, $\mu(s)$, versus the logarithm of the inverse temperature, $\log(s)$—to guide the temperature change. The idea is to control the inverse temperature so that $\mu(s)$ decreases in a uniform manner. From the slope of the annealing curve,

$$\frac{\partial \mu(s)}{\partial \log(s)} = s \frac{\partial \mu(s)}{\partial s},$$

which upon substitution of (1.18), leads to the relation

$$\frac{\partial \mu(s)}{\partial \log(s)} = -s\,\sigma^2(s).$$

The left hand side of this relation can be approximated by $(\mu(s_{n+k})-\mu(s_n))$ / $(\log(s_{n+k})-\log(s_n))$ to obtain

$$\frac{\mu(s_{n+k}) - \mu(s_n)}{\log(s_{n+k}) - \log(s_n)} = -s_n\,\sigma^2(s_n)$$

(In this scheme the inverse temperature is changed after every k steps with the value of k dynamically determined by the equilibrium detection mechanism.) To maintain quasi-equilibrium, Huang *et al.* require that the new inverse temperature, s_{n+k}, be chosen such that

$$\mu(s_n) - \mu(s_{n+k}) = \lambda\sigma(s_n) \tag{1.20}$$

with $\lambda \leq 1$. This leads to an annealing schedule of the form

$$s_{n+k} = s_n\, e^{\frac{\lambda}{s_n \sigma(s_n)}}. \tag{1.21}$$

Note that for small values of $|\,\lambda\,/\,(s_n\,\sigma(s_n))\,|$, the exponential function in the preceding expression can be approximated by $1 + \lambda\,/\,(s_n\,\sigma(s_n))$. Huang's annealing schedule then becomes

$$s_{n+k} = s_n + \frac{\lambda}{\sigma(s_n)}$$

which seems equivalent to Aarts's annealing schedule in (1.13). However, as mentioned earlier, there is a significant difference: the selection of k, the number of steps before the inverse temperature is changed, is static for Aarts's schedule and dynamic for Huang's schedule. Another difference between the two schedules is that Huang *et al.* assume that $\sigma(s_n) = \sigma(0)$ in their implementation, although in typical applications of simulated annealing the values of $\sigma(s_n)$ can differ by a couple orders of magnitude. In order to compensate for this discrepancy, we shall see, in Chapter 5, that the value of λ for Huang's schedule has to be set a couple orders of magnitude larger than 1 to result in an acceptable amount of computation time.

Huang's choice of initial inverse temperature is similar to White's. The system is first randomized at $s_0 = 0$ by accepting all proposed moves. Then, the next inverse temperature is chosen using the formula

$$s_1 = \frac{1}{k\,\sigma(0)} \tag{1.22}$$

with $k = 20$. The rationale behind this formula is as follows. Assume that the energy density function can be approximated by a normal density function as in (1.19). The equilibrium average energy at inverse temperature s can be computed from (1.16) and (1.17) and is given by

$$\mu(s) = \mu - s\,\sigma^2. \tag{1.23}$$

Substituting the inverse temperature s_1 into the preceding expression, we obtain

$$\mu(s_1) = \mu - \frac{\sigma}{k}.$$

Thus, $\mu(s_1)$ is within a fraction of a standard deviation from μ, and the transition from an inverse temperature of zero to the first inverse temperature, s_1, is smooth.

To determine whether the system is frozen, Huang *et al.* propose the following method. When the system is considered in thermal equilibrium, they compare the difference between the maximum and the minimum energy values at that inverse

temperature with the maximum change in energy in any accepted move at that inverse temperature. If the two are equal, they consider that all the states visited are of comparable energy values (all the states visited can be reached in one move) and, therefore, the system is frozen.

Thus far, we have discussed two quasi-equilibrium criteria that depend only on the equilibrium properties of the system. Although Huang's equilibrium detection scheme accounts for the dynamics of the system (the dependency of the time to reach thermal equilibrium on the inverse temperature and the move generation strategy), it is better and cleaner to take care of the dynamics of the system in a quasi-equilibrium criterion. Let $\overline{X}(s_{n-1})$ be the average energy of the system—not necessarily in thermal equilibrium—at step $n-1$. We consider the system is in quasi-equilibrium at inverse temperature s_n [Lam86] if it satisfies the relation

$$\left|\overline{X}(s_{n-1}) - \mu(s_n)\right| \leq \lambda\sigma(s_n). \tag{1.24}$$

In contrast to Aarts's and Huang's criteria which depend only on the stationary statistics of the system, our quasi-equilibrium criterion also depends on the average energy of the system which, in turn, depends on the move generation strategy. Since a good move generation strategy lets the system reach quasi-equilibrium faster, we can change the inverse temperature faster or in larger increments. This dependency between the move generation strategy and the temperature change is accounted for by our quasi-equilibrium criterion. From this quasi-equilibrium criterion, we derive, in Chapter 2, an annealing schedule that raises the inverse temperature *at every step* and keeps the system in quasi-equilibrium at all times, thus avoiding the need of an equilibrium detection scheme. Furthermore, for a given move generation strategy, a given λ and a given number of steps, this schedule is also shown to give the *minimum* final average energy among all schedules that maintain the system in quasi-equilibrium at all times. The intermediate form of this annealing schedule is

$$s_{n+1} = s_n + \lambda \frac{\rho_2(s_n)}{2\sigma^3(s_n)},$$

where $\rho_2(s_n)$ is the variance of the actual energy change in one step at inverse temperature s_n. The reader is referred to [Salam87] for insights behind this annealing schedule from the thermodynamic point of view.

In Chapter 3 we introduce two models: one for the energy density function and one for the move generation strategy. Based on these two models, we derive the final form of our annealing schedule:

$$s_{n+1} = s_n + \left[\frac{\lambda}{\sigma(s_n)}\right]\left[\frac{1}{{s_n}^2\sigma^2(s_n)}\right]\left[\frac{4\rho_0(1-\rho_0)^2}{(2-\rho_0)^2}\right], \tag{1.25}$$

where ρ_0 is the acceptance ratio (the probability that a proposed move is accepted). The increment in inverse temperature can be decomposed into three factors. The first factor, which depends on the quasi-equilibrium criterion, is the same as the factors in Aarts's and Huang's annealing schedules. The second factor relates the specific heat, which is the rate of change of the equilibrium average energy with respect to the temperature,

$$\frac{\partial\mu(s)}{\partial(1/s)} = {s}^2\sigma^2(s),$$

to the temperature decrement. The presence of this factor is in agreement with the common belief that the larger is the specific heat, the slower should be the cooling. The third factor relates the acceptance ratio and, hence, the move generation strategy to the temperature decrement. As will be derived in Chapter 3, to maximize this factor, the move generation strategy should be controlled in order to arrive at a desired acceptance ratio of 44%. This is similar to the suggestion made by Binder [Binde86, p. 11] on Monte Carlo methods in statistical physics that the magnitude of the proposed energy change should be chosen such that the acceptance ratio is close to 50%. Thus, not only the dynamics of the system is accounted for by our annealing schedule, but also the move generation strategy is actively controlled.

Our choice of initial inverse temperature is similar to Huang's with $s_0 = 0$ and $s_1 = 1 / (2\sigma)$, but our choice of frozen detection scheme is different. We have experimented with Aarts's and Huang's frozen detection methods as well as the simpler scheme of stopping annealing when the energy remains unchanged for the last few hundred moves. We did not observe any significant difference among the final results of these schemes. We, therefore, consider that the system is frozen and stop annealing when the energy of the last few hundred moves remains unchanged.

Other annealing schedules have been developed using the Markov chain and the quasi-equilibrium approaches. The four schedules discussed were selected because they either represent a novel approach or perform well experimentally. For an extensive survey of different annealing schedules, the interested reader is referred to [Laarh87, Chapter 5].

We shall begin the derivation of our annealing schedule in the next chapter.

References

[Aarts85] E. Aarts, and P. van Laarhoven, "Statistical Cooling Algorithm: A General Approach to Combinatorial Optimization Problems," *Philips Journal of Research,* Vol. 40, No. 4, 193-226, 1985.

[Aarts86] E. Aarts, F. de Bont, E. Habers, and P. van Laarhoven, "Parallel Implementations of the Statistical Cooling Algorithm," *INTEGRATION, the VLSI Journal,* Vol. 4, 209-238, 1986.

[Baner86] P. Banerjee, and M. Jones, "A Parallel Simulated Annealing Algorithm for Standard Cell Placement on a Hypercube Computer," *Proceedings of the IEEE International Conference on Computer-Aided Design,* 34-37, 1986.

[Beenk85] G. Beenker, T. Claasen, and P. Hermens, "Binary Sequences with Maximally Flat Amplitude Spectrum," *Philips Journal of Research,* Vol. 40, 289-304, 1985.

[Binde86] K. Binder, *Monte Carlo Methods in Statistical Physics,* 2nd Edition, Springer-Verlag, 1986.

[Casot86] A. Casotto, F. Romeo, and A. Sangiovanni-Vincentelli, "A Parallel Simulated Annealing Algorithm for the Placement of Macro-Cells," *Proceedings of the IEEE International Conference on Computer-Aided Design,* 30-33, 1986.

[Cerny85] V. Cerny, "A Thermodynamical Approach to the Traveling Salesman Problem: an Efficient Simulation Algorithm," *Journal of Optimization Theory and Applications,* Vol. 45, 41-51, 1985.

[Felle70] W. Feller, *An Introduction to Probability Theory and Applications,* Wiley, 3rd Edition, 1970.

[Hajek85] B. Hajek, "A Tutorial Survey of Theory and Applications of Simulated Annealing," *Proceedings of the 24th Conference on Decision and Control,* 755-760, 1985.

[Huang86] M. Huang, F. Romeo, and A. Sangiovanni-Vincentelli, "An Efficient General Cooling Schedule for Simulated Annealing," *Proceedings of the IEEE International Conference on Computer-Aided Design,* 381-384, 1986.

[Isaac76] D. Isaacson, and R. Madsen, *Markov Chains: Theory and Applications,* John Wiley, 1976.

[Jakob87] M. Jakobsen, K. Mosegaard, and J. Pedersen, "Global Model Optimization in Reflection Seismology by Simulated Annealing," *Mathematical Geophysics 5th International Seminar on Model Optimization in Exploration Geophysics,* Berlin, 1987.

[Kirkp82] S. Kirkpatrick, C. Gelatt Jr., and M. Vecchi, "Optimization by Simulated Annealing," *IBM Research Report, RC 9355,* 1982.

[Kravi86] S. Kravitz, and R. Rutenbar, "Multiprocessor-Based Placement by Simulated Annealing," *Proceedings of the 23rd IEEE Design Automation Conference,* 567-573, 1986.

[Laarh87] P. van Laarhoven, and E. Aarts, *Simulated Annealing: Theory and Applications,* D. Reidel Publishing Company, 1987.

[Lam86] J. Lam, and J.-M. Delosme, "An Adaptive Annealing Schedule," Report 8608, Department of Electrical Engineering, Yale University, 1986.

[Metro53] M. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *Journal of Chemical Physics,* Vol. 21, 1087-1092, 1953.

[Mitra85] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and Finite-time Behavior of Simulated Annealing," *Proceedings of the 24th Conference on Decision and Control,* 761-767, 1985.

[Romeo85] F. Romeo, and A. Sangiovanni-Vincentelli, "Probabilistic Hill Climbing Algorithms: Properties and Applications," *Proceedings of the 1985 Chapel Hill Conference on VLSI,* 393-417, 1985.

[Salam87] P. Salamon, J. Nulton, J. Robinson, J. Pedersen, G. Ruppeiner, and L. Liao, "Simulated Annealing with Constant Thermodynamic Speed," *unpublished manuscript,* Department of Mathematical Sciences, San Diego State University, 1987.

[Seche85] C. Sechen, and A. Sangiovanni-Vincentelli, "The TimberWolf Placement and Routing Package," *IEEE Journal of Solid-State Circuits,* Vol. 20, No. 2, 510-522 1985.

[Sonta85] E. Sontag, and H. Sussmann, "Image Restoration and Segmentation using the Annealing Algorithm," *Proceedings of the 24th Conference on Decision and Control,* 768-773, 1985.

[White84] S. White, "Concepts of Scale in Simulated Annealing," *Proceedings of the IEEE International Conference on Computer Design,* 646-651, 1984.

CHAPTER 2

Evolution of the Average Energy in Simulated Annealing

The most important quantity in simulated annealing is the energy of the system after the annealing process terminates; this quantity, called the *final energy*, represents the quality of the final solution. Since simulated annealing is a probabilistic algorithm, the final energy varies among different executions on the same problem. Hence, the *final average energy*, representing the average quality of the final solution, is a more appropriate quantity to analyze and optimize.

The final average energy depends on the evolution of the average energy during the execution of simulated annealing; the evolution of the average energy, in turn, depends on the annealing schedule as well as the move generation strategy. Therefore, to minimize the final average energy with respect to *both* the annealing schedule and the move generation strategy, we carry out a two step process in two chapters. In this chapter we minimize the final average energy with respect to the annealing schedule—assuming that the move generation strategy is fixed—to obtain a class of annealing schedules called the efficient λ-schedules. In the next chapter we minimize the final average energy with respect to the move generation strategy for this class of schedules.

We first define a class of annealing schedules called λ-schedules that keep a system in an approximate equilibrium condition called quasi-equilibrium. Then, we develop a formula characterizing the evolution of the average energy for a system in quasi-equilibrium. Using this formula, we prove that there exists, among all λ-schedules, a λ-schedule that leads to a minimum final average energy, and we give a formula for this efficient λ-schedule. We assume, throughout this chapter, that the move generation strategy is fixed. Furthermore, we also assume that the irreducibility condition in (1.6)

and the reversibility condition in (1.7) are satisfied so that the equilibrium average energy of the system is given by (1.9).

2.1. Quasi-stationary process

The simulated annealing method is based on the observation that annealing is successful if the system is kept close to thermal equilibrium as the temperature is lowered. However, to keep the system in equilibrium at all times requires that the temperature decrements be infinitesimal; a *long* time would have passed before the system is frozen and annealing is stopped. From a practical standpoint, a good annealing schedule must, therefore, achieve a compromise between the final average energy and the computation time. To determine when the system is in equilibrium so that the temperature could be lowered, we need an equilibrium criterion.

Before introducing this criterion, we have to understand the relationship between equilibrium and stationary process. In simulated annealing time (or step) is discrete and is incremented whenever a proposed move is either accepted or rejected. The energy, $X(s_n)$, of a system is a stochastic process whose statistics depend on step n as well as the inverse temperature s_n, $s_n \equiv 1/T_n$. Thus, given a system, there exists a stochastic process corresponding to each annealing schedule. Let us consider a special class of annealing schedules where the inverse temperature, s_n, is constant. As $n \rightarrow \infty$, the system approaches equilibrium, and the process $X(s_n)$ becomes stationary. (Keeping the inverse temperature constant results in a time-homogeneous Markov chain discussed in Chapter 1. But since we are dealing with continuous energy values rather than discrete states, we adopt a different notation here.) We denote this kind of stationary process by $\underline{X}(s_n)$ and its mean and variance by $\mu(s_n)$ and $\sigma^2(s_n)$, respectively. The stationary process $\underline{X}(s_n)$ is of interest because comparison with its statistics provides information on how far the system is away from equilibrium at any inverse temperature. Since one cannot afford to wait for the system to reach equilibrium at a given inverse temperature, the need for an approximate equilibrium criterion that controls how far the system is allowed to deviate from equilibrium is evident.

We know that if the inverse temperature, s, is fixed, the average energy, $\overline{X}(s)$, converges monotonically towards $\mu(s)$. Thus, we would like to have an approximate equilibrium (or quasi-equilibrium) criterion of the form

$$\left|\overline{X}(s) - \mu(s)\right| \leq \epsilon.$$

With this criterion once the system reaches quasi-equilibrium, it stays in quasi-equilibrium until the inverse temperature is changed. However, we observe that the criterion is invariant with respect to translation, but not to scaling of the energy. In order to have an approximate equilibrium criterion which is invariant with respect to both transformations, we replace ϵ by $\lambda\sigma(s)$. We say a process, $X(s_{n-1})$, is *quasi-stationary at inverse temperature* s_n if it satisfies the *quasi-stationarity* (or quasi-equilibrium) criterion:

$$\left|\overline{X}(s_{n-1}) - \mu(s_n)\right| \leq \lambda\sigma(s_n), \tag{2.1}$$

where $\overline{X}(s_{n-1})$ is the average energy of the system at step $n-1$ (after a proposed move is either accepted or rejected), and λ is a user specified parameter. (Note that $\overline{X}(s_{n-1})$ depends on s_{n-1}, not s_n.) A *quasi-stationary process* is a process that satisfies the quasi-stationarity criterion at *all* inverse temperatures in an annealing schedule; a schedule that gives rise to such a process is called a *λ-schedule*. The λ-schedules are important, for they result in carefully controlled executions of simulated annealing. Note that a unique quasi-stationary process is associated with each λ-schedule (assuming the move generation strategy is fixed). A λ-schedule is *n-step efficient* if among all λ-schedules, it minimizes the final average energy at step n. If a λ-schedule is n-step efficient for all $n > 0$, we call it the *efficient λ-schedule*. Our goal in this chapter is to show that the efficient λ-schedule exists and to construct it by studying the behavior of the associated quasi-stationary process.

Since it is not clear a priori that a constant λ is the best choice, we tested temperature varying $\lambda(s_n)$ functions for a 100-city traveling salesman problem. These functions come from the following observation. If the energy density function, $P(x)$, can be approximated by a normal density function,

Group	k	λ_0	δ (%)
A		0.4	2.8
B	∞	0.2	2.1
C		0.1	1.5
A		0.4	3.6
B	0.014	0.2	2.0
C		0.1	1.6
A		0.4	3.1
B	0.0014	0.2	2.3
C		0.1	1.2
A		0.4	3.1
B	0.00014	0.2	2.8
C		0.1	1.6

Table 2.1: Test results of $\lambda(s) = \lambda_0(1+ks) \,/\, (\sqrt{2}+ks)$ for different values of k for a 100-city traveling salesman problem. These are average results of *eight* executions, where δ is the percentage above the best solution ever found. The difference in the total number of proposed moves between any two cases in the same group is less than 10%.

$$P(x) = N_x(\mu, \sigma) \equiv \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where $\mu = \mu(0)$ and $\sigma^2 = \sigma^2(0)$ are, respectively, the stationary mean and variance of the energy at an inverse temperature of zero, then it can be shown that the stationary probability density function of the energy at inverse temperature s, $P_s(x)$, is given by the normal density function $N_x(\mu(s), \sigma)$, where $\mu(s) = \mu - s\,\sigma^2$. Thus, $P_s(x)$ can be viewed as $P(x)$ with a translation of $-s\,\sigma^2$. Since the shape of $P_s(x)$ does not change with respect to the inverse temperature, the quasi-stationarity criterion that depends on the *shape* of $P_s(x)$ should not change with respect to the inverse temperature. The choice of a constant λ makes the quasi-stationarity criterion invariant with respect not only to translation and scaling of the energy but also to the inverse temperature.

But instead of approximating $P(x)$ by a normal density function, we find in, Chapter 3, that it is preferable to approximate $P(x)$ by a gamma density function,

$$P(x) = G_x(N, b) \equiv \frac{b^N}{\Gamma(N)} x^{N-1} e^{-bx}, \quad x > 0,$$

where $\Gamma(N) = \int_0^\infty x^{N-1} e^{-x}\, dx$ is the gamma function. With this approximation, $P_s(x)$ is

given by $G_x(N, b+s)$. Unlike the shape of a normal density function which is symmetric about the mean, the shape of a gamma density function is skewed. As the inverse temperature is raised from zero to infinity, the shape of $P_s(x)$ changes from almost symmetric about the mean $\mu(s)$ to almost one sided. Let $\sigma_L^2(s)$ and $\sigma_R^2(s)$ be the contributions to $\sigma^2(s)$ from the left side and the right side of $P_s(x)$, respectively, with $\sigma^2(s) = \sigma_L^2(s) + \sigma_R^2(s)$. As the inverse temperature is raised from zero to infinity, the value of $\sigma^2(s)$ changes from $2\sigma_R^2(s)$ (symmetric with $\sigma_L^2(s) = \sigma_R^2(s)$) to $\sigma_R^2(s)$ (one sided with $\sigma_L^2(s) = 0$). Since the inverse temperature is chosen such that $\overline{X}(s)$ is always greater than $\mu(s)$, we would like to use $\sigma_R(s)$ in place of $\sigma(s)$ in the criterion. Let

$$\left|\overline{X}(s) - \mu(s)\right| \leq \lambda\sigma_R(s).$$

be the skew-compensated quasi-stationarity criterion. Expressed in terms of $\sigma(s)$, the criterion can be written as

$$\left|\overline{X}(s) - \mu(s)\right| \leq \lambda(s)\sigma(s),$$

with $\lambda(s)$ changing from $1 / \sqrt{2}$ at $s = 0$ to 1 at $s = \infty$. Because of the difficulty in finding the exact form of $\lambda(s)$, we use the function

$$\lambda(s) = \lambda_0 \frac{1 + ks}{\sqrt{2} + ks}$$

with different values of k in our test. The case of a constant λ corresponds to $k = \infty$. The average results of eight executions using the efficient λ-schedule from Chapter 3 are tabulated in Table 2.1. (Theoretically, this annealing schedule may not be the efficient λ-schedule for temperature varying $\lambda(s)$, for the derivation of this schedule depends upon that $\lambda(s)$ is constant. However, because $\lambda(s)$ is a slowly varying function of s, $\partial\lambda(s) / \partial s$ is small. Hence, this annealing schedule is a good approximation to the efficient one.) According to this table, a constant λ gives at least as good results as the temperature varying $\lambda(s_n)$ functions that we experimented with. Since we could not find a $\lambda(s_n)$ function that improves on the constant function, we impose that λ be constant to get

$$\left| \overline{X}(s_{n-1}) - \mu(s_n) \right| \leq \lambda\sigma(s_n) \tag{2.2}$$

as the quasi-stationarity criterion at inverse temperature s_n.

The parameter λ, invariant with respect to translation and scaling of the energy, can be viewed as the maximum allowable "distance" between a quasi-stationary process $X(s_{n-1})$ and the stationary process $\underline{X}(s_n)$ at inverse temperature s_n. This parameter, which can be made as small as desired to ensure a good approximation of equilibrium, realizes the compromise between the quality of the final average energy and the computation time: the smaller the λ; the better the final average energy; the longer the computation time.

2.2. Autoregressive process

Before deriving the efficient λ-schedule, we need to know how the average energy of a system evolves. We model the stationary process $\underline{X}(s)$ as a first order autoregressive process [Marpl87, Chapter 6]:

$$\underline{X}_n(s) = r(s)(\underline{X}_{n-1}(s) - \mu(s)) + \mu(s) + \mathcal{N}_n, \tag{2.3}$$

where $r(s)$, which depends on the move generation strategy, is the first autocorrelation coefficient of the stationary process $\underline{X}(s)$, and the random fluctuation $\mathcal{N}_n$ is uncorrelated (white noise). The value of the first autocorrelation coefficient is close to 1 because the energy difference between moves is usually small when compared with the standard deviation, $\sigma(s)$, of the energy, as the proposed solution is obtained by slightly perturbing the current solution.

There are two reasons behind the model in (2.3). First, given the current state (not energy) of a system, the next state of the system does not depend on its previous states; it only depends on the current state, the move generation strategy, and the inverse temperature. This Markov property follows directly from the simulated annealing method. The grouping of states according to the values of their energy, however, may not preserve this property: given the current value of the energy of a system, the next value may depend on the previous values. (As noted in Chapter 1, the

term energy values in this context refers to non-overlapping intervals of the energy.) Yet the dependency must be slight: since the states with a given energy are *numerous*, the probability is small that

1. a sequence of previous energy values favors a particular group of states, and
2. the next energy for this particular group of states is different from the expected next energy of the system given the current energy.

Consequently, the previous values of the energy provide little extra information, and we assume that the next value of the energy does not depend on the previous values given the current value, that is, $\underline{X}_n(s)$ is a function of $\underline{X}_{n-1}(s)$, but not $\underline{X}_{n-2}(s)$, $\underline{X}_{n-3}(s)$,.... This Markov property at two different levels is often used in the physical sciences. For instance, Van Kampen [Kampe81, p. 79] models the dissociation of a gas of binary molecules, $AB \rightarrow A + B$, as follows:

> On a coarse scale one argues that each molecule AB has a certain probability per unit time to be broken up by some collision. Hence the change in the concentration between t and $t + \Delta t$ has a certain probability distribution, which depends on the concentration at t, but not on its previous values. Thus on this level the concentration is a Markov process. In a more detailed description of the mechanism of breaking up, one distinguishes the vibrational states of the molecule, and studies the way by which successive collisions kick it back and forth between these states, until it happens to go over the top of the potential barrier. This random walk over the vibrational levels is another Markov process ...

Second, we assume that the next energy depends linearly on the current energy. This assumption, leading to the model in (2.3), is imposed for simplicity. It would be extremely difficult if not impossible to derive an efficient λ-schedule with a nonlinear dependency. Furthermore, as the following experiments illustrate, a nonlinear dependency is unlikely to improve the accuracy of our model significantly.

To assess the quality of the model, we manipulate (2.3) to get an expression for the random fluctuation:

$$\mathcal{N}_n = (\underline{X}_n(s) - \mu(s)) - r(s)(\underline{X}_{n-1}(s) - \mu(s)). \tag{2.4}$$

The average value of $\mathcal{N}_n$ is zero by construction. We can test the quality of this model

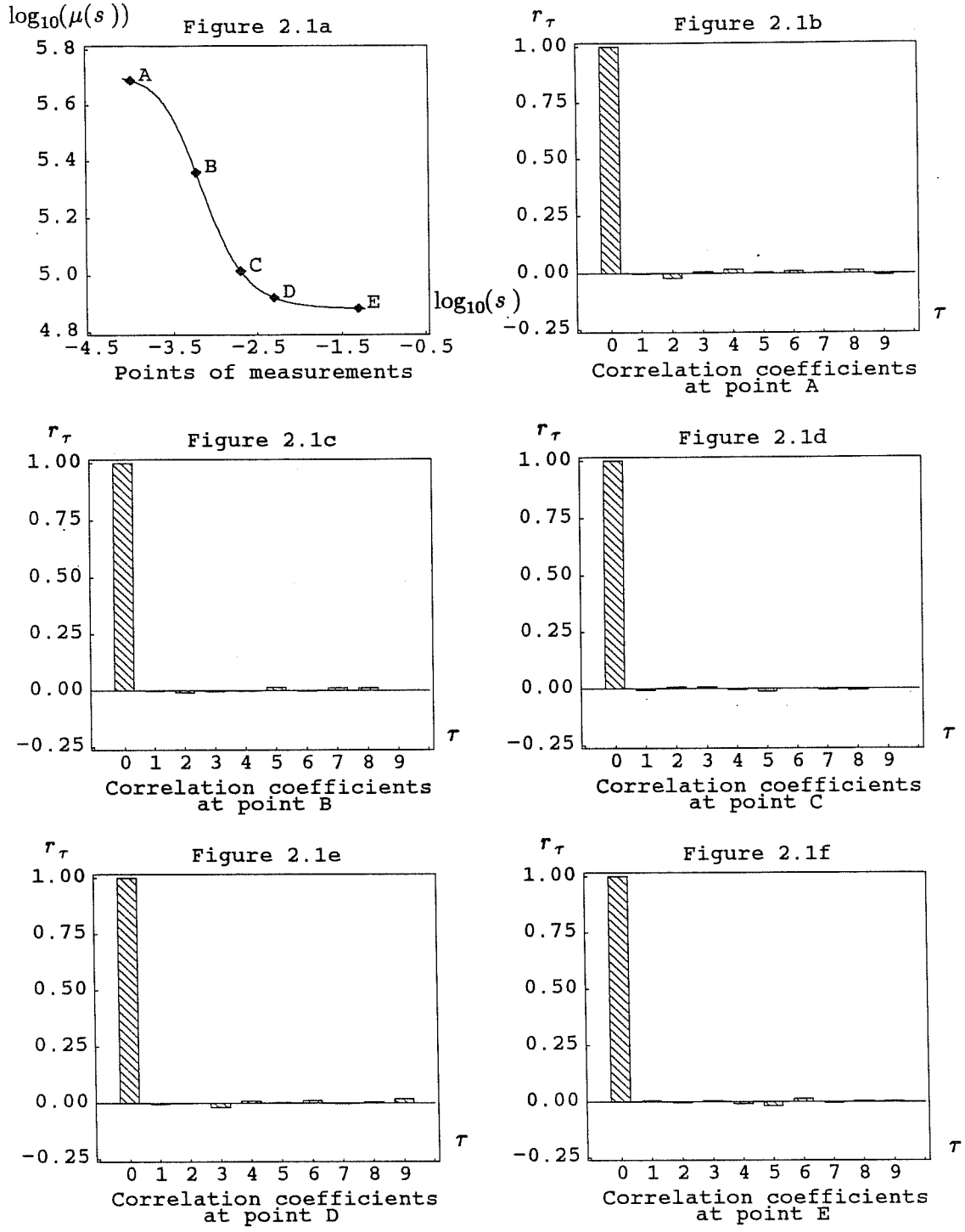

Figure 2.1: Autocorrelation coefficients of the random fluctuations for a 100-city traveling salesman problem. The inverse temperatures at which the tests were performed are indicated on the annealing curve in Figure 2.1a while the autocorrelation coefficients are shown in Figure 2.1b to Figure 2.1f.

by measuring the autocorrelation coefficients of the random fluctuation. If the autocorrelation coefficient, r_τ, between $\mathcal{N}_n$ and $\mathcal{N}_{n+\tau}$ is close to zero for $\tau \neq 0$, the model is good. Otherwise, the random fluctuation is not a white process and the model is poor. We performed the tests on a 100-city traveling salesman problem and a graph partition problem with $N = 500$ (number of vertices) and $d = 5$ (average degree). (See Chapter 5 for details of the test cases.) The tests were conducted by first running simulated annealing for a long time at a fixed inverse temperature to obtain a close to stationary process. Then, the energy of the system, $\underline{X}(s)$, was measured for the next 10,000 steps. The τ^{th} autocovariance, R_τ, between $\underline{X}_n$ and $\underline{X}_{n+\tau}$ was estimated for $\tau = 0$ and 1 according to the formula [Marpl87, p. 147]

$$R_\tau = \frac{1}{M} \sum_{i=0}^{M-\tau-1} (\underline{X}_i - \hat{\mu})(\underline{X}_{i+\tau} - \hat{\mu}), \tag{2.5}$$

where $\underline{X}_i$ is the i^{th} piece of data, M is the number of data points, and $\hat{\mu} = \sum_{j=0}^{M-1} \underline{X}_j \;/\; M$ is the estimated average energy. From the autocovariance of $\underline{X}(s)$ we computed the first autocorrelation coefficient,

$$r(s) = \frac{R_1}{R_0}. \tag{2.6}$$

Then, using (2.4), we obtained a sequence of values, $\mathcal{N}_n$, from the sequence of data, $\underline{X}_n$. Finally, the autocorrelation coefficients, r_τ, were computed as the ratio of the τ^{th} autocovariance to the 0^{th} autocovariance of the random fluctuation, with the autocovariance being estimated using the same formula as for $\underline{X}_n$. The results for the traveling salesman problem are shown in Fig. 2.1 while the results for the graph partition problem are shown in Fig. 2.2. The inverse temperatures at which the tests were performed are indicated on the annealing curves in Fig. 2.1a and Fig. 2.2a while the autocorrelation coefficients are shown in Fig. 2.1b to Fig. 2.1f and Fig. 2.2b to Fig. 2.2f. Since r_τ for $\tau \geq 1$ are close to zero, we conclude that the model is good.

The properties of the quasi-stationary process $X(s_{n-1})$ will be close to that of the stationary process $\underline{X}(s_n)$ if λ is made small enough. Consequently, we can use (2.3),

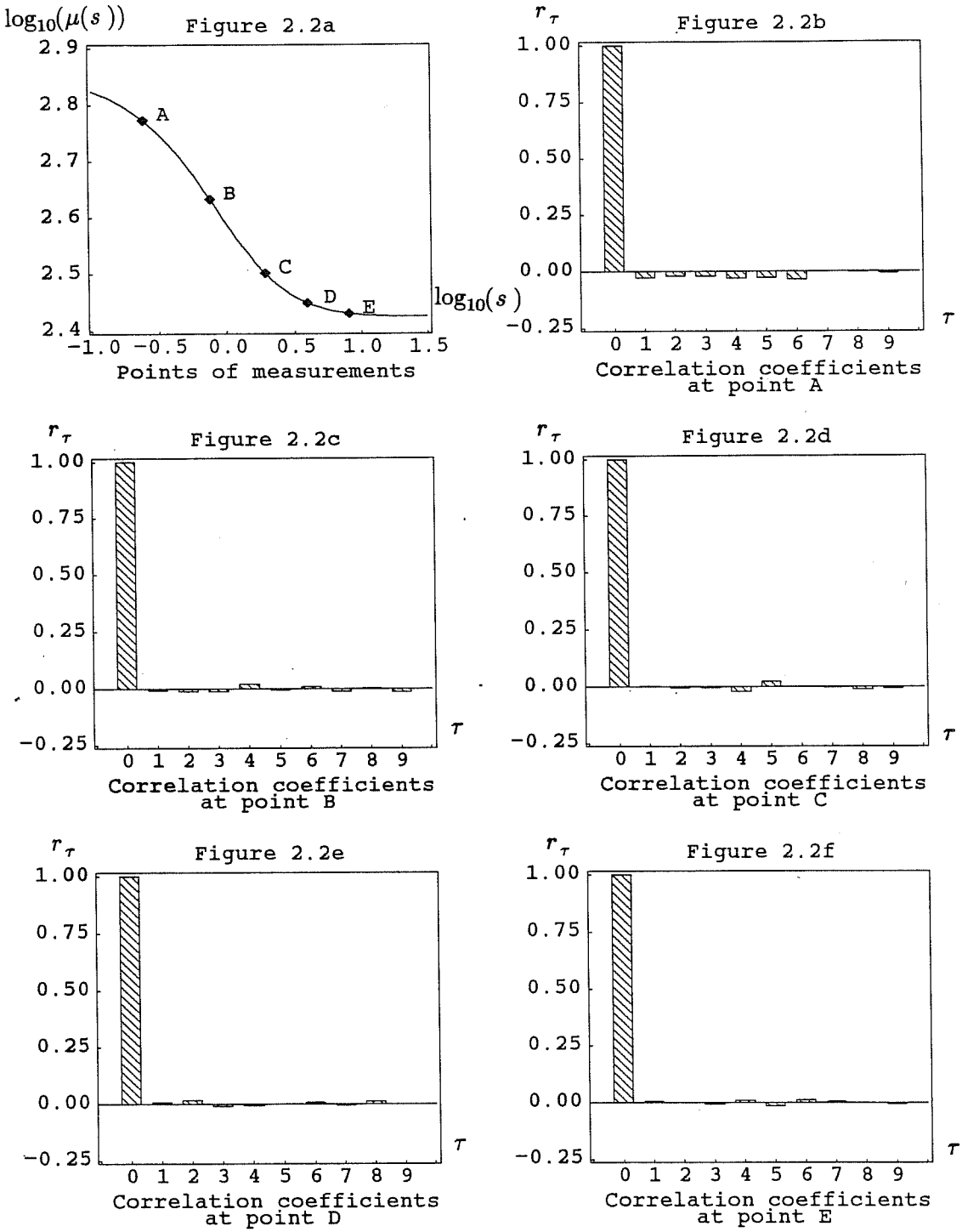

Figure 2.2: Autocorrelation coefficients of the random fluctuations for a graph partition problem with $N = 500$, $d = 5$. The inverse temperatures at which the tests were performed are indicated on the annealing curve in Figure 2.2a while the autocorrelation coefficients are shown in Figure 2.2b to Figure 2.2f.

$$X(s_n) = r(s_n)(X(s_{n-1}) - \mu(s_n)) + \mu(s_n) + \mathcal{N}_n,$$

with $\mathcal{N}_n$ a zero-mean white noise, to model the evolution of the quasi-stationary process $X(s_{n-1})$. From the preceding expression we obtain, after taking expectations, an expression for *the evolution of the average energy*:

$$\overline{X}(s_n) = r(s_n)(\overline{X}(s_{n-1}) - \mu(s_n)) + \mu(s_n), \tag{2.7}$$

where $\overline{X}(s_{n-1})$ is the average energy of the quasi-stationary process $X(s_{n-1})$, which may differ from $\mu(s_n)$, the average energy of the stationary process $\underline{X}(s_n)$. Equation (2.7) specifies the evolution of the average energy not only at a fixed inverse temperature but also at all inverse temperatures in a λ-schedule. This claim is best illustrated by describing the execution of simulated annealing. We start by proposing a move after the inverse temperature is raised from s_{n-1} to s_n. If the process $X(s_{n-1})$ satisfies the quasi-stationarity criterion at inverse temperature s_n,

$$\left|\overline{X}(s_{n-1}) - \mu(s_n)\right| \leq \lambda\sigma(s_n), \tag{2.8}$$

the average energy at step n is given by (2.7):

$$\overline{X}(s_n) = r(s_n)(\overline{X}(s_{n-1}) - \mu(s_n)) + \mu(s_n). \tag{2.9}$$

Since a quasi-stationary process satisfies the quasi-stationarity criterion at all inverse temperatures encountered, the preceding expression, after replacing s_n with s_{n+1} and s_{n-1} with s_n, also applies to the process at inverse temperature s_{n+1}. This procedure can be carried on for inverse temperatures s_{n+2}, s_{n+3},.... Therefore, equation (2.9) completely specifies the evolution of the average energy for any λ-schedule. Furthermore, we can replace the quasi-stationarity criterion in (2.8) with the simpler expression

$$\overline{X}(s_{n-1}) - \mu(s_n) \leq \lambda\sigma(s_n). \tag{2.10}$$

To prove this claim, we observe that the inverse temperature in an annealing schedule does not decrease with respect to time, and that $\partial\mu(s) / \partial s = -\sigma^2(s) \leq 0$ in (1.18). These two conditions imply $\mu(s_n)$ is a non-increasing function of s_n, $\mu(s_n) \leq \mu(s_{n-1})$. If $\overline{X}(s_{n-2}) \geq \mu(s_{n-1})$, we deduce from (2.9) that $\overline{X}(s_{n-1}) \geq \mu(s_{n-1})$ and, hence,

$\overline{X}(s_{n-1}) \geq \mu(s_n)$. If the system is in equilibrium to start with, $\overline{X}(s_o) = \mu(s_o)$, $\overline{X}(s_{n-1})$ will always be greater than or equal to $\mu(s_n)$ for any n. Consequently, we can replace (2.8) with (2.10).

2.3. Efficient annealing schedule

Now exploiting our formula for the evolution of the average energy, we can claim that the n-step efficient λ-schedule consists of the $(n-1)$-step efficient λ-schedule followed by the inverse temperature s_n that maximizes the decrement in average energy at step n subject to the quasi-stationarity constraint,

$$\overline{X}(s_{n-1}) - \mu(s_n) \leq \lambda\sigma(s_n),$$

that is, the n-step efficient λ-schedule is also k-step efficient for $k = 1, 2, \ldots, n-1$. This property, which can be viewed as an instance of Bellman's principle of optimality [Coope81], is essential, for it enables us to compute the efficient λ-schedule iteratively.

To derive this property, we note that the Markov property and the separability of the objective function [Coope81, p. 38] are necessary and sufficient conditions for the application of the principle of optimality. A system possesses the Markov property if after decision δ_k is made at step k, the state π_k that results from that decision depends only on state π_{k-1} and decision δ_k. This condition is satisfied if we identify the average energy $\overline{X}(s_{k-1})$ as the state π_{k-1} and the inverse temperature s_k as the decision δ_k to be made at step k. The objective function is separable if for all k, the effect of the final k steps on the objective function of an n-step process depends only on state π_{n-k} and upon the final k decisions. In our case the objective function is the average energy after n steps, $\overline{X}(s_n)$, and it is indeed separable. Consequently, the principle of optimality applies.

