CWI Tracts

Managing Editors

J.W. de Bakker (CWI, Amsterdam)
M. Hazewinkel (CWI, Amsterdam)
J.K. Lenstra (CWI, Amsterdam)

Editorial Board

W. Albers (Maastricht)
P.C. Baayen (Amsterdam)
R.T. Boute (Nijmegen)
E.M. de Jager (Amsterdam)
M.A. Kaashoek (Amsterdam)
M.S. Keane (Delft)
J.P.C. Kleijnen (Tilburg)
H. Kwakernaak (Enschede)
J. van Leeuwen (Utrecht)
P.W.H. Lemmens (Utrecht)
M. van der Put (Groningen)
M. Rem (Eindhoven)
A.H.G. Rinnooy Kan (Rotterdam)
M.N. Spijker (Leiden)

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

The CWI is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

Theoretical and computational aspects of simulated annealing

P.J.M. van Laarhoven

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

1980 Mathematics Subject Classification (1985): 60J10, 65K05, 90B35, 62F15, 33A35.
ISBN 90 6196 352 4
NUGI-code: 811

Acknowledgements

This tract is a slightly revised version of my doctoral thesis, which was written under the supervision of professors Jan Karel Lenstra and Alexander Rinnooy Kan. I feel fortunate to have had the opportunity to benefit from their vast knowledge of combinatorial optimization; their meticulous reading of several drafts of the thesis has yielded many substantial improvements.

It is with even more pleasure that I acknowledge the contributions of Emile Aarts. Not only did he initiate the research described here, he was also an active participant in it. I have found our many discussions always inspiring and illuminating and I can only hope that the future will hold many a promise for further fruitful co-operation.

I would also like to acknowledge the numerous contributions of Guus Boender to chapter 5 and the appendices; in addition, I am grateful for his suggestions and for the many stimulating discussions we had.

I am obliged to Eric van Utteren and to the board of directors of the Philips Research Laboratories, in particular to Theo Claasen, for the opportunity afforded to carry out the research described here, to my colleagues Gerard Beenker, Ronan Burgess, Ton Kalker and Jan Korst for their comments and contributions while the work was in progress, and to the other members of the Ph.D. committee, professors L.F.M. de Haan, F.A. Lootsma, J.A.E.E. van Nunen and J. Wessels, who read drafts of the thesis and made constructive comments.

Finally, I thank the Centre for Mathematics and Computer Science for the opportunity to publish this tract in their series CWI tracts and all those who have contributed to its technical realization.

P.J.M. van Laarhoven

Few people of attainments take easily to a plan of self-improvement. Some discover very early their perfection cannot endure the insult. Others find their intellectual pleasure lies in the theory, not the practice. Only a few stubborn ones will blunder on, painfully, out of the luxuriant world of their pretensions into the desert of mortification and reward.

Patrick White

Contents

Chapter 1

Introduction and summary

In this tract we are concerned with *combinatorial optimization* [Lawler, 1976; Papadimitriou & Steiglitz, 1982], the search for optima of functions of discrete variables. Combinatorial optimization problems are nowadays ubiquitous in such diverse areas as, for example, design of algorithms [Aho, Hopcroft & Ullman, 1974], of integrated circuits [Breuer, 1972] and operations research [Wagner, 1975].
An instance of a combinatorial optimization problem is formalized as a pair $(\mathcal{R}, C)$, where $\mathcal{R}$ is the finite - or possibly countably infinite - *set of configurations* (also called *configuration space* or *solution space*) and C a *cost function*, $C : \mathcal{R} \rightarrow \mathbb{R}$, which assigns a real number to each configuration. For convenience, only minimization problems are considered (which can be done without loss of generality). Thus, the problem is to find a configuration $i_0 \in \mathcal{R}$, for which C takes its minimum value, i.e. such that

$$C_{opt} = C(i_0) = \min_{i \in \mathcal{R}} C(i), \tag{1.1}$$

where C_{opt} denotes the minimum cost value.
When dealing with a combinatorial optimization problem Π, there are two ways to go. One can try to construct either an *optimization algorithm* for Π, i.e. an algorithm that returns a globally minimal configuration for every instance of Π, or an *approximation algorithm*, i.e. an algorithm that merely returns a configuration for each instance [Garey & Johnson, 1979]. Preferably, the latter algorithm should have the property that for all instances the returned configu-