An alternate proof, by induction, is given next. First, for $k = 1$ the 1-step efficient λ-schedule consists of a single inverse temperature s_1 that maximizes the decrement in average energy at step 1 subject to the quasi-stationarity constraint. This follows directly from the definition of the n-step efficient λ-schedule.

Second, we show that the k-step λ-schedule consisting of the $(k-1)$-step efficient λ-schedule followed by the inverse temperature s_k that maximizes the decrement in average energy at step k subject to the quasi-stationarity constraint is k-step efficient. Let the fore-mentioned k-step λ-schedule be schedule A and s_k^A denote the inverse temperature of schedule A at step k. In order for schedule A *not* be the k-step efficient λ-schedule, there must exist a λ-schedule B that leads to a lower average energy at step k:

$$\overline{X}(s_k^A) > \overline{X}(s_k^B). \tag{2.11}$$

Since schedule A consists of the $(k-1)$-step efficient λ-schedule, the average energy using schedule A at step $k-1$ must be less than or equal to the average energy using schedule B at step $k-1$:

$$\overline{X}(s_{k-1}^A) \leq \overline{X}(s_{k-1}^B). \tag{2.12}$$

Yet the two inequalities cannot both be true.

To show the contradiction, we construct a λ-schedule C such that

$$\overline{X}(s_{k-1}^C) = \overline{X}(s_{k-1}^A) \quad \text{and} \quad s_k^C = s_k^B. \tag{2.13}$$

The existence of this λ-schedule is guaranteed by the existence of λ-schedule B since that schedule B satisfies the quasi-stationarity constraint

$$\overline{X}(s_{k-1}^B) - \mu(s_k^B) \leq \lambda\sigma(s_k^B)$$

implies, after substituting s_k^C for s_k^B and noting that $\overline{X}(s_{k-1}^C) = \overline{X}(s_{k-1}^A) \leq \overline{X}(s_{k-1}^B)$ from (2.12) and (2.13), that schedule C satisfies the quasi-stationarity constraint

$$\overline{X}(s_{k-1}^C) - \mu(s_k^C) \leq \lambda\sigma(s_k^C).$$

Since the decrement in average energy for schedule A is maximized, it must be greater than or equal to the decrement in average energy for schedule C:

$$\overline{X}(s_{k-1}^A) - \overline{X}(s_k^A) \geq \overline{X}(s_{k-1}^C) - \overline{X}(s_k^C)$$

and, hence,

$$\overline{X}(s_k^A) \leq \overline{X}(s_k^C). \tag{2.14}$$

The evolution of the average energy is given by

$$\overline{X}(s_k^B) = r(s_k^B)(\overline{X}(s_{k-1}^B) - \mu(s_k^B)) + \mu(s_k^B) \tag{2.15}$$

for schedule B and

$$\overline{X}(s_k^C) = r(s_k^C)(\overline{X}(s_{k-1}^C) - \mu(s_k^C)) + \mu(s_k^C) \tag{2.16}$$

for schedule C. Since $s_k^C = s_k^B$, $r(s_k^B)$ and $\mu(s_k^B)$ must be equal to $r(s_k^C)$ and $\mu(s_k^C)$ because they are statistics of a stationary process. Thus, we obtain, after subtracting (2.15) from (2.16),

$$\overline{X}(s_k^C) - \overline{X}(s_k^B) = r(s_k^B)(\overline{X}(s_{k-1}^C) - \overline{X}(s_{k-1}^B)).$$

Replacing $\overline{X}(s_{k-1}^C)$ with $\overline{X}(s_{k-1}^A)$ in the preceding expression and noting that $\overline{X}(s_{k-1}^A) \leq \overline{X}(s_{k-1}^B)$ in (2.12), we get

$$\overline{X}(s_k^C) - \overline{X}(s_k^B) \leq 0$$

which, combined with (2.14), implies

$$\overline{X}(s_k^A) \leq \overline{X}(s_k^B).$$

This inequality contradicts the inequality in (2.11). Therefore, schedule B cannot exist and schedule A is the k-step efficient λ-schedule.

Based on the principle of induction, we conclude that schedule A is the n-step efficient λ-schedule. Since the n-step efficient λ-schedule is also k-step efficient for $k = 1, 2, \ldots, n-1$, schedule A is the efficient λ-schedule.

2.4. Derivation of the efficient annealing schedule

To compute the n-step efficient λ-schedule, we start with the $(n-1)$-step efficient λ-schedule and find the inverse temperature s_n that maximizes the decrement in average energy at step n subject to the quasi-stationarity constraint

$$\overline{X}(s_{n-1}) - \mu(s_n) \leq \lambda\sigma(s_n).$$

For ease of presentation, we denote s_{n-1} by s, s_n by s_+, $\overline{X}(s_{n-1})$ by $\overline{X}(s)$, and $\overline{X}(s_n)$ by $\overline{X}(s_+)$. The decrement in average energy at step n can be obtained by subtracting $\overline{X}(s_+)$ from $\overline{X}(s)$ using (2.9):

$$\overline{X}(s) - \overline{X}(s_+) = (1 - r(s_+))(\overline{X}(s) - \mu(s_+)). \tag{2.17}$$

If the two constraints (to be checked in Chapter 3)

$$\sigma(s_+) \geq \frac{\lambda}{1 - r(s_+)} \left| \frac{\partial r(s_+)}{\partial s_+} \right| \tag{2.18}$$

and

$$\sigma^2(s_+) \gg \lambda \left| \frac{\partial \sigma(s_+)}{\partial s_+} \right| \tag{2.19}$$

are satisfied, then it can be shown that the decrement in average energy is a non-decreasing function of s_+,

$$\frac{\partial}{\partial s_+}(\overline{X}(s) - \overline{X}(s_+)) \geq 0. \tag{2.20}$$

Consequently, maximizing the decrement in average energy subject to the quasi-stationarity constraint is equivalent to maximizing the inverse temperature, s_+, subject to the same constraint. Furthermore, this maximum occurs when

$$\overline{X}(s) - \mu(s_+) = \lambda\sigma(s_+). \tag{2.21}$$

To show that the condition in (2.20) holds, we carry out the differentiation using (2.17) and substitute $-\sigma^2(s_+)$ for $\partial\mu(s_+) / \partial s_+$ to obtain

$$-\frac{\partial r(s_+)}{\partial s_+}(\overline{X}(s) - \mu(s_+)) + (1 - r(s_+))\sigma^2(s_+) \geq 0$$

and, hence,

$$\sigma^2(s_+) \geq \frac{1}{1 - r(s_+)} \frac{\partial r(s_+)}{\partial s_+}(\overline{X}(s) - \mu(s_+)).$$

If $\partial r(s_+) / \partial s_+$ is negative, the preceding inequality holds; if it is positive, we need to show that

$$\sigma(s_+) \geq \frac{\lambda}{1 - r(s_+)} \frac{\partial r(s_+)}{\partial s_+}, \tag{2.22}$$

which is obtained by noting that $\overline{X}(s) - \mu(s_+) \leq \lambda\sigma(s_+)$ and canceling common terms. This inequality clearly holds if the constraint in (2.18) is satisfied.

To show that the maximum occurs when (2.21) is satisfied, we rewrite the quasi-stationarity constraint as

$$\overline{X}(s) \leq \mu(s_+) + \lambda\sigma(s_+).$$

The right hand side of the preceding expression decreases monotonically with respect to s_+ since its first derivative with respect to s_+ is given by

$$\frac{\partial \mu(s)}{\partial s_+} + \lambda \frac{\partial \sigma(s_+)}{\partial s_+}$$

which, upon substitution of $-\sigma^2(s_+)$ for $\partial\mu(s_+) / \partial s_+$ and noting the inequality in (2.19), is a negative quantity. Therefore, the minimum value of $\mu(s_+) + \lambda\sigma(s_+)$ and, hence, the maximum value of s_+ that satisfies the quasi-stationarity constraint occurs when

$$\overline{X}(s) = \mu(s_+) + \lambda\sigma(s_+).$$

Equation (2.17) can be manipulated using (2.21) to get

$$\overline{X}(s) - \overline{X}(s_+) = \lambda(1 - r(s_+))\sigma(s_+) \tag{2.23}$$

which, upon replacement of $\overline{X}(s)$ by $\mu(s_+) + \lambda\sigma(s_+)$, leads to

$$\overline{X}(s_+) = \mu(s_+) + \lambda r(s_+)\sigma(s_+)$$

and, therefore,

$$\overline{X}(s) = \mu(s) + \lambda r(s)\sigma(s)$$

for any inverse temperature s in the schedule except the initial inverse temperature s_0. Substituting the above expressions of $\overline{X}(s)$ and $\overline{X}(s_+)$ into (2.23) and re-grouping terms, we get

$$\mu(s) - \mu(s_+) = \lambda \left[\frac{\sigma(s_+)}{\sigma(s)} - r(s) \right] \sigma(s). \tag{2.24}$$

(Note the difference between this result and Huang's criterion in (1.20).) The Taylor expansion of $\mu(s_+)$ around s is

$$\mu(s_+) = \mu(s) + (s_+ - s)\frac{\partial \mu(s)}{\partial s} + \frac{(s_+ - s)^2}{2}\frac{\partial^2 \mu(\dot{s})}{\partial \dot{s}^2}, \quad s \leq \dot{s} \leq s_+,$$

which, upon replacement of $\partial \mu(s) / \partial s$ with $-\sigma^2(s)$, results in

$$\mu(s_+) = \mu(s) - (s_+ - s)\sigma^2(s) - (s_+ - s)^2 \sigma(\dot{s}) \frac{\partial \sigma(\dot{s})}{\partial \dot{s}}.$$

Identifying (2.24) with the above expression, we obtain

$$\lambda \left[\frac{\sigma(s_+)}{\sigma(s)} - r(s) \right] \sigma(s) = (s_+ - s)\sigma^2(s) + (s_+ - s)^2 \sigma(\dot{s}) \frac{\partial \sigma(\dot{s})}{\partial \dot{s}}. \tag{2.25}$$

An expression for $\sigma(s_+) / \sigma(s)$ can be computed from the Taylor expansion of $\sigma(s_+)$ around s and is given by

$$\frac{\sigma(s_+)}{\sigma(s)} = 1 + \frac{s_+ - s}{\sigma(s)} \frac{\partial \sigma(\overline{s})}{\partial \overline{s}},$$

where $s \leq \overline{s} \leq s_+$. Substitution of this expression into (2.25) leads to

$$s_+ = s + \lambda \frac{1 - r(s)}{\sigma(s)} + \frac{(s_+ - s)}{\sigma^2(s)} \left[\lambda \frac{\partial \sigma(\overline{s})}{\partial \overline{s}} - (s_+ - s)\sigma(\dot{s}) \frac{\partial \sigma(\dot{s})}{\partial \dot{s}} \right]. \tag{2.26}$$

Although s_+ can theoretically be computed from (2.26)—an implicit expression for s_+—if analytical expressions for $\sigma(\dot{s})$ and $\sigma(\overline{s})$ are known, such a computation is difficult to carry out. Thus, we would like to omit the last term in (2.26) to obtain an explicit expression for s_+:

$$s_+ = s + \lambda \frac{1 - r(s)}{\sigma(s)}. \tag{2.27}$$

This approximation is good if we can show that the resulting error in $s_+ - s$ is small,

that is,

$$\lambda\frac{1-r(s)}{\sigma(s)} \gg \frac{(s_+ - s)}{\sigma^2(s)}\left|\lambda\frac{\partial\sigma(\overline{s})}{\partial\overline{s}} - (s_+ - s)\sigma(\dot{s})\frac{\partial\sigma(\dot{s})}{\partial\dot{s}}\right|$$

or, equivalently, after making use of (2.27),

$$\frac{1}{\sigma^2(s)}\left|\lambda\frac{\partial\sigma(\overline{s})}{\partial\overline{s}} - \lambda(1-r(s))\frac{\sigma(\dot{s})}{\sigma(s)}\frac{\partial\sigma(\dot{s})}{\partial\dot{s}}\right| \ll 1. \tag{2.28}$$

Now if we assume either $\sigma(s) \geq \sigma(\overline{s})$, a property usually satisfied over most of if not all the temperatures, or $\sigma(s) \approx \sigma(\overline{s})$, it follows from (2.19) that

$$\left|\frac{\lambda}{\sigma^2(s)}\frac{\partial\sigma(\overline{s})}{\partial\overline{s}}\right| \ll 1.$$

Similarly, if $\sigma(s) \geq \sigma(\dot{s})$ or $\sigma(s) \approx \sigma(\dot{s})$, inequality (2.19) implies

$$\left|\frac{\lambda}{\sigma^2(s)}\frac{\sigma(\dot{s})}{\sigma(s)}\frac{\partial\sigma(\dot{s})}{\partial\dot{s}}\right| \ll 1$$

and, since $|1-r(s)| < 1$, also

$$\left|\frac{\lambda}{\sigma^2(s)}(1-r(s))\frac{\sigma(\dot{s})}{\sigma(s)}\frac{\partial\sigma(\dot{s})}{\partial\dot{s}}\right| \ll 1.$$

Because both contributions to (2.28) are much smaller than 1, the approximation in (2.27) used for the proof was legitimate and the last term in (2.26) may be neglected. Formula (2.27) can, therefore, be used to approximate formula (2.26).

An alternate form for (2.27), which is used in Chapter 3, is

$$s_+ = s + \lambda\frac{\rho_2(s)}{2\sigma^3(s)}, \tag{2.29}$$

where $\rho_2(s)$ is the variance of the energy increment, $\rho_2(s) \equiv E\left\{(\underline{X}(s_n) - \underline{X}(s_{n-1}))^2\right\}$. ($E\{\cdot\}$ is the expectation operator. Since $\underline{X}(s_n)$ is a stationary process, $s_n = s_{n-1} = s$.) To arrive at this form, we make use of the expression

$$r(s) = 1 - \frac{\rho_2(s)}{2\sigma^2(s)} \tag{2.30}$$

which we obtain by substituting the equality

$$E\left\{\underline{X}(s_{n-1})\underline{X}(s_n)\right\} = E\left\{\underline{X}(s_n)\underline{X}(s_n)\right\} - \frac{1}{2}\rho_2(s)$$

into the definition of the first autocorrelation coefficient:

$$r(s) \equiv \frac{E\left\{\underline{X}(s_{n-1})\underline{X}(s_n)\right\} - \mu^2(s)}{E\left\{\underline{X}(s_n)\underline{X}(s_n)\right\} - \mu^2(s)}.$$

The alternate form in (2.29) follows upon substitution of (2.30) into (2.27).

To summarize, we have an explicit update formula,

$$s_+ = s + \lambda \frac{\rho_2(s)}{2\sigma^3(s)}, \tag{2.31}$$

for the efficient λ-schedule if the two constraints

$$\sigma(s_+) \geq \frac{\lambda}{1 - r(s_+)} \left| \frac{\partial r(s_+)}{\partial s_+} \right| \tag{2.32}$$

and

$$\sigma^2(s_+) \gg \lambda \left| \frac{\partial \sigma(s_+)}{\partial s_+} \right| \tag{2.33}$$

are satisfied. These constraints are satisfied in most cases for the entire temperature range. In the unlikely event that the constraints are not satisfied, formula (2.31) can still be used and may give a good annealing schedule. We shall return to investigate these constraints in Chapter 3.

Thus far, we have derived the efficient λ-schedule that minimizes the final average energy with respect to a sequence of inverse temperatures assuming that the move generation strategy is fixed. Note that associated with each move generation strategy, there is an efficient λ-schedule. In Chapter 3, we shall minimize the final average energy with respect to the move generation strategy for the class of efficient λ-schedules. Before proceeding, we show that minimizing the final average energy is equivalent to

maximizing $\rho_2(s_+)$ for all inverse temperatures s_+, and, hence, we shall maximize $\rho_2(s_+)$ with respect to the move generation strategy in order to minimize the final average energy.

To show the equivalence, we observe from Section 2.3 that minimizing the final average energy for the class of efficient λ-schedules is equivalent to maximizing the decrement in average energy at every step. Furthermore, from the expression for the decrement in average energy

$$\overline{X}(s) - \overline{X}(s_+) = \lambda \frac{\rho_2(s_+)}{2\sigma(s_+)}$$

obtained after substituting (2.30) into (2.23), we observe that maximizing the decrement in average energy is equivalent to maximizing $\rho_2(s_+)$ since $\sigma(s_+)$ does not depend on the move generation strategy (see Chapter 1). Therefore, we conclude that minimizing the final average energy is indeed equivalent to maximizing $\rho_2(s_+)$ for the class of efficient λ-schedules. Consequently, we shall maximize $\rho_2(s_+)$ with respect to the move generation strategy in Chapter 3.

References

[Coope81] L. Cooper and M. Cooper, *Introduction to Dynamic Programming,* Pergamon Press, 1981.

[Kampe81] N. Van Kampen, *Stochastic Processes in Physics and Chemistry,* North-Holland, 1981.

[Marpl87] S. Marple, *Digital Spectral Analysis with Applications,* Prentice-Hall Inc., 1987.

CHAPTER 3

Models for Simulated Annealing

The performance of simulated annealing depends on the annealing schedule as well as the move generation strategy. Although a good annealing schedule gives a good performance, a good annealing schedule with a cooperating move generation strategy gives a better performance. In Chapter 2 we have derived the efficient λ-schedule,

$$s_+ = s + \lambda \frac{\rho_2(s)}{2\sigma^3(s)},$$

that minimizes the final average energy for a given move generation strategy. We have also shown that the final average energy can further be minimized by maximizing the variance of the energy increment, ρ_2, at every step. Since ρ_2 depends on the move generation strategy, the preceding expression represents the class of efficient λ-schedules corresponding to different move generation strategies. In this chapter we derive the conditions that a move generation strategy should satisfy in order to maximize ρ_2 at every step and, hence, to minimize the final average energy. We assume, throughout this chapter, that the annealing schedule in use belongs to the class of efficient λ-schedules shown in the preceding expression.

Since move generation strategies are difficult to analyze as they vary widely among problems, we need to model these strategies in order to carry out the analysis. The models that we propose in this chapter fall into two categories: *move generation models*, which model the move generation strategies, and *energy density models*, which model the energy density functions. The energy density models are needed because the move generation models depend on them.

To obtain the best performance with the efficient λ-schedule, we have to find an expression for ρ_2 and the conditions under which ρ_2 is maximized at every step. In order to accomplish this goal, we first derive an intermediate expression for ρ_2 from the move generation models. Then using the energy density models, we simplify this expression and find the conditions that maximize ρ_2. Lastly, we arrive at the final form of the efficient λ-schedule using the closed form expression for ρ_2.

3.1. Move generation models

The choice of move generation strategies strongly influences the performance of simulated annealing; a move generation strategy that is matched to an annealing schedule improves the performance of simulated annealing. In order to analyze these move generation strategies, we need a model that is simple, yet able to capture their essential properties. Since the goal of simulated annealing is to minimize the energy of the system, only those aspects of move generation strategies that affect the energy need to be modeled. The conditional probability density function of the proposed energy given the current energy, modeled in this section, completely specifies the evolution of the energy and is adequate for that purpose.

For ease of presentation, we let the random variable $\xi(s)$ be the energy increment, and the random variable ξ_p be the proposed energy increment. We also use X, X_p and X_+ to symbolize $\underline{X}(s_{n-1})$, $\underline{X}(s_{n-1}) + \xi_p$ and $\underline{X}(s_n)$, respectively; thus, X_p represents the proposed—but not necessarily accepted—energy which is different from X_+, the energy at the next time step. Moreover, we use lower case for the values of these random variables.

3.1.1. Class of separable models

What does the proposed energy depend on assuming the current energy is known? It is reasonable to assume that the frequency with which an energy is proposed depends on the number of states possessing that energy (specified by the energy density function): the more states with a given energy, the more likely that energy is to be

proposed. Also, it is expected that the distance between the proposed energy and the current energy affects the frequency: energy values at about the same distance from the current energy are proposed with nearly equal frequencies. Formulating both ideas into a separable function leads to the class of *separable models*:

$$f_p(x_p \mid x) \propto G(\mid x_p - x \mid)Q(x_p),$$

where $f_p(x_p \mid x)$ is the conditional probability density function of the proposed energy, x_p, given the current energy, x. The function $Q(x_p)$ models the effect of the energy density function, $P(x_p)$, on $f_p(x_p \mid x)$ while the function $G(\mid x_p - x \mid)$ models the effect of distance between x_p and x on $f_p(x_p \mid x)$. Rewritting this class of models to include the normalization factor, $K(x)$, we have

$$f_p(x_p \mid x) = \frac{1}{K(x)} G(\mid x_p - x \mid)Q(x_p). \tag{3.1}$$

Since $f_p(x_p \mid x)$ is a conditional probability density function, it must satisfy

$$\int_{-\infty}^{\infty} f_p(x_p \mid x)dx_p = 1.$$

Carrying out the integration results in the expression for the normalization factor:

$$K(x) = \int_{-\infty}^{\infty} G(\mid y - x \mid)Q(y)dy = G(\mid x \mid) * Q(x), \tag{3.2}$$

where $*$ denotes convolution.

Based on $f_p(x_p \mid x)$, the conditional probability density function of the proposed energy, we compute $f(x_+ \mid x)$, the conditional probability density function of *the energy at the next time step*, needed for the computation of ρ_2.

The probability that a move with proposed energy x_p is accepted given the current energy x is determined by the Metropolis criterion:

$$\min(1, e^{-s(x_p - x)}).$$

Since the probability that such a move is proposed is $f_p(x_p \mid x)$, the probability that the move is proposed and accepted is

$$f_p(x_p \mid x)\min(1, e^{-s(x_p - x)})$$

while the probability that the move is proposed and rejected is

$$f_p(x_p \mid x)[1 - \min(1, e^{-s(x_p - x)})].$$

Summing up the rejection probability of all possible moves,

$$\int_x^\infty f_p(y \mid x)(1 - e^{-s(y-x)})dy\,,$$

and combining the probabilities of rejection and acceptance, we obtain

$$f(x_p \mid x) = \begin{cases} f_p(x_p \mid x)e^{-s(x_p - x)} & \text{if } x_p > x \\ f_p(x_p \mid x) + \delta(x_p - x)\int_x^\infty f_p(y \mid x)(1 - e^{-s(y-x)})dy & \text{if } x_p \leq x \end{cases}$$

as the conditional probability density function of the energy at the next time step. The delta function, $\delta(x_p - x)$, in the preceding expression ensures that the integral, representing the rejection probability, contributes to $f(x_p \mid x)$ only when the system remains at the same energy. Substituting the expression for $f_p(x_p \mid x)$ into the above equation and replacing x_p with x_+ lead to the final form of the conditional probability density function of the energy at the next time step:

$$f(x_+ \mid x) = \frac{1}{K(x)}\Big[A_s(x_+ - x)Q(x_+) + \delta(x_+ - x)\int_x^\infty [G(\mid y - x \mid) - A_s(y - x)]Q(y)dy\Big] \tag{3.3}$$

with

$$A_s(x_+ - x) = \begin{cases} G(\mid x_+ - x \mid)e^{-s(x_+ - x)} & \text{if } x_+ \geq x \\ G(\mid x_+ - x \mid) & \text{otherwise .} \end{cases} \tag{3.4}$$

The function $Q(x)$ is determined from the properties of the stationary process $\underline{X}(s)$. (Refer to Chapter 2, Section 2.1 for the properties of a stationary process.) Since the process $\underline{X}(s)$ is stationary, for any two energy values x and y with $x \neq y$, the transition probability from energy x to energy y must be equal to the transition probability from energy y to energy x. Thus,

$$f(y \mid x)P_s(x) = f(x \mid y)P_s(y)$$

where $P_s(x)$ (respectively, $P_s(y)$) denotes the probability that the system has energy x (y). Assuming, without loss of generality, that $y > x$ and substituting (3.3) into the above expression, we obtain

$$\frac{1}{K(x)}G(\mid y - x \mid)e^{-s(y-x)}Q(y)P_s(x) = \frac{1}{K(y)}G(\mid x - y \mid)Q(x)P_s(y).$$

Canceling $G(\mid y - x \mid)$ and multiplying both sides by $e^{-sx}K(x)K(y)$, we get

$$K(y)Q(y)e^{-sy}P_s(x) = K(x)Q(x)e^{-sx}P_s(y). \tag{3.5}$$

However, from equation (1.15), the probability density function of $\underline{X}(s)$ is

$$P_s(x) = \frac{P(x)e^{-sx}}{Z(s)} \tag{3.6}$$

which, upon division by $P_s(y) = P(y)e^{-sy}/Z(s)$ and cross-multiplication, yields

$$P(y)e^{-sy}P_s(x) = P(x)e^{-sx}P_s(y).$$

Comparison of this expression with (3.5) shows that $P(x)$ is proportional to $K(x)Q(x)$, or equivalently

$$Q(x) = \kappa\frac{P(x)}{K(x)} \tag{3.7}$$

where κ is a proportionality constant. We call $Q(x)$ the *implicit* energy density function because of its relationship with the energy density function, $P(x)$. Since $Q(x)$ is a probability density function, κ has to be chosen such that $\int_{-\infty}^{\infty} Q(x)dx = 1$.

3.1.2. Moments of the energy increment

We now derive an intermediate expression for ρ_2 using the properties of separable models. From the joint probability density function of the energy y at the next time step and of the current energy x,

$$f(y, x) = f(y \mid x) P_s(x)$$

where $P_s(x)$ denotes the probability that the system has energy x, we compute the n^{th} order moment of the energy increment,

$$E\{\xi^n(s)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(y - x)^n f(y, x)\,dy\,dx = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(y - x)^n f(y \mid x) P_s(x)\,dy\,dx,$$

or equivalently, letting $z = y - x$,

$$E\{\xi^n(s)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} z^n f(y \mid y - z) P_s(y - z)\,dy\,dz.$$

Substituting (3.3) into the above expression and noting that the integral associated with the delta function vanishes for $n > 0$, we obtain

$$E\{\xi^0(s)\} = 1$$

and

$$E\{\xi^n(s)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} z^n \frac{A_s(z) Q(y)}{K(y - z)} P_s(y - z)\,dy\,dz \quad \text{for } n > 0.$$

The *modified* moments of the energy increment, ρ_n, defined as

$$\rho_n(s) \equiv \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} z^n \frac{A_s(z) Q(y)}{K(y - z)} P_s(y - z)\,dy\,dz \quad \text{for } n \geq 0,$$

are identical to $E\{\xi^n(s)\}$ except for $n = 0$; while $E\{\xi^0(s)\} = 1$ is the probability of accepting or rejecting a move, ρ_0, called the *acceptance ratio*, is the probability of accepting a move.

In order to simplify the preceding expression for ρ_n, we replace $P_s(y-z)$ with $P(y-z)e^{-s(y-z)} / Z(s)$ using (3.6) and replace $P(y-z) / K(y-z)$ with $Q(y-z) / \kappa$

using (3.7) to obtain

$$\rho_n(s) = \frac{1}{\kappa Z(s)} \int_{-\infty}^{\infty} z^n A_s(z) \int_{-\infty}^{\infty} Q(y) Q(y - z) e^{-s(y-z)} dy \, dz.$$

Defining $Q_s(x)$ as

$$Q_s(x) \equiv \frac{Q(x) e^{-sx}}{Z_Q(s)} \tag{3.8}$$

with

$$Z_Q(s) = \int_{-\infty}^{\infty} Q(x) e^{-sx} \, dx, \tag{3.9}$$

we rewrite ρ_n as

$$\rho_n(s) = \frac{Z_Q(s)}{\kappa Z(s)} \int_{-\infty}^{\infty} z^n A_s(z) H_s(z) dz \tag{3.10}$$

where

$$H_s(z) = \int_{-\infty}^{\infty} Q(y) Q_s(y-z) dy = Q(z) * Q_s(-z). \tag{3.11}$$

Finally, we eliminate the factor $Z_Q(s) / (\kappa Z(s))$ in (3.10) (see Appendix 3.A) and arrive at

$$\rho_n(s) = \begin{cases} \dfrac{2 \int_0^{\infty} z^n G(|z|) H_s(-z) dz}{\int_{-\infty}^{\infty} G(|z|) H_s(-z) dz} & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd.} \end{cases} \tag{3.12}$$

Note that if the conditional probability density function $f_p(x_p \mid x)$ is a separable function of the variables X, X_p and $|X_p - X|$, the preceding equation is the *only* solution for ρ_n, the modified moments of the energy increment.

3.1.3. Exponential locality model

We have already determined the implicit energy density function, Q, which models the dependency of $f_p(x_p \mid x)$ on the energy density function, but we have not yet determined the function G which models the dependency of $f_p(x_p \mid x)$ on the distance between the proposed energy and the current energy. In order to specify a move generation model completely, we introduce the *exponential locality model*: a separable model in which

$$G(\mid z \mid) \equiv e^{-\beta \mid z \mid}, \quad \beta > 0, \tag{3.13}$$

where $z = \mid x_p - x \mid$ and β depends on the move generation strategy. The parameter β models the effect of locality: the closer the proposed energy is to the current energy, the higher the probability that it is proposed. The exponential locality model is chosen because it is simple to use and easy to analyze.

Using this model, we rewrite the expression for the modified moments of the energy increment, ρ_n, into a form that is used in Section 3.2. We first decompose $G(\mid z \mid) = e^{-\beta \mid z \mid}$ into two parts,

$$g_+(z) = \begin{cases} e^{-\beta z} & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad g_-(z) = \begin{cases} e^{\beta z} & \text{if } z < 0 \\ 0 & \text{otherwise,} \end{cases}$$

and obtain, using the following definition of the Fourier transform pair (See [Butko68, Chapter 7]),

$$F\{g(x)\} = \phi(\omega) = \int_{-\infty}^{\infty} g(x) e^{j\omega x}\, dx$$

$$F^{-1}\{\phi(\omega)\} = g(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \phi(\omega) e^{-j\omega x}\, d\omega,$$

the Fourier transforms of the functions $g_+(z)$ and $g_-(z)$,

$$F\{g_+(z)\} = \frac{1}{\beta - j\omega} \quad \text{and} \quad F\{g_-(z)\} = \frac{1}{\beta + j\omega}.$$

Then, we define $L_+^{(0)}$ and $L_-^{(0)}$ as

$$L_{+}^{(0)}(s) \equiv \int_{0}^{\infty} G(|z|) H_s(-z)\,dz \quad \text{and} \quad L_{-}^{(0)}(s) \equiv \int_{-\infty}^{0} G(|z|) H_s(-z)\,dz$$

and substitute (3.13), part of the exponential locality model, into these definitions to obtain

$$\begin{aligned} L_{+}^{(0)}(\beta,s) &= \int_{0}^{\infty} e^{-\beta z} H_s(-z)\,dz = \int_{-\infty}^{\infty} g_{+}(z) H_s(\beta,-z)\,dz \\ L_{-}^{(0)}(\beta,s) &= \int_{-\infty}^{0} e^{\beta z} H_s(-z)\,dz = \int_{-\infty}^{\infty} g_{-}(z) H_s(\beta,-z)\,dz. \end{aligned} \tag{3.14}$$

Applying Parseval's second theorem [Butko68, p. 267] for real functions,

$$\int_{-\infty}^{\infty} F\{f(z)\} F^{*}\{g(z)\}\,d\omega = \int_{-\infty}^{\infty} f(z) g(z)\,dz$$

where F^{*} denotes the complex conjugate of F, and substituting the complex conjugate of the Fourier transform of equation (3.11) into the preceding expressions for $L_{+}^{(0)}$ and $L_{-}^{(0)}$, we get

$$\begin{aligned} L_{+}^{(0)}(\beta,s) &= \int_{-\infty}^{\infty} \frac{F\{Q(\beta,x)\} F^{*}\{Q_s(\beta,x)\}}{\beta - j\omega}\,d\omega \\ L_{-}^{(0)}(\beta,s) &= \int_{-\infty}^{\infty} \frac{F\{Q(\beta,x)\} F^{*}\{Q_s(\beta,x)\}}{\beta + j\omega}\,d\omega, \end{aligned} \tag{3.15}$$

where the dependency of $Q(\beta,x)$ on β can be observed from (3.2) and (3.7). Finally, substituting these expressions back into (3.12), we arrive at

$$\rho_n(\beta,s) = \begin{cases} \dfrac{2L_{+}^{(n)}(\beta,s)}{L_{+}^{(0)}(\beta,s) + L_{-}^{(0)}(\beta,s)} & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd} \end{cases} \tag{3.16}$$

where $L_{+}^{(n)}$ is defined as

$$L_{+}^{(n)}(\beta,s) \equiv \int_0^\infty z^n e^{-\beta z} H_s(-z)dz = (-1)^n \left[\frac{\partial^n}{\partial \alpha^n} \int_0^\infty e^{-\alpha z} H_s(-z)dz \right]_{\alpha=\beta}$$

and can be rewritten in the form

$$L_{+}^{(n)}(\beta,s) = (-1)^n \left[\frac{\partial^n}{\partial \alpha^n} \int_{-\infty}^{\infty} \frac{F\{Q(\beta,x)\}F^*\{Q_s(\beta,x)\}}{\alpha - j\omega} d\omega \right]_{\alpha=\beta} \tag{3.17}$$

using (3.14) and (3.15).

Thus far, the only constraint for β is $\beta > 0$. If we allow β to change by dynamically controlling move generation, theoretically, we can obtain a β that improves the performance of simulated annealing by maximizing ρ_2. The necessary condition under which ρ_2 is maximized for a given s is

$$\frac{\partial \rho_2(\beta,s)}{\partial \beta} = 0. \tag{3.18}$$

In Section 3.2 we use this formula together with (3.15) to (3.17) and the energy density models to construct the final form of the efficient λ-schedule.

3.1.4. Test of the exponential locality model

To assess the quality of the exponential locality model, we compare the computed and measured conditional probability density function $f_p(x_p \mid x)$ for different values of x. The tests were performed on a 100-city traveling salesman problem and a graph partition problem with $N = 500$ (number of vertices) and $d = 20$ (average degree). The inverse temperatures at which the tests were performed are indicated on the annealing curves shown in Fig. 3.1a and Fig. 3.2a, while the test results for each temperature are shown in Fig. 3.1b to Fig. 3.1f and Fig. 3.2b to Fig. 3.2f. We compute the conditional probability density function $f_p(x_p \mid x)$ for different values of x from the expression

$$f_p(x_p \mid x) \propto e^{-\beta |x_p - x|} [P_s(x_p) e^{s x_p}]^{1/2}. \tag{3.19}$$

This expression is obtained by substituting (3.13) into (3.1),

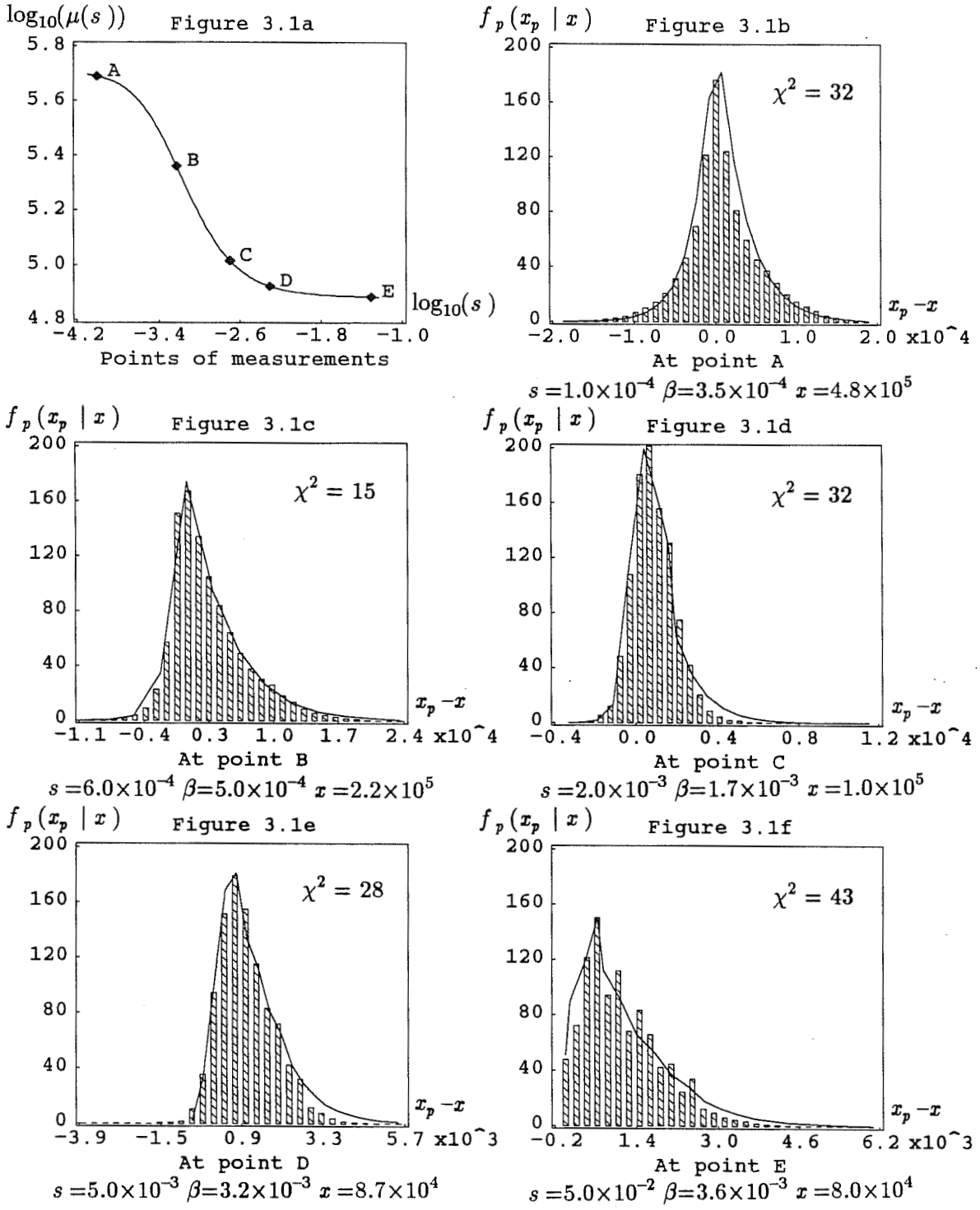

Figure 3.1: Test results of the conditional probability density function $f_p(x_p \mid x)$ for different values of x for a 100-city traveling salesman problem. The inverse temperatures at which the tests were performed are indicated in Figure 3.1a while the measured $f_p(x_p \mid x)$ (histograms) and the computed $f_p(x_p \mid x)$ (curves) are shown in Figure 3.1b to Figure 3.1f.

$$f_p(x_p \mid x) \propto e^{-\beta \mid x_p - x \mid} Q(x_p),$$

replacing $Q(x_p)$ with $P(x_p)^{1/2}$ using (3.26) (derived in Section 3.2),

$$f_p(x_p \mid x) \propto e^{-\beta \mid x_p - x \mid} [P(x_p)]^{1/2},$$

and substituting $P_s(x_p)e^{sx_p}$ for $P(x_p)$ using (3.6). Since direct measurements of β are not possible, we estimate its values at different temperatures from (3.37) (derived in Section 3.2),

$$\rho_2(s) \approx \frac{4(2\beta - s)}{\beta(2\beta + s)^2},$$

using measured values of ρ_2, the variance of the energy increment. The histograms and the curves in Fig. 3.1b to Fig. 3.1f and Fig. 3.2b to Fig. 3.2f represent, respectively, the measured and computed $f_p(x_p \mid x)$ for different values of x. The computed curves agree reasonably well with the measured data throughout the entire temperature range. This agreement suggests that the exponential locality model, although simple, captures the essence of move generation strategies. Note that these experiments also tested the validity of the models developed in Section 3.2.

We also performed goodness of fit tests [Walpo78, p. 266] on the histograms. A goodness of fit test between the observed and computed frequency is based on the quantity

$$\chi^2 = \sum_{i=1}^{k} \frac{(o_i - c_i)^2}{c_i},$$

where o_i and c_i represent the observed and computed frequencies, respectively, for the i^{th} cell of the histogram, k is the number of cells (shaded rectangles) in the histogram and the distribution of χ^2 is chi-square with $k - 2$ degrees of freedom. The hypothesis which was tested is that the model is good. Based on the chi-square values shown in the figures, we fail to reject the hypothesis at the 0.05 level of significance. Therefore, the model is accepted.

A difficulty arose when we attempted to measure $P_s(x_p)$ used in (3.19). Since the first autocorrelation coefficient, $r(s)$, is close to 1 at low temperature, with typical

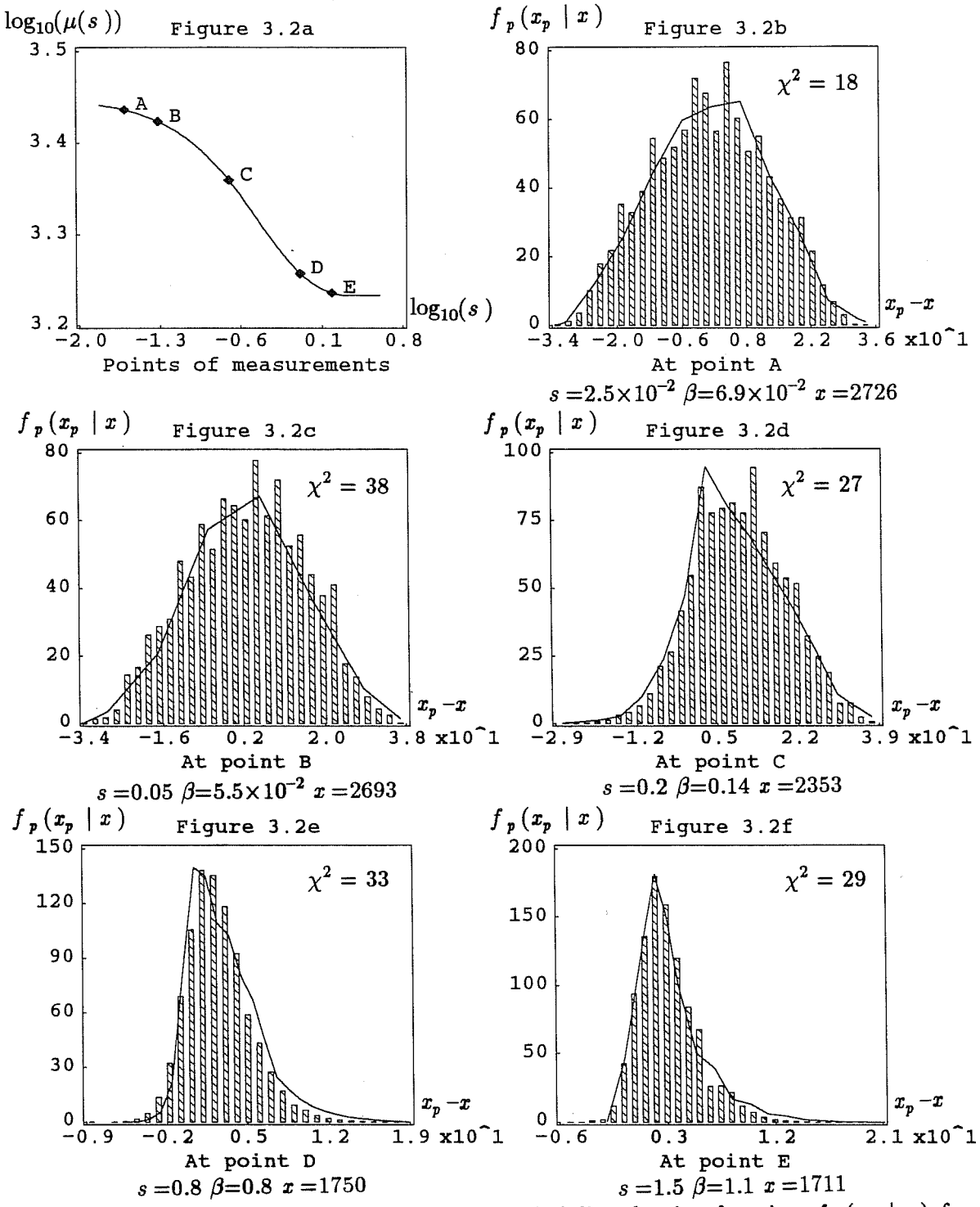

Figure 3.2: Test results of the conditional probability density function $f_p(x_p \mid x)$ for different values of x for a graph partition problem with $N = 500$ and $d = 20$. The inverse temperatures at which the tests were performed are indicated in Figure 3.2a while the measured $f_p(x_p \mid x)$ (histograms) and the computed $f_p(x_p \mid x)$ (curves) are shown in Figure 3.2b to Figure 3.2f.

values near 0.99, sequential measurements of the energy would yield highly correlated data, giving us little information about $P_s(x_p)$. So instead of obtaining the data sequentially in one execution, we collected the data in 60 different executions with 1,000 data points in each execution. This collection procedure reduces the correlation among data and enables us to obtain more accurate descriptions of $P_s(x_p)$ for different values of x.

3.2. Energy density models

The implicit energy density function, $Q(x)$ which is part of the exponential locality model, depends on the energy density function, $P(x)$. In order to find a closed form expression for ρ_2 using the exponential locality model, we need to know $P(x)$. However, owing to the complexity of a typical optimization problem, we rarely know $P(x)$ and must, therefore, resort to modeling.

In this section, we first model the energy density function, $P(x)$, by a gamma density function and develop a closed form expression for ρ_2, the expression that leads to the final form of the efficient λ-schedule. Then, we extend the gamma density model by representing $P(x)$ with a Laguerre series, and we call the representation the close-to-gamma density model because a one-term Laguerre series is a gamma density function. Finally, we show that the gamma density model and the close-to-gamma density model lead to the same expression for ρ_2 and, consequently, the same efficient λ-schedule. This result suggests that the expression for ρ_2 and the annealing schedule are not very sensitive to the detailed features of the energy density function.

3.2.1. Gamma density model

The gamma density function [Fisz67, p. 151] with parameters N and b is

$$\frac{b^N}{\Gamma(N)}(x - x_0)^{N-1}e^{-b(x-x_0)}, \quad x > x_0,$$

where $\Gamma(N) = \int_0^\infty x^{N-1}e^{-x}\,dx$ is the gamma function. The mean and variance of this

probability density function are, respectively,

$$\mu = \frac{N}{b} + x_0 \quad \text{and} \quad \sigma^2 = \frac{N}{b^2}.$$

The main reasons for using the gamma density function to model $P(x)$ are that it is simple, it leads to manageable analysis, and unlike the normal density function it has a bounded left tail which corresponds to a finite global minimum (at $x = x_0$). The last reason is important, for all minimization problems have a finite global minimum that must be modeled in order to give accurate descriptions of the energy density functions, $P_s(x)$, at large values of s. The normal density function is rejected as a model for $P(x)$ because the resulting $\sigma^2(s)$ is invariant with respect to s, whereas $\sigma^2(s)$ should be zero for infinite s.

Without loss of generality, we assume that the global minimum is at $x_0 = 0$. The case $x_0 \neq 0$ can be treated similarly via a simple substitution of variable. From equation (3.7),

$$Q(\beta,x) = \kappa \frac{P(x)}{K(\beta,x)}, \tag{3.20}$$

and equation (3.2) with the exponential locality model,

$$K(\beta,x) = \int_{-\infty}^{\infty} e^{-\beta|y-x|} Q(\beta,y)\,dy, \tag{3.21}$$

we observe that if $Q(\beta,x)$ is specified, $P(x)$ can be obtained from $Q(\beta,x)$ by computing an integral; however, if $P(x)$ is specified, $Q(\beta,x)$ must be obtained from $P(x)$ by solving an integral equation. Therefore, we start with the assumption that $Q(\beta,x)$ is a gamma density function and show that $P(x)$ is also a gamma density function.

Let $Q(\beta,x)$ be a gamma density function with parameters $N(\beta)$ and $b(\beta)$,

$$Q(\beta,x) = \frac{b^N}{\Gamma(N)} x^{N-1} e^{-bx}, \quad x > 0, \tag{3.22}$$

where $N = N(\beta)$ and $b = b(\beta)$. The mean and variance of $Q(\beta,x)$ are

$$\mu_Q = \frac{N}{b} \quad \text{and} \quad \sigma^2{}_Q = \frac{N}{b^2}, \tag{3.23}$$

respectively. Multiplying (3.20) by $K(\beta,x)$ and substituting (3.21) into the expression, we obtain

$$P(x) = \frac{1}{\kappa} Q(\beta,x) \int_{-\infty}^{\infty} e^{-\beta|y-x|} Q(\beta,y)\,dy. \tag{3.24}$$

If the variance of $\beta e^{-\beta|y-x|}/2$ (the normalized form of $e^{-\beta|y-x|}$ in the above expression) is much smaller than the variance of $Q(\beta,y)$, specifically

$$\frac{2}{\beta^2} \ll \sigma^2{}_Q, \tag{3.25}$$

we can approximate $\beta e^{-\beta|y-x|}/2$ by the delta function $\delta(y-x)$ to get

$$P(x) \approx \frac{1}{\kappa} Q(\beta,x) \int_{-\infty}^{\infty} \delta(y-x) Q(\beta,y)\,dy = \frac{1}{\kappa} Q(\beta,x)^2 = \frac{1}{\kappa} Q(x)^2. \tag{3.26}$$

This approximation suggests that the dependency of $Q(\beta,x)$ on β is insignificant since $P(x)$ is independent of β. Because the condition in (3.25) is easily satisfied, we use $Q(x)$ in place of $Q(\beta,x)$ hereafter! Substituting into the preceding relation the expression of $Q(x)$ in (3.22) yields

$$P(x) \propto \frac{b^{2N}}{\Gamma(N)^2} x^{2N-2} e^{-2bx}, \quad x > 0.$$

Including the normalization constant, which can be found easily, into the expression, we arrive at

$$P(x) = \frac{\overline{b}^{\overline{N}}}{\Gamma(\overline{N})} x^{\overline{N}-1} e^{-\overline{b}x}, \quad x > 0,$$

a gamma density function with parameters $\overline{N} = 2N-1$ and $\overline{b} = 2b$, whose mean and variance are

$$\mu = \frac{\overline{N}}{\overline{b}} \quad \text{and} \quad \sigma^2 = \frac{\overline{N}}{\overline{b}^2}, \tag{3.27}$$

respectively. From the variance of $Q(x)$ in (3.23) and the variance of $P(x)$, we obtain

the relation $\sigma^2{}_Q = 2\sigma^2$. Hence, we rewrite the condition in (3.25) as

$$\frac{1}{\beta^2} \ll \sigma^2. \tag{3.28}$$

3.2.1.1. Moments of the energy increment

Now that we have modeled $Q(x)$ and $P(x)$, we would like to find closed form expressions for ρ_0, the acceptance ratio, and ρ_2, the variance of the energy increment. We start with equation (3.16):

$$\rho_n(\beta,s) = \frac{2L_+^{(n)}(\beta,s)}{L_+^{(0)}(\beta,s) + L_-^{(0)}(\beta,s)} \quad \text{if } n \text{ is even} \tag{3.29}$$

in which

$$L_+^{(n)}(\beta,s) = (-1)^n \frac{\partial^n}{\partial\beta^n} \int_{-\infty}^{\infty} \frac{\phi(\omega)\phi_s^*(\omega)}{\beta - j\omega} d\omega \tag{3.30}$$

and

$$L_-^{(0)}(\beta,s) = \int_{-\infty}^{\infty} \frac{\phi(\omega)\phi_s^*(\omega)}{\beta + j\omega} d\omega \tag{3.31}$$

where $\phi(\omega)$ and $\phi_s^*(\omega)$ are the Fourier transforms of $Q(x)$ and $Q_s(-x)$, respectively. The expression for $Q_s(x)$,

$$Q_s(x) = \frac{1}{Z_Q(s)} \left[\frac{b^N}{\Gamma(N)} x^{N-1} e^{-bx} \right] e^{-sx}, \quad x > 0,$$

where $c = b + s$, is obtained by substituting (3.22) into (3.8),

$$Q_s(x) = \frac{Q(x)e^{-sx}}{Z_Q(s)}. \tag{3.32}$$

Since $Q_s(x)$ is a probability density function, the normalization constant $Z_Q(s)$ must be chosen so that the sum of probabilities for all events equals one. Thus, $Q_s(x)$, like $P(x)$ and $Q(x)$, is a gamma density function,

$$Q_s(x) = \frac{c^N}{\Gamma(N)} x^{N-1} e^{-cx}, \quad x > 0, \tag{3.33}$$

with parameters N and c.

To find expressions for ρ_0 and ρ_2, we must first find expressions for $L_+^{(0)}$, $L_-^{(0)}$ and $L_+^{(2)}$. The expressions for $L_+^{(0)}$ and $L_-^{(0)}$ (see Appendix 3.B) are

$$L_+^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{b+c}{2\beta+s}(1+\Re_+^{(0)}), \quad \beta < 3b+s, \tag{3.34}$$

and

$$L_-^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{b+c}{2\beta-s}(1+\Re_-^{(0)}), \quad \frac{s}{2} < \beta < 3b+2s, \tag{3.35}$$

with

$$|\Re_+^{(0)}| \le \left|\frac{2z_+(1-z_+)}{(N+1)(4z_+(1-z_+)-1)}\right|,$$

$$|\Re_-^{(0)}| \le \left|\frac{2z_-(1-z_-)}{(N+1)(4z_-(1-z_-)-1)}\right|,$$

where $z_+ = (b-\beta)/(b+c)$ and $z_- = (c-\beta)/(b+c)$. The expression for $L_+^{(2)}$ (see Appendix 3.D) is

$$L_+^{(2)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{8(b+c)}{(2\beta+s)^3}(1+\Re_+^{(2)}), \quad \beta < 3b+s,$$

with

$$\Re_+^{(2)} = R_1 - \frac{5}{2(1-2z)^2(N+1)}(1+R_2) + \frac{3}{(1-2z)^4(N+1)(N+2)}(1+R_3),$$

where

$$|R_1| \leq \left| \frac{2z(1-z)}{(N+1)(4z(1-z)-1)} \right|,$$

$$|R_2| \leq \left| \frac{12z(1-z)}{(N+2)(4z(1-z)-1)} \right|,$$

$$|R_3| \leq \left| \frac{30z(1-z)}{(N+3)(4z(1-z)-1)} \right|,$$

and $z = (b-\beta) / (b+c)$. Substituting the expressions for $L_+^{(0)}$, $L_-^{(0)}$ and $L_+^{(2)}$ into (3.29), we get an expression for the acceptance ratio,

$$\rho_0(\beta,s) = \frac{2L_+^{(0)}(\beta,s)}{L_+^{(0)}(\beta,s) + L_-^{(0)}(\beta,s)} = \frac{2\beta - s}{2\beta}(1 + \Re_0), \tag{3.36}$$

and an expression for the variance of the energy increment,

$$\rho_2(\beta,s) = \frac{2L_+^{(2)}(\beta,s)}{L_+^{(0)}(\beta,s) + L_-^{(0)}(\beta,s)} = \frac{4(2\beta - s)}{\beta(2\beta + s)^2}(1 + \Re_2), \tag{3.37}$$

subject to the condition $s/2 < \beta < 3b+s$, where the bounds for $\Re_0$ and $\Re_2$ can be found from the bounds for $L_+^{(0)}$, $L_-^{(0)}$ and $L_+^{(2)}$. Since β is not a directly measurable parameter, it is advantageous to express ρ_2 in terms of ρ_0 which may be measured directly. Eliminating β from (3.37) using (3.36), we finally arrive at the closed form expression for ρ_2:

$$\rho_2(s) \approx \frac{8\rho_0(1-\rho_0)^2}{s^2(2-\rho_0)^2}, \quad \frac{s}{2} < \beta < 3b+s. \tag{3.38}$$

3.2.1.2. Efficient annealing schedule

To obtain the condition under which ρ_2 is maximized, we use the necessary condition in (3.18),

$$\frac{\partial \rho_2(\beta,s)}{\partial \beta} = 0.$$

Applying this formula to (3.37) and noting that the second derivative of ρ_2 with respect

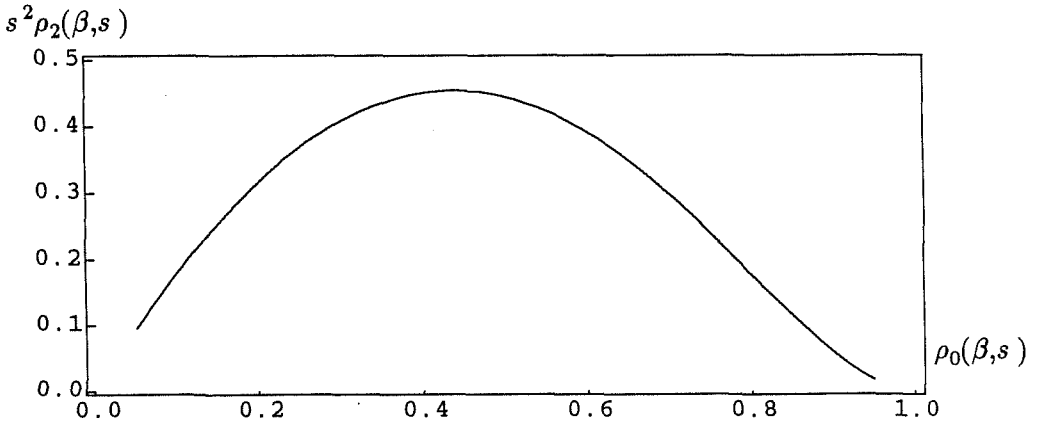

Figure 3.3: A plot of $s^2\rho_2(\beta,s)$ versus $\rho_0(\beta,s)$. Note that the maximum occurs at $\rho_0(\beta,s) = 0.44$.

to β is always negative, we obtain an expression for the *desired* β that maximizes ρ_2:

$$\beta_{des} \approx 0.89s. \tag{3.39}$$

A plot of $s^2\rho_2(\beta,s)$ versus $\rho_0(\beta,s)$ is shown in Fig. 3.3. Since the annealing schedule should operate with β close to β_{des}, the condition, $s/2 < \beta < 3b + s$, for the validity of (3.38) is clearly satisfied. Substituting β_{des} into (3.36) and (3.37), we obtain the *desired* acceptance ratio,

$$\rho_{0,des}(s) \approx 0.44, \tag{3.40}$$

and the *desired* variance of the energy increment,

$$\rho_{2,des}(s) \approx \frac{0.45}{s^2}. \tag{3.41}$$

Moreover, upon substitution of (3.41) into the efficient λ-schedule,

$$s_+ = s + \lambda\frac{\rho_2(s)}{2\sigma^3(s)}, \tag{3.42}$$

we arrive at

$$s_+ = s + \lambda\frac{0.45}{s^2\sigma^3(s)}$$

as the optimized (with respect to move generation) efficient λ-schedule.

In general, we cannot use the optimized version for the entire temperature range because our means of controlling β so that β is close to β_{des} are limited and imprecise. Therefore, instead of using the optimized version, we try to keep β close to β_{des} and employ the annealing schedule

$$s_+ = s + \lambda \frac{4\rho_0(1-\rho_0)^2}{s^2(2-\rho_0)^2\sigma^3(s)}, \tag{3.43}$$

obtained by substituting (3.38) into (3.42), so that deviations from the optimized version are compensated for automatically. We shall discuss practical aspects of applying this annealing schedule in Chapter 4.

To check experimentally that an acceptance ratio of 0.44 is indeed desirable, we performed tests on a 100-city traveling salesman problem in which different values of the desired acceptance ratio, $\rho_{0,des}$, were used. Since the plot of $s^2\rho_2(\beta,s)$ versus $\rho_0(\beta,s)$ in Fig. 3.3 is relatively flat around the maximum, we need to deviate from $\rho_{0,des} = 0.44$ sufficiently in order to observe any difference in performance. Therefore, we chose the values of $\rho_{0,des}$ from 0.04 to 0.84 in increments of 0.20 in our tests. From Table 3.1 we observe that in agreement with our derivation $\rho_{0,des} = 0.44$ gives the best overall result.

λ	Tour Length (10^4)				
	$\rho_{0,des}$=0.04	$\rho_{0,des}$=0.24	$\rho_{0,des}$=0.44	$\rho_{0,des}$=0.64	$\rho_{0,des}$=0.84
0.400	8.38	8.15	8.11	8.11	8.15
0.200	8.21	8.07	8.05	8.04	8.07
0.100	8.13	8.05	8.00	8.02	8.01
0.050	8.08	7.99	7.97	7.97	7.98
0.025	7.95	7.97	7.91	7.93	7.97

Table 3.1: Test results of using different values of desired acceptance ratio, $\rho_{0,des}$, on a 100-city traveling salesman problem. The efficient λ-schedule is used in these tests. Note that between any two cases in a row, the total number of proposed moves differs by less than 10%. Each entry is an average of at least eight executions.

We have delayed the investigation of the constraints for the efficient λ-schedule. Now is the time to study them. Since the efficient λ-schedule should operate close to its optimized version, investigations of the constraints for which β assumes its desired value β_{des} are appropriate.

We start with the three constraints required in this chapter, needed for the derivation and simplification of ρ_2. First, we investigate the condition on the approximation required for the derivation of $L_{+}^{(0)}$ in (3.34). This approximation (see Appendix 3.C) requires the following restriction on the inverse temperature:

$$s \geq \frac{7}{\sigma(0)}. \tag{3.44}$$

Second, we show that the condition in (3.28),

$$\frac{1}{\beta^2} \ll \sigma^2(0),$$

needed for the validity of (3.26), is satisfied if (3.44) is satisfied. Substituting β_{des} into (3.28), we get

$$\frac{1.26}{s^2} \ll \sigma^2(0)$$

which, upon substitution of the restriction in (3.44), yields

$$\frac{1.26\sigma^2(0)}{7^2} \ll \sigma^2(0).$$

This inequality is obviously satisfied.

Third, we replace the condition that $N = (\overline{N} + 1) / 2$ is large, required for the approximation of the hypergeometric function (see Appendix 3.B), with

$$\overline{N} \gg 1.$$

Since $P(x)$ is a gamma density function with parameters $\overline{N}$ and $\overline{b}$, we can easily show that $P_s(x) = P(x)e^{-sx} / Z(s)$ is also a gamma density function with parameters $\overline{N}$ and $\overline{b} + s$. Thus, the mean and variance of $P_s(x)$ are

$$\mu(s) = \frac{\overline{N}}{\overline{b} + s} \quad \text{and} \quad \sigma^2(s) = \frac{\overline{N}}{(\overline{b} + s)^2}. \tag{3.45}$$

Using these expressions, we rewrite the condition as

$$\frac{\mu^2(0)}{\sigma^2(0)} \gg 1. \tag{3.46}$$

Now, we investigate the two constraints derived in Section 2.4 of Chapter 2. We first show that the constraint in (2.33),

$$\sigma^2(s) \gg \lambda \left| \frac{\partial \sigma(s)}{\partial s} \right|, \tag{3.47}$$

is satisfied. To prove this inequality, we use the expression for the variance of $P_s(x)$. Differentiation of $\sigma^2(s)$ with respect to s results in

$$\frac{\partial \sigma^2(s)}{\partial s} = -\frac{2\overline{N}}{(\overline{b} + s)^3}$$

which, upon substitution of the equality

$$\frac{\partial \sigma^2(s)}{\partial s} = 2\sigma(s)\frac{\partial \sigma(s)}{\partial s},$$

yields

$$\left| \frac{\partial \sigma(s)}{\partial s} \right| = \frac{\sqrt{\overline{N}}}{(\overline{b} + s)^2}.$$

Substituting the preceding expression and the expression for $\sigma^2(s)$ into (3.47), we obtain

$$\frac{\overline{N}}{(\overline{b} + s)^2} \gg \lambda \frac{\sqrt{\overline{N}}}{(\overline{b} + s)^2}$$

and, hence, the inequality

$$\sqrt{\overline{N}} \gg \lambda.$$

Since λ is usually much less than 1, we assume the inequality in (3.46) supersedes this inequality.

Next, we show that the constraint in (2.32),

$$\sigma(s) \geq \frac{\lambda}{1 - r(s)} \left| \frac{\partial r(s)}{\partial s} \right|, \tag{3.48}$$

can be replaced with

$$s \geq \frac{2\lambda}{\sigma(0)}.$$

We obtain this alternate constraint by first substituting $\rho_{2,des}$ into (2.30),

$$1 - r(s) = \frac{\rho_2(s)}{2\sigma^2(s)},$$

to get

$$1 - r(s) = \frac{0.23(\overline{b} + s)^2}{\overline{N}s^2}. \tag{3.49}$$

Then, substituting the preceding formula and the expression for $\sigma^2(s)$ into the constraint in (3.48), we obtain

$$\frac{\sqrt{\overline{N}}}{\overline{b} + s} \geq \lambda \frac{s^2\overline{N}}{0.23(\overline{b} + s)^2} \left| \frac{\partial r(s)}{\partial s} \right|. \tag{3.50}$$

Differentiating (3.49) and taking absolute values, we get

$$\left| \frac{\partial r(s)}{\partial s} \right| = \frac{0.45\overline{b}(\overline{b} + s)}{\overline{N}s^3}$$

which is substituted into (3.50) to obtain

$$\frac{\sqrt{\overline{N}}}{\overline{b} + s} \geq \lambda \frac{s^2\overline{N}}{0.23(\overline{b} + s)^2} \frac{0.45\overline{b}(\overline{b} + s)}{\overline{N}s^3}.$$

Canceling common terms and multiplying both sides by $s(\overline{b}+s) / \sqrt{\overline{N}}$, we arrive at

$$s \geq \frac{2\lambda\overline{b}}{\sqrt{\overline{N}}},$$

which is rewritten in the form of

$$s \geq \frac{2\lambda}{\sigma(0)} \tag{3.51}$$

by making use of the expression for $\sigma^2(s)$ in (3.45).

Finally, combining the constraints from both chapters and taking into account the non-zero global minimum, i.e., $x_0 = \mu(\infty)$, we arrive at

$$s_+ = s + \lambda \frac{4\rho_0(1-\rho_0)^2}{s^2(2-\rho_0)^2\sigma^3(s)}, \quad s \geq \frac{7}{\sigma(0)} \text{ and } \frac{(\mu(0)-\mu(\infty))^2}{\sigma^2(0)} \gg 1. \tag{3.52}$$

Note that the conditions were derived assuming $\beta = \beta_{des}$; in practice, β may differ from β_{des} leading to slightly different conditions.

3.2.2. Close-to-gamma density model

The gamma density model, although reasonably easy to analyze, does not usually represent the *left tail* of an energy density function well; an accurate representation of the left tail is important at low temperatures. To remedy this problem, we introduce the close-to-gamma density model in which an energy density function defined in the interval $I = [0,\infty)$ is approximated by a series of orthogonal functions associated with the Laguerre polynomials. This series, which we call the Laguerre series, is chosen because of the following reason. We want a series that can approximate a gamma density function with only a few terms because a gamma density function typically models an energy density function well at a few standard deviations around the mean. The Laguerre series enjoys this property since a one-term Laguerre series is a gamma density function. Unlike the gamma density model for which the parameters N and b are fixed, we are free to choose the parameters of the Laguerre series at different inverse temperatures. The goal is to get a good approximation of an energy density function with as small a number of terms as possible. Since the left tail of an energy density function becomes increasingly important as the temperature is lowered, we want the accuracy of the approximation for the left tail to increase as well.

We first define the Laguerre series. Then, we show that if the energy density function, $P(x)$, and the implicit energy density function, $Q(x)$, are approximated by

Laguerre series, the accuracy of the approximation for $P(x)$ improves linearly with the accuracy of the approximation for $Q(x)$. Finally, we show that if $Q(x)$ can be approximated by a Laguerre series with a small number of terms, the expressions for ρ_0 and ρ_2 in (3.36) and (3.37), and for the efficient λ-schedule in (3.52) remain valid.

As in the case of the gamma density model, a shift of origin in $P(x)$ and, hence, $Q(x)$ does not affect our results. Hence, for derivation purposes, we can assume without loss of generality that $P(x)$ is defined in the interval $[0,\infty)$. (See [Sanso59] for a thorough treatment of orthogonal functions and Laguerre polynomials.)

3.2.2.1. Laguerre series

Let the distance between two functions $f(x)$ and $g(x)$ in the interval $I = [0,\infty)$ be defined as

$$D(f, g) \equiv \int_I [f(x) - g(x)]^2 dx.$$

A system of functions, $\psi_n(x)$, $n = 0, 1,...,$ forms an orthogonal Cartesian system in this interval if it satisfies the following orthogonality relations:

$$\int_I \psi_i(x)\psi_j(x)dx = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise .} \end{cases} \tag{3.53}$$

Furthermore, it can be shown that for any energy density function $P(x)$ in the interval I,

$$D(P, p_{n+1}) \leq D(P, p_n)$$

and

$$\lim_{n \to \infty} D(P, p_n) = 0$$

where $p_n(x) = \sum_{i=0}^{n} a_i \psi_i(x)$ is the $(n+1)$-term expansion of $P(x)$ in the orthogonal Cartesian system and the a_i's, defined by $a_i = \int_I P(x)\psi_i(x)dx$, are the *Fourier coefficients* with respect to that system. By increasing the number of terms in the expansion, we improve the accuracy of the approximation. In the limit as $n \to \infty$,

$p_n(x) \to P(x)$ except possibly at a finite number of points.

Because $P(x)$ is expected to be close to a gamma density function, we shall use the orthogonal Cartesian system formed by the system of functions

$$\psi_n(N,b;x) = \left[\frac{\Gamma(n+1)}{\Gamma(2N+n-1)}\right]^{1/2} (2b)^{N-1/2} x^{N-1} e^{-bx} \mathcal{L}_n^{(2N-2)}(2bx)$$

$$n = 0, 1,..., \quad b > 0, \quad N > 1/2, \tag{3.54}$$

where the Laguerre polynomials, $\mathcal{L}_n^{(\alpha)}(x)$, are defined recursively as

$$\begin{aligned} &\mathcal{L}_{-1}^{(\alpha)}(x) = 0, \quad \mathcal{L}_o^{(\alpha)}(x) = 1, \\ &(n+1)\,\mathcal{L}_{n+1}^{(\alpha)}(x) - [2n+\alpha+1-x]\,\mathcal{L}_n^{(\alpha)}(x) + (n+\alpha)\,\mathcal{L}_{n-1}^{(\alpha)}(x) = 0, \quad n = 0, 1,..., \end{aligned} \tag{3.55}$$

and are given by

$$\mathcal{L}_n^{(\alpha)}(x) = \sum_{i=0}^{n} \gamma_i^{(\alpha,n)} x^i. \tag{3.56}$$

(To simplify the notation, we drop the index α associated with $\gamma_i^{(\alpha,n)}$ in the sequel.) We call the series

$$\Theta_n(N,b;x) = \sum_{i=0}^{n} c_i\,\psi_i(N,b;x)$$

an $(n+1)$-term *Laguerre series*, named after the Laguerre polynomials. (See Appendix 3.E for the proof that the system of functions $\psi_n(N,b;x)$, $n = 0, 1,...$, forms an orthogonal Cartesian system.)

From this orthogonal Cartesian system, we show, in Appendix 3.F, that when $Q(x)$ is approximated by an $(M+1)$-term Laguerre series $q_M(x) = \Theta_M(N,b;x)$ with a distance of $D(Q, q_M)$, $P(x)$ can be approximated by an $(\overline{M}+1)$-term Laguerre series $p_{\overline{M}}(x) = \Theta_{\overline{M}}(\overline{N},\overline{b};x)$ with a distance bounded from above by $4D(Q, q_M) / \kappa$, where $\overline{M} = 2M$, $\overline{N} = 2N-1$, $\overline{b} = 2b$ and according to (3.26) $\kappa \approx \int_0^\infty Q(x)^2 dx$. Note that the coefficients for $q_M(x)$ are the Fourier coefficients, while the coefficients for $p_{\overline{M}}(x)$ are not.

3.2.2.2. Moments of the energy increment

The expressions for ρ_0 and ρ_2 are found by first computing the expressions for $L_+^{(0)}$, $L_-^{(0)}$, and $L_+^{(2)}$. We start with an expansion for $Q(x)$ as a Laguerre series:

$$q_M(x) = \sum_{i=0}^{M} d_i \psi_i(N,b\,;x), \tag{3.57}$$

where d_i's are the Fourier coefficients. Note that the parameters, N and b (hence, the basis of the expansion and the coefficients, d_i), can vary with the inverse temperature, s. From (3.32), we have

$$Q_s(x) = \frac{Q(x)e^{-sx}}{Z_Q(s)} \approx \sum_{i=0}^{M} f_i \psi_i(N,b\,;x)e^{-sx} \tag{3.58}$$

with $f_i = d_i/Z_Q(s)$. If M, the number terms in the expansion, is much less than N, then the expressions for $L_+^{(0)}$ and $L_-^{(0)}$ (see Appendix 3.G) are given by

$$L_+^{(0)}(\beta,s) \approx U(N,M,b,s)\,\frac{b+c}{2\beta+s}, \quad \beta < 3b + s,\ M \ll N,$$

and

$$L_-^{(0)}(\beta,s) \approx U(N,M,b,s)\,\frac{b+c}{2\beta-s}, \quad \frac{s}{2} < \beta < 3b + 2s,\ M \ll N,$$

where

$$U(N,M,b,s) = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} (2b)^j \sum_{m=0}^{M} h_m \sum_{i=0}^{m} \gamma_i^{(m)} (2b)^i \frac{\Gamma(2N+i+j)}{(N+i)(b+c)^{2N+i+j}},$$

with

$$g_n = f_n (2b)^{N-1/2} \left[\frac{\Gamma(n+1)}{\Gamma(2N+n-1)}\right]^{1/2}$$

and

$$h_m = d_m (2b)^{N-1/2} \left[\frac{\Gamma(m+1)}{\Gamma(2N+m-1)}\right]^{1/2}.$$

The expression for $L_+^{(2)}$ is obtained from (3.30),

$$L_{+}^{(n)}(\beta,s) = (-1)^n \frac{\partial^n}{\partial\beta^n} L_{+}^{(0)}(\beta,s),$$

by differentiating $L_{+}^{(0)}$ twice to obtain

$$L_{+}^{(2)}(\beta,s) \approx U(N,M,b,s) \frac{8(b+c)}{(2\beta+s)^3}, \quad M \ll N. \tag{3.59}$$

Since the differentiation is taken with respect to β, the parameters, N and b, can vary with s and the result still holds. Substituting the expressions for $L_{+}^{(0)}$, $L_{-}^{(0)}$ and $L_{+}^{(2)}$ into (3.29), we get, for the acceptance ratio,

$$\rho_0(\beta,s) = \frac{2L_{+}^{(0)}(\beta,s)}{L_{+}^{(0)}(\beta,s) + L_{-}^{(0)}(\beta,s)} \approx \frac{2\beta - s}{2\beta} \tag{3.60}$$

and, for the variance of the energy increment,

$$\rho_2(\beta,s) = \frac{2L_{+}^{(2)}(\beta,s)}{L_{+}^{(0)}(\beta,s) + L_{-}^{(0)}(\beta,s)} \approx \frac{4(2\beta - s)}{\beta(2\beta+s)^2} \tag{3.61}$$

subject to the conditions $s/2 < \beta < 3b + s$ and $M \ll N$. These expressions are identical to the corresponding expressions obtained for the gamma density model.

3.2.2.3. Efficient annealing schedule

Since the expressions for ρ_0 and ρ_2 are the same as those obtained for the gamma density model, the schedule has to be the same also. Therefore,

$$s_+ = s + \lambda \frac{4\rho_0(1-\rho_0)^2}{s^2(2-\rho_0)^2\sigma^3(s)},$$

is the efficient λ-schedule.

The selection of parameters, N and b, of the Laguerre series $q_M(x) = \Theta_M(N,b;x)$, needed for the approximation of $Q(x)$, is now discussed. The only condition for N and b we have so far is that $N \geq 50$ for small values of s, which can be relaxed for large values of s (see Appendix 3.C). Thus, we are tempted to let $N \to \infty$ and take Laguerre series with a large number of terms. However, this does not work out nicely, for $\psi_n(N,b;x)$, $n = 0, 1, \ldots$ become delta functions at infinity if we let

$N \rightarrow \infty$. In order to approximate any practical energy density function well with delta functions, we would have to take an infinite number of terms. Therefore, the good choices for N and b are those that let $\psi_0(N, b\,;\, x)$ approximate $Q(x)$ well in the first place so that few extra terms are needed in order to have the necessary accuracy. Equating the mean and variance of $Q(x)$ with those of $\psi_0(N, b\,;\, x)$, we arrive at

$$\mu_Q = \frac{N}{b} \quad \text{and} \quad {\sigma_Q}^2 = \frac{N}{b^2}$$

which yield

$$N = \frac{{\mu_Q}^2}{{\sigma_Q}^2} \quad \text{and} \quad b = \frac{\mu_Q}{{\sigma_Q}^2}$$

as our choice of N and b.

We now investigate the constraints. Except for

$$\sigma^2(s) \gg \lambda \left| \frac{\partial \sigma(s)}{\partial s} \right| \tag{3.62}$$

and

$$\sigma(s) \geq \frac{\lambda}{1 - r(s)} \left| \frac{\partial r(s)}{\partial s} \right|, \tag{3.63}$$

all other constraints yield similar conditions as in the case of the gamma density model. For the two constraints in (3.62) and (3.63), we need to know the coefficients of the Laguerre series in order to find the exact relationship. But if a probability density function can be approximated by a Laguerre series with a small number of terms, it is likely that the variance, $\sigma^2(s)$, would not deviate much from that of a gamma density function. Hence, the constraints in (3.62) and (3.63) should be close to

$$s \geq \frac{2\lambda}{\sigma(0)} \quad \text{and} \quad \sqrt{\overline{N}} \gg \lambda,$$

derived for the gamma density model. Furthermore, these constraints are less restrictive than the constraints,

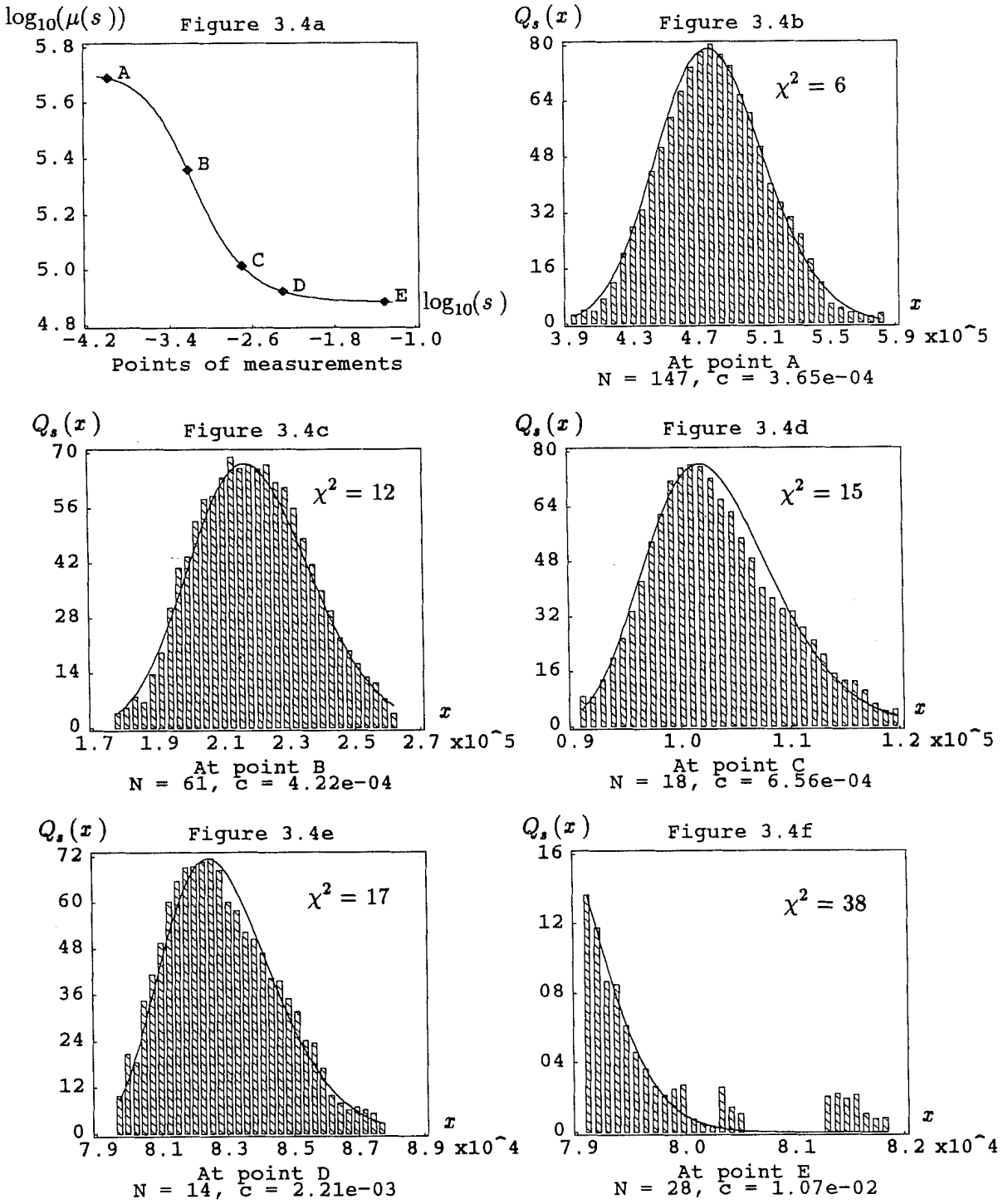

Figure 3.4: Test results for $Q_s(x)$ at different values of s for a 100-city traveling salesman problem. The inverse temperatures at which the tests were performed are indicated in Figure 3.4a while the measured $Q_s(x)$ (histograms) and the fitted one-term Laguerre series (curves) with parameters N and c are shown in Figure 3.4b to Figure 3.4f.

$$s \geq \frac{7}{\sigma(0)} \quad \text{and} \quad \frac{(\mu(0) - \mu(\infty))^2}{\sigma^2(0)} \gg 1, \tag{3.64}$$

that we impose on the efficient λ-schedule. Consequently, the efficient λ-schedule for the gamma density model,

$$s_+ = s + \lambda \frac{4\rho_0(1-\rho_0)^2}{s^2(2-\rho_0)^2\sigma^3(s)}, \quad s \geq \frac{7}{\sigma(0)}, \frac{(\mu(0) - \mu(\infty))^2}{\sigma^2(0)} \gg 1, \tag{3.65}$$

is also the efficient λ-schedule for the close-to-gamma density model.