ration is "close" to a globally minimal configuration.
The reason why many combinatorial optimization problems are tackled by constructing approximation rather than optimization algorithms is related to the fact that many combinatorial optimization problems are *NP-hard*, or that the *decision version* of many combinatorial problems is *NP-complete* [Garey & Johnson, 1979]. For such problems it is commonly believed that no algorithm can be constructed that solves each instance of the problem to optimality with an amount of computational effort bounded by a polynomial function of the *input length*[1] of such an instance. Indeed, it would be a major breakthrough in complexity theory if a polynomial-time algorithm could be found for an *NP*-complete problem, since in that case all *NP*-complete problems would be solvable in polynomial time.
Furthermore, the distinction between *NP*-hard problems and problems solvable in polynomial time seems to be closely related to the distinction between hard and easy problems. Computational experience has increased evidence for this relation: though there is, of course, the possibility of such contrasts as an $O(1.001^n)$ exponential-time algorithm and an $O(n^{100})$ polynomial-time algorithm, these kinds of complexities hardly ever seem to occur in practice [Cook, 1983; Johnson & Papadimitriou, 1985].
The aforementioned facts have led to the belief that large *NP*-hard combinatorial optimization problems cannot be solved to optimality in acceptable amounts of computation time. A more reasonable goal then is to find an approximation algorithm that runs in low-order polynomial time and has the property that final configurations are "close" to globally minimal ones. For many combinatorial optimization problems such algorithms are nowadays available, but usually they suffer from the fact that they are only applicable to the particular problem they are designed for (in this connection, the notion *tailored algorithm* is used). As soon as a new combinatorial optimization problem arises, a new algorithm has to be constructed. General approximation algorithms, able to find near-optimal configurations for a wide variety of combinatorial optimization problems, are rare. The *simulated annealing algorithm*, the subject of this tract, is such a

[1] The input length is a formal measure of instance size [Garey & Johnson, 1979].

generally applicable approximation algorithm.
The simulated annealing algorithm can be viewed as a randomized version of an *iterative improvement algorithm*[2]. The application of an iterative improvement algorithm presupposes the definition of configurations, a cost function and a *generation mechanism*, i.e. a simple prescription to generate a *transition* from one configuration to another. The generation mechanism defines a *neighbourhood* $\mathcal{R}_i$ for each configuration i, consisting of all configurations that can be reached from i in a single transition. Iterative improvement is therefore also known as *neighbourhood search* or *local search* [Papadimitriou & Steiglitz, 1982]. Alternatively, instead of defining a generation mechanism, one can define a *neighbourhood structure*, i.e. a description of the neighbourhood of each configuration. In many cases, it is then implicitly assumed that the generation mechanism is such that each transition from a configuration to one of its neighbours is generated with equal probability.
The iterative improvement algorithm can be formulated as follows. Starting off at a given configuration, a sequence of trials is generated. In each trial a configuration is selected from the neighbourhood of the current configuration. If this neighbouring configuration has a lower cost, the current configuration is replaced by this neighbour, otherwise another neighbour is selected and compared for its cost value. The algorithm terminates when a configuration is obtained whose cost is no worse than any of its neighbours.
Iterative improvement algorithms possess the following disadvantages:

- By definition, iterative improvement algorithms terminate in the first local minimum encountered; generally, such a local minimum deviates substantially in cost from a global minimum.
- The returned local minimum depends on the initial configuration, for the choice of which generally no guidelines are available.
- In general, it is not possible to give an upper bound on the computation time. For instance, the worst-case complexity of the iterative improvement algorithm for the *travelling salesman*

[2]Strictly speaking, an iterative improvement algorithm need not be completely deterministic itself, since the neighbourhood can be searched in a random order.

problem based on *Lin's 2-change strategy* [Lin, 1965] is an open problem, see [Johnson, Papadimitriou & Yannakakis, 1985] and the references therein.

It should be clear, however, that iterative improvement does have the advantage of being generally applicable: the main ingredients, viz. configurations, a cost function and a neighbourhood structure, are usually easy to define. Besides, though upper bounds for computation times are missing, a single run of an iterative improvement algorithm can **on the average** be executed in a small amount of computation time. For instance, dealing with Lin's 2-change strategy for the travelling salesman problem, Kern [1986a] recently showed that for a certain class of problem instances the expected computation time is bounded by a polynomial function of the input length of the instance. Because of the small computational cost of one run of an iterative improvement algorithm, it is customary to execute the algorithm for a large number of initial configurations, drawn independently from the configuration space $\mathcal{R}$. In this way, the first two of the disadvantages mentioned can be removed.

We recall that the reason why iterative improvement algorithms terminate in the first local minimum they encounter is that only transitions corresponding to a **decrease** in cost are accepted by the algorithm. Alternatively, we might think of an algorithm which also accepts, in some limited way, transitions corresponding to an **increase** in cost. Simulated annealing is an example of the latter approach: in addition to cost-decreasing transitions, cost-increasing transitions are accepted with a non-zero probability, which gradually decreases as the algorithm continues its execution. Ever since its introduction, independently by Kirkpatrick, Gelatt & Vecchi [1983] and Černy [1985], the algorithm has attracted much attention, partly because it is based on an intriguing combination of ideas from completely different and at first sight totally unrelated fields of science, and partly because it is claimed by many authors not to exhibit any of the aforementioned disadvantages of the iterative improvement algorithm, whilst maintaining its advantages. Thus, it is often asserted that the simulated annealing algorithm is a generally applicable, high-quality combinatorial optimization tool.