3.2.2.4. Test of the close-to-gamma density model

How do we test for the accuracy of the approximation for $Q(x)$, the implicit energy density function? We can, of course, measure $Q(x)$ and compare the resulting histogram with its approximation. However, this method only gives an accurate picture of $Q(x)$ at a few standard deviations around the mean, and this is not good enough. As the temperature drops, the system spends more and more time in regions of low energy; the left tail of $Q(x)$ becomes increasingly important. An alternate method must be devised to get a good description of the left tail.

The function $Q_s(x)$, defined in (3.8),

$$Q_s(x) = \frac{Q(x)e^{-sx}}{Z_Q(s)},$$

emphasizes the importance of the left tail of $Q(x)$ by scaling $Q(x)$ unevenly with e^{-sx}. Thus, comparing the histogram of $Q_s(x)$ with its approximation is a better method. To construct the histogram of $Q_s(x)$ from the histogram of $P_s(x)$, the quantity that can be measured directly, we need the following relation:

$$Q_{s/2}(x) = \left[\frac{\kappa Z(s)}{Z_Q(s/2)^2} P_s(x)\right]^{1/2}. \tag{3.66}$$

This relation is obtained by first substituting the definition of $P_s(x)$ in (3.6) into (3.26),

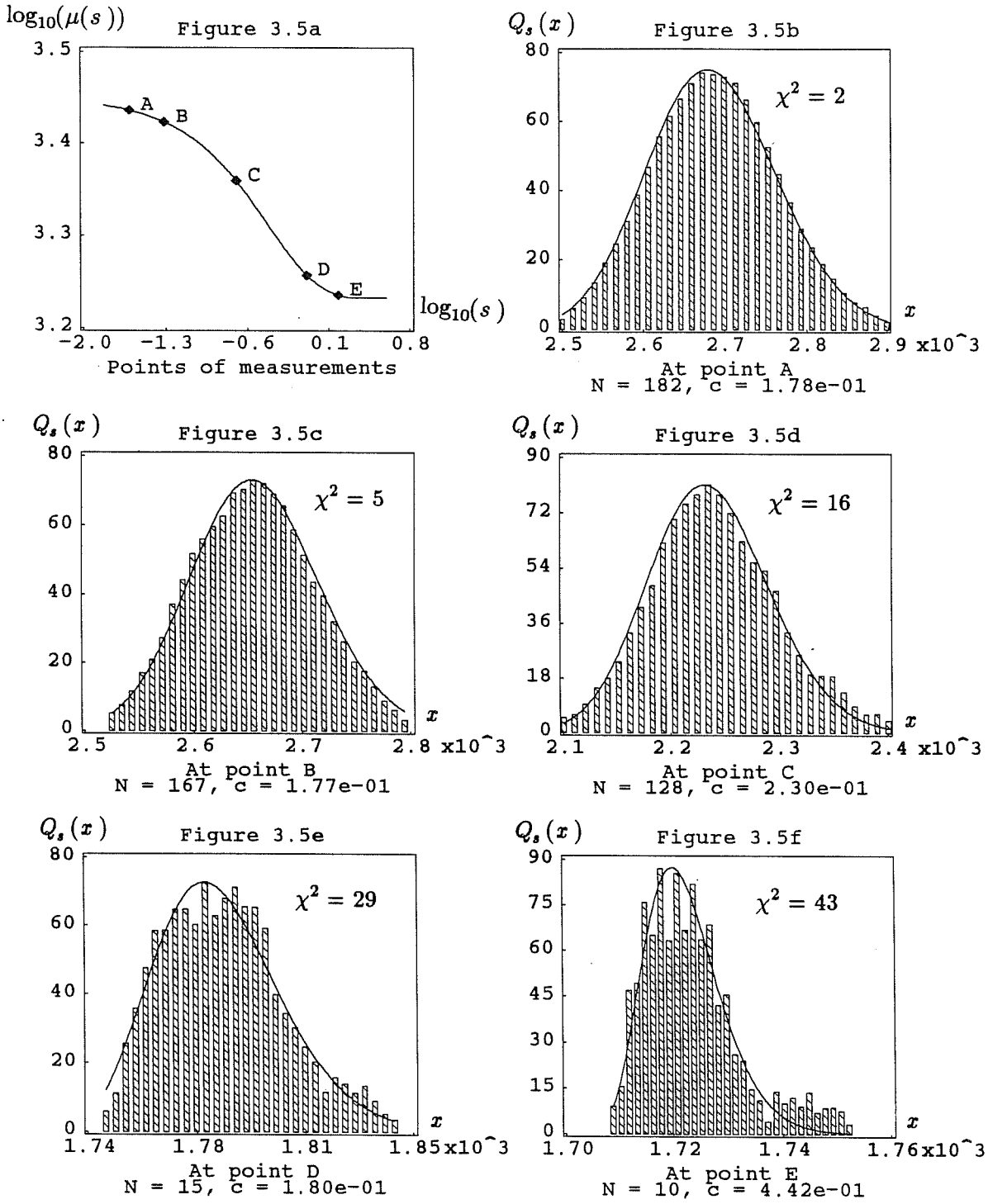

Figure 3.5: Test results for $Q_s(x)$ at different values of s for a graph partition problem with $N = 500$ and $d = 20$. The inverse temperatures at which the tests were performed are indicated in Figure 3.5a while the measured $Q_s(x)$ (histograms) and the fitted one-term Laguerre series (curves) with parameters N and c are shown in Figure 3.5b to Figure 3.5f.

$$P(x) = \frac{1}{\kappa} Q(x)^2,$$

to get

$$P_s(x) = \frac{1}{\kappa Z(s)} Q(x)^2 e^{-sx}.$$

Then, from the definition of $Q_s(x)$ in (3.8), we obtain

$$P_s(x) = \frac{Z_Q(s/2)^2}{\kappa Z(s)} Q_{s/2}(x)^2$$

and, hence, the relation (3.66).

Since we need to measure $P_s(x)$, the problem of highly correlated data associated with the testing of the exponential locality model would occur. Therefore, we need to combine data from multiple executions, as described in Section 3.1.4, in order to obtain an accurate description of $Q_s(x)$. We performed the tests on a 100-city travel salesman problem and a graph partition problem with $N = 500$ (number of vertices) and $d = 20$ (average degree). The inverse temperatures at which the tests were performed are indicated on the annealing curves shown in Fig. 3.4a and Fig. 3.5a, respectively. The test results are displayed in Fig. 3.4b to Fig. 3.4f and Fig. 3.5b to Fig. 3.5f in which the histograms of $Q_s(x)$ for different values of s together with the fitted 1-term Laguerre series (gamma density function) are shown. The chi-square values, χ^2, for the goodness of fit tests are also shown in the figures. Based on these values, we fail to reject the model at the 0.05 level of significance. Therefore, the model is accepted. Note that the parameters N and $c = b + s$ are different at different inverse temperatures. Despite that $Q(x)$ is invariant with respect to s, the parameters of the approximation for $Q(x)$ can be a function of s as long as N is greater than 50 for small values of s or is greater than certain small value close to 3.2 for large values of s.

We have derived the efficient λ-schedule that minimizes the final average energy with respect to a sequence of inverse temperatures for a given move generation strategy and a given number of steps. We have also derived the conditions that the move

generation strategy should satisfy in order to minimize the final average energy further. In Chapter 4, we shall discuss practical aspects of applying the efficient λ-schedule in (3.65), including parameter estimation and move generation control method.

References

[Butko68] E. Butkov, *Mathematical Physics,* Addison-Wesley, 1968.

[Fisz67] M. Fisz, *Probability Theory and Mathematical Statistics,* 3rd Edition, Wiley Publishing Company, 1967.

[Grads80] I.S. Gradshteyn and I.M. Ryzhik, *Table of Integrals, Series, and Products,* Academic Press, 1980.

[Luke69] Y. Luke, *The Special Functions and Their Approximations,* Vol. 1, Academic Press, 1969.

[Sanso59] G. Sansone, *Orthogonal Functions,* Interscience Publishers, Inc., 1959.

[Walpo78] R. Walpole, and R. Myers, *Probability and Statistics for Engineers and Scientists,* 2nd Edition, Macmillan Publishing Co., Inc., 1978.

Appendix 3.A: Simplification of the expression for ρ_n

Show that the factor $Z_Q(s) / (\kappa Z(s))$ in

$$\rho_n(s) = \frac{Z_Q(s)}{\kappa Z(s)} \int_{-\infty}^{\infty} z^n A_s(z) H_s(z) dz, \tag{A.1}$$

where

$$H_s(z) = \int_{-\infty}^{\infty} Q(y) Q_s(y-z) dy = Q(z) * Q_s(-z), \tag{A.2}$$

can be eliminated to obtain

$$\rho_n(s) = \begin{cases} \dfrac{2\int_0^{\infty} z^n G(|z|) H_s(-z) dz}{\int_{-\infty}^{\infty} G(|z|) H_s(-z) dz} & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd}. \end{cases} \tag{A.3}$$

First, we show that

$$H_s(-z) = e^{-sz} H_s(z).$$

To prove this equality, we observe from (A.2) that

$$H_s(-z) = \int_{-\infty}^{\infty} Q(y) Q_s(y + z) dy.$$

Substituting (3.8) into the above expression and replacing $y + z$ with x, we obtain

$$H_s(-z) = \int_{-\infty}^{\infty} \frac{Q(y) Q(y + z) e^{-s(y+z)}}{Z_Q(s)} dy = \int_{-\infty}^{\infty} \frac{Q(x - z) Q(x) e^{-sx}}{Z_Q(s)} dx.$$

Pulling out e^{-sz} and replacing $Q(x - z) e^{-s(x-z)} / Z_Q(s)$ with $Q_s(x - z)$ using (3.8), we get the equality

$$H_s(-z) = e^{-sz} \int_{-\infty}^{\infty} Q(x)Q_s(x - z)dx = e^{-sz} H_s(z).$$

Second, from this equality and (3.4), we observe that for $z > 0$, we can manipulate $A_s(z)H_s(z)$, part of the integrand in (A.1), to obtain

$$A_s(z)H_s(z) = G(|z|)e^{-sz} H_s(z) = G(|z|)H_s(-z) = A_s(-z)H_s(-z).$$

Therefore, $A_s(z)H_s(z)$ is an even function. Consequently, the integrand in (A.1) is an even or odd function depending on whether n is even or odd. Using this property and the equality $A_s(z)H_s(z) = G(|z|)H_s(-z)$ for $z > 0$, we rewrite ρ_n in (A.1) to get

$$\rho_n(s) = \begin{cases} \dfrac{2Z_Q(s)}{\kappa Z(s)} \displaystyle\int_0^{\infty} z^n G(|z|)H_s(-z)dz & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd}. \end{cases} \tag{A.4}$$

Third, we simplify the preceding expression for ρ_n by removing the factor $Z_Q(s) / (\kappa Z(s))$. To carry out this simplification, we need the following definition of the Fourier transform pair:

$$F\{g(x)\} = \phi(\omega) = \int_{-\infty}^{\infty} g(x)e^{j\omega x} dx$$

$$F^{-1}\{\phi(\omega)\} = g(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \phi(\omega)e^{-j\omega x} d\omega.$$

From these it can be shown that

$$F\{f(x) * g(x)\} = F\{f(x)\}F\{g(x)\},$$

known as the *convolution theorem.* Replacing $P(z)$ with $K(z)Q(z) / \kappa$ using (3.7) and $Q(z)e^{-sz}$ by $Z_Q(s)Q_s(z)$ using (3.8), we rewrite (1.16),

$$Z(s) = \int_{-\infty}^{\infty} P(z)e^{-sz} dz,$$

as

$$Z(s) = \frac{1}{\kappa} \int_{-\infty}^{\infty} K(z) Q(z) e^{-sz}\, dz = \frac{Z_Q(s)}{\kappa} \int_{-\infty}^{\infty} K(z) Q_s(z)\, dz.$$

Upon multiplication by $\kappa / Z_Q(s)$ and application of Parseval's second theorem [Butko68, p. 267] for real functions,

$$\int_{-\infty}^{\infty} F\{f(z)\} F^*\{g(z)\}\, d\omega = \int_{-\infty}^{\infty} f(z) g(z)\, dz$$

where F^* denotes the complex conjugate of F, we obtain

$$\frac{\kappa Z(s)}{Z_Q(s)} = \int_{-\infty}^{\infty} K(z) Q_s(z)\, dz = \int_{-\infty}^{\infty} F^*\{K(z)\} F\{Q_s(z)\}\, d\omega.$$

We can express the preceding equation as

$$\frac{\kappa Z(s)}{Z_Q(s)} = \int_{-\infty}^{\infty} F^*\{G(|z|)\} F^*\{Q(z)\} F\{Q_s(z)\}\, d\omega$$

by substituting the Fourier transform of equation (3.2) and applying the convolution theorem. Using the complex conjugate of the Fourier transform of equation (A.2), and applying Parseval's second theorem once more, we get an expression for the factor $\kappa Z(s) / Z_Q(s)$,

$$\frac{\kappa Z(s)}{Z_Q(s)} = \int_{-\infty}^{\infty} F^*\{G(|z|)\} F\{H_s(-z)\}\, d\omega = \int_{-\infty}^{\infty} G(|z|) H_s(-z)\, dz.$$

Finally, substituting this factor back into (A.4), we arrive at

$$\rho_n(s) = \begin{cases} \dfrac{2\int_0^{\infty} z^n G(|z|) H_s(-z)\, dz}{\int_{-\infty}^{\infty} G(|z|) H_s(-z)\, dz} & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd.} \end{cases}$$

Appendix 3.B: Expressions for $L_{\pm}^{(0)}$ for the gamma density model

Show that the expressions for $L_{+}^{(0)}$ and $L_{-}^{(0)}$ are

$$L_{+}^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{b+c}{2\beta+s}(1+\Re_{+}^{(0)}), \quad \beta < 3b+s,$$

$$L_{-}^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{b+c}{2\beta-s}(1+\Re_{-}^{(0)}), \quad \frac{s}{2} < \beta < 3b+2s,$$

with

$$\begin{aligned} |\Re_{+}^{(0)}| &\le \left|\frac{2z_+(1-z_+)}{(N+1)(4z_+(1-z_+)-1)}\right|, \\ |\Re_{-}^{(0)}| &\le \left|\frac{2z_-(1-z_-)}{(N+1)(4z_-(1-z_-)-1)}\right|, \end{aligned} \tag{B.1}$$

where $z_+ = (b-\beta)/(b+c)$ and $z_- = (c-\beta)/(b+c)$.

We start with (3.30) for $L_{+}^{(0)}$:

$$L_{+}^{(0)}(\beta,s) = \int_{-\infty}^{\infty} \frac{\phi(\omega)\phi_s^*(\omega)}{\beta - j\omega} d\omega.$$

Using Parseval's second theorem for real functions,

$$\int_{-\infty}^{\infty} F\{f(z)\}F^*\{g(z)\}d\omega = \int_{-\infty}^{\infty} f(z)g(z)dz,$$

with $F = \phi(\omega)$ and $F^* = \phi_s^*(\omega)/(\beta - j\omega)$, we get

$$L_{+}^{(0)}(\beta,s) = \int_{-\infty}^{\infty} Q(x)F^{-1}\left\{\frac{\phi_s(\omega)}{\beta + j\omega}\right\}dx.$$

Since $Q(x) = 0$ for $x < 0$, we rewrite the preceding equation as

$$L_{+}^{(0)}(\beta,s) = \int_{0}^{\infty} Q(x)F^{-1}\left\{\frac{\phi_s(\omega)}{\beta + j\omega}\right\}dx. \tag{B.2}$$

Applying the convolution theorem to $\phi_s(\omega) \,/\, (\beta + j\omega)$,

$$F^{-1}\left\{\frac{\phi_s(\omega)}{\beta + j\omega}\right\} = Q_s(x) * F^{-1}\left\{\frac{1}{\beta + j\omega}\right\},$$

and noting that

$$F^{-1}\left\{\frac{1}{\beta + j\omega}\right\} = \begin{cases} e^{\beta x} & \text{if } x < 0 \\ 0 & \text{otherwise,} \end{cases}$$

we obtain

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta + j\omega}\right\} = \int_x^\infty e^{\beta(x-z)} \frac{c^N}{\Gamma(N)} z^{N-1} e^{-cz}\, dz, \quad x > 0,$$

using the formula for $Q_s(x)$ in (3.33). Pulling $e^{\beta x} c^N / \Gamma(N)$ out of the integral in the preceding expression and replacing the dummy variable z with $y = (c+\beta)z$ lead to

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta + j\omega}\right\} = \left[\frac{c}{c+\beta}\right]^N \frac{e^{\beta x}\Gamma(N,(c+\beta)x)}{\Gamma(N)}, \quad x > 0,$$

where $\Gamma(N,x) \equiv \int_x^\infty y^{N-1} e^{-y}\, dy$ is the incomplete gamma function. Substitution of the above expression for F^{-1} in (B.2) yields

$$L_+^{(0)}(\beta,s) = \frac{1}{\Gamma(N)^2}\left[\frac{bc}{c+\beta}\right]^N \int_0^\infty x^{N-1} e^{-(b-\beta)x}\Gamma(N,(c+\beta)x)\,dx.$$

Using [Grads80, p. 663], we evaluate the integral to obtain

$$L_+^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N {}_2F_1(1, 2N; N+1; \frac{b-\beta}{b+c}) \qquad \text{(B.3)}$$

where ${}_2F_1$ is the Gauss's hypergeometric function [Grads80, p. 1093], defined as

$${}_2F_1(\alpha, \beta; \gamma; z) = \sum_{k=0}^{\infty} \frac{(\alpha)_k (\beta)_k}{(\gamma)_k} \frac{z^k}{k!} \quad \text{with} \quad (\alpha)_k = \alpha(\alpha+1)\cdots(\alpha+k-1).$$

Similarly, we evaluate $L_-^{(0)}$ starting from (3.31) to obtain

$$L_{-}^{(0)}(\beta,s) = \int_{-\infty}^{\infty} Q(x) F^{-1}\left\{\frac{\phi_s(\omega)}{\beta - j\omega}\right\} dx = \int_{0}^{\infty} Q(x) F^{-1}\left\{\frac{\phi_s(\omega)}{\beta - j\omega}\right\} dx. \qquad \text{(B.4)}$$

Applying the convolution theorem to $\phi_s(\omega)$ / $(\beta - j\omega)$,

$$F^{-1}\left\{\frac{\phi_s(\omega)}{\beta - j\omega}\right\} = Q_s(x) * F^{-1}\left\{\frac{1}{\beta - j\omega}\right\},$$

and noting that

$$F^{-1}\left\{\frac{1}{\beta - j\omega}\right\} = \begin{cases} e^{-\beta x} & \text{if } x > 0 \\ 0 & \text{otherwise,} \end{cases}$$

we get

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta - j\omega}\right\} = \int_{0}^{x} e^{-\beta(x-z)} \frac{c^N}{\Gamma(N)} z^{N-1} e^{-cz}\, dz, \quad x > 0.$$

Pulling $e^{-\beta x} c^N / \Gamma(N)$ out of the integral and replacing the dummy variable z with $y = (c - \beta)z$, we obtain

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta - j\omega}\right\} = \left\{\frac{c}{c - \beta}\right\}^N \frac{e^{-\beta x}\gamma(N, (c-\beta)x)}{\Gamma(N)}, \quad x > 0,$$

where $\gamma(N, x) \equiv \Gamma(N) - \Gamma(N, x)$. Putting this expression back into (B.4) yields

$$L_{-}^{(0)}(\beta,s) = \frac{1}{\Gamma(N)^2}\left\{\frac{bc}{c - \beta}\right\}^N \int_{0}^{\infty} x^{N-1} e^{-(b+\beta)x} \Gamma(N, (c-\beta)x)\, dx.$$

Again, using [Grads80, p. 663], we arrive at

$$L_{-}^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left\{\frac{bc}{(b+c)^2}\right\}^N {}_2F_1(1, 2N; N+1; \frac{c-\beta}{b+c}). \qquad \text{(B.5)}$$

We can simplify the preceding expressions for $L_{+}^{(0)}$ and $L_{-}^{(0)}$. But first, we need an approximation for the hypergeometric function

$${}_2F_1(1, 2N; N+1; z) = 1 + \frac{2N}{N+1} z + \frac{2N}{N+1}\frac{2N+1}{N+2} z^2 + \cdots$$

which, in general, has no closed form representation. However, if N is sufficiently large,

we can approximate the individual factors $2N\ /\ (N+1)$, $(2N+1)\ /\ (N+2)$,..., by 2 to obtain

$$ {}_2F_1(1,\ 2N;\ N+1;\ z\ ) \approx 1 + 2z + (2z)^2 + \cdots . $$

(See Appendix 3.C for the error of this approximation.) Moreover, if $|\ z\ | < 1/2$, we have

$$ {}_2F_1(1,\ 2N;\ N+1;\ z\ ) \approx \frac{1}{1-2z}, \quad |\ z\ | < \frac{1}{2}. \tag{B.6} $$

For the range of $-1 < z \leq -1/2$, we apply to ${}_2F_1(1,\ 2N;\ N+1;\ z\ )$ the transformations from [Grads80, p. 1043]: first

$$ {}_2F_1(2\alpha,\ 2\beta;\ \alpha+\beta+\frac{1}{2};\ z\ ) = {}_2F_1(\alpha,\ \beta;\ \alpha+\beta+\frac{1}{2};\ 4z\,(1-z\ )\ ) $$

to obtain

$$ {}_2F_1(1,\ 2N;\ N+1;\ z\ ) = {}_2F_1(\frac{1}{2},\ N;\ N+1;\ 4z\,(1-z\ )\ ), $$

then

$$ {}_2F_1(\alpha,\ \beta;\ \gamma;\ z\ ) = (1-z\ )^{-\alpha}\ {}_2F_1(\alpha,\ \gamma-\beta;\ \gamma;\ \frac{z}{z-1}) $$

to obtain

$$ {}_2F_1(\frac{1}{2},\ N;\ N+1;\ 4z\,(1-z)) = (1-4z\,(1-z))^{-1/2}\ {}_2F_1(\frac{1}{2},\ 1;\ N+1;\ \frac{4z\,(1-z\ )}{4z\,(1-z\ )-1}). \tag{B.7} $$

Since $|\ 4z\,(1-z\ )\ /\ (4z\,(1-z\ )-1)\ | < 1$, we have, for large N,

$$ {}_2F_1(\frac{1}{2},\ 1;\ N+1;\ \frac{4z\,(1-z\ )}{4z\,(1-z\ )-1}) = 1 + \frac{1/2}{N}\frac{4z\,(1-z\ )}{4z\,(1-z\ )-1} + \cdots \approx 1. $$

Substituting this expression into (B.7), we arrive at

$$ {}_2F_1(\frac{1}{2},\ N;\ N+1;\ 4z\,(1-z\ )\ ) \approx (1-4z\,(1-z))^{-1/2} = \frac{1}{1-2z} $$

which is combined with (B.6) to get

$$ {}_2F_1(\frac{1}{2},\, N;\, N+1;\, 4z\,(1-z)\,) \approx \frac{1}{1-2z}, \quad -1 < z < \frac{1}{2}. $$

The error bound for the preceding expression can be obtained using (C.4):

$$ {}_2F_1(1,\, 2N;\, N+1;\, z) = \frac{1}{1-2z}(1 + R_1), \quad -1 < z < \frac{1}{2} \tag{B.8} $$

with

$$ |\, R_1 \,| \;\le\; \left| \frac{2z\,(1-z)}{(N+1)(4z\,(1-z)-1)} \right|. $$

Finally, applying this formula to $L_{+}^{(0)}$ and $L_{-}^{(0)}$, we arrive at

$$ L_{+}^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\,\Gamma(N)^2}\left\{\frac{bc}{(b+c)^2}\right\}^N \frac{b+c}{2\beta+s}(1+\Re_{+}^{(0)}), \quad \beta < 3b+s\,, $$

$$ L_{-}^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\,\Gamma(N)^2}\left\{\frac{bc}{(b+c)^2}\right\}^N \frac{b+c}{2\beta-s}(1+\Re_{-}^{(0)}), \quad \frac{s}{2} < \beta < 3b+2s\,, $$

where $\Re_{+}^{(0)}$ and $\Re_{-}^{(0)}$ are given by (B.1).

Appendix 3.C: Error of the approximation for ${}_2F_1$

We are interested in the relationship between the value of z and the error of the approximation

$$ {}_2F_1(1,\, 2N;\, N+1;\, z) \approx \frac{1}{1-2z} \tag{C.1} $$

for large N with $z \to 1/2$ from $z = 0$.

We first find a formula that bounds the approximation error. Application of the transformation in [Grads80, p. 1043]

$$ {}_2F_1(2\alpha,\, 2\beta;\, \alpha+\beta+\frac{1}{2};\, z) = {}_2F_1(\alpha,\, \beta;\, \alpha+\beta+\frac{1}{2};\, 4z\,(1-z)\,) $$

to ${}_2F_1(1,\, 2N;\, N+1;\, z)$ yields

$$ {}_2F_1(1,\, 2N\,;\, N+1;\, z\,) = {}_2F_1(\frac{1}{2},\, N\,;\, N+1;\, 4z\,(1-z\,)\,) $$

which, upon application of

$$ {}_2F_1(\alpha,\, \beta;\, \gamma;\, z\,) = (1-z\,)^{-\alpha}\, {}_2F_1(\alpha,\, \gamma-\beta;\, \gamma;\, \frac{z}{z-1}), $$

a transformation in [Grads80, p. 1043], leads to

$$ {}_2F_1(1,\, 2N\,;\, N+1;\, z\,) = (1-4z\,(1-z\,))^{-1/2}\, {}_2F_1(\frac{1}{2},\, 1;\, N+1;\, \frac{4z\,(1-z\,)}{4z\,(1-z\,)-1}). \quad \text{(C.2)} $$

But from [Luke69, p. 235], we have, for large γ,

$$ {}_2F_1(\alpha,\, \beta;\, \gamma;\, z\,) = \sum_{k=0}^{m} \frac{(\alpha)_k\,(\beta)_k\, z^{\,k}}{(\gamma)_k\, k\,!} + R_{m+1}, \quad z < 1,\ \alpha,\beta > -1, \qquad \text{(C.3)} $$

where the residual R_{m+1} satisfies

$$ |\, R_{m+1}\,| \;\le\; |\, z\,|^{\,m+1} \frac{(\alpha)_{m+1}(\beta)_{m+1}}{(m\,+\,1)!} \left| \frac{\Gamma(\gamma)}{\Gamma(\gamma-\beta)} \right| \gamma^{-\beta-m-1}[1 + O\,(\gamma^{-1})]. \qquad \text{(C.4)} $$

Letting $m = 0$ and applying the preceding formula to the ${}_2F_1$ on the right hand side of (C.2), we get

$$ {}_2F_1(\frac{1}{2},\, 1;\, N+1;\, \frac{4z\,(1-z\,)}{4z\,(1-z\,)-1}) = 1 + R_1 $$

with a bounded residual R_1:

$$ |\, R_1\,| \;\le\; \left| \frac{4z\,(1-z\,)}{4z\,(1-z\,)-1} \right| \left| \frac{1}{2} \frac{\Gamma(N+1)}{\Gamma(N\,)} (N+1)^{-2} \le \right| \frac{4z\,(1-z\,)}{2N\,(4z\,(1-z\,)-1)} \Bigg|. $$

Substituting the two preceding expressions into (C.2) and noting that $1-4z\,(1-z\,) = (1-2z\,)^2$, we have

$$ {}_2F_1(1,\, 2N\,;\, N+1;\, z\,) = {}_2F_1(\frac{1}{2},\, N\,;\, N+1;\, 4z\,(1-z\,)\,) = \frac{1}{1-2z}(1 + R_1) $$

where

$$|R_1| \leq \left| \frac{4z(1-z)}{2N(4z(1-z)-1)} \right|.$$

Next, we find the restriction on z as $z \rightarrow 1/2$ so that the residual R_1 is bounded by a constant k:

$$|R_1| \leq \left| \frac{4z(1-z)}{2N(4z(1-z)-1)} \right| \leq k.$$

Since $z \rightarrow 1/2$ from $z = 0$, we have $0 < 4z(1-z) < 1$. Therefore, we can rewrite the bound as

$$4z(1-z) \leq 2Nk[1-4z(1-z)].$$

Noting that $1-4z(1-z) = (1-2z)^2$ in the preceding expression, we get

$$1-(1-2z)^2 \leq 2Nk(1-2z)^2.$$

Factorizing $(1-2z)^2$ out of the expression and taking square roots on both sides, we obtain

$$\frac{1}{r} \leq 1-2z, \tag{C.5}$$

where $r = \sqrt{2Nk+1}$. Since $z_+ = (b-\beta)/(b+c)$ is always less than or equal to $z_- = (c-\beta)/(b+c)$, we only need to investigate the bound for

$$z = z_- = \frac{c-\beta}{b+c}$$

in (B.5) of Appendix 3.B for β close to $\beta_{des} = 0.89s$. Substituting β_{des} for β in the preceding expression and noting that $c = b+s$, we obtain

$$z = \frac{b+0.11s}{2b+s}.$$

Putting this expression back into (C.5), we get

$$\frac{1}{r} \leq 1-\frac{2b+0.22s}{2b+s} \tag{C.6}$$

which leads to

$$s \geq \frac{2b}{r(0.78 - 1/r)}.$$

Assuming N is large, we get $r \approx \sqrt{2Nk}$ and, hence,

$$s \geq \frac{2b}{\sqrt{2Nk}\,(0.78 - 1/\sqrt{2Nk}\,)} \tag{C.7}$$

from which we obtain, letting $k = 0.1$, a 10% bound on the residual, and assuming $N \geq 50$,

$$s \geq 9.64\frac{b}{\sqrt{N}}.$$

Finally, applying (3.45), $\sigma^2(0) = (2N-1) / (4b^2) \approx 2N / (4b^2)$, to the preceding expression, we arrive at a condition on s:

$$s \geq \frac{7}{\sigma(0)}. \tag{C.8}$$

The condition that $N \geq 50$ can be relaxed for large s. Cross-multiplication of (C.6) leads to

$$r \geq \frac{2b + s}{0.78s}.$$

If s is large so that $s \gg 2b$, we can approximate the preceding expression by

$$\sqrt{2Nk + 1} \geq 1.28$$

which yields

$$N \geq \frac{0.32}{k}.$$

For $k = 0.1$, this condition becomes

$$N \geq 3.2.$$

Note that the omission of $O(\gamma^{-1})$ in (C.4) may not be justified for such a small value of N.

Appendix 3.D: Expression for $L_+^{(2)}$ for the gamma density model

Show that the expression for $L_+^{(2)}$ is

$$L_+^{(2)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{8(b+c)}{(2\beta+s)^3}(1+\Re_+^{(2)})$$

with

$$\Re_+^{(2)} = R_1 - \frac{5}{2(1-2z)^2(N+1)}(1+R_2) + \frac{3}{(1-2z)^4(N+1)(N+2)}(1+R_3), \quad \text{(D.1)}$$

where

$$\begin{aligned} |R_1| &\le \left|\frac{2z(1-z)}{(N+1)(4z(1-z)-1)}\right|, \\ |R_2| &\le \left|\frac{12z(1-z)}{(N+2)(4z(1-z)-1)}\right|, \\ |R_3| &\le \left|\frac{30z(1-z)}{(N+3)(4z(1-z)-1)}\right|, \end{aligned} \quad \text{(D.2)}$$

and $z = (b-\beta)\,/\,(b+c)$.

We start with the expression for $L_+^{(0)}$ in (B.3),

$$L_+^{(0)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N {}_2F_1(1,\,2N;\,N+1;\,\frac{b-\beta}{b+c}),$$

which can be manipulated using (C.2) into an alternate form:

$$L_+^{(0)}(\beta,s) = K(N,b,s)\,\frac{1}{1-2z}\;{}_2F_1(\frac{1}{2},\,1;\,N+1;\frac{4z(1-z)}{4z(1-z)-1}),$$

where

$$K(N,b,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N$$

and $z = (b-\beta)\,/\,(b+c)$. According to equation (3.30), we have, for $n = 2$,

$$L_{+}^{(2)}(\beta,s) = (-1)^2 \frac{\partial^2}{\partial\beta^2} \int_{-\infty}^{\infty} \frac{\phi(\omega)\phi_s^*(\omega)}{\beta - j\omega} d\omega = \frac{\partial^2}{\partial\beta^2} L_{+}^{(0)}(\beta,s).$$

The first derivative of $L_{+}^{(0)}$ with respect to β is

$$\frac{\partial}{\partial\beta} L_{+}^{(0)}(\beta,s) = K(N,b,s) \frac{\partial z}{\partial\beta} \left[\frac{2}{(1-2z)^2} {}_2F_1(\frac{1}{2}, 1; N+1; \frac{4z(1-z)}{4z(1-z)-1}) \right.$$

$$\left. + \frac{1}{1-2z} \frac{\partial}{\partial z} {}_2F_1(\frac{1}{2}, 1; N+1; \frac{4z(1-z)}{4z(1-z)-1}) \right],$$

which, upon substitution of the elementary relation [Luke69, p. 44]

$$\frac{\partial^n}{\partial z^n} {}_2F_1(\alpha, \beta; \gamma; z) = \frac{(\alpha)_n (\beta)_n}{(\gamma)_n} {}_2F_1(\alpha+n, \beta+n; \gamma+n; z)$$

and noting that $\partial z / \partial\beta = -1 / (b+c)$, leads to

$$\frac{\partial}{\partial\beta} L_{+}^{(0)}(\beta,s) = K(N,b,s) \frac{-1}{b+c} \left[\frac{2}{(1-2z)^2} {}_2F_1(\frac{1}{2}, 1; N+1; \frac{4z(1-z)}{4z(1-z)-1}) \right.$$

$$\left. + \frac{-2}{(1-2z)^4(N+1)} {}_2F_1(\frac{3}{2}, 2; N+2; \frac{4z(1-z)}{4z(1-z)-1}) \right].$$

After some straightforward but tedious manipulations, we get an expression for the second derivative with respect to β:

$$\frac{\partial^2}{\partial\beta^2} L_{+}^{(0)}(\beta,s) = K(N,b,s) \frac{1}{(b+c)^2} \frac{8}{(1-2z)^3} \left[{}_2F_1(\frac{1}{2}, 1; N+1; \frac{4z(1-z)}{4z(1-z)-1}) \right.$$

$$- \frac{5}{2(1-2z)^2(N+1)} {}_2F_1(\frac{3}{2}, 2; N+2; \frac{4z(1-z)}{4z(1-z)-1})$$

$$\left. + \frac{3}{(1-2z)^4(N+1)(N+2)} {}_2F_1(\frac{5}{2}, 3; N+3; \frac{4z(1-z)}{4z(1-z)-1}) \right].$$

The formula in (C.3) can be used to approximate these hypergeometric functions to obtain

$$\frac{\partial^2}{\partial\beta^2} L_+^{(0)}(\beta,s) = K(N,b,s)\frac{1}{(b+c)^2}\frac{8}{(1-2z)^3}(1+\Re_+^{(2)}) \tag{D.3}$$

with

$$\Re_+^{(2)} = R_1 - \frac{5}{2(1-2z)^2(N+1)}(1+R_2) + \frac{3}{(1-2z)^4(N+1)(N+2)}(1+R_3),$$

where R_1, R_2 and R_3 are the residuals of the hypergeometric functions. The bounds for these residuals can be found easily using equation (C.4). They are given by (D.2). Substituting $(b-\beta)\ /\ (b+c)$ for z in (D.3), we arrive at the expression for $L_+^{(2)}$:

$$L_+^{(2)}(\beta,s) = \frac{\Gamma(2N)}{N\Gamma(N)^2}\left[\frac{bc}{(b+c)^2}\right]^N \frac{8(b+c)}{(2\beta+s)^3}(1+\Re_+^{(2)}).$$

Appendix 3.E: Orthogonal Cartesian system

Show that the system of functions

$$\psi_n(N,b;x) = \left[\frac{\Gamma(n+1)}{\Gamma(2N+n-1)}\right]^{1/2}(2b)^{N-1/2}x^{N-1}e^{-bx}\mathcal{L}_n^{(2N-2)}(2bx),$$

$$n = 0, 1,..., \quad b > 0, \quad N > 1/2, \tag{E.1}$$

forms an orthogonal Cartesian system, where $\mathcal{L}_n^{(\alpha)}(x)$, the Laguerre polynomials, are defined recursively as

$$\begin{aligned} &\mathcal{L}_{-1}^{(\alpha)}(x) = 0, \quad \mathcal{L}_0^{(\alpha)}(x) = 1, \\ &(n+1)\,\mathcal{L}_{n+1}^{(\alpha)}(x) - [2n+\alpha+1-x]\,\mathcal{L}_n^{(\alpha)}(x) + (n+\alpha)\,\mathcal{L}_{n-1}^{(\alpha)}(x) = 0, \quad n = 0, 1,..., \end{aligned} \tag{E.2}$$

and are solutions to the differential equation

$$\frac{d}{dx}\left[x^{\alpha+1}e^{-x}\frac{d\,\mathcal{L}_n^{(\alpha)}(x)}{dx}\right] + nx^{\alpha}e^{-x}\mathcal{L}_n^{(\alpha)}(x) = 0. \tag{E.3}$$

We first show that the system of functions

$$\left[\frac{\Gamma(n+1)}{\Gamma(n+\alpha+1)}\right]^{1/2} x^{\alpha/2}e^{-x/2}\mathcal{L}_n^{(\alpha)}(x), \quad n = 0, 1,...,$$

forms an orthogonal Cartesian system. Then, by simple transformations of variables, we prove that the system of functions $\psi_n(N, b; x)$, $n = 0, 1,...,$ forms another orthogonal Cartesian system.

Multiplication of (E.3) by $\mathcal{L}_m^{(\alpha)}$ leads to

$$\mathcal{L}_m^{(\alpha)}(x)\frac{d}{dx}\left[x^{\alpha+1}e^{-x}\frac{d\mathcal{L}_n^{(\alpha)}(x)}{dx}\right] + nx^{\alpha}e^{-x}\mathcal{L}_n^{(\alpha)}(x)\mathcal{L}_m^{(\alpha)}(x) = 0$$

from which we subtract the equation obtained by interchanging n and m in the preceding expression to get

$$\frac{d}{dx}\left[x^{\alpha+1}e^{-x}\left\{\mathcal{L}_m^{(\alpha)}(x)\frac{d\mathcal{L}_n^{(\alpha)}(x)}{dx} - \mathcal{L}_n^{(\alpha)}(x)\frac{d\mathcal{L}_m^{(\alpha)}(x)}{dx}\right\}\right] + x^{\alpha}e^{-x}(n-m)\mathcal{L}_n^{(\alpha)}(x)\mathcal{L}_m^{(\alpha)}(x) = 0.$$

Integrating between 0 and ∞ for $\alpha > -1$, we obtain

$$(n-m)\int_0^{\infty} x^{\alpha}e^{-x}\mathcal{L}_n^{(\alpha)}(x)\mathcal{L}_m^{(\alpha)}(x)dx = 0;$$

therefore, the second part of the orthogonality relations in (3.53),

$$\int_0^{\infty} x^{\alpha}e^{-x}\mathcal{L}_n^{(\alpha)}(x)\mathcal{L}_m^{(\alpha)}(x)dx = 0, \quad n \neq m, \alpha > -1, \tag{E.4}$$

follows.

We next show that the first part of the orthogonality relations in (3.53),

$$\frac{\Gamma(n+1)}{\Gamma(n+\alpha+1)}\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_n^{(\alpha)}(x)]^2dx = 1, \quad n = 0, 1,..., \alpha > -1, \tag{E.5}$$

holds. Multiplying (E.2) by $x^{\alpha}e^{-x}\mathcal{L}_{n-1}^{(\alpha)}(x)dx$, integrating between 0 to ∞, and making use of (E.4), we get

$$\int_0^{\infty} x^{\alpha+1}e^{-x}\mathcal{L}_n^{(\alpha)}(x)\mathcal{L}_{n-1}^{(\alpha)}(x)dx + (n+\alpha)\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_{n-1}^{(\alpha)}(x)]^2dx = 0.$$

Changing n to $n-1$ in (E.2), multiplying by $x^{\alpha}e^{-x}\mathcal{L}_n^{(\alpha)}(x)dx$, and integrating between 0 to ∞, we obtain

$$\int_0^{\infty} x^{\alpha+1}e^{-x}\mathcal{L}_n^{(\alpha)}(x)\mathcal{L}_{n-1}^{(\alpha)}(x)dx + n\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_n^{(\alpha)}(x)]^2dx = 0.$$

From the preceding two equations follows the recurrence relation

$$\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_n^{(\alpha)}(x)]^2dx = \frac{n+\alpha}{n}\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_{n-1}^{(\alpha)}(x)]^2dx.$$

Since

$$\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_o^{(\alpha)}(x)]^2dx = \int_0^{\infty} x^{\alpha}e^{-x}dx = \Gamma(\alpha+1),$$

we get, from the recurrence relation,

$$\int_0^{\infty} x^{\alpha}e^{-x}[\mathcal{L}_n^{(\alpha)}(x)]^2dx = \frac{(n+\alpha)\cdots(\alpha+1)\Gamma(\alpha+1)}{\Gamma(n+1)}$$

and, hence, equation (E.5). Consequently, the system of functions

$$\left[\frac{\Gamma(n+1)}{\Gamma(n+\alpha+1)}\right]^{1/2} x^{\alpha/2}e^{-x/2}\mathcal{L}_n^{(\alpha)}(x), \quad n = 0, 1,...,$$

forms an orthogonal Cartesian system.

To show that the system of functions in (E.1) forms an orthogonal Cartesian system, we start with the orthogonality relations:

$$\int_0^{\infty} \psi_n(N,b;x)\psi_m(N,b;x)dx = \begin{cases} 1 & \text{if } n = m \\ 0 & \text{otherwise.} \end{cases}$$

Using (E.1) and replacing $2N-2$ with α and $2bx$ with y in the preceding expressions, we obtain

$$\frac{\Gamma(n+1)}{\Gamma(n+\alpha-1)}\int_0^\infty y^\alpha e^{-y}[\mathcal{L}_n^{(\alpha)}(y)]^2 dy = 1, \quad n = 0, 1,...,$$

and

$$\left[\frac{\Gamma(n+1)\Gamma(m+1)}{\Gamma(n+\alpha-1)\Gamma(m+\alpha-1)}\right]^{1/2}\int_0^\infty y^\alpha e^{-y}\mathcal{L}_n^{(\alpha)}(y)\mathcal{L}_m^{(\alpha)}(y)dy = 0, \quad n \neq m,$$

which are (E.5) and (E.4), respectively. Therefore, the system of functions $\psi_n(N,b;x)$, $n = 0, 1,...,$ forms an orthogonal Cartesian system.

Appendix 3.F: Accuracy of the approximations for Q and P

Show that when $Q(x)$, the implicit energy density function, is approximated by the first $(M+1)$ terms of the Laguerre series $\Theta(N,b;x)$,

$$q_M(x) = \sum_{i=0}^{M} d_i \psi_i(N,b;x),$$

with a distance of $D(Q, q_M)$, $P(x)$, the energy density function, can be approximated by the first $(\overline{M}+1)$ terms of the Laguerre series $\Theta(\overline{N},\overline{b};x)$,

$$p_{\overline{M}}(x) = \sum_{i=0}^{\overline{M}} k_i \psi_i(\overline{N},\overline{b};x),$$

with a distance bounded from above by $4D(Q, q_M) / \kappa$, where $\overline{M} = 2M$, $\overline{N} = 2N-1$ and $\overline{b} = 2b$.

Let $Q(x)$ be approximated by

$$q_M(x) = \sum_{i=0}^{M} d_i \psi_i(N,b;x) \tag{F.1}$$

where

$$\psi_n(N,b\,;\,x) = \left[\frac{\Gamma(n+1)}{\Gamma(2N+n-1)}\right]^{1/2}(2b)^{N-1/2}x^{N-1}e^{-bx}\,\mathcal{L}_n^{(2N-2)}(2bx),$$

$$n = 0, 1,..., \quad b > 0, \quad N > 1/2, \tag{F.2}$$

and the Laguerre polynomials, $\mathcal{L}_n^{(\alpha)}$, are

$$\mathcal{L}_n^{(\alpha)}(x) = \sum_{i=0}^{n}\gamma_i^{(n)}x^i\,. \tag{F.3}$$

Also, let $G(N,b)$ be a gamma density function with parameters N and b, respectively,

$$G(N,b) = \frac{b^N}{\Gamma(N)}x^{N-1}e^{-bx}, \quad x > 0.$$

Since $\mathcal{L}_n^{(\alpha)}$ are polynomials of degree n, we can rewrite the $\psi_i(N,b\,;\,x)$'s in (F.1) in terms of gamma density functions and re-group to obtain

$$q_M(x) = \sum_{i=0}^{M}d_i\,\psi_i(N,b\,;\,x) = \sum_{i=0}^{M}f_i\,G(N+i,b)$$

where the f_i's are uniquely determined constants. Substituting the preceding expression into (3.26) provides the following approximation for $P(x)$,

$$p_{\overline{M}}(x) = \frac{1}{\kappa}q_M(x)^2 = \frac{1}{\kappa}\left[\sum_{i=0}^{M}f_i\,G(N+i,b)\right]^2.$$

From this expression we get, noting that $G(\overline{N}+i+j,\overline{b}) = G(2N+i+j-1,2b)$ is proportional to $G(N+i,b)\,G(N+j,b)$ and collecting common terms, an alternate expression for the approximation:

$$p_{\overline{M}}(x) = \sum_{i=0}^{\overline{M}}g_i\,G(\overline{N}+i,\overline{b}) \tag{F.4}$$

where $\overline{M} = 2M$, $\overline{N} = 2N-1$, $\overline{b} = 2b$, and the g_i's are again uniquely determined constants. That expression can be rewritten as

$$p_{\overline{M}}(x) = \sum_{i=0}^{\overline{M}}k_i\,\psi_i(\overline{N},\overline{b}\,;\,x).$$

The distance between $P(x)$ and its approximation $p_{\overline{M}}(x)$ is

$$D(P, p_{\overline{M}}) = \int_0^\infty [P(x) - p_{\overline{M}}(x)]^2 dx$$

from which we obtain, using the expressions $P(x) \approx Q(x)^2/\kappa$ and $p_{\overline{M}}(x) = q_M(x)^2/\kappa$,

$$D(P, p_{\overline{M}}) \approx \frac{1}{\kappa^2} \int_0^\infty [Q(x)^2 - q_M(x)^2]^2 dx.$$

Factorizing $Q(x)^2 - q_M(x)^2$ into $[Q(x)+q_M(x)]\,[Q(x)-q_M(x)]$, we have

$$D(P, p_{\overline{M}}) \approx \frac{1}{\kappa^2} \int_0^\infty [Q(x) - q_M(x)]^2 [Q(x) + q_M(x)]^2 dx,$$

which, upon application of the inequality

$$\int_0^\infty f(x) g(x) dx \le \int_0^\infty |f(x)|\, dx \int_0^\infty |g(x)|\, dx,$$

leads to

$$D(P, p_{\overline{M}}) \le \frac{1}{\kappa^2} \int_0^\infty [Q(x) - q_M(x)]^2 dx \int_0^\infty [Q(x) + q_M(x)]^2 dx.$$

Noting that $Q(x) \approx q_M(x)$, we obtain

$$D(P, p_{\overline{M}}) \le \frac{4}{\kappa^2} \int_0^\infty [Q(x) - q_M(x)]^2 dx \int_0^\infty Q(x)^2 dx.$$

The constant κ can be computed from (3.26) and is given by

$$\kappa \approx \int_0^\infty Q(x)^2 dx.$$

Therefore, we arrive at

$$D(P, p_{\overline{M}}) \le \frac{4}{\kappa} \int_0^\infty [Q(x) - q_M(x)]^2 dx.$$

Appendix 3.G: Expressions for $L_{\pm}^{(0)}$ for the close-to-gamma density model

Show that the expressions for $L_{+}^{(0)}$ and $L_{-}^{(0)}$ are

$$L_{+}^{(0)}(\beta,s) \approx U(N,M,b,s)\,\frac{b+c}{2\beta+s}, \quad \beta < 3b+s,$$

$$L_{-}^{(0)}(\beta,s) \approx U(N,M,b,s)\,\frac{b+c}{2\beta-s}, \quad \frac{s}{2} < \beta < 3b+2s,$$

where

$$U(N,M,b,s) = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} (2b)^j \sum_{m=0}^{M} h_m \sum_{i=0}^{m} \gamma_i^{(m)} (2b)^i \frac{\Gamma(2N+i+j)}{(N+i)(b+c)^{2N+i+j}}.$$

We start with (3.30) for $L_{+}^{(0)}$,

$$L_{+}^{(0)}(\beta,s) = \int_0^{\infty} Q(x) F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta+j\omega}\right\} dx. \tag{G.1}$$

Since, by the convolution theorem,

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta+j\omega}\right\} = Q_s(x) * F^{-1}\left\{\frac{1}{\beta+j\omega}\right\},$$

using (3.58) and noting that

$$F^{-1}\left\{\frac{1}{\beta+j\omega}\right\} = e^{\beta x} \quad \text{for } x < 0,$$

we get

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta+j\omega}\right\} = \int_x^{\infty} e^{\beta(x-z)} \left[\sum_{n=0}^{M} f_n \psi_n(N,b;z) e^{-sz}\right] dz \quad \text{for } x > 0.$$

Substituting (3.54) and (3.56) into the preceding expression, replacing $f_n (2b)^{N-1/2}[\Gamma(n+1)/\Gamma(2N+n-1)]^{1/2}$ with g_n, and interchanging the order of summation and integration, we obtain

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta + j\omega}\right\} = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} (2b)^j e^{\beta x} \int_x^{\infty} z^{N+j-1} e^{-(c+\beta)z}\, dz.$$

Replacing z with $y = (c+\beta)z$ as the dummy variable and noting that $\Gamma(N,x) \equiv \int_x^{\infty} y^{N-1} e^{-y}\, dy$, we get

$$F^{-1}\left\{\frac{\phi_s(\beta,\omega)}{\beta + j\omega}\right\} = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} (2b)^j e^{\beta x} \frac{\Gamma(N+j,(c+\beta)x)}{(c+\beta)^{N+j}}.$$

Substituting the preceding expression into (G.1) and replacing $d_m (2b)^{N-1/2}[\Gamma(m+1)/\Gamma(2N+m-1)]^{1/2}$ with h_m lead to

$$L_+^{(0)}(\beta,s) = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} \frac{(2b)^j}{(c+\beta)^{N+j}} \times$$

$$\int_0^{\infty} \left[\sum_{m=0}^{M} h_m \sum_{i=0}^{m} \gamma_i^{(m)} (2b)^i x^{N+i-1} e^{-bx} \right] e^{\beta x} \Gamma(N+j,(c+\beta)x)\, dx$$

from which we obtain, after interchanging the order of integration and summation,

$$L_+^{(0)}(\beta,s) = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} \frac{(2b)^j}{(c+\beta)^{N+j}} \sum_{m=0}^{M} h_m \sum_{i=0}^{m} \gamma_i^{(m)} (2b)^i \times$$

$$\int_0^{\infty} x^{N+i-1} e^{-(b-\beta)x} \Gamma(N+j,(c+\beta)x)\, dx.$$

From [Grads80, p. 663], we have

$$L_+^{(0)}(\beta,s) = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{(n)} (2b)^j \sum_{m=0}^{M} h_m \sum_{i=0}^{m} \gamma_i^{(m)} (2b)^i \frac{\Gamma(2N+i+j)}{(N+i)(b+c)^{2N+i+j}} \times$$

$${}_2F_1\left(1,\, 2N+i+j;\, N+i+1;\, \frac{b-\beta}{b+c}\right).$$

Since the indices i and j are less than M, we have, if $N \gg M$,

$${}_2F_1(1,\, 2N+i+j;\, N+i+1;\, z) \approx \frac{1}{1-2z}, \qquad |z| < \frac{1}{2}, \tag{G.2}$$

by following the approximation in (B.8). (A bound similar to the one in (B.8) cannot be

obtained since the validity of that bound depends on the transformation in (C.2), which cannot be applied to the preceding ${}_2F_1$.) Substituting the preceding formula into the expression for $L_{+}^{(0)}$ and letting

$$U(N,M,b,s) = \sum_{n=0}^{M} g_n \sum_{j=0}^{n} \gamma_j^{\{n\}} (2b)^j \sum_{m=0}^{M} h_m \sum_{i=0}^{m} \gamma_i^{\{m\}} (2b)^i \frac{\Gamma(2N+i+j)}{(N+i)(b+c)^{2N+i+j}},$$

we arrive at

$$L_{+}^{(0)}(\beta,s) \approx U(N,M,b,s) \frac{b+c}{2\beta+s}, \quad \beta < 3b + s. \tag{G.3}$$

Similarly, we find an expression for $L_{-}^{(0)}$:

$$L_{-}^{(0)}(\beta,s) \approx U(N,M,b,s) \frac{b+c}{2\beta-s}, \quad \frac{s}{2} < \beta < 3b + 2s. \tag{G.4}$$

CHAPTER 4

Estimation and Control of Simulated Annealing Parameters

As shown in Chapter 3, the efficient λ-schedule relates move generation to temperature decrement. A good move generation strategy will cooperate with the efficient λ-schedule to maximize the temperature decrements at every step. Since the efficient λ-schedule,

$$s_+ = s + \lambda \frac{4\rho_0(1-\rho_0)^2}{s^2(2-\rho_0)^2\sigma^3(s)},$$

depends on the acceptance ratio, ρ_0, and the variance of the energy, $\sigma^2(s)$, successful application of the schedule requires proper estimation of these parameters. This chapter is devoted to these estimation problems, which must be solved before the efficient λ-schedule can be applied, and to the method of move generation control, which strongly influences the performance of the efficient λ-schedule.

Section 4.1 covers the temperature control and parameter estimation problems associated with the use of the efficient λ-schedule. Several estimation procedures are discussed, and experimental results given. Section 4.2 presents a move generation control method. We first group optimization problems into two classes: those for which we know how to design "good" move generation strategies and those for which we do not. A procedure for controlling the move generation strategy is described for the former class. Finally, Section 4.3 gives a complete picture of the interaction among the efficient λ-schedule, the move generation strategy, and the rest of the system as well as practical issues associated with the use of the efficient λ-schedule.

4.1. Temperature control

One of the questions that must be raised when attempting to use the efficient λ-schedule is what are the values of ρ_0 and $\sigma(s)$ for a given optimization problem. In general these values are not known and must be estimated. Since the acceptance ratio varies slowly from one inverse temperature to another, the ratio of the number of accepted moves to the number of proposed moves in the last τ steps, with τ between 100 to 1,000, provides a good approximation for the current ρ_0. However, a similar procedure cannot be used for the variance of the energy, $\sigma^2(s)$. We know that if the inverse temperature is fixed and the system is allowed to run for a *long* time to reach equilibrium, the estimator

$$\hat{\sigma}^2(s) = \frac{1}{\tau}\sum_{i=1}^{\tau}(x_i - \mu(s))^2,$$

where x_i is the energy at step i and $\mu(s)$ is the expected energy, gives an unbiased estimate of $\sigma^2(s)$ provided that $\mu(s)$ is known. Unfortunately, neither the system is in equilibrium nor $\mu(s)$ is known. Since with the efficient λ-schedule the inverse temperature is updated at every step, we do not wait for the system to reach equilibrium. Therefore, the preceding formula alone cannot be used to estimate $\sigma^2(s)$. We need an estimator that would give the current value of $\sigma^2(s)$ using measurements made at previous values of the inverse temperature. Such an estimator can be constructed using the energy density model introduced in Chapter 3.

4.1.1. Estimators based on the energy density model

We describe two methods to construct the estimator for $\sigma(s)$. As a first try, let us construct an estimator based on the close-to-gamma density model. Given the energy density function, $P(x)$, we can compute the mean and variance of the energy, respectively, at different inverse temperatures using (1.17),

$$\mu(s) = -\frac{1}{Z(s)}\frac{\partial Z(s)}{\partial s}, \tag{4.1}$$

and (1.18),

$$\sigma^2(s) = -\frac{\partial \mu(s)}{\partial s}, \tag{4.2}$$

where

$$Z(s) = \int_{-\infty}^{\infty} P(x) e^{-sx}\, dx.$$

Carrying out such computations for the close-to-gamma density model, we obtain, after substituting (F.4) in Appendix 3.F of Chapter 3 for $P(x)$ into the preceding expression,

$$Z(s) = \int_{-\infty}^{\infty} \left[\sum_{i=0}^{\overline{M}} g_i\, G(\overline{N}+i, \overline{b}) \right] e^{-sx}\, dx.$$

Expanding $G(\overline{N}+i, \overline{b})$,

$$Z(s) = \int_{0}^{\infty} \left[\sum_{i=0}^{\overline{M}} g_i \frac{\overline{b}^{\overline{N}+i}}{\Gamma(\overline{N}+i)} x^{\overline{N}+i-1} e^{-\overline{b}x} \right] e^{-sx}\, dx,$$

letting $\overline{c} = \overline{b}+s$ and evaluating the integral, we have

$$Z(s) = \sum_{i=0}^{\overline{M}} g_i \frac{\overline{b}^{\overline{N}+i}}{\overline{c}^{\overline{N}+i}},$$

which can be substituted into the expression for $\mu(s)$ in (4.1) to obtain

$$\mu(s) = \frac{\sum_{i=0}^{\overline{M}} g_i (\overline{N}+i) \frac{\overline{b}^{\overline{N}+i}}{\overline{c}^{\overline{N}+i+1}}}{\sum_{j=0}^{\overline{M}} g_j \frac{\overline{b}^{\overline{N}+j}}{\overline{c}^{\overline{N}+j}}}.$$

Multiplying both the numerator and the denominator of the above expression by $\overline{c}^{\overline{M}+1}$, expanding powers of $\overline{c}$ into powers of s, and grouping terms according to their powers, we arrive at

$$\mu(s) = \frac{\sum_{i=0}^{\overline{M}} a_i s^i}{\sum_{j=0}^{\overline{M}+1} b_j s^j}, \tag{4.3}$$

where a_i and b_j are coefficients independent of s. (Since there are $2\overline{M}+3$ coefficients

and only $\overline{M} + 3$ independent variables, namely $\overline{N}$, $\overline{b}$, and g_i for $i = 0, 1, \ldots, \overline{M}$, the coefficients are not independent of each other for $\overline{M} \geq 1$.) If these coefficients can be found, the variance of the energy, $\sigma^2(s)$, can be computed using (4.2) to get

$$\sigma^2(s) = \frac{\sum_{i=0}^{\overline{M}} a_i s^i \sum_{j=1}^{\overline{M}+1} j b_j s^{j-1} - \sum_{i=1}^{\overline{M}} i a_i s^{i-1} \sum_{j=0}^{\overline{M}+1} b_j s^j}{(\sum_{j=0}^{\overline{M}+1} b_j s^j)^2}. \tag{4.4}$$

To estimate the coefficients a_i and b_j, we can use the method of least squares. Let us choose a τ for a given λ such that $\mu(s)$ for the last τ steps changes very little. Then, we can use

$$u_i = \frac{1}{\tau} \sum_{j=i-\tau+1}^{i} x_j \tag{4.5}$$

as the measured average energy of the system at step i. We want to find the coefficients such that the weighted square error on $\mu(s)$,

$$\sum_i w_i \left[u_i - \mu(s_i) \right]^2, \quad i = \tau, 2\tau, 3\tau, \ldots, \tag{4.6}$$

is minimized. Unfortunately, according to (4.3), $\mu(s)$ is a *nonlinear* function of the coefficients. Although there are methods, such as Levenberg-Marquard's [Press87, pp. 526-528], that converge to a minimum of the summation in (4.6), these methods are computationally intensive. Furthermore, they do not necessarily converge to the global minimum. What we need is a method that is fast and gives a reasonable estimate of $\sigma^2(s)$. That leads us to the second method.

We observe from Fig. 3.5 and Fig. 3.6 that only a one-term Laguerre series is needed to get an acceptable approximation of $Q_s(x)$ at each inverse temperature, provided that N and b are allowed to vary as the inverse temperature changes. Now, if we need a one-term Laguerre series, i.e., let $\overline{M} = 0$, so that (4.3) becomes

$$\mu(s) = \frac{a_0}{b_0 + b_1 s}, \tag{4.7}$$

and minimize the weighted square error on $1 / \mu(s)$,

$$\sum_i w_i \left[\frac{1}{u_i} - \frac{1}{\mu(s_i)} \right]^2, \quad i = \tau, 2\tau, 3\tau, \ldots, \tag{4.8}$$

instead of the weighted square error on $\mu(s)$, the least squares problem becomes *linear*. The difficulty with this approach is that $\sigma^2(s)$ can no longer be computed from the coefficients of $\mu(s)$ using (4.4). This is because by using the one-term Laguerre series, we approximate $\mu(s)$ locally. The coefficients, a_i and b_j, are no longer constants; they depend in an unknown way on the inverse temperature s. Since the derivatives of a_i and b_j with respect to s are not known, $\sigma^2(s)$ cannot be computed from the first derivative of $\mu(s)$ using (4.2). A way around this difficulty is to use (4.4) to compute $\sigma^2(s)$ with the coefficients for $\sigma^2(s)$ in (4.4) estimated rather than computed from the coefficients for $\mu(s)$ in (4.3).

Letting $\overline{M} = 0$ in (4.4), we obtain an expression for $\sigma^2(s)$ of the form

$$\sigma^2(s) = \left[\frac{f_0}{h_0 + h_1 s} \right]^2$$

and, consequently, an expression for $\sigma(s)$,

$$\sigma(s) = \frac{f_0}{h_0 + h_1 s}, \tag{4.9}$$

where f_0, h_0 and h_1 are slowly varying functions of s. This expression has the same form as the expression in (4.7). Therefore, finding an estimate for $\sigma(s)$ that minimizes the weighted square error on $1 / \sigma(s)$,

$$\sum_i w_i \left[\frac{1}{v_i} - \frac{1}{\sigma(s_i)} \right]^2, \quad i = \tau, 2\tau, 3\tau, \ldots, \tag{4.10}$$

is a linear least squares problem. The measured standard deviation of the energy, v_i, can be obtained using the following expression

$$v_i = \left[\frac{1}{\tau} \sum_{j=i-\tau+1}^{i} (x_j - \hat{\mu}(s_j))^2 \right]^{1/2}, \tag{4.11}$$

where $\hat{\mu}(s_j)$ is the estimate of $\mu(s_j)$. We are tempted to use u_j, the measured average energy, in place of $\hat{\mu}(s_j)$ in the preceding expression in order to avoid estimating $\mu(s)$.

However, using u_j in place of $\hat{\mu}(s_j)$ would cause an underestimation of v_i because the sequence of measurements for x_i is highly correlated. The estimator

$$\dot{v}^2{}_i = \frac{1}{\tau} \sum_{j=i-\tau+1}^{i} (x_j - u_j)^2$$

is biased and its expected value is given by [Ander71, p. 448]

$$E\left\{\dot{v}^2{}_i\right\} = \sigma^2(s_i) - \sigma^2(s_i)\frac{1}{\tau}\left\{1 + 2\sum_{j=1}^{\tau-1}\left[1 - \frac{j}{\tau}\right] r_j(s_i)\right\},$$

where $r_j(s_i)$ is the j^{th} autocorrelation coefficient of the energy at inverse temperature s_i. Since the sequence of measurements is highly correlated with $r_1(s_i)$ typically close to 0.99 and $r_n(s_i) \approx [r_1(s_i)]^n$, the resulting underestimation can be quite large. Therefore, we need to estimate $\mu(s)$ in order to get a better estimate of $\sigma(s)$.

After some straightforward manipulations, the resulting estimates for $\mu(s)$ and $\sigma(s)$ that minimize the weighted square errors in (4.8) and (4.10) are

$$\hat{\mu}(s) = \frac{1}{As + B}$$

and

$$\hat{\sigma}(s) = \frac{1}{Ds + E}$$

(where $A = b_1 / a_0$, $B = b_0 / a_0$, $D = h_1 / f_0$ and $E = h_0 / f_0$) with

$$A = \frac{\sum_{i=0}^{n} w_i \sum_{i=0}^{n} w_i \frac{s_i}{u_i} - \sum_{i=0}^{n} w_i s_i \sum_{i=0}^{n} w_i \frac{1}{u_i}}{\sum_{i=0}^{n} w_i \sum_{i=0}^{n} w_i s_i{}^2 - \sum_{i=0}^{n} w_i s_i \sum_{i=0}^{n} w_i s_i},$$

$$B = \frac{\sum_{i=0}^{n} w_i \frac{1}{u_i} - A \sum_{i=0}^{n} w_i s_i}{\sum_{i=0}^{n} w_i},$$

$$D = \frac{\sum_{i=0}^{n} w_i \sum_{i=0}^{n} w_i \frac{s_i}{v_i} - \sum_{i=0}^{n} w_i s_i \sum_{i=0}^{n} w_i \frac{1}{v_i}}{\sum_{i=0}^{n} w_i \sum_{i=0}^{n} w_i s_i^{\,2} - \sum_{i=0}^{n} w_i s_i \sum_{i=0}^{n} w_i s_i},$$

$$E = \frac{\sum_{i=0}^{n} w_i \frac{1}{v_i} - D \sum_{i=0}^{n} w_i s_i}{\sum_{i=0}^{n} w_i},$$

where $i = 0, \tau, 2\tau, \ldots, n$.

Since we use a one-term Laguerre series to approximate $\mu(s)$ and $\sigma(s)$ locally, it is important that the coefficients be able to adapt as the temperature changes. This is accomplished by allowing the weight factor w_i to increase as a function of time so that heavier weights are associated with the more recent measurements. We let $w_i = a^i$ for $\mu(s)$ and $w_i = b^i$ for $\sigma(s)$, where a and b are numbers greater than 1. Thus, the larger the values of a and b, the faster the estimators adapt, but also the larger the variance of the estimates becomes. The computational formulas for the efficient λ-schedule are summarized in Table 4.1 in Section 4.3. A description of how to use the formulas is also given in that section.

4.1.2. Test of the estimators

There are three parameters related to temperature control that need to be set in Table 4.1: λ, which controls the trade-off between the quality of the final solution and the computation time, and the weight factors a and b, which control how fast the parameters can vary. The weight factors a and b are computed using $a^\tau = L_a / (L_a - \tau)$, and $b^\tau = L_b / (L_b - \tau)$, where L_a and L_b are the "memory lengths" [Cowan85, Chapter 3] of the estimators for the average and the standard deviation of the energy, respectively. The settings of the memory lengths, L_a and L_b, influence the accuracy of the estimators. Since it is difficult to determine accurately $\sigma(s)$ for any optimization problem, we tested the estimators by applying the efficient λ-schedule in Table 4.1 to a highly correlated sequence of random numbers with probability density

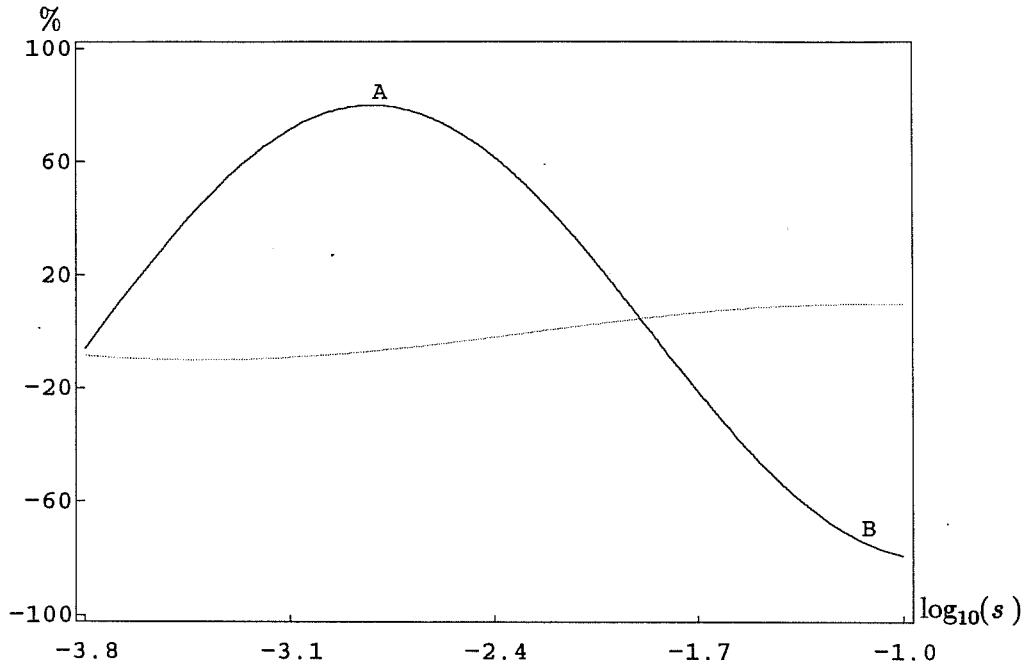

Figure 4.1: Percentage variations of N (solid curve) and b (dotted curve) from N_0 and b_0.

function given by the gamma density function

$$G_x(N,c) = \frac{c^N}{\Gamma(N)} x^{N-1} e^{-cx}, \quad x > 0, \tag{4.12}$$

where the parameters N and $c = b + s$ are temperature dependent. These parameters were varied as a function of the inverse temperature according to the formulas

$$N = N_0[1 + k_N \sin(0.67\pi \log(s))]$$

$$b = b_0[1 + k_b \sin(0.57\pi \log(s))]$$

with $N_0 = 100$, $b_0 = 0.0003$, $k_N = 0.8$, and $k_b = 0.1$, which lead to variations, a half sinusoid on the log scale (from point A to point B in Fig. 4.1), similar to those observed in Fig. 3.5 for the 100-city traveling salesman problem. The percentage variations of N and b from N_0 and b_0, i.e.,

$$\frac{N - N_0}{N_0} \quad \text{and} \quad \frac{b - b_0}{b_0},$$

as a function of the inverse temperature are shown in Fig. 4.1 while the percentage variations of $\mu(s)$ and $\sigma(s)$ from $\mu_0(s)$ and $\sigma_0(s)$, i.e.,

$$\frac{\mu(s) - \mu_0(s)}{\mu_0(s)} \quad \text{and} \quad \frac{\sigma(s) - \sigma_0(s)}{\sigma_0(s)},$$

where $\mu(s) = N / (b+s)$, $\sigma(s) = \sqrt{N} / (b+s)$, $\mu_0(s) = N_0 / (b_0+s)$ and $\sigma_0(s) = \sqrt{N_0} / (b_0+s)$, are shown in Fig. 4.2. By comparing the estimated $\sigma(s)$ with

$$\sigma(s) = \frac{\sqrt{N}}{b + s}$$

computed from the gamma density function, we can assess the accuracy of the estimators and their tracking capability. Note that the variation of $\mu(s)$ and $\sigma(s)$ depends on two sources: the change in s assuming that the gamma density model is perfect (constant N and b) and the variation of N and b as a function of s. Only the variation due to the latter source is shown in Fig. 4.2.

Before proceeding to test the estimators, we first describe how to generate a highly correlated sequence of random numbers with a gamma distribution. Let X be a random variable whose probability density function is given by the gamma density function, $G_x(N,c)$, in (4.12). We know that X can be formulated as the sum of N independent random variables,

$$X = \sum_{i=1}^{N} Y_i,$$

whose probability density function is given by the exponential density function,

$$p(y) = ce^{-cy}, \quad y > 0.$$

Unlike random numbers with gamma distribution, exponentially distributed random numbers are easy to generate. They can be obtained using the formula

$$y_i = -\frac{1}{c}\log(\xi_i),$$

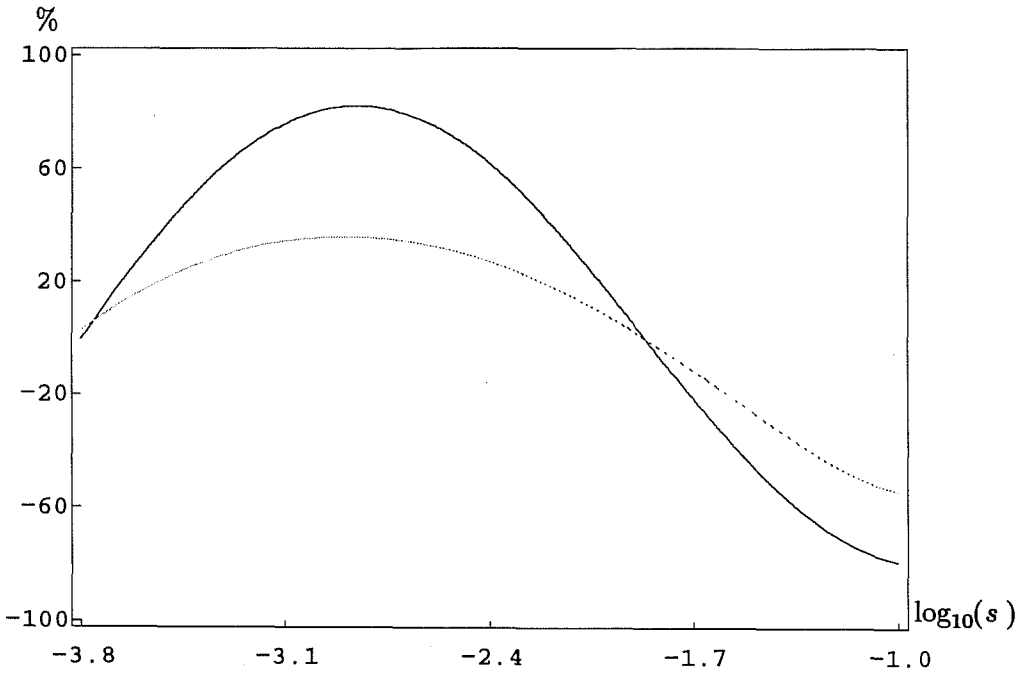

Figure 4.2: Percentage variations of $\mu(s)$ (solid curve) and $\sigma(s)$ (dotted curve) from $\mu_0(s)$ and $\sigma_0(s)$.

where ξ_i is a random number uniformly distributed between 0 and 1. To obtain a random number with a probability density function given by $G_x(N,c)$, we just add up N independent exponentially distributed random numbers,

$$x = -\frac{1}{c}\sum_{i=1}^{N}\log(\xi_i).$$

To generate a sequence of random numbers with autocorrelation coefficients $r_n = r^n$ for $n = 1, 2,...,$ we randomly and independently select and replace $(1-r)N$ out of the N possible $\log(\xi_i)$'s to generate a new random number from the current one.

The estimators have been tested by applying the formulas in Table 4.1 to the forementioned sequence of highly correlated random numbers. The experimental results serve two purposes: to assess the accuracy of the estimators and to guide the selection of the memory lengths, L_a and L_b. Throughout the experiments, r for the random sequence was set at 0.99 while the sample size, τ, was set at 1,000. Since the efficient λ-schedule depends on the measured acceptance ratio, $\hat{\rho}_n$, and the random numbers are

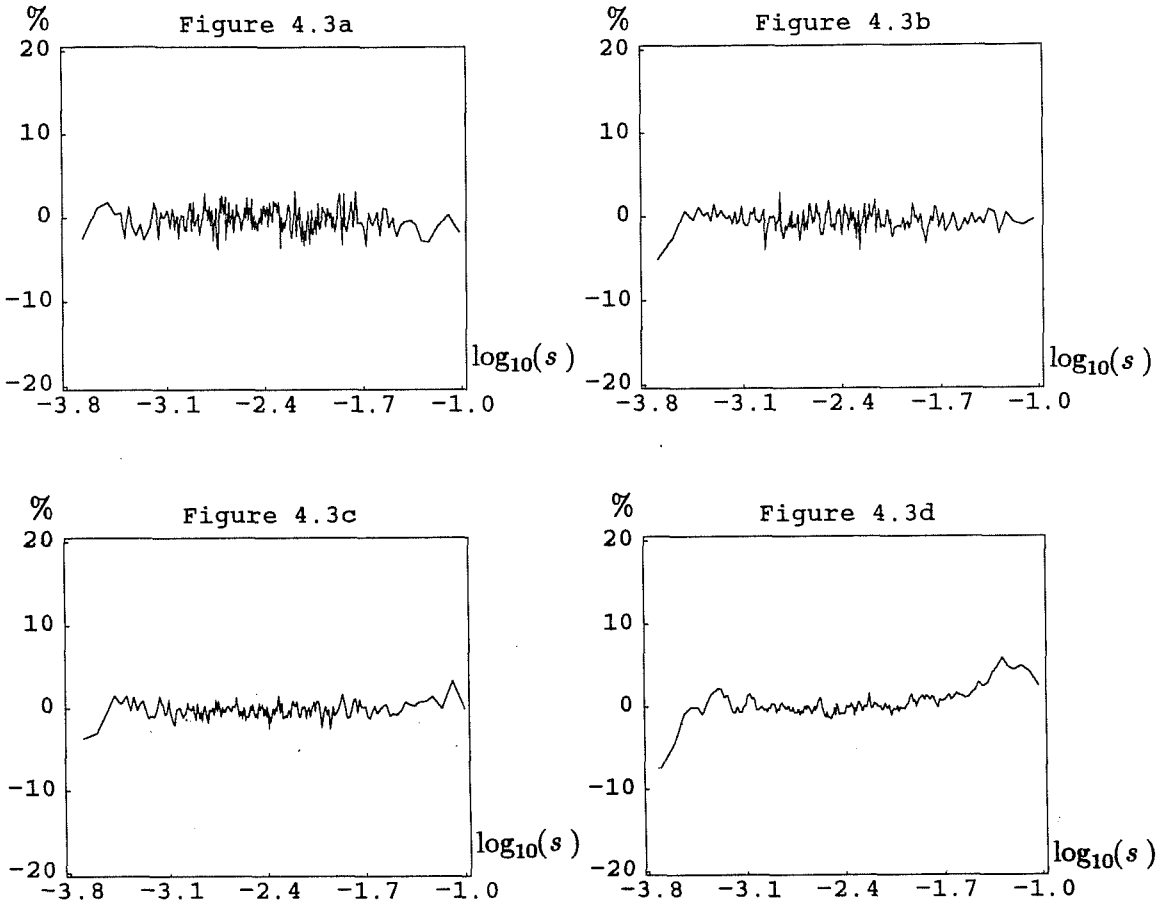

Figure 4.3: Percentage error on the estimation of $\mu(s)$ for different values of L_a. Figure 4.3a: L_a = 5,000. Figure 4.3b: L_a = 10,000. Figure 4.3c: L_a = 20,000. Figure 4.3d: L_a = 40,000.

taken as energy values bypassing the discrimination by the Metropolis criterion, we assume that $\hat{\rho}_n = 0.44$ in the tests.