The major contribution of this tract is threefold:

1. An implementation of the algorithm is described which provably leads to polynomial-time execution.

2. Extensive computational evidence is presented to support the aforementioned assertion that simulated annealing is a generally applicable, high-quality optimization tool. At the same time it is shown not to be a panacea, since it is often not able to compete with tailored algorithms for a particular problem.

3. A novel (Bayesian) approach to the analysis of the algorithm is presented.

The tract is organized as follows. The algorithm itself is extensively discussed in chapters 2 and 3. In chapter 2, after an introduction to the algorithm and the analogy on which it is based, simulated annealing is mathematically described as the generation of a sequence of homogeneous Markov chains. Necessary and sufficient conditions are derived to ensure that asymptotically the algorithm finds a globally minimal configuration with probability 1. Thus, these conditions relate to the *asymptotic behaviour* as an optimization algorithm.

In chapter 3, the *finite-time behaviour* of simulated annealing is addressed. Since the aforementioned necessary and sufficient conditions cannot be satisfied in finite time, the finite-time behaviour of simulated annealing is that of an approximation algorithm. The problem then is to find values for certain parameters of the algorithm (referred to as a *cooling schedule*) that ensure that near-optimal configurations are returned. A cooling schedule which tries to achieve the latter by closely imitating the aforementioned asymptotic behaviour is described in chapter 3. This schedule is compared with other schedules from the literature on the basis of numerical results obtained by solving instances of the *graph partitioning* and *travelling salesman problems*.

The quality of an approximation algorithm can be judged on its performance in terms of the quality of the configuration returned by the algorithm and the computation time needed by the algorithm to find that configuration (for the moment, we discard other criteria such as ease of implementation, flexibility and simplicity). When theoretical results with respect to these criteria are lacking, as is largely the case

with simulated annealing, one has to resort to numerical tests of the quality of an algorithm by running the algorithm on a large set of representative problem instances and measuring computation times and quality of returned configurations. Such tests are described in chapter 4, where results are presented obtained by running simulated annealing on instances of the *travelling salesman*, the *job shop scheduling* and the *football pool* problems. Where possible, simulated annealing is pitted against other approximation and optimization algorithms for these problems.

In chapter 5, we consider simulated annealing from a *Bayesian* point of view. In this approach, the generation of each Markov chain is seen as a random experiment with unknown parameters characterizing the outcome of the experiment. In the case of simulated annealing, the unknown parameters are the probability of occurrence of values of the cost function and the minimum value of the cost function. Assuming a probability distribution on the values of these unknown parameters (referred to as the *prior distribution*) and given the outcome of the experiment (the sequence of configurations resulting from the generation of a Markov chain), we use Bayes's theorem to derive the *posterior distribution* on the values of the parameters. Numerical experiments are described in which the posterior distribution firstly is shown to predict accurately the behaviour of the algorithm during the next Markov chain and secondly is used to compute the (a posteriori) expectation of the minimum value of the cost function. Furthermore, the Bayesian information is used to derive optimal rules for choosing some of the parameters of a cooling schedule.

The tract is ended with some conclusions and remarks.

Chapter 2

Simulated annealing

2.1 Introduction of the algorithm

The simulated annealing algorithm [Kirkpatrick, Gelatt & Vecchi, 1983; Černy, 1985] originates from the analogy between two problems: that of finding the *ground state* of a solid and that of finding a globally minimal configuration in a combinatorial optimization problem.
In condensed matter physics, *annealing* denotes a physical process by which, if carried out sufficiently slowly, the ground state of a solid can be found. The simulated annealing algorithm takes its name from the fact that it is based on an algorithm to simulate (parts of) the annealing process.
The physical process denoted by annealing is one in which a solid in a *heat bath* is heated up by increasing the temperature of the heat bath to a value at which all particles of the solid randomly arrange themselves in the liquid phase, followed by cooling through slowly lowering the temperature of the heat bath. In this way, the particles arrange themselves in the low-energy ground state, provided the cooling is carried out sufficiently slowly. Starting off at a given value of the temperature, the cooling phase of the annealing process can be described as follows. At each temperature value T, the solid is allowed to reach *thermal equilibrium*. In thermal equilibrium the probability of occurrence of a state with energy E is given by the *Boltzmann*

distribution:

$$Pr\{\mathbf{E} = E\} = \frac{1}{Z(T)} \cdot \exp\left(-\frac{E}{k_B T}\right), \tag{2.1}$$

where $Z(T)$ is the *partition function* and k_B the *Boltzmann constant*. The factor $\exp\left(-\frac{E}{k_B T}\right)$ is known as the *Boltzmann factor*. As the temperature decreases, the Boltzmann distribution concentrates on the low-energy