In the first group of experiments, we study the effect of L_a on the accuracy of the estimation of $\mu(s)$. The percentage error for the estimation of $\mu(s)$ computed using the formula

$$\frac{\hat{\mu}(s) - \mu(s)}{\mu(s)}$$

is depicted in Fig. 4.3 for different values of L_a. As expected, the memory length L_a

realizes the trade-off between the variance and the bias of the estimate; a smaller value of L_a leads to an estimate with a smaller variance, but with a larger bias, while a larger value of L_a leads to an estimate with a larger variance, but with a smaller bias. Among the four values of L_a, the results indicate that L_a = 20,000 gives a slightly better compromise between the bias and the variance of the estimate. However, in all cases the estimation error is small.

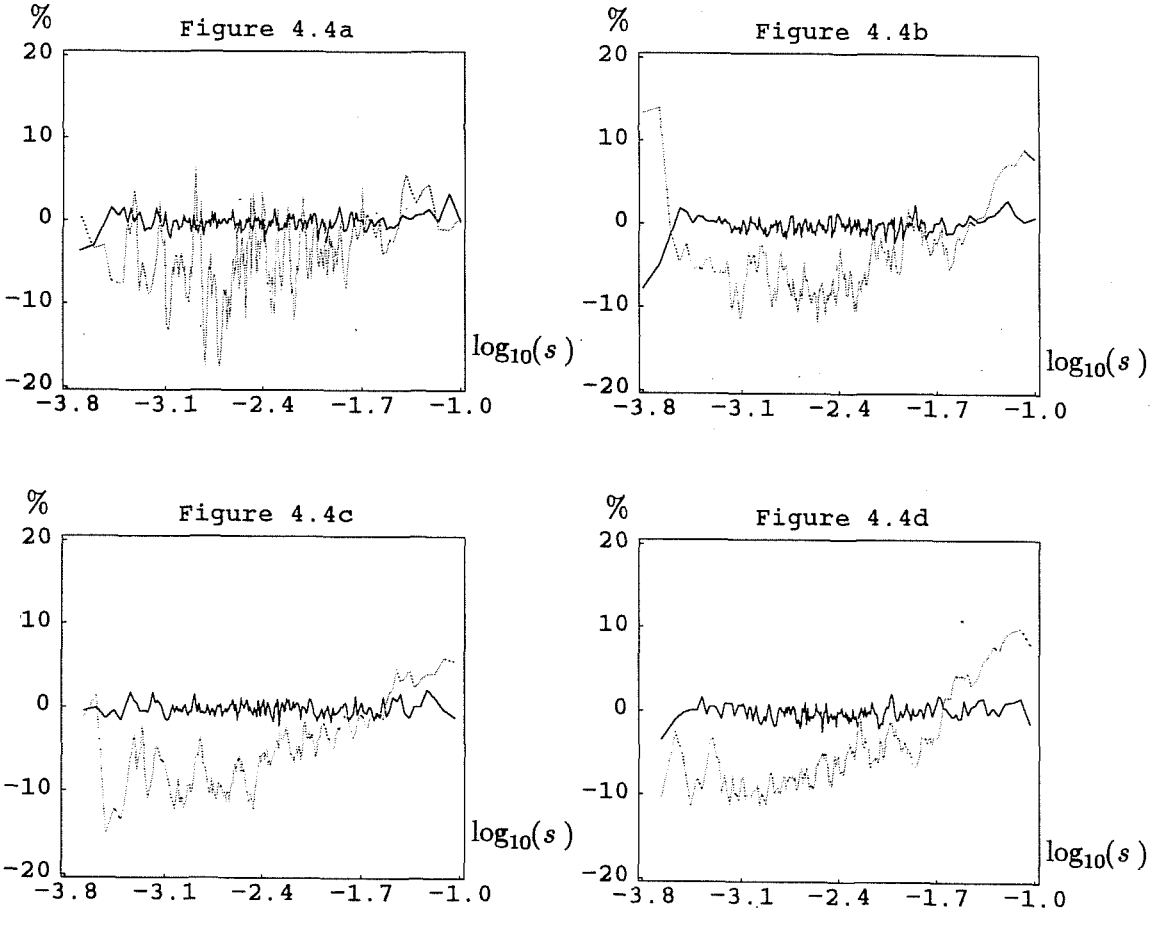

Figure 4.4: Percentage error on the estimation of $\sigma(s)$ for different values of L_b. The solid curves represent the estimation errors on $\mu(s)$ while the dotted curves represent the estimation errors on $\sigma(s)$. Figure 4.4a: L_b = 40,000. Figure 4.4b: L_b = 100,000. Figure 4.4c: L_b = 140,000. Figure 4.4d: L_b = 200,000.

In the second group of experiments, we study the effect of L_b on the accuracy of the estimation of $\sigma(s)$. Since the measured standard deviation depends on $\hat{\mu}(s)$, (4.11), the accuracy of $\hat{\sigma}(s)$ depends strongly on the accuracy of $\hat{\mu}(s)$. The best value of L_a, $L_a = 20{,}000$, observed in the first set of experiments was used in this group of experiments. The percentage errors in the estimates of $\sigma(s)$, computed using the formula

$$\frac{\hat{\sigma}(s) - \sigma(s)}{\sigma(s)},$$

and of $\mu(s)$ are depicted in Fig. 4.4 for different values of L_b. We observe that $L_b = 200{,}000$ gives the best result, a 10% error on the estimation of $\sigma(s)$. Since $\sigma^3(s)$ rather than $\sigma(s)$ is used in the efficient λ-schedule, a 10% error on $\sigma(s)$ would mean a 30% error on $\sigma^3(s)$. This estimation error on $\sigma^3(s)$ can be viewed as variation in λ with an exact estimate of $\sigma^3(s)$. Thus, although λ is a constant, because of the estimation error on $\sigma^3(s)$, λ can have, in effect, a 30% variation. This may explain the test results in Table 2.1 where only small differences in the quality of the final solutions are observed between a constant λ and λ's with up to 30% variation.

There is room for improvement for the current set of estimators. However, it seems difficult to obtain significantly better estimators while maintaining the same level of computational complexity. Since the main reason for an efficient annealing schedule is to minimize the computation time of simulated annealing, we are forced to use estimators that are computationally efficient. A more accurate set of estimators would be less desirable if it increases the computation time significantly.

4.2. Move generation control

Now that we know how to estimate $\sigma(s)$ and, therefore, to control the inverse temperature using the formulas in Table 4.1, we are going to tackle the problem of move generation control. The goal of this section is to communicate ideas on how to design good move generation strategies and how to control them. These ideas seem to work well in practice, especially with the efficient λ-schedule, but they are by no means the

only possible approach.

To propose a move in simulated annealing is to perturb the current solution to the optimization problem to obtain a new one. This perturbation is highly problem specific; different optimization problems are likely to have different methods for obtaining new solutions. For example, in a traveling salesman problem one may want to modify sections of a tour to obtain a new tour, while in the standard cell placement problem one may want to move a cell from one position to another. (Refer to Chapter 5 for a description of these optimization problems.) In order to apply the move generation control method described in this section, we classify two kinds of move generation strategies: *controllable* move generation strategies and *non-controllable* move generation strategies. A controllable move generation strategy is one for which there exists a parameter θ that increases with the *average magnitude of the proposed energy change.* For example, in our implementation of the N-city traveling salesman problem, each city maintains a list of the neighboring cities sorted in ascending order according to their distances from that city. A move is proposed by picking two cities A and B, and modifying the tour as shown in Fig. 4.5. City A is always picked randomly with equal probability from the N cities while city B is picked from the sorted list of neighbors of city A. Since we believe that the *average magnitude of the proposed energy change increases with the distance between cities A and B*, we choose the index value of the

Figure 4.5: How a tour is modified in the traveling salesman problem.

sorted list from which city B is picked as the parameter θ. Thus, the bigger is the index, the further apart is city B from city A , and, most likely, the larger is the average magnitude of the proposed energy change.

By increasing (or decreasing) θ, we increase (or decrease) the average magnitude of the proposed energy change and, therefore, decrease (or increase) the acceptance ratio. Note that by controlling the acceptance ratio, we are controlling, in effect, β since there is a one to one relationship between β and the acceptance ratio, i.e.,

$$\rho_0 = \frac{2\beta - s}{2\beta}.$$

Controllable move generation strategies are in general preferable to non-controllable move generation strategies, for they decrease the execution time of simulated annealing while maintaining comparable results. However, for some optimization problems, it is difficult to find a controllable move generation strategy. In that case, we will have to settle with no control on the average magnitude of the proposed energy change and, hence, the acceptance ratio.

Once a controllable move generation strategy has been selected, the problem of move generation control becomes the problem of θ control. Since the goal of move generation control is to achieve a desired acceptance ratio of 0.44 (see Section 3.2.1.2 of Chapter 3), we would like to control θ in order to achieve that goal. Let the parameter θ be picked according to the formula

$$\theta = -\ \overline{\theta}\log(\xi),$$

where ξ is a random number uniformly distributed between 0 and 1. (In the event that θ exceeds its maximum allowable value, $\theta_{\max}$, θ will be set to $\xi * \theta_{\max}$.) Using this formula, the probability that a given θ will be picked is specified by the exponential density function with an average value of $\overline{\theta}$,

$$p(\theta) = \frac{1}{\overline{\theta}} e^{-\frac{\theta}{\overline{\theta}}}. \tag{4.13}$$

This function is chosen because of its simplicity and the fact that any positive θ has a

non-zero probability of getting picked.

Since the desired acceptance ratio is 0.44, we would like to keep that acceptance ratio for as long as possible during the course of simulated annealing. Comparison of the acceptance ratio measured for the last τ steps,

$$\hat{\rho}_n = \frac{\textit{number of accepted moves in the last } \tau \textit{ steps}}{\tau}, \tag{4.14}$$

with the desired acceptance ratio indicates how $\overline{\theta}$ should be modified. If $\hat{\rho}_n$ is less than the desired value, we decrease the value of $\overline{\theta}$ and, hence, the average magnitude of the proposed energy change, which would lead to an increase in the acceptance ratio. Likewise, if $\hat{\rho}_n$ is greater than the desired value, we increase the value of $\overline{\theta}$. (In light of the estimation procedure for $\sigma(s)$, we have also experimented with a more elaborate scheme for $\hat{\rho}_n$ in which the current $\hat{\rho}_n$ is the weighted average of the last few $\hat{\rho}_n$'s. However, we did not observe any significant difference between the final results of the two methods.)

A simple negative feedback controller for $\overline{\theta}$ in order to achieve the desired acceptance ratio of 0.44 is the proportional control

$$\overline{\theta}_n = \overline{\theta}_{n-\tau} + K(\hat{\rho}_n - 0.44),$$

where the positive gain K, which is problem as well as implementation dependent, controls the rate at which $\overline{\theta}$ could be changed. Since $\overline{\theta}$ cannot be negative because of the exponential density function in (4.13), we require that

$$\overline{\theta}_n = \max(\overline{\theta}_{n-\tau} + K(\hat{\rho}_n - 0.44), \overline{\theta}_{\min}),$$

where $\overline{\theta}_{\min}$ is a positive constant. Care must be taken in choosing the gain K. Too large a gain makes $\hat{\rho}_n$ oscillate around 0.44 while too small a gain makes $\hat{\rho}_n$ lag behind 0.44. The proper gain either has to be found experimentally for each optimization problem or has to be estimated on the fly using adaptive control methods [Landa79].

The move generation control method described in this section has been implemented on the traveling salesman problem, the graph partition problem and the standard cell placement problem. Refer to Chapter 5 for details.

4.3. Application procedure and practical issues

We have discussed the temperature control and move generation control problems separately. Now, we give a complete picture of the interaction among these controllers and the rest of the system as well as practical issues associated with the use of the efficient λ-schedule. To use the schedule in Table 4.1, we first initialize the inverse temperature, s_0, to 0 and $\overline{\theta}_n$ to its maximum value. The system is allowed to run at this inverse temperature until it is sufficiently randomized and initial measurements of the average and the standard deviation of the energy can be computed using the formulas

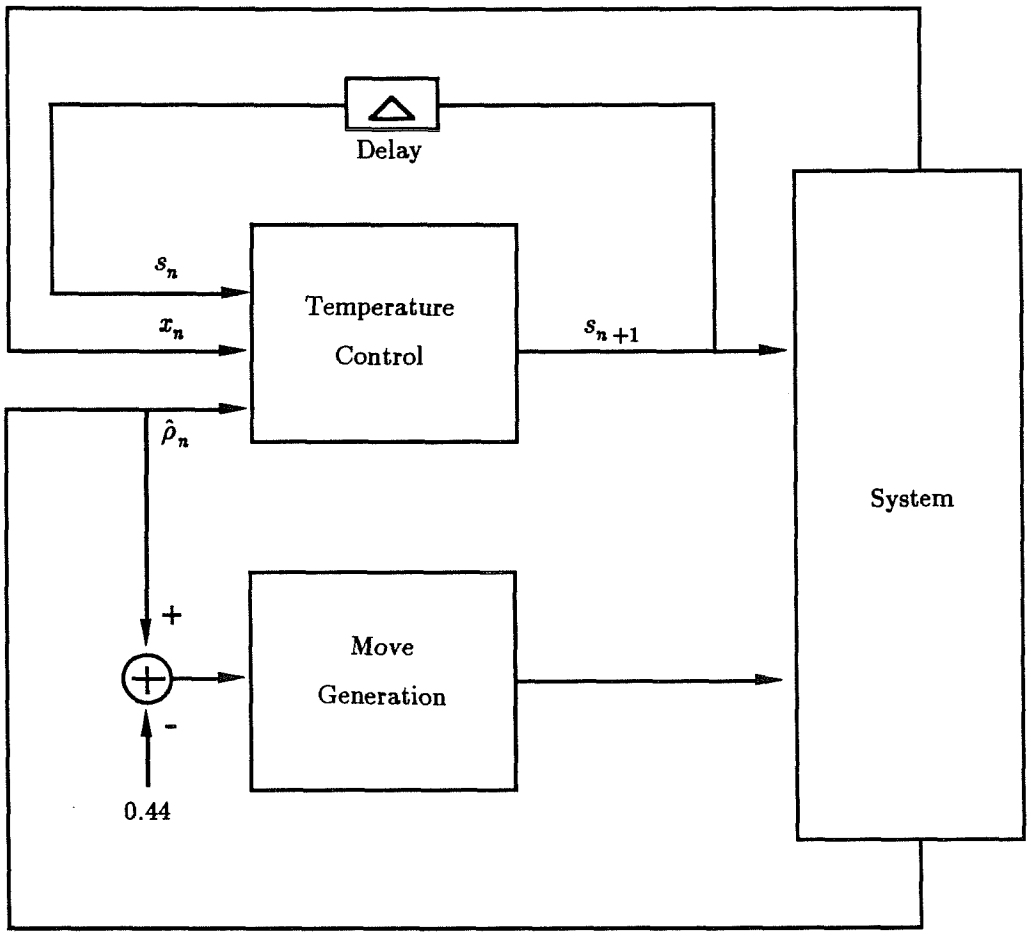

Figure 4.6: A block diagram that shows the interaction among temperature control, move generation control, and the rest of the system.

$$u_0 = \frac{1}{M}\sum_{i=1}^{M} x_i$$

and

$$v_0 = \left[\frac{1}{M}\sum_{i=1}^{M}(x_i - u_0)^2\right]^{1/2},$$

where M is the number of steps executed at s_0, and x_i is the energy of the system at step i. These measurements are used to compute the initial values of the parameters A, B, D, and E using the parameter initialization formulas in Table 4.1. Then, after every step, the new inverse temperature is computed using the temperature updating formulas while after every τ steps, the parameters A, B, D, and E are adjusted using the parameter updating formulas, $\hat{\rho}_n$ is re-computed using the measurement formula, and $\overline{\theta}_t$ is updated using the move generation control formula. This process is repeated until the measured average energy, u_i, remains unchanged for the last $k\tau$ steps. At this point, the system is considered frozen and annealing is stopped. We have also experimented with the more elaborate frozen detection schemes proposed by Aarts and van Laarhoven in [Aarts85] and Huang *et al.* in [Huang86]. Although these two schemes are logically sound, they do not lead to a significant difference in performance. We, therefore, use the simpler frozen detection scheme. Note that the initial value of $\hat{\rho}_n$ is equal to 1 since all proposed moves are accepted when $s_0 = 0$. A block diagram that shows the interaction among temperature control, move generation control, and the rest of the system is depicted in Fig. 4.6, while an implementation of the efficient λ-schedule in the C programming language is listed in Appendix 4.A.

The inverse temperature $s_1 = 1 \,/\, (2\sigma(0))$ in Table 4.1 is chosen such that it is high enough to ensure a smooth transition from infinite temperature to finite temperature. Since the efficient λ-schedule is valid for $s \geq 7 \,/\, \sigma(0)$, (3.66), the system undergoes sub-optimal cooling within the range

$$\frac{1}{2\sigma(0)} \leq s \leq \frac{7}{\sigma(0)}.$$

This is tolerable since the system spends only a very small percentage of its time within

Parameter initialization formulas

$$A = \frac{v_0^2}{u_0^2} \qquad D = \frac{v_0}{u_0}$$

$$B = \frac{1}{u_0} - As_0 \qquad E = \frac{1}{v_0} - Ds_0$$

Temperature updating formulas $n = 1, 2, 3, \ldots$

$$s_1 = \frac{1}{2\hat{\sigma}(s_0)} \qquad s_{n+1} = s_n + \lambda \frac{4\hat{\rho}_n (1 - \hat{\rho}_n)^2}{s_n^2 (2 - \hat{\rho}_n)^2 \hat{\sigma}^3(s_n)}$$

$$\hat{\mu}(s_n) = \frac{1}{As_n + B} \qquad \hat{\sigma}(s_n) = \frac{1}{Ds_n + E}$$

Parameter updating formulas $i = 0, \tau, 2\tau, \ldots$

$$A = \frac{f(1)f(\frac{s}{u}) - f(s)f(\frac{1}{u})}{f(1)f(s^2) - f(s)f(s)} \qquad D = \frac{g(1)g(\frac{s}{v}) - g(s)g(\frac{1}{v})}{g(1)g(s^2) - g(s)g(s)}$$

$$B = \frac{f(\frac{1}{u}) - Af(s)}{f(1)} \qquad E = \frac{g(\frac{1}{v}) - Dg(s)}{g(1)}$$

$$f(z) = \sum_{i=0}^{n} z_i a^i \qquad g(z) = \sum_{i=0}^{n} z_i b^i$$

Measurement formulas $i = t-\tau+1, t-\tau+2, \ldots, n$

$$u_n = \frac{1}{\tau} \sum_{i=n-\tau+1}^{n} x_i \qquad v_n = \left(\frac{1}{\tau} \sum_{i=n-\tau+1}^{n} [x_i - \hat{\mu}(s_i)]^2 \right)^{1/2}$$

$$\hat{\rho}_n = \frac{\textit{number of accepted moves in the last } \tau \textit{ steps}}{\tau}$$

Move generation control formula

$$\overline{\theta}_n = \max(\overline{\theta}_{n-\tau} + K(\hat{\rho}_n - 0.44), \overline{\theta}_{\min})$$

Table 4.1: Computational formulas for the efficient λ-schedule.

this range.

As noted in Section 4.1.2, the factors a^τ and b^τ in the parameter updating formulas are computed using $a^\tau = L_a \ / \ (L_a - \tau)$, and $b^\tau = L_b \ / \ (L_b - \tau)$, where L_a and L_b are the memory lengths of the estimators. Smaller memory lengths give larger weight factors which, in turn, make the system forget its history faster. Finite memory lengths are desirable because our model is imperfect. By selecting L_a different from L_b, we allow

the model parameters to vary at different speeds as annealing proceeds.

Occasionally, instead of starting at zero inverse temperature, we may want to start at a higher inverse temperature. Let us take the standard cell placement problem as an example. If we only modify a few cells of a 3,000 cell problem, we may want to use the placement obtained from a previous annealing execution and start annealing at a higher inverse temperature so that most of the original placement remains intact. This is accomplished easily by setting s_0 to any desired inverse temperature instead of 0. Alternatively, we may want to specify the initial acceptance ratio rather than the initial inverse temperature and compute the inverse temperature for that ratio as described in Section 1.2.2 using formula (1.14). Yet another way is to measure the effective inverse temperature of the current solution and use that as the initial inverse temperature as described in [Rose88].

As noted in Chapter 2, the user specified parameter λ realizes the trade-off between the quality of the final solution and the computation time. Ideally, we would like to have a procedure that computes the proper λ for either a given quality of final solution or an allotted computation time. Unfortunately, such a procedure cannot be found analytically. Since λ controls the quasi-stationarity criterion at different inverse temperatures, its effect on the quality of the final solution and the computation time is indirect. Nevertheless, we observe the following empirical relation in the standard cell placement problem:

$$M = \frac{aN^b}{\lambda^c}, \tag{4.15}$$

where N is the number of standard cells in the placement problem, and M is the total number of proposed moves. The constants a, b, and c for the standard cell placement problem are estimated to be 247, 0.7, and 1, respectively. Using equation (4.15) to predict the total number of proposed moves, we observe an average error of 15%. A similar relation has also been observed in the traveling salesman problem. Therefore, we suggest the following method to find the constants. First, experiment with different values of λ and N and measure the resulting total number of proposed moves in each

case. Then, find the constants by solving the linear least squares problem in $\log a$, b and c

$$\log M = \log a + b \log N - c \log \lambda.$$

With the empirical relation in (4.15), we can compute the proper λ for a given total number of proposed moves and, with a little extra effort, for an allotted computation time.

Of all practical concerns, speed is one of the most important. We observe from Table 4.1 that the efficient λ-schedule uses floating point computations heavily. To reduce the number of floating point operations, we approximate the efficient λ-schedule by computing the new inverse temperature after every ten steps instead of one,

$$s_{n+10} = s_n + 10\lambda \frac{4\rho_0(1-\rho_0)^2}{s_n{}^2(2-\rho_0)^2\sigma^3(s_n)}.$$

This speeds up the efficient λ-schedule by a few percent without degrading the quality of the final solutions.

We also pre-compute the values of the exponential function, which are needed in deciding whether a proposed move should be accepted or rejected, as in [Johns87, pp. 39-40], and the values of the logarithm function, which are used to generate exponentially distributed random numbers. These values are stored in tables and looked up when needed. Since the evaluation of the exponential and the logarithm functions occurs quite often, a significant portion of computation time is typically saved.

Now that we know how to use the efficient λ-schedule, it is time to assess its performance. In Chapter 5, we shall assess the performance of the efficient λ-schedule by comparing it with other annealing schedules available in the literature. Furthermore, we shall also assess the performance of simulated annealing as a general combinatorial optimization method by comparing it with other problem specific heuristics on some combinatorial optimization problems.

References

[Aarts85] E. Aarts, and P. van Laarhoven, "Statistical Cooling Algorithm: A General Approach to Combinatorial Optimization Problems," *Philips Journal of Research,* Vol. 40, No. 4, 193-226, 1985.

[Ander71] T. Anderson, *The Statistical Analysis of Time Series,* John Wiley and Sons Inc., 1971.

[Cowan85] C. Cowan, and P. Grant, *Adaptive Filters,* Prentice-Hall, 1985.

[Huang86] M. Huang, F. Romeo, and A. Sangiovanni-Vincentelli, "An Efficient General Cooling Schedule for Simulated Annealing," *Proceedings of the IEEE International Conference on Computer-Aided Design,* 381-384, 1986.

[Johns87] D. Johnson, C. Aragon, L. McGeoch, and C. Schevon, "Optimization by Simulated Annealing: an Experimental Evaluation (Part I)," *Draft,* 1987.

[Landa79] I. Landau, *Adaptive Control: The Model Reference Approach,* Dekker, New York, 1979.

[Press87] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes,* Press Syndicate of the University of Cambridge, 1987.

[Rose88] J. Rose, W. Klebsch, and J. Wolf, "Equilibrium Detection and Temperature Measurement of Simulated Annealing Placement," *Proceedings of the International Workshop on Placement and Routing,* 1988.

Appendix 4.A: A sample simulated annealing program

```
/*
*   FILE: sample.c
*       An implementation of the efficient simulated annealing schedule.
*
*       The following problem specific procedures need to be written:
*       1. double initial_move():
*               initialize the data structure and return the current ENERGY
*               of the system.
*       2. double make_move():
*               propose a new solution and return the ENERGY CHANGE.
*       3. accept_move():
*               make the proposed solution the current solution.
*       4. final_move():
*               nothing is required in this procedure.
*       5. update_control(acc_ratio):
*               update move generation control.
*
*       The constants M, LAMBDA_MEM_LENGTH_U, and LAMBDA_MEM_LENGTH_V are
*       tuned for the traveling salesman problem only.  They need to be
*       modified for other problems.
*
*       Lambda needs to be set.
*/

#include <stdio.h>
#include <math.h>

#define RANDFACT (1.0 / 0x7fffffff)
#define dice() (rand() * RANDFACT)

#define SKIP_TIME 10        /* change S every 10 iterations instead of 1 */
#define FROZEN_NO 5
#define TAU 100

/* the following three parameters are problem dependent */
#define M 1000      /* initial # of moves, depends on problem */
#define LAMBDA_MEM_LENGTH_U 600.0       /* lambda memory length product */
#define LAMBDA_MEM_LENGTH_V 30000.0     /* lambda memory length product */

/* the following five problem specific procedures need to be written. */
double    initial_move();
double    make_move();
int       accept_move();
int       final_move();
int       update_control();

/* lambda needs to be set. */
double    lambda = 0.01;
double    energy, mean, vari, estimate_mean = 0.0, estimate_sd;
double    S, dS, alpha, acc_ratio;
int       skip = -1;

double    w_b, vsyy, vsxy, vsxx, vsx, vsy, vsum, D, E;
double    w_a, usyy, usxy, usxx, usx, usy, usum, A, B;

main()
{
        /* need to set lambda */
        simulated_annealing();
}
```

```
simulated_annealing()
{
        initialize();
        loop();
}

/*
 * initialization: randomize the system and collect statistics to initialize
 * the parameters.
 */
initialize()
{
        int     i;

        energy = initial_move();
        mean = vari = 0.0;
        for (i = 0; i < M; i++) {
                energy += make_move();
                mean += energy;
                vari += energy * energy;
                accept_move();
        }
        mean /= (double) M;
        vari = vari / ((double) M) - mean * mean;
        acc_ratio = 1.0;
        initialize_parameter();
}

/*
 * repeat until the system is considered frozen.
 */
loop()
{
        int     i, success;
        double  energy_change, d;

        while (1) {
                mean = vari = 0.0;
                success = 0;
                for (i = 0; i < TAU; i++) {
                        energy_change = make_move();
                        if ((energy_change <= 0.0) || exp(-S * energy_change) >
                            dice()) {
                                energy += energy_change;
                                accept_move();
                                success ++;
                        }
                        mean += energy;
                        d = energy - estimate_mean;
                        vari += d * d;
#ifdef SKIP_TIME
                        if (--skip <= 0)
#endif
                                update_S();
                }
                mean /= (double) TAU;
                vari /= (double) TAU;
                acc_ratio = ((double) success) / ((double) TAU);
                if (frozen()) {
                        final_move();
                        return;
                }
                update_parameter();
                update_control(acc_ratio);
```

```
        }
}

/*
 * initialize parameters A,B,D, and E, and the estimate_mean and estimate_sd.
 * initial inverse temperature is set to 0, and acceptance ratio to 1.
 */
initialize_parameter()
{
        double    d;

        estimate_sd = sqrt(vari);
        estimate_mean = mean;
        A = estimate_sd * estimate_sd / (estimate_mean * estimate_mean);
        B = 1.0 / estimate_mean;
        D = estimate_sd / estimate_mean;
        E = 1.0 / estimate_sd;
        usum = vsum = 1.0;
        usxy = usxx = usx = 0.0;
        usy = 1.0 / estimate_mean;
        usyy = usy * usy;
        vsxy = vsxx = vsx = 0.0;
        vsy = 1.0 / estimate_sd;
        vsyy = vsy * vsy;
        S = 0.0;
        dS = 0.5 / estimate_sd;
        d = (1.0 - acc_ratio) / (2.0 - acc_ratio);
        alpha = 4.0 * acc_ratio * d * d;
        initialize_weights();
}

/*
 * update parameters A,B,D, and E, and the estimate_mean and estimate_sd
 * for the current S value.
 */
update_parameter()
{
        register double      d;

        d = 1.0 / mean;
        usyy *= w_a;
        usxy *= w_a;
        usy *= w_a;
        usx *= w_a;
        usxx *= w_a;
        usum *= w_a;
        usyy += d * d;
        usxy += S * d;
        usy += d;
        usx += S;
        usxx += S * S;
        usum += 1.0;
        A = (usum * usxy - usx * usy) / (usum * usxx - usx * usx);
        B = (usy - A * usx) / usum;
        estimate_mean = 1.0 / (A * S + B);
        if (vari > 0.0) {
                d = 1.0 / sqrt(vari);
                vsyy *= w_b;
                vsxy *= w_b;
                vsy *= w_b;
                vsx *= w_b;
                vsxx *= w_b;
                vsum *= w_b;
                vsyy += d * d;
```

```
                vsxy += S * d;
                vsy += d;
                vsx += S;
                vsxx += S * S;
                vsum += 1.0;
                D = (vsum * vsxy - vsx * vsy) / (vsum * vsxx - vsx * vsx);
                E = (vsy - D * vsx) / vsum;
        }
        estimate_sd = 1.0 / (D * S + E);
        d = (1.0 - acc_ratio) / (2.0 - acc_ratio);
        alpha = 4.0 * acc_ratio * d * d;
}

/*
 * update inverse temperature S at every step or every 10 steps depending on
 * SKIP_TIME being defined or not.
 */
update_S()
{
        register double     d;

        S += dS;
        estimate_mean = 1.0 / (A * S + B);
        estimate_sd = 1.0 / (D * S + E);
        d = S * estimate_sd;
        dS = lambda * alpha / (d * d * estimate_sd);
#ifdef SKIP_TIME
        dS *= SKIP_TIME;
        skip = SKIP_TIME;
#endif
}

/*
 * if mean remains unchanged for FROZEN_NO of samples, with TAU moves in a
 * sample, the system is considered frozen.
 */
int     frozen()
{
        static double       old_mean = 0.0;
        static int  counter = 0;

        if (mean == old_mean) {
                counter ++;
        } else {
                counter = 0;
                old_mean = mean;
        }
        return(counter >= FROZEN_NO);
}

/*
 * initialize weights a and b.
 * these weights are computed from the lambda memory length product.
 */
initialize_weights()
{
        w_a = LAMBDA_MEM_LENGTH_U / lambda;
        w_a = 1.0 - TAU / w_a;
        if (w_a < 0.0) w_a = 0.0;
        w_b = LAMBDA_MEM_LENGTH_V / lambda;
        w_b = 1.0 - TAU / w_b;
        if (w_b < 0.0) w_b = 0.0;
}
```

CHAPTER 5

Performance of the Efficient Simulated Annealing Schedule

The figure of merit for any algorithm is its performance which, in our case, is the quality (energy) of the final solution. The goals of this chapter are to assess the performance of the efficient λ-schedule, when compared with annealing schedules available in the literature, and to assess the performance of simulated annealing as a general method for solving combinatorial optimization problems.

Since the introduction of simulated annealing by Kirkpatrick *et al.* [Kirkp82] and, independently, by Cerny [Cerny85], the construction of an efficient annealing schedule has been of great concern to proponents of simulated annealing. At first, most practical annealing schedules were designed to work on specific problems and constructed based on experimental results; not until recently did annealing schedules of wide applicability appear in the literature. Among these general annealing schedules, Aarts' schedule [Aarts85] and Huang's schedule [Huang86] stand out as schedules reported to give good average results, while the logarithmic schedule derived independently in [Hajek85], [Mitra85] and others was shown to give solutions that converge to the global optima. (Refer to Chapter 1 for a discussion of these annealing schedules.) To assess how well the efficient λ-schedule performs—where it stands among the competition, we compare it with the best of the three annealing schedules, selected on the basis of preliminary results on the traveling salesman problem. The test cases for the comparison cover three well-known problems: the traveling salesman problem, the graph partition problem and the standard cell placement problem.

Most of the practical applications of simulated annealing have been in complicated problem domains, where other algorithms either did not exist or performed poorly. To

assess the performance of simulated annealing as a general method for solving combinatorial optimization problems, we have to compare the method with efficient heuristics on well-studied problems. The traveling salesman and the graph partition problems are chosen for this reason because they have been studied extensively for two decades resulting in a number of well-known heuristics. Among these heuristics, we choose to compete with simulated annealing the ones by Lin and Kernighan [Lin73], and by Karp [Karp77] for the traveling salesman problem, and the one by Fiduccia and Mattheyses [Fiduc82] for the graph partition problem.

This chapter is divided into three sections with each section devoted to one optimization problem. The implementation details and the test results of this problem as well as other practical issues are discussed. Only specifics related to the optimization problem are mentioned since the application procedure for the efficient λ-schedule was described in Section 4.3. The traveling salesman problem is covered first, followed by the graph partition problem and the standard cell placement problem. The last problem is the most interesting because one of the motivations for our study of simulated annealing is to speed up the method significantly so that it can be applied, with acceptable amount of computation time, to optimization problems in computer-aided design of VLSI integrated circuits. As reported by Sechen and Sangiovanni-Vincentelli [Seche85], the simulated annealing based standard cell and gate array placement program TimberWolf, when compared with other industrial programs, achieved up to 57% reduction in final chip area at the expense of computation time: a 2,700 standard cell problem required 84 CPU hours on a VAX780! Our study of simulated annealing, which leads to the efficient λ-schedule, is one step in the direction of cutting down this massive computation time requirement. The traveling salesman and graph partition problems, although less commonly associated with computer-aided design of VLSI integrated circuits, also have applications in that area. For example, power and ground routing in PI, a placement and interconnect system [Ullma84], is formulated as a traveling salesman problem while min-cut placement [Dunlo85] and logic partitioning [Green86] are formulated as graph partition problems.

The test results given in this chapter were measured on a Sun 3/280-s8 with MC68881 floating point option. Unless stated otherwise, they are average results of *at least eight* executions, with the number of executions chosen to minimize the standard deviations of the measurements to an acceptable level. All programs were implemented in the C programming language. Refer to Appendix 4.A for the implementation of the simulated annealing method.

5.1. Traveling salesman problem

One of the classical NP-complete combinatorial optimization problems is the traveling salesman problem. The goal of this problem is to minimize the length of a tour starting from any city, visiting each of the N cities once and only once, and returning to the original place of departure. We restrict our attention to problems where the cities lie in the plane, and the distances between cities are symmetric and Euclidean. We first cover the implementation details, followed by a discussion of the test results.

5.1.1. Implementation details

The cost to be minimized in a N-city traveling salesman problem is the length of a tour. In this implementation, each city maintains a list of up to M neighboring cities

Figure 5.1: How a tour is modified in the traveling salesman problem.

sorted in ascending order according to their distances from that city. A move is proposed by first picking two cities A and B, then modifying the tour as shown in Fig. 5.1. City A is always picked randomly with equal probability from the N cities while city B is picked using one of the two following methods. In the first method called *standard control*, city B is picked randomly with equal probability from the first θ cities on the neighbor list of city A. The parameter θ, which roughly controls the maximum allowable distance between A and B, is lowered as a function of the inverse temperature, s, according to the empirical formula

$$\theta = 0.5N \left[\log_{10}(10 + \frac{1}{s\sqrt{N}}) - 1 \right]^2.$$

This method implements the widely held idea that the acceptance ratio should be allowed to drop almost linearly as a function of the total number of proposed moves [Seche85] [Huang86]. In the second method called *feedback control*, city B is the θ^{th} entry on the neighbor list of city A with

$$\theta = -\overline{\theta}\log(\xi),$$

where ξ is a random number uniformly distributed between 0 and 1, and $\overline{\theta}$ is a control parameter between 2 and N. Because ξ is uniformly distributed, θ will be exponentially distributed,

$$p(\theta) = \frac{1}{\overline{\theta}} e^{-\frac{\theta}{\overline{\theta}}},$$

with an average value of $\overline{\theta}$. If the value of θ exceeds the value of M, a condition that happens frequently when $\overline{\theta}$ is large, θ will be set to ξN. This is because for large values of $\overline{\theta}$, $p(\theta)$ is essentially uniform. Consequently, θ should be set to ξN. The control parameter $\overline{\theta}$ is adjusted after every τ moves according to the formula

$$\overline{\theta}_n = \max(\overline{\theta}_{n-\tau} + 100(\hat{\rho}_n - 0.44), 2),$$

where $\hat{\rho}_n$ is the measured acceptance ratio for the last τ moves. If $\hat{\rho}_n$ is less than 0.44, the value of $\overline{\theta}$ is lowered. If $\hat{\rho}_n$ is greater than 0.44, the value of $\overline{\theta}$ is raised. This updating procedure enables us to keep the acceptance ratio close to the *desired*

acceptance ratio of 44%.

Care must be taken in limiting the value of $\overline{\theta}$. We are tempted to let $\overline{\theta} \to 0$ as $s \to \infty$ so that the acceptance ratio can be kept at 44% for a longer period of time. However, if the value of $\overline{\theta}$ is too close to 0, the value of θ will be restricted to a small set. This restriction is undesirable since the number of possible moves is so small that simulated annealing may terminate prematurely.

For the sake of efficiency, the distances between any two cities are pre-computed and stored in a table. Their values are looked up when needed. This $O(N^2)$ storage requirement is usually not a problem for a small number of cities, but prohibitive for a large number of cities. For problems in which the cities amount to thousands, the table lookup scheme cannot be used; we have to compute the distance every time we need it.

The settings of the λ-schedule parameters in Table 4.1 for the traveling salesman problem are $\lambda L_a = 600$, $\lambda L_b = 30{,}000$, $\tau = 100$, and $k = 5$. Ideally, M, the number of cities in the neighbor list, should increase as N increases if we can afford the $O(N^2)$ storage requirement. But for $N = 10{,}000$, this $O(N^2)$ storage requirement is too high. We, therefore, set the value of M to the minimum of $N - 1$ and 250.

Huang's annealing schedule was implemented as in [Huang86]. Due to the difference in move generation controls, a few maximum generation limits have been tried. We settled on a value of $N\theta/2$, corresponding to the number of possible moves when standard control was used.

Size	Tour Length (10^4)			Total Moves (10^5)		
	Huang	Aarts	Log.	Huang	Aarts	Log.
100	7.99	8.51	8.28	2.08	2.29	2.02
200	11.45	12.12	11.60	4.07	4.09	4.02
300	13.12	14.20	13.40	8.33	7.92	8.02
400	15.24	15.99	15.41	14.20	14.10	14.00

Table 5.1: Test results of the three annealing schedules on the traveling salesman problem with 100 to 400 uniformly distributed cities. Feedback control is used with all three annealing schedules.

5.1.2. Test results

To assess the performance of the efficient λ-schedule, we need to compare it with the best general annealing schedule available in the literature. This best schedule is selected from the logarithmic schedule, the schedules by Huang *et al.*, and by Aarts and van Laarhoven, based on a preliminary experiment on the traveling salesman problem with 100 to 400 uniformly distributed cities—the coordinates of such a city are a pair of random numbers uniformly distributed in the interval of 0 to 10,000. To keep the competition fair, we picked one instance of the traveling salesman problem for each problem size and tested the different annealing schedules and heuristics on that same instance. From the test results shown in Table 5.1, we observe that Huang's annealing schedule performs significantly better than the others; we, therefore, select Huang's annealing schedule to compete with the efficient λ-schedule.

We compare the efficient λ-schedule and Huang's annealing schedule on the traveling salesman problem with 100 to 400 uniformly distributed cities. The quality of the solutions from both schedules is kept within 0.2% so that the speedup can be measured. The experimental results are displayed in Table 5.2 in which δ, representing the average quality of the final solution, is the percentage of the average cost above the estimated best cost. This best cost was found by carrying out a sequence of careful annealing executions with the run-time doubled after every few executions until the average cost was stabilized. Then the best cost in the sequence was used as the estimated best cost. A typical sequence for a 300-city traveling salesman problem using the efficient λ-schedule coupled with feedback control is depicted in Fig. 5.2. The

Size	Speedup			
	$\delta = 3.6\%$	$\delta = 2.9\%$	$\delta = 2.2\%$	$\delta = 1.5\%$
100	2.09	3.39	6.00	8.35
200	2.54	4.09	7.50	10.61
300	2.90	4.38	10.86	24.20
400	3.37	7.77	17.61	>21.00

Table 5.2: Comparison between the efficient λ-schedule with feedback control and Huang's annealing schedule with standard control. The cities are uniformly distributed.

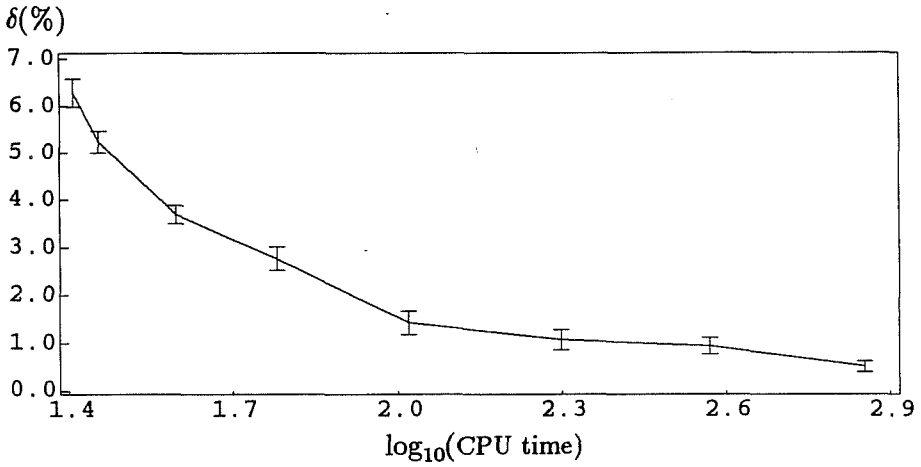

Figure 5.2: A sequence of careful annealing executions with the run-time doubled after every few executions for a 300-city traveling salesman problem. The average cost, in percent above the estimated best cost, versus the logarithm of the CPU time is shown. The estimated standard deviations of the average cost are also depicted as vertical bars in the figure (with two standard deviations in a vertical bar).

estimated standard deviations of the average cost are also depicted as vertical bars in this figure (with two standard deviations in a vertical bar).

The speedups associated with a particular δ as the number of cities increases are shown along each column in Table 5.2 while the speedups associated with a particular problem size as the cost improves are shown along each row. The speedup is defined as the ratio of the CPU time used by Huang's annealing schedule with standard control to the CPU time used by the efficient λ-schedule with feedback control. Although the efficient λ-schedule is 10% slower per move than Huang's, we still observe a speedup of

Size	Speedup			
	$\delta = 3.6\%$	$\delta = 2.9\%$	$\delta = 2.2\%$	$\delta = 1.5\%$
100	1.74	3.27	3.43	3.87
200	1.37	1.75	2.23	2.76
300	1.94	1.76	3.00	5.00
400	1.37	1.87	2.82	4.90

Table 5.3: An experiment with move generation control. Comparison between the efficient λ-schedule with feedback control and the same schedule with standard control. The cities are uniformly distributed.

up to 24 for the 300-city traveling salesman problem!

To isolate the effect of feedback control from the efficient λ-schedule, we experimented on both annealing schedules with both controls. The speedups, computed as the CPU time spent by the efficient λ-schedule with standard control over the CPU time spent by the same schedule with feedback control, are displayed in Table 5.3, while the speedups, computed as the CPU time spent by Huang's annealing schedule over the CPU time spent by the efficient λ-schedule when feedback control was employed in both cases, are displayed in Table 5.4. We observe that the use of feedback control speeds up the efficient λ-schedule by a factor of up to 5. However, even when the same control method is used with both annealing schedules, the efficient λ-schedule still out-performs Huang's by a factor of up to 3.

We also compare simulated annealing, using the efficient λ-schedule and feedback control, with other heuristics for the traveling salesman problem. We chose the heuristics by Lin and Kernighan [Lin73], and by Karp [Karp77] as the competitors. The comparison with Lin and Kernighan's heuristic covers four types of city distributions: uniformly distributed cities, super-clustered cities, cities on a lattice, and cities on a lattice with small perturbation. In the case of uniformly distributed cities, an instance for each problem size was selected randomly as the test case. Examples of city distributions are shown in Fig. 5.3 while some optimal solutions are shown in Fig. 5.4. All results tabulated are the average of *at least eight* executions. In the case of Lin and Kernighan's heuristic, each execution consists of *three* iterations and within each

Size	Speedup			
	$\delta = 3.6\%$	$\delta = 2.9\%$	$\delta = 2.2\%$	$\delta = 1.5\%$
100	1.95	1.99	2.04	2.21
200	2.19	2.02	2.06	2.26
300	2.55	2.26	2.36	2.70
400	2.55	2.27	2.80	3.01

Table 5.4: Comparison between the efficient λ-schedule and Huang's annealing schedule when feedback control was used in both cases. The cities are uniformly distributed.

Figure 5.3a

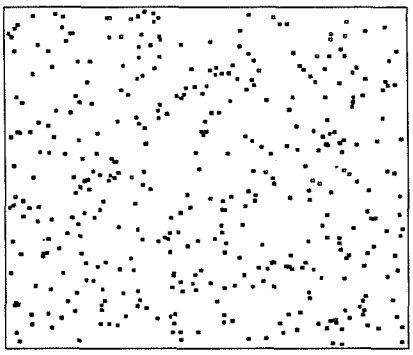

Figure 5.3b

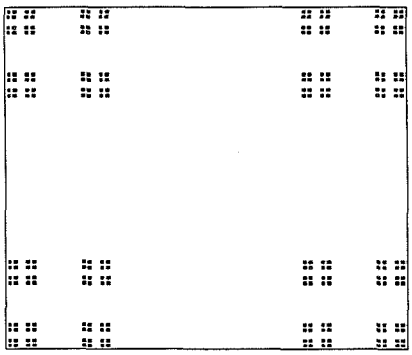

Figure 5.3c

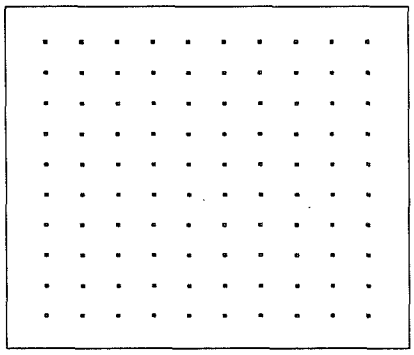

Figure 5.3d

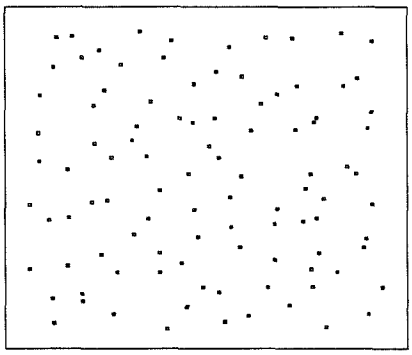

Figure 5.3: Examples of city distributions for the traveling salesman problem. Figure 5.3a: 400 uniformly distributed cities. Figure 5.3b: 256 super-clustered cities (clusters of four). Figure 5.3c: 100 cities on a lattice. Figure 5.3d: 100 cities on a lattice with perturbation (up to 50%).

iteration, the local optima found in the preceding iterations are used to speed up the search at the current iteration.

The speedups, defined as the ratio of the CPU time used by Lin and Kernighan's heuristic, t', to the CPU time used by simulated annealing, t, are shown in Table 5.5 to Table 5.8. In Table 5.5 and Table 5.8, where the optimal costs are not known, δ' and δ are the percentages above the estimated best costs respectively for Lin and Kernighan's heuristic and for simulated annealing while in Table 5.6 and Table 5.7, where the optimal costs are known, δ' and δ are the percentages above the optimal costs. To give

an idea of the typical number of moves proposed by simulated annealing, we note that the result in Table 5.5 for a 100-city traveling salesman problem was obtained by executing an average of 2.3×10^5 moves in 52.9 CPU seconds.

From these tables we observe that Lin and Kernighan's heuristic performs very poorly on super-clustered cities and does a very good job on cities on a lattice. This comes as no surprise since Lin and Kernighan's heuristic produces 3-optimal solutions—an n-optimal solution is a local optimum with respect to moves that exchange n sections of a tour instead of only two sections as in Fig. 5.1. In the case of super-clustered cities, the number of locally optimal solutions that are 2-optimal is large. Therefore, Lin and Kernighan's heuristic spends lots of time going from one 2-optimal solution to another until it reaches a 3-optimal solution. In the case of cities on a lattice, the number of globally optimal solutions is large: any tour formed by connecting cities either vertically or horizontally is a globally optimal solution. Consequently, the probability of finding one of these optimal solutions is high and the heuristic terminates quickly.

To check that the number of globally optimal solutions influences the speed of Lin and Kernighan's heuristic, we perturb the coordinates of the cities on a lattice by up to

Figure 5.4a

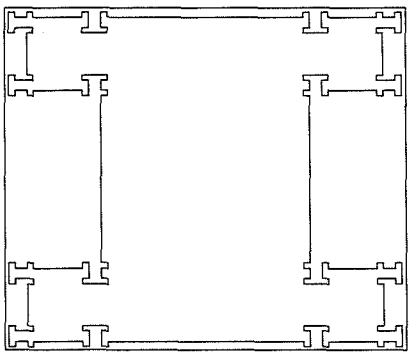

Figure 5.4b

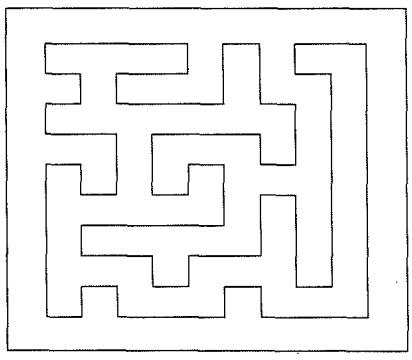

Figure 5.4: Optimal solutions for some traveling salesman problems. Figure 5.4a: Optimal solution for 256 super-clustered cities. Figure 5.4b: Optimal solution for 100 cities on a lattice.

Size	CPU time (sec.)			Tour length	
	t'	t	t' / t	$\delta'(\%)$	$\delta(\%)$
100	69.8	52.9	1.32	1.01	1.01
200	296.0	101.6	2.91	1.25	1.21
300	686.1	197.5	3.47	1.32	1.32
400	1492.7	330.8	4.51	1.46	1.40

Table 5.5: Comparison with Lin and Kernighan's heuristic on the traveling salesman problem for uniformly distributed cities.

Size	CPU time (sec.)			Tour length	
	t'	t	t' / t	$\delta'(\%)$	$\delta(\%)$
64	57.9	9.3	6.23	0.58	0.58
128	428.2	17.4	24.61	1.13	1.06
256	2113.6	61.5	34.37	2.52	2.58
512	13883.1	296.3	46.85	4.11	4.03

Table 5.6: Comparison with Lin and Kernighan's heuristic on the traveling salesman problem for super-clustered cities.

Size	CPU time (sec.)			Tour length	
	t'	t	t' / t	$\delta'(\%)$	$\delta(\%)$
100	13.0	78.2	0.17	0.20	0.31
200	70.7	492.0	0.14	0.15	0.67
300	190.3	1338.4	0.14	0.10	1.09
400	415.6	3125.5	0.13	0.25	1.09

Table 5.7: Comparison with Lin and Kernighan's heuristic on the traveling salesman problem for cities on a lattice.

P percent of the normal inter-neighbor distance so that the number of globally optimal solutions is small. The new coordinates of a city (x', y') are computed from the old coordinates (x, y) using the following formulas:

$$x' = x + \xi M_P$$

$$y' = y + \xi M_P,$$

where M_P is $P\%$ of the inter-neighbor distance and ξ is a uniformly distributed random number between -1 and 1. The test results for a 200-city problem with different perturbations are shown in Table 5.8. We observe that the CPU time spent by Lin and Kernighan's heuristics increases from 71 seconds to 181 seconds with only a 5%

perturbation, but does not increase further with larger perturbations. This observation confirms our conjecture that the abundance of globally optimal solutions speeds up Lin and Kernighan's heuristic. We also experimented with different numbers of cities on a lattice with up to 50% perturbation. As expected, the speedups shown in Table 5.9 fall in between those for the uniformly distributed cities and the cities on a lattice.

Contrary to the sensitivity of Lin and Kernighan's heuristic, simulated annealing is relatively insensitive to the structure and regularity of an optimization problem; the CPU time spent by simulated annealing for different city distributions stays fairly constant for a given quality of solution. This combination of the insensitivity of simulated annealing towards the structure of a problem and the sensitivity of Lin and Kernighan's heuristic towards the number of global optima and the number of 2-optimal solutions explains the drastic speedup (up to 46) for problems with super-clustered cities as well as the superiority of Lin and Kernighan's heuristic over simulated annealing on problems with cities on a lattice.

P	CPU time (sec.)			Tour length	
	t'	t	t' / t	$\delta'(\%)$	$\delta(\%)$
0%	70.7	492.0	0.14	0.15	0.67
5%	180.8	491.1	0.37	0.21	1.79
10%	179.2	491.1	0.36	0.33	2.27
20%	154.6	297.2	0.52	0.67	0.72
50%	182.3	205.6	0.89	0.73	0.70

Table 5.8: Comparison with Lin and Kernighan's heuristic on a 200-city traveling salesman problem for cities on a lattice with up to P percent perturbation.

Size	CPU time (sec.)			Tour length	
	t'	t	t' / t	$\delta'(\%)$	$\delta(\%)$
100	32.7	104.9	0.31	0.70	0.70
200	182.3	205.6	0.89	0.73	0.70
300	510.6	270.8	1.89	1.42	1.38
400	978.8	417.5	2.34	1.43	1.42

Table 5.9: Comparison with Lin and Kernighan's heuristic on the traveling salesman problem for cities on a lattice with up to 50% perturbation.

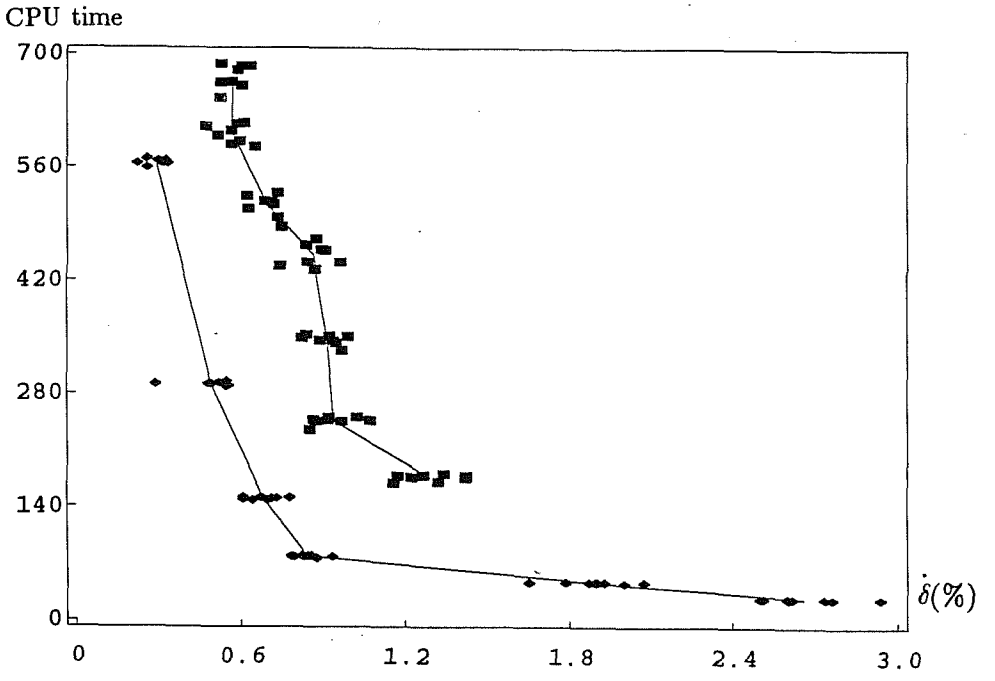

Figure 5.5: Test results for eight instances of 200-city traveling salesman problems. The cities are uniformly distributed on a square. The rectangles represent the $\dot{\delta}$'s for Lin and Kernighan's heuristic while the diamonds represent the $\dot{\delta}$'s for simulated annealing.

As mentioned earlier, only one instance for each problem size was chosen for uniformly distributed cities to obtain the test results shown in Table 5.5. In order to check that similar speedups can be expected for any random instance, we carried out the following experiment. We first randomly picked eight instances of a 200-city traveling salesman problem. Then, Lin and Kernighan's heuristic was used to solve each instance once, resulting in a total of eight executions. Let us refer to these eight executions as a group. Since the CPU time spent by Lin and Kernighan's heuristic is controlled by the number of iterations, we can vary the number of iterations to obtain different groups. Similarly, we can obtain different groups of executions for these eight instances using simulated annealing by controlling the value of λ and, hence, the CPU time. Let $\delta_{i,j}$ be the percentage above the estimated best cost for instance i in group j, and $\dot{\delta}_{i,j}$ be the average among all instances in group j except instance i,

$$\dot{\delta}_{i,j} = \frac{1}{M-1} \sum_{\forall k : k \neq i} \delta_{k,j}, \tag{5.1}$$

where $M = 8$ is the number of instances. In the next two figures, we display $\dot{\delta}$'s rather than δ's since the $\dot{\delta}$'s form tighter clusters than the δ's and, therefore, give a better indication of the trend. It is easy to verify that the average of $\dot{\delta}_{i,j}$ in group j is equal to the average of $\delta_{i,j}$ in group j, that is,

$$\mu_j = \frac{1}{M} \sum_{\forall k} \dot{\delta}_{k,j} = \frac{1}{M} \sum_{\forall k} \delta_{k,j},$$

and the standard deviation of $\dot{\delta}_{i,j}$, $\dot{\sigma}_j$, is equal to $1 / (M-1)$ of the standard deviation of $\delta_{i,j}$, σ_j, where

$$\dot{\sigma}_j = \left[\frac{1}{M(M-1)} \sum_{\forall k} (\dot{\delta}_{k,j} - \mu_j)^2 \right]^{1/2}$$

and

$$\sigma_j = \left[\frac{1}{M(M-1)} \sum_{\forall k} (\delta_{k,j} - \mu_j)^2 \right]^{1/2}.$$

A plot of the CPU time spent (in seconds) versus $\dot{\delta}$ is shown in Fig. 5.5, where the number of iterations for Lin and Kernighan's heuristic varies from 2 to 9 and the value of λ for simulated annealing varies from 0.2 to 0.00625. The group averages, μ_j, are connected and displayed as two solid curves. (The method of grouping data using (5.1) is employed in the bias estimation technique called jackknife [Efron82].) A similar experiment has also been conducted on a slightly different city distribution. Instead of uniformly distributed on a square with sides of length $2R$, the coordinates of the cities in the experiment are restricted to a disc with a diameter of $2R$. The test results are depicted in Fig. 5.6. From these two figures, we conclude that similar speedups to those shown in Table 5.5 can be expected for other instances of the traveling salesman problem with uniformly distributed cities.

Except for the case of cities on a lattice, simulated annealing offers a substantial speedup over Lin and Kernighan's heuristic for high quality solutions. This observation

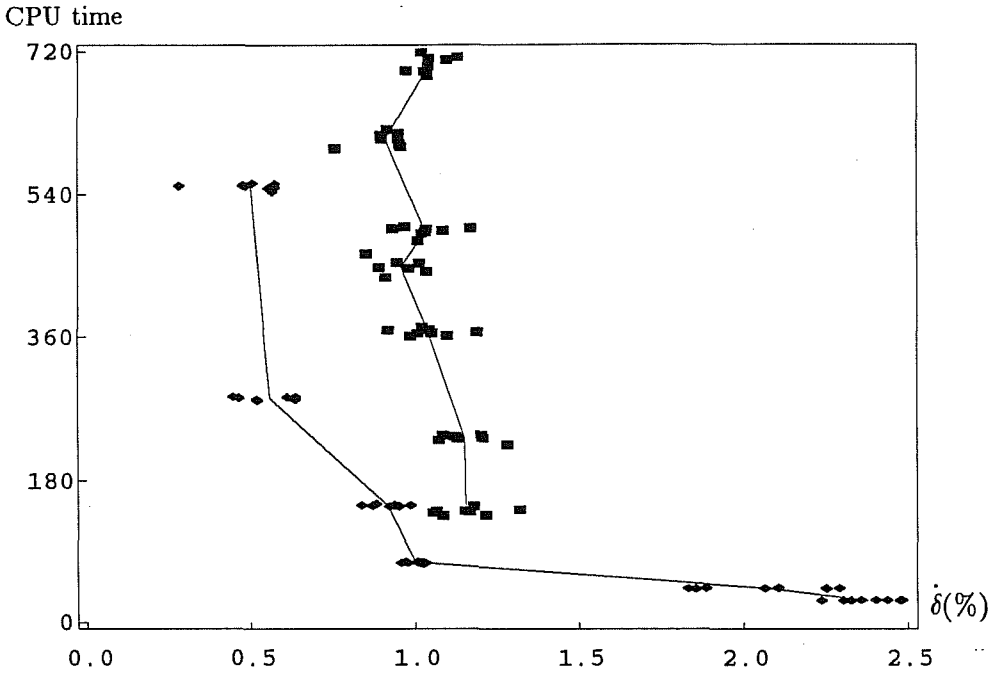

Figure 5.6: Test results for eight instances of 200-city traveling salesman problems. The cities are uniformly distributed on a disc. The rectangles represent the δ's for Lin and Kernighan's heuristic while the diamonds represent the δ's for simulated annealing.

indicates that for problems where globally optimal solutions are sparse and high quality solutions are desired, simulated annealing coupled with a good move generation strategy is an efficient and highly competitive method.

Lin and Kernighan's heuristic is only good for small problems since its computation time becomes unacceptable for large problems. Karp designed a heuristic that handles large problems. The comparison with Karp's heuristic was performed on a traveling salesman problem with 10,000 cities whose coordinates are integers uniformly distributed in the interval of 0 to 1,000,000. Karp's heuristic consists of three stages: partition a problem into M smaller size sub-problems, solve the sub-problems using another heuristic, and patch the resulting solutions to form a tour. In this experiment, we used *three* iterations of Lin and Kernighan's heuristic to solve the partitioned sub-problems.

Two modifications are needed in order for simulated annealing to run on a 10,000-city traveling salesman problem. Ideally, we would like to let M = 9,999 and perform

M	CPU time (sec.)			Tour length	
	t'	t	t' / t	$\delta'(\%)$	$\delta(\%)$
128	4850	11050	0.44	6.98	6.66
64	10330	16070	0.64	3.52	3.41
32	26460	21610	1.22	2.03	1.97

Table 5.10: Comparison with Karp's heuristic coupled with Lin and Kernighan's heuristic on the traveling salesman problem with 10,000 uniformly distributed cities. In this table, M is the number of sub-problems in the partition, t' and t represent the CPU time for Karp's heuristic and simulated annealing, respectively, while δ' and δ represent the percentages above the best cost estimated from formula (5.2). These are the average results of *four* executions.

table lookup for the distances between cities. This is, however, impossible due to the $O(N^2)$ storage requirement. As a result, we compute the distances as the need arises and store the distances of only 250 nearest neighbors. The result of the comparison between simulated annealing and Karp's heuristic is shown in Table 5.10 while the pictures of tours obtained via both methods are shown in Fig. 5.7. The estimated best cost for this problem is based on the formula [Beard59]

$$\lim_{N \to \infty} \frac{C_{opt}}{\sqrt{NI^2}} = 0.749, \tag{5.2}$$

where C_{opt} is the optimal tour length and I = 1,000,000 is the interval from which the coordinates of the cities are picked. Table 5.10 is constructed by varying the number of partitions used in Karp's heuristic and comparing the resulting solutions with simulated annealing for different values of λ.

From this table we observe that simulated annealing out-performs Karp's heuristic for high quality solutions. This evidence strongly supports our belief that an efficient annealing schedule coupled with a good move generation strategy makes simulated annealing an extremely effective method.

Figure 5.7a

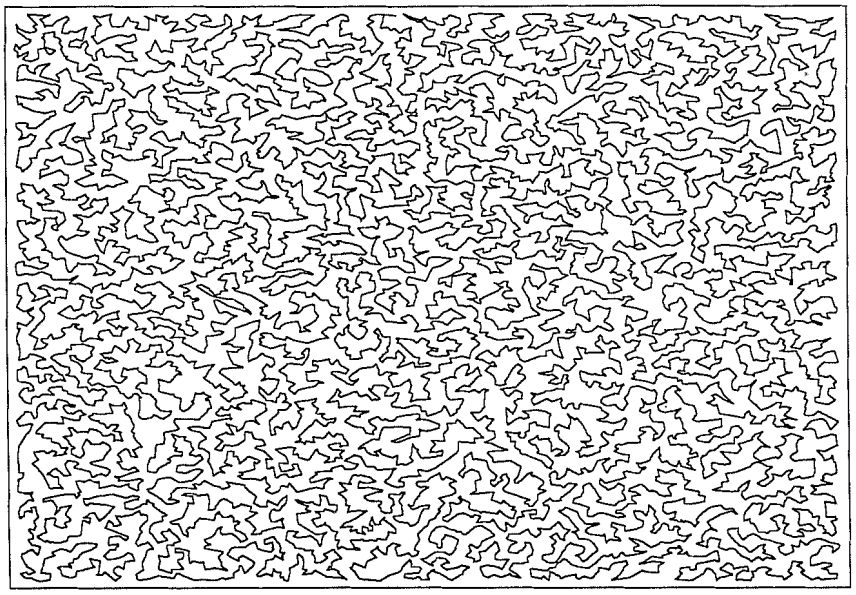

Figure 5.7b

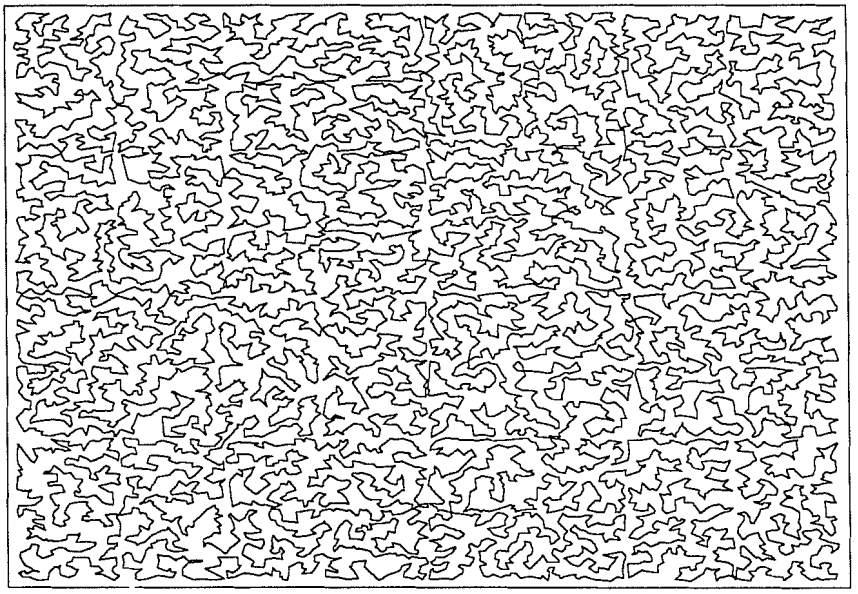

Figure 5.7: Tours of a 10,000-city traveling salesman problem. Figure 5.7a: Tour found using simulated annealing, with length 7.59×10^7. Figure 5.7b: Tour found using Karp's heuristic, with length 7.64×10^7; the edges of the 64 squares in the partition can be made out from the tour.

5.2. Graph partition problem

The graph partition problem (or graph partitioning) is NP-complete [Garey79]. The goal of this problem is to partition the vertices of a graph into two equal size vertex sets, V_A and V_B, so as to minimize the number of edges with end-points in both sets. We first describe the implementation details of the move generation strategy for simulated annealing, followed by a discussion of the test results. Our results are similar to those obtained by Johnson *et al.* [Johns87] with one essential difference. By using the efficient λ-schedule and a good move generation strategy, we are able to speed up simulated annealing significantly. Thus, for the same high quality solution, the CPU time spent by simulated annealing compares favorably with the CPU time spent by multiple executions of Fiduccia and Mattheyses's heuristic [Fiduc82]. The first part of the discussion covers the comparison between the efficient λ-schedule and Huang's annealing schedule while the second part covers the comparison between simulated annealing and Fiduccia and Mattheyses's heuristic.

5.2.1. Implementation details

The cost to be minimized in a graph partition problem is the size of the cutset—the number of edges with end-points in both vertex sets. In order to increase flexibility in move generation, we, following the work of Johnson *et al.*, introduce an additional imbalance factor into the cost function:

$$Cost = cutsize + \xi(\mid V_A \mid - \mid V_B \mid)^2,$$

where *cutsize* is the size of the cutset, ξ is the imbalance factor, and $\mid V_A \mid$ and $\mid V_B \mid$ are, respectively, the number of vertices in sets V_A and V_B. Ideally, we would allow $\mid V_A \mid$ to differ from $\mid V_B \mid$ in order to get more potential moves and, thus, increasing the chance of climbing out from a local optimum. However, if this size difference is not controlled, we may end up with a cutsize of zero but with all vertices in one set while with no vertex in the other. The introduction of the imbalance factor serves two purposes. The first one is to control the difference in size (the number of vertices) between V_A and V_B. A big difference in size is unlikely to occur at low temperature

because moves that reduce this difference would likely be accepted while moves that increase this difference would likely be rejected. The second one is to increase flexibility in move generation by allowing moves that result in a difference between $| V_A |$ and $| V_B |$. This increase in flexibility increases the number of available moves and, hence, the chance of climbing out from a local optimum. Since these two purposes are contradictory, we must choose the imbalance factor carefully: too small a value will unduly favor the first purpose while too large a value will unduly favor the second purpose. The value of the imbalance factor used in our experiments is either 0.005 or 0.02. The former value is used for problems in which the average edge degree is low while the latter value is used for problems in which the average edge degree is high.

A move is proposed in two steps involving two vertices. First, a vertex is picked from an array of double link lists and moved to the other vertex set. Then, a second vertex is picked independently from the array and moved to the other vertex set. Since the two vertices are picked independently, we may end up moving two vertices from the same set to the other, a perfectly acceptable operation. The reason for moving two vertices instead of one is as follows. The desired acceptance ratio of 44% is difficult to achieve at high temperature because almost all proposed moves are accepted at this temperature. Thus, we would like to move a larger number of vertices at high temperature since moving a larger number of vertices is likely to result in a larger proposed energy change and, hence, a smaller acceptance ratio. As the temperature is lowered, the acceptance ratio decreases. When it is below 44%, we would like to move a smaller number of vertices to obtain a smaller proposed energy change and, hence, a larger acceptance ratio. However, moving a constant number of vertices instead of a variable number has the advantage of reducing the program complexity. Furthermore, although moving a large number of vertices is desirable at high temperature, the computation time per move is unacceptable. We, therefore, achieve a compromise by moving two vertices per move.

In order to understand how move generation is controlled, we must first know how the double link lists are organized. Associated with each vertex, there is a variable called *gain* that keeps track of the change in cutsize if this vertex is moved from its

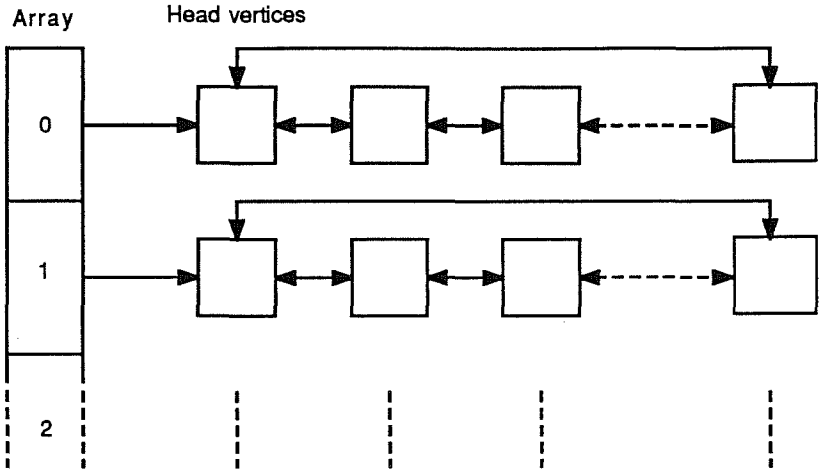

Figure 5.8: Organization of double link lists.

current vertex set to the other. All vertices with the same absolute gain are grouped to form a double link list. These lists can then be indexed by an array in which the n^{th} element of the array points to the double link list of vertices with absolute gain n. See Fig. 5.8 for an illustration of the data structure. We shall call *head vertex* of the list the vertex to which the array points.

Move generation is controlled using either of the two following methods. In the first method called *standard control*, the two vertices are picked randomly and independently with equal probability from the head vertices of the first θ double link lists. After the head vertex of a list is picked, it loses its head position to the next vertex in the list even if the proposed move is rejected. The parameter θ is lowered as a function of the inverse temperature, s, according to the empirical formula

$$\theta = M \left\lceil \log_{10}(10 + \frac{0.0005}{sC_i}) - 1 \right\rceil^2,$$

where M is the maximum absolute gain possible and C_i is the initial cutsize. The idea behind this method is to control the proposed energy change in such a way that the acceptance ratio drops almost linearly as a function of the total number of proposed moves. In the second method called *feedback control*, the two vertices are picked

independently from the head vertices of two double link lists selected using the formula

$$\theta = - \overline{\theta}\log(\xi),$$

where ξ is a random number uniformly distributed between 0 and 1, $\overline{\theta}$ is a control parameter and θ denotes the θ^{th} list in the array. If the value of θ exceeds the value of M, a condition that happens less frequently as $\overline{\theta}$ gets smaller, θ will be set to ξM. The control parameter $\overline{\theta}$ is adjusted after every τ moves according to the formula

$$\overline{\theta}_n = \max(\overline{\theta}_{n-\tau} + 5(\hat{\rho}_n - 0.44), 1.5),$$

where $\hat{\rho}_n$ is the measured acceptance ratio for the last τ moves. If $\hat{\rho}_n$ is less than 0.44, the value of $\overline{\theta}$ is lowered. If $\hat{\rho}_n$ is greater than 0.44, the value of $\overline{\theta}$ is raised. This updating procedure enables us to keep the acceptance ratio close to the desired acceptance ratio of 44%.

The settings of the λ-schedule parameters in Table 4.1 for the graph partition problem are $\lambda L_a = 400$, $\lambda L_b = 20{,}000$, $\tau = 100$, and $k = 5$.

Huang's annealing schedule was implemented as in [Huang86]. Due to the difference in move generation controls, a few maximum generation limits have been tried. We settled on a value of $0.04N^2$, where $N = \mid V_A \mid + \mid V_B \mid$ is the number of vertices in the graph.

In the implementation of Fiduccia and Mattheyses's heuristic, we considered two procedures in which the size of the two vertex sets was allowed to differ either by at most 1 or by at most $0.1N$, where N is the total number of vertices. This size difference is eliminated at the end of an execution by moving vertices that lead to the smallest increase in cutsize from the larger vertex set to the smaller one. Preliminary experimentation did not indicate any significant difference between the final results of the two procedures. We, therefore, restrict the maximum size difference of the two vertex sets to 1.

5.2.2. Test results

To assess the performance of the efficient λ-schedule, we compare it with Huang's annealing schedule on the graph partition problem. The experiment was conducted on *random graphs* with $N = 500$ or $N = 1{,}000$ vertices and an average edge degree, d, of 5 or 20. These graphs are random in the sense that the probability that any pair of vertices constitutes an edge is given by $d \,/\, (N-1)$. Since for any vertex, there are $N-1$ such pairs, the average edge degree is d. Four instances of random graphs with the following N and d values are chosen in the test: $N = 500$, $d = 5$; $N = 500$, $d = 20$; $N = 1{,}000$, $d = 5$; $N = 1{,}000$, $d = 20$. The experimental results are shown in Table 5.11 in which γ is the excess of the found cutsize, C, over the estimated best cutsize, C_0, i.e., $\gamma = C - C_0$. This best cutsize was found by carrying out a sequence of careful annealing executions as described in the traveling salesman problem.

N	d	Speedup		
		$\gamma = 12$	$\gamma = 8$	$\gamma = 5$
500	5	3.2	4.6	4.2
1000	5	2.5	2.7	4.8
500	20	9.8	13.7	>100.0
1000	20	13.1		

Table 5.11: Comparison between the efficient λ-schedule with feedback control and Huang's annealing schedule with standard control on random graphs. The excess of the cutsize, C, over the estimated best cutsize, C_0, is denoted by γ, i.e., $\gamma = C - C_0$.

N	d	CPU time (sec.)		Cutsize	
		t'	t	γ'	γ
500	5	65.7	8.3	26	17
1000	5	371.4	24.4	61	41
500	20	87.3	22.6	43	29
1000	20	424.2	57.0	83	65

Table 5.12: Comparison between Kernighan and Lin's heuristic and the best of 10 executions of Fiduccia and Mattheyses's heuristic. The results from Kernighan and Lin's heuristic are denoted by t' and γ', while the results from Fiduccia and Mattheyses's heuristic are denoted by t and γ.

The speedups associated with a particular γ as the number of vertices or the average edge degree increases are shown along each column in Table 5.11 while the speedups associated with a particular problem size as the cutsize improves are shown along each row. The speedup is defined as the ratio of the CPU time used by Huang's annealing schedule with standard control to the CPU time used by the efficient λ-schedule with feedback control. We observe a speedup of up to 100 on a random graph with $N = 500$ and $d = 20$.

We also compare simulated annealing using the efficient λ-schedule and feedback control with another heuristic for the graph partition problem. Our choice of heuristics consists of Kernighan and Lin's heuristic [Kerni70], and multiple executions of Fiduccia and Mattheyses's heuristic (a sped up version of the Kernighan and Lin's heuristic). Since Fiduccia and Mattheyses's heuristic is very fast (linear time), and Kernighan and Lin's heuristic gives slightly better results than Fiduccia and Mattheyses's heuristic, we would like to know whether the best of multiple executions of Fiduccia and Mattheyses's heuristic gives better results than Kernighan and Lin's heuristic. Based on the preliminary results in Table 5.12, we chose multiple executions of Fiduccia and

Figure 5.9a

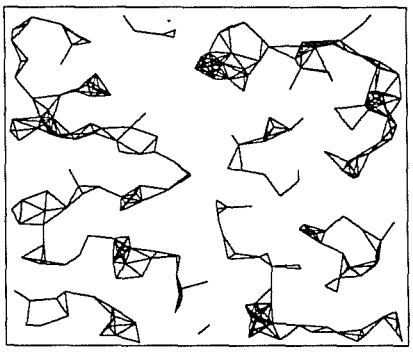

Figure 5.9b

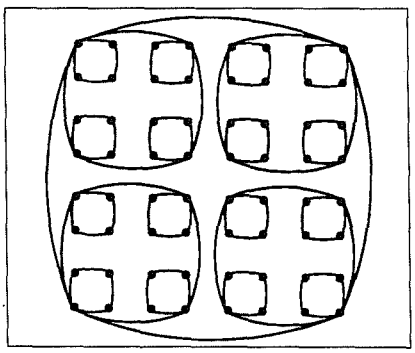

Figure 5.9a: A geometric random graph with 500 vertices and an average edge degree of 5. Figure 5.9b: A hierarchical graph with 64 vertices. The minimum cutsize for such graphs is always 2.

Mattheyses's heuristic as the competitor. Three types of graphs were used in the comparison between simulated annealing and Fiduccia and Mattheyses's heuristic: random graph, geometric random graph, and hierarchical graph. The construction of random graphs has been described. To construct a geometric random graph with N vertices and an average edge degree d as described in [Johns87], we first independently pick N pairs of random numbers uniformly distributed in the interval of 0 to 1. These N pairs of numbers are the coordinates of the N vertices. Then, we add an edge between any two vertices whose distance apart is less than r with $r = \sqrt{d \ / \ N\pi}$. This formula for r is obtained by noting that all N vertices are contained in an area of 1. Hence, the average area per vertex is $1 \ / \ N$. To find an average of d vertices whose distances are less than r apart from a given vertex (i.e., in an area of πr^2), we let $r = \sqrt{d \ / \ N\pi}$. See Fig. 5.9a for the picture of a geometric random graph with $N = 500$ and $d = 5$. To construct a hierarchical graph with 4^k vertices where $k = 1, 2,...,$ we apply the following algorithm.

1. let $i = k$.
2. partition the 4^i vertices into 4^{i-1} groups with four vertices in each group.
3. add four edges to each group to make a cycle.
4. select one vertex from each of these groups to obtain a total of 4^{i-1} vertices.
5. let $i = i - 1$ and repeat step 2 to step 5 on these 4^i vertices until the graph is connected.

A hierarchical graph with 64 vertices is depicted in Fig. 5.9b. It is easy to see that the minimum cutsize for these graphs is always 2.

The test results for the three types of graphs are displayed in Table 5.13 to Table 5.15 in which C_0 is the best cutsize found and γ' and γ are the cutsizes in excess of C_0 for Fiduccia and Mattheyses's heuristic and for simulated annealing, respectively. The speedups, $t' \ / \ t$, are defined as the ratio of the CPU time used by Fiduccia and Mattheyses's heuristic, t', to the CPU time used by simulated annealing, t. All results

N	d	CPU time (sec.)			Cutsize		
		t'	t	t' / t	γ'	γ	C_0
500	5	64.7	41.5	1.56	10	10	247
1000	5	154.0	42.1	3.66	31	32	463
500	20	209.7	49.4	4.24	11	12	1706
1000	20	487.4	91.7	5.32	45	47	3374

Table 5.13: Comparison with Fiduccia and Mattheyses's heuristic on random graphs.

N	d	CPU time (sec.)			Cutsize		
		t'	t	t' / t	γ'	γ	C_0
500	5	66.4	1050.1	0.06	6	11	5
1000	5	204.2	905.7	0.23	14	14	20
500	20	174.5	2823.5	0.06	0	49	100
1000	20	388.5	1315.3	0.30	2	369	181

Table 5.14: Comparison with Fiduccia and Mattheyses's heuristic on geometric random graphs.

N	CPU time (sec.)			Cutsize	
	t'	t	t' / t	γ'	γ
256	20.1	248.3	0.08	0	0
1024	128.6	126.3	1.02	7	7
4096	488.7	161.4	3.03	43	43

Table 5.15: Comparison with Fiduccia and Mattheyses's heuristic on hierarchical graphs. The minimum cutsize C_0 for hierarchical graphs is always 2.

tabulated for Fiduccia and Mattheyses's heuristic are the average results of *at least eight* groups, where the CPU time of a group is the cumulative CPU time for 100 executions and the cutsize of a group is the best cutsize found in those 100 executions. All results tabulated for simulated annealing are the average results of *at least eight* executions.

From Table 5.13 we observe a speedup of up to 5 for a random graph with 1,000 vertices and an average degree of 20. To better understand the relationship between the speedup and the quality of the final solution, we pick a random graph with $N = 500$ and $d = 5$ and vary the execution time. The test results on such a graph are shown in Table 5.16 in which the number of executions within a group is varied from 10 to 2,000. We observe a speedup of up to 15 for high quality solutions.

Exec.	CPU time (sec.)			Cutsize	
	t'	t	t' / t	γ'	γ
10	8.6	24.0	0.36	13	13
100	64.7	41.5	1.56	10	10
500	308.4	76.4	4.04	7	6
2000	1226.4	82.0	14.96	5	5

Table 5.16: Detailed comparison with Fiduccia and Mattheyses's heuristic on a random graph with 500 vertices and an average edge degree of 5. The best cutsize found, C_0, for this problem is 247.

In contrast, simulated annealing performs relatively poorly on geometric random graphs. We observe from Table 5.14 that simulated annealing runs up to 17 times slower and gives worse results than Fiduccia and Mattheyses's heuristic. The superiority of Fiduccia and Mattheyses's heuristic over simulated annealing on geometric graphs may be related to the special structure of these graphs. As shown in Fig. 5.9a, all edges of a geometric random graph are between pairs of vertices that are close by. Simulated annealing, being insensitive to any structure, performs poorly on this type of graph when compared with Fiduccia and Mattheyses's heuristic, which takes advantage of this structure.

Another kind of graph for which simulated annealing does not perform well is hierarchical graphs. The optimal cutsize for these graphs is always 2 as can be observed easily from Fig. 5.9b. The test results for hierarchical graphs are displayed in Table

N	CPU time (sec.)		Cutsize	
	t'	t	γ'	γ
256	20.1	248.3	0	0
1024	2574.2	1745.7	5	1
4096	9890.5	2755.5	42	18

Table 5.17: The best average results obtained (with reasonable amount of CPU time) by simulated annealing and Fiduccia and Mattheyses's heuristic on hierarchical graphs with 256 to 4,096 vertices. The CPU time tabulated for Fiduccia and Mattheyses's heuristic is the cumulative CPU time of all the executions in a group. By varying the number of executions in a group, we vary the total CPU time. The minimum cutsize for hierarchical graphs is always 2.

5.15. Although simulated annealing performs poorly on this kind of graph, Fiduccia and Mattheyses's heuristic does even worse. We observe a speedup of up to 3 on a hierarchical graph with 4,096 vertices. Furthermore, by comparing Table 5.15 with Table 5.17, we notice that the quality of the solutions obtainable by simulated annealing is much higher than that obtainable by Fiduccia and Mattheyses's heuristic if we increase the allotted CPU time. Both methods find the optimal cutsize for $N = 256$. However, as the number of vertices increases, the problem becomes increasingly difficult. Except for specially designed heuristics that exploit this kind of structure, we believe finding high quality solutions for hierarchical graphs is a very difficult problem because of the small optimal cutsize.

5.3. Standard cell placement problem

One of the most difficult and important NP-complete combinatorial optimization problems in integrated circuit layout is the standard cell placement problem [Shing86]. Given a set of fixed height variable width standard cells with pins on their boundaries and a net list of interconnections among pins, the goal of a standard cell placement problem is to place the cells on rows so that the interconnection length and, hopefully, the resulting area are minimized. In this section, we restrict our attention to the comparison of the efficient λ-schedule with a hand-crafted annealing schedule and Huang's annealing schedule. We use TimberWolfSC version 4.1 and version 4.2 [Seche87], a standard cell placement and global routing program, as the "arena". We first cover the implementation details, followed by a discussion of the test results. Note that this experiment also demonstrates the ease of replacing an existing annealing schedule with the efficient λ-schedule in a simulated annealing based program.

5.3.1. Implementation details

The primary cost to be minimized in TimberWolfSC is the total length of the interconnecting wires. To increase flexibility in move generation which involves either single cell movement or pairwise cell exchange, the authors of TimberWolfSC allow overlapping of standard cells. This complicates matter as a valid placement does not

permit cell overlaps. To remedy this problem, the authors of TimberWolfSC introduce an additional time-varying overlap penalty factor in the cost function:

$$Cost = total\_wire\_length + c(i) \times penalty.$$

The factor $c(i)$ discussed below is adjusted in such a way that the cell overlap penalty becomes an increasingly important contribution to the total cost of the system. Consequently, as the temperature drops, the system spends more of its time removing cell overlaps than minimizing total wire length. After simulated annealing terminates, the remaining cell overlaps are removed by a heuristic procedure.

The time-varying cost function in TimberWolfSC violates our fundamental assumption that the cost function for a given problem is invariant—the energy density function does not change with respect to time. This assumption is made implicitly in order to derive the efficient λ-schedule. However, if the cost function is *slowly varying*, the analysis carried out in Chapter 2 and Chapter 3 will remain valid locally. Therefore, the annealing schedule that we obtained should still be a good approximation to the actual efficient λ-schedule—assuming that such schedule exists.

Implementation of the efficient λ-schedule in TimberWolfSC version 4.1 involves four modifications. We replaced TimberWolfSC's annealing schedule with the efficient λ-schedule and also modified the computation of the coefficient of the penalty function, $c(i)$, the move generation controller, and the number of bins in a row. The modification of the computation of $c(i)$ is necessary because the cost function depends on $c(i)$ and, hence, the current iteration number i. TimberWolfSC always executes 104 iterations and changes temperature after every iteration. Within each iteration a fixed number of moves depending on the problem size is proposed. Since the efficient λ-schedule runs faster than TimberWolfSC's hand-crafted annealing schedule, it executes less moves than TimberWolfSC's. In order to arrive at similar values of $c(i)$, we need to modify the way $c(i)$ is computed. This was done by making an educated guess of the last iteration number for the efficient λ-schedule and scaling the values of $c(i)$ accordingly. The settings of the last iteration number, i, for both Huang's annealing schedule and the efficient λ-schedule are shown in Table 5.20.

The move generation controller was also modified. In TimberWolfSC, rows are divided into rectangular regions called bins. A move is proposed by first picking a cell, say cell A, randomly. Then a window, whose size is independent of the cell and is equal to $2\times mean\_cell\_width + 6\times standard\_deviation$ by three rows, is centered around cell A. A bin is picked randomly within that window. If the bin contains no cell, a single cell movement, which involves moving cell A to a random position in the bin, is proposed. If the bin contains one or more cells, a cell is picked randomly within that bin and a pairwise exchange is proposed. Instead of picking a bin in such a way, we modified the move generation controller by allowing the window size to change dynamically and picking a bin according to the formulas

$$\theta_x = \pm\, \overline{\theta}_x \log(\xi) \quad \text{and} \quad \theta_y = \pm\, \overline{\theta}_y \log(\xi),$$

in which ξ is a uniformly distributed random number between 0 and 1, θ_x and θ_y are, respectively, the x and y distances of the new bin from cell A, and $\overline{\theta}_x$ and $\overline{\theta}_y$ are the control parameters. (The symbol $\pm$ indicates the signs of θ_x and θ_y are either positive or negative with equal probability.) If bins are picked this way, the probability that a bin at distances θ_x and θ_y from cell A is picked is proportional to $e^{-\theta_x/\overline{\theta}_x}$ in the x-direction, and $e^{-\theta_y/\overline{\theta}_y}$ in the y-direction; therefore, bins that are closer to cell A have a higher probability of being picked. The control parameters $\overline{\theta}_x$ and $\overline{\theta}_y$ are adjusted after every τ moves according to the formulas

$$\overline{\theta}_x \leftarrow \max(\overline{\theta}_x + \frac{M_x - N_x}{30}(\hat{\rho}_n - 0.44),\, N_x),$$

$$\overline{\theta}_y \leftarrow \max(\overline{\theta}_y + \frac{M_y - N_y}{30}(\hat{\rho}_n - 0.44),\, N_y),$$

where $\hat{\rho}_n$ is the measured acceptance ratio for the last τ moves, $N_x = mean\_cell\_width + 3\times standard\_deviation$, M_x is equal to the maximum row width, $N_y = 0.75$, and M_y is equal to the maximum number of rows. If $\hat{\rho}_n$ is less than 0.44, the values of $\overline{\theta}_x$ and $\overline{\theta}_y$ are lowered. If $\hat{\rho}_n$ is greater than 0.44, the values of $\overline{\theta}_x$ and $\overline{\theta}_y$ are raised.

The last modification was to change the number of bins in a row. In TimberWolfSC the total number of bins is roughly equal to the number of standard cells. Since the efficient λ-schedule starts at infinite temperature, all proposed moves will be accepted indiscriminately. Whenever a single cell movement is proposed, the number of empty bins can only decrease. This is because when a cell is moved from a bin occupied by more than one cell to a vacant bin, the number of vacant bins decreases by one. And there is no way to get a vacant bin back! Since single cell movements are proposed only when empty bins are found, the percentage of single cell movements proposed at later temperatures will be small. On the contrary, TimberWolfSC starts at an initial temperature of 500. At this temperature, not all proposed moves are accepted so that typically 10% to 20% of the bins remain empty. In order to allow the efficient λ-schedule to propose single cell movements at a higher rate, we increase the number of bins by decreasing the bin size. Different total numbers of bins have been tried; we settled on a ratio of 1.5 for the total number of bins to the total number of cells. This arrangement guarantees a 33% chance for single cell movements to be proposed.

The major differences between the implementations of the efficient λ-schedule in TimberWolfSC version 4.1 and version 4.2 are how $c(i)$ is adjusted and how to choose the initial temperature. In TimberWolfSC version 4.1, $c(i)$ is adjusted by scaling the values of $c(i)$ on the basis of an educated guess of the last iteration number. In

Name	Size	CPU time (sec.)			Wire length		
		t_1	t_0	t_1 / t_0	w_1	w_0	w_1 / w_0
example	183	719	437	1.65	218049	216285	1.01
8870	286	1463	881	1.66	181621	177399	1.02
ic	347	1839	1550	1.19	83232	83027	1.00
sd2	469	2092	2034	1.03	262318	262427	1.00
primary1	752	4491	3187	1.41	986564	974308	1.01
5655	800	4880	3318	1.47	958758	946914	1.01
sd1	2357	19891	12412	1.60	2165850	2179570	0.99
ha	2965	29798	20940	1.42	2006360	1832440	1.09

Table 5.18: Comparison with TimberWolfSC version 4.1's hand-crafted annealing schedule. The subscripts 1 and 0 are used to indicate results from TimberWolfSC's annealing schedule and the efficient λ-schedule, respectively.

TimberWolfSC version 4.2, $c(i)$ is adjusted automatically after every iteration using the measured acceptance ratio as the reference. In addition, we modified TimberWolfSC version 4.2 so that it can start at a lower temperature as described in Section 4.3 to take advantage of certain good initial placements. This option must be used with caution. Starting at a lower temperature makes the final result depend on the initial placement. Although good initial placements produced by *certain* methods tend to give good results systematically, not all good initial placements give good final results. Unless the initial placement is produced by a method which is known to be favored by simulated annealing, starting at an infinite temperature is recommended.

The settings of the λ-schedule parameters in Table 4.1 for the standard cell placement problem are $\lambda L_a = 60$, $\lambda L_b = 3{,}000$, $\tau = 100$, and $k = 5$.

5.3.2. Test results

We tested the efficient λ-schedule in TimberWolfSC version 4.1 on eight instances of the standard cell placement problem with size (the total number of cells) ranging from 183 to 2,965. The test results when compared with TimberWolfSC's annealing schedule and Huang's annealing schedule are shown in Table 5.18 and Table 5.19, respectively. The subscripts 1 and 0 in Table 5.18 are used to indicate results from TimberWolfSC's annealing schedule and the efficient λ-schedule, respectively, while the subscripts 2 and 0 in Table 5.19 are used to indicate results from Huang's annealing schedule and the

Name	Size	CPU time (sec.)			Wire length		
		t_2	t_0	t_2 / t_0	w_2	w_0	w_2 / w_0
example	183	630	437	1.44	251567	216285	1.16
8870	286	1396	881	1.58	202731	177399	1.14
ic	347	1521	1550	0.98	98842	83027	1.19
sd2	469	2091	2034	1.03	371500	262427	1.42
primary1	752	5145	3187	1.61	1235140	974308	1.27
5655	800	4433	3318	1.34	1154030	946914	1.22

Table 5.19: Comparison with Huang's annealing schedule (in TimberWolfSC version 4.1). The subscripts 2 and 0 are used to indicate results from Huang's annealing schedule and the efficient λ-schedule, respectively.

efficient λ-schedule. In these tables t represents the CPU time and w represents the total wire length after cell overlaps are removed. The test results for **sd1** and **ha** are the average results of *four* executions while the test results for the others are the average results of *eight* executions.

From Table 5.18 we observe that the efficient λ-schedule gives a speedup of 1.03 to 1.66 when compared with TimberWolfSC's. The test cases **ic** and **sd2** give a much smaller speedup than the others. This is due to two reasons. The first and the most important one is the very good initial placement of **sd2**. The initial temperature of TimberWolfSC is set at 500; this is not high enough to destroy the structure of the initial placement. Consequently, a good initial solution helps TimberWolfSC give a

Name	Size	x:y	Parameters			
			λ_2	i_2	λ_0	i_0
example	183	12.0:1	70.0	104	0.03	54
8870	286	1.4:1	10.0	104	0.014	54
ic	347	0.4:1	15.0	104	0.01	60
sd2	469	0.8:1	100.0	104	0.013	90
primary1	752	1.0:1	100.0	104	0.009	54
5655	800	2.2:1	100.0	104	0.008	54

Table 5.20: Parameters for Huang's annealing schedule and the efficient λ-schedule (in TimberWolfSC version 4.1). The subscripts 2 and 0 are used to indicate settings for Huang's annealing schedule and the efficient λ-schedule, respectively. The parameters λ_0, λ_2 and i_0, i_2 control the quasi-equilibrium criteria and the cost function, respectively. Note that λ_2 is set to a value much greater than 1 since $\lambda_2 \leq 1$ results in an unacceptable amount of computation time.

A.P.C.	CPU time (sec.)			Wire length		
	t_1	t_0	t_1 / t_0	w_1	w_0	w_1 / w_0
10	594	694	0.86	183901	182771	1.01
25	1463	881	1.66	181621	177399	1.02
50	2958	1318	2.24	177012	178610	0.99
75	4408	1745	2.53	176149	175191	1.01

Table 5.21: Detailed comparison with TimberWolfSC version 4.1's hand-crafted annealing schedule on **8870**, a 286 cell example. The number of moves (per iteration) proposed by TimberWolfSC's annealing schedule is varied and is given by $no\_of\_cells \times A.P.C.$ in which $A.P.C.$ stands for *attempts per cell.*

better final placement. To test this hypothesis, we performed an experiment in which a randomized placement of **sd2** was used as the initial solution. Instead of giving an average result of 262,318, TimberWolfSC gives an average of 267,250. Though the difference is only 2%, it is significant since the efficient λ-schedule runs 20% to 30% faster if an average of 267,250 is considered sufficient. The second reason has to do with the aspect ratios of **ic** and **sd2**. We observe from the **x:y** column of Table 5.20 that the ratios of the width of the desired placement (**x**) to the height of the desired placement (**y**) are less for **ic** and **sd2** than for the rest. The modified move generation controller works better when the set of values it can control is larger. Since standard cells are placed on rows, their y-positions can only take on discrete values imposed by the y-positions of the rows. Their x-positions, however, can take on any integral value between 0 and M, where M is a large number—a much larger set. Therefore, a smaller **x:y** ratio gives less freedom for the controller to work with. This hinders the operation of the efficient λ-schedule.

Since the speedup depends on the allotted CPU time, it is interesting to know how the speedup varies with the quality of solution on the same problem. So, we performed another experiment on **8870**, a 286 cell problem. In this experiment, the CPU time spent by TimberWolfSC's annealing schedule was allowed to increase by proposing more moves per iteration. Then for the same quality of solution, the CPU times spent by the

Name	Size	CPU time (sec.)			Wire length		
		t_1	t_0	t_1 / t_0	w_1	w_0	w_1 / w_0
example	183	1134	732	1.55	206775	200660	1.03
8870	286	1758	1332	1.32	173254	174645	0.99
ic	347	2155	1550	1.39	83350	82279	1.01
sd2	469	2714	1925	1.41	357911	351277	1.02
primary1	752	5540	3682	1.50	987050	984829	1.00
5655	800	5730	5209	1.10	955483	954058	1.00
sd1	2357	23869	12304	1.94	1849220	1813530	1.02
ha	2965	35757	27089	1.32	2006720	1994300	1.01

Table 5.22: Comparison with TimberWolfSC version 4.2's hand-crafted annealing schedule. The subscripts 1 and 0 are used to indicate results from TimberWolfSC's annealing schedule and the efficient λ-schedule, respectively.

efficient λ-schedule were measured. The results are shown in Table 5.21. We observe that the speedup increases as the quality of the solution improves. This is consistent with the test results for the traveling salesman problem that the better the quality of the solution, the larger the speedup.

Huang's annealing schedule was implemented as in [Huang86]. From Table 5.19, we observe that the efficient λ-schedule gives a speedup of 0.98 to 1.61 with solutions that are 14% to 42% better than those obtained with Huang's annealing schedule. Due to the excessive computation time used by Huang's annealing schedule, we were not able to obtain comparable solutions so that a better measure of the speedup could be made. The results displayed for Huang's annealing schedule were obtained using TimberWolfSC's move generation controller. Due to the difference in move generation control between this version of TimberWolfSC and the older version used in [Huang86], a few maximum generation limits have been tried. We settled on a value of $5.5 \times no\_of\_cells$, which corresponds roughly to the number of possible moves when the move generation controller is used. The discrepancy between Huang *et al.*'s results in [Huang86] and our results is probably due to the different versions of TimberWolfSC used in the tests; TimberWolfSC has been sped up significantly since the version used in [Huang86].

The efficient λ-schedule has also been tested in TimberWolfSC version 4.2. The test results when compared with TimberWolfSC's annealing schedule are shown in Table 5.22. The test results for **sd1** and **ha** are the average results of *four* executions while the test results for the others are the average results of *eight* executions. From this table we observe that the efficient λ-schedule speeds up TimberWolfSC by a factor of up to 2. The test cases **sd1** and **sd2** are allowed to start at a lower temperature at which the measured acceptance ratio is close to 44% and the control parameters $\overline{\theta}_x$ and $\overline{\theta}_y$ of the move generation controller are at their minimum values. Comparing Table 5.18 with Table 5.22, we observe that starting at a lower temperature leads to better speedups for **sd1** and **sd2**. We also tried to start at a lower temperature for other test cases. The results are less favorable. This experiment suggests that although initial placements produced by certain methods tend to give good final results, not all good initial

Figure 5.10a

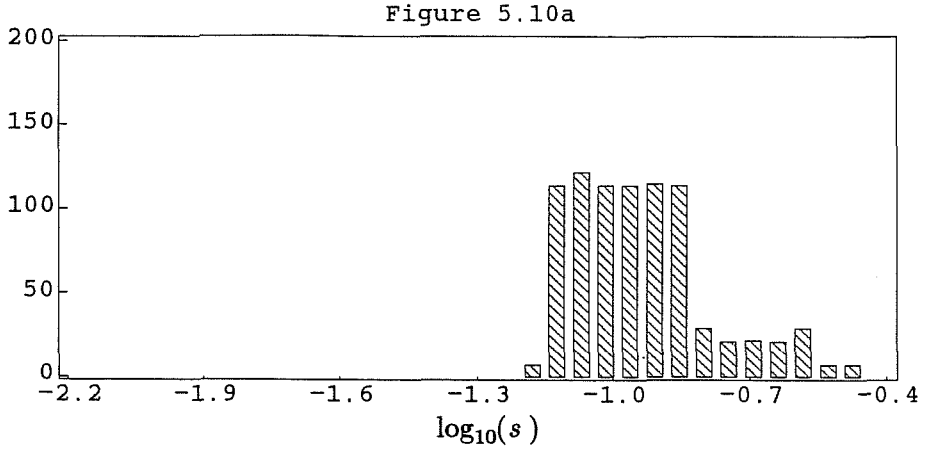

Figure 5.10b

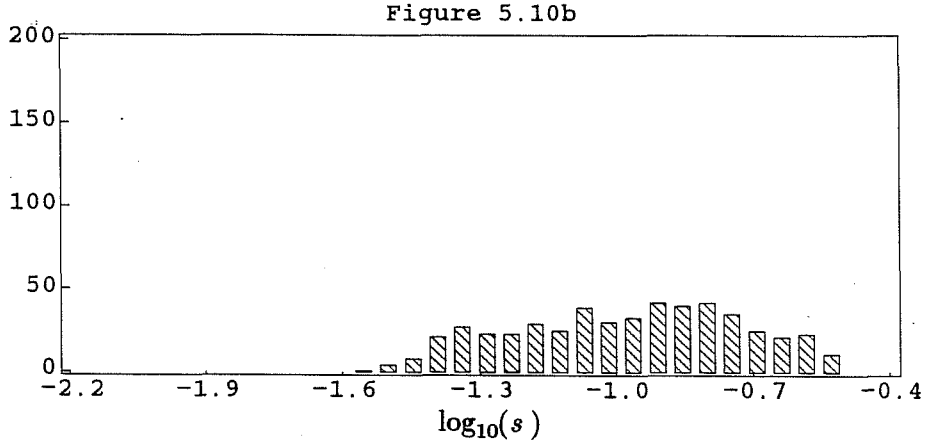

Figure 5.10c

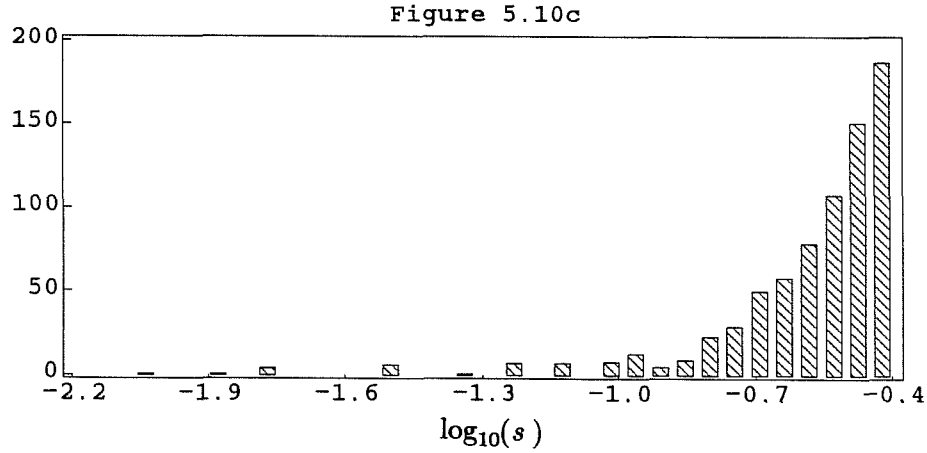

Figure 5.10: How different annealing schedules spend their time at different inverse temperatures. Figure 5.10a: The histogram of the time spent by TimberWolfSC version 4.1's annealing schedule. Figure 5.10b: The histogram of the time spent by the efficient λ-schedule. Figure 5.10c: The histogram of the time spent by Huang's annealing schedule.

placements give good final results.

To better understand how the different annealing schedules spend their time at different inverse temperatures, $\log_{10}(s)$ is divided into equal increments and a histogram of the number of inverse temperatures that fall within each of these increments is constructed. The histograms for TimberWolfSC version 4.1's annealing schedule, the efficient λ-schedule, and Huang's annealing schedule are shown in Fig. 5.10, respectively. Note that Huang's annealing schedule spends most of its time at very low temperature, while the efficient λ-schedule spends its time more evenly. One of the reasons for that is because in Huang's implementation, $\sigma^2(s)$ is kept constant at all temperatures, whereas the actual $\sigma^2(s)$ is much smaller at low temperature; consequently, the temperature is lowered more slowly than it should be.

Before we end this section, we should point out that TimberWolfSC's annealing schedule has been hand-crafted over the years and tailored towards the placement problem, while the efficient λ-schedule is a completely general schedule; applicable to a wide variety of problems. The fact that the efficient λ-schedule can achieve a speedup over TimberWolfSC's annealing schedule is significant.

We have shown that the efficient λ-schedule coupled with a good move generation strategy achieves substantial speedups when compared with schedules available in the literature. Since the efficient λ-schedule uses statistical quantities only, it is applicable to general combinatorial optimization problems. In addition, for problems that have little structure, simulated annealing has been shown to out-perform tailored heuristics for the traveling salesman and the graph partition problems for high quality solutions. In Chapter 6 we shall give our concluding remarks on the efficient λ-schedule and simulated annealing as a general method for solving combinatorial optimization problems.

References

[Aarts85] E. Aarts and P. van Laarhoven, "Statistical Cooling Algorithm: A General Approach to Combinatorial Optimization Problems," *Philips Journal of Research;* Vol. 40, No. 4, 193-226, 1985.

[Beard59] J. Beardwood, J. Halton, and J. Hammersley, "The Shortest Path Through Many Points," *Proceedings of the Cambridge Philosophical Society,* Vol. 55, 299-327, 1959.

[Cerny85] V. Cerny, "A Thermodynamical Approach to the Traveling Salesman Problem: an Efficient Simulation Algorithm," *Journal of Optimization Theory and Applications,* Vol. 45, 41-51, 1985.

[Dunlo85] A. Dunlop and B. Kernighan, "A Procedure for Placement of Standard Cell VLSI Circuits," *IEEE Transactions on Computed-Aided Design,* Vol. 4, No. 1, 92-98, 1985.

[Efron82] B. Efron, *The Jackknife, the Bootstrap and Other Resampling Plans,* CBMS 38, Society for Industrial and Applied Mathematics, 1982.

[Fiduc82] C. Fiduccia and R. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," *Proceedings of the 19th Design Automation Conference,* 175-181, 1982.

[Garey79] M. Garey and D. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness,* Freeman and Company, 1979.

[Green86] J. Greene and K. Supowit, "Simulated Annealing Without Rejected Moves," *IEEE Transactions on Computed-Aided Design,* Vol. 5, No. 1, 221-228, 1986.

[Hajek85] B. Hajek, "A Tutorial Survey of Theory and Applications of Simulated Annealing," *Proceedings of the 24th Conference on Decision and Control,* 755-760, 1985.

[Huang86] M. Huang, F. Romeo, and A. Sangiovanni-Vincentelli, "An Efficient General Cooling Schedule for Simulated Annealing," *Proceedings of the*

IEEE International Conference on Computer-Aided Design, 381-384, 1986.

[Johns87] D. Johnson, C. Aragon, L. McGeoch, and C. Schevon, "Optimization by Simulated Annealing: an Experimental Evaluation (Part I)," *Draft,* 1987.

[Karp77] R. Karp, "Probabilistic Analysis of Partitioning Algorithms for the Traveling Salesman Problem in the Plane," *Mathematics of Operations Research,* Vol. 2, No. 3, 209-224, 1977.

[Kerni70] B. Kernighan, and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *Bell System Technical Journal,* Vol. 49, 291-307, 1970.

[Kirkp82] S. Kirkpatrick, C. Gelatt Jr., and M. Vecchi, "Optimization by Simulated Annealing," *IBM Research Report, RC 9355,* 1982.

[Lin73] S. Lin and B. Kernighan, "An Effective Heuristic Algorithm for the Traveling Salesman Problem," *Operations Research,* Vol. 21, 498-516, 1973.

[Mitra85] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and Finite-time Behavior of Simulated Annealing," *Proceedings of the 24th Conference on Decision and Control,* 761-767, 1985.

[Seche85] C. Sechen, and A. Sangiovanni-Vincentelli, "The TimberWolf Placement and Routing Package," *IEEE Journal of Solid-State Circuits,* Vol. 20, No. 2, 510-522, 1985.

[Seche87] C. Sechen and K.-W. Lee, "An Improved Simulated Annealing Algorithm for Row-Based Placement," *Proceedings of the IEEE International Conference on Computer-Aided Design,* 478-481, 1987.

[Shing86] M. Shing and T. Hu, "Computational Complexity of Layout Problems," *Advances in CAD for VLSI,* Vol. 4, North-Holland, 1986.

[Ullma84] J. Ullman, *Computational Aspects of VLSI,* Computer Science Press, 1984.

CHAPTER 6

Summary and Conclusions

We have presented a new simulated annealing schedule derived from a quasi-equilibrium (or quasi-stationarity) criterion:

$$\left| \overline{X}(s_{n-1}) - \mu(s_n) \right| \leq \lambda \sigma(s_n),$$

where $\overline{X}(s_{n-1})$ is the average energy of the system at step $n-1$, λ is a user specified parameter, and $\mu(s_n)$ and $\sigma(s_n)$ are, respectively, the equilibrium (or stationary) mean and standard deviation of the energy at inverse temperature s_n, $s_n \equiv 1 / T_n$. Unlike Aarts's quasi-equilibrium criterion which depends only on the equilibrium statistics of the system, our criterion depends on the average energy of the system and, hence, the move generation strategy. Since a good move generation strategy lets the system reach quasi-equilibrium faster, we can lower the temperature faster; a good quasi-equilibrium criterion, therefore, should take into account this dependency between the move generation strategy and the temperature decrements.

From our quasi-equilibrium criterion, we have derived an annealing schedule that lowers the temperature at every step and keeps the system in quasi-equilibrium at all times, thus avoiding the need of an equilibrium detection scheme. Furthermore, for a given move generation strategy, a given λ and a given number of steps, this schedule has been shown to give the minimum final average energy among all schedules that maintain the system in quasi-equilibrium at all times.

Two models have been introduced in order to relate move generation strategy to temperature decrements. Based on these models, we have derived the final form of the annealing schedule,

$$s_{n+1} = s_n + \left[\frac{\lambda}{\sigma(s_n)}\right]\left[\frac{1}{s_n{}^2\sigma^2(s_n)}\right]\left[\frac{4\rho_0(1-\rho_0)^2}{(2-\rho_0)^2}\right],$$

where ρ_0 is the acceptance ratio. We have also derived the condition on the move generation strategy that leads to the largest decrement in temperature (and, hence, the fastest schedule) while maintaining the system in quasi-equilibrium. The decrement in temperature can be grouped into three factors. The first factor which is the same as those factors in Aarts's and Huang's schedules depends on the quasi-equilibrium criterion. The second factor relates the specific heat, the rate of change of equilibrium average energy with respect to the temperature, $s^2\sigma^2(s)$, to the temperature decrement. The presence of this factor is in agreement with the common belief that the larger is the specific heat, the slower should be the cooling. The third factor relates the acceptance ratio, which depends on the temperature and the move generation strategy, to the temperature decrement. The maximum of this factor occurs at $\rho_0 = 0.44$. Consequently, in order to have the largest decrement in temperature, the move generation strategy should be controlled such that an acceptance ratio of 44% results. This is similar to the suggestion made by Binder [Binde86, p. 11] on Monte Carlo methods in statistical physics that the magnitude of the proposed energy change should be chosen such that the acceptance ratio is close to 50%. Indeed, we have shown in Table 3.1 of Section 3.2.1.2 that a close to 44% acceptance ratio gives the best results.

The new annealing schedule has been tested extensively on the traveling salesman problem, the graph partition problem and the standard cell placement problem. Based on a preliminary experiment on the traveling salesman problem, we selected Huang's annealing schedule, the best among the three annealing schedules including Aarts's schedule and the logarithmic schedule, as the competitor. The comparison with Huang's annealing schedule gives a speedup of up to 24 for the traveling salesman problem and a speedup of up to 100 for the graph partition problem. Due to the excessive computation time required by Huang's annealing schedule, we were not able to obtain solutions to the standard cell placement problem whose quality was close enough to that obtained with the efficient λ-schedule so that an accurate assessment of the speedup could be made.

We have also compared simulated annealing with other heuristics on the traveling salesman and the graph partition problems. The results depend on the types of city distributions and the types of graphs. For the traveling salesman problem and when compared with Lin and Kernighan's heuristic, we observed a speedup of up to 47 for problems in which the cities are clustered. However, for problems in which the cities lie on a grid, Lin and Kernighan's heuristic gives significantly better solutions in much shorter time. For the graph partition problem and when compared with the best of multiple executions of Fiduccia and Mattheyses's heuristic, we observed a speedup of up to 5 for random graphs. Yet for geometric random graph, Fiduccia and Mattheyses's heuristic out-performed simulated annealing on both the quality of the final solutions and the computation time. These results lead us to believe that for problems where optimal or close to optimal solutions are sparse and high quality solutions are desired, simulated annealing coupled with a good move generation strategy can be a very effective heuristic.

One naturally wonders whether to use simulated annealing or not for a given optimization problem. In order to make the decision, one needs answers to the following questions. The first two questions are whether high quality solutions are desired and whether a large amount of computation time can be afforded. If high quality solutions are not needed or computation time is scarce, a fast heuristic may be a better choice. Simulated annealing typically performs well on problems where many degrees of freedom (or independent variables) exist or the constraints on the solutions are conflicting. Because this kind of problems has a large number of local minima, a probabilistic hill climbing method like simulated annealing would likely to be better than other heuristics in avoiding these local minima. Thus, the third question is to determine whether the optimization problem has many degrees of freedom or conflicting constraints.

After one decides to use simulated annealing, the next problem is to select the cost function. The cost function that one wants to minimize may not be favorable to simulated annealing or may be difficult to evaluate. In that case, one may use a different cost function that reflects a slightly different objective, but is more favorable to simulated annealing or more efficient computationally. For example, in the standard cell

placement problem, the primary objective is to minimize the area of the layout. However, if one uses the area as the cost function, either single cell movement or pairwise cell exchange will show little change in the area [Seche88]. Such a combination of cost function and move generation strategy is undesirable since it does not point to the direction of the search. A slightly different objective would be the total length of the interconnecting wires. Either a single cell movement or a pairwise cell exchange will give a significant change in this new cost function. Furthermore, the total wire length is related to the area of the layout; usually, the shorter is the total wire length, the smaller is the area. Consequently, the total wire length is a better objective for simulated annealing than the area. To avoid the computation complexity of finding out the exact total wire length, we may use the sum of the half perimeter of the bounding box for each net as an approximation for the total wire length. Once the decision on the cost function has been made, the procedure described in Section 4.3 may be followed.

Despite the speedup due to the new annealing schedule, simulated annealing still requires a significant amount of computation time; future work includes an extension of the new annealing schedule to multi-processor implementation. For example, a placement problem may be partitioned into regions with each processor responsible for one region. There are two types of possible moves: intra-processor move and inter-processor move. For intra-processor moves, each region may be thought of as a placement problem and, hence, associated with it a temperature. For inter-processor moves, the two regions involved in a move may be at different temperatures. The goal of the extension is to incorporate the diffusion of temperature between regions at different temperatures in the new annealing schedule. For a review of current research in parallel implementation of simulated annealing, the interested reader is referred to [Laarh87, Chapter 8].

References

[Binde86] K. Binder, *Monte Carlo Methods in Statistical Physics,* 2nd Edition, Springer-Verlag, 1986.

[Laarh87] P. van Laarhoven, and E. Aarts, *Simulated Annealing: Theory and Applications,* D. Reidel Publishing Company, 1987.

[Seche88] C. Sechen, *private communications,* 1988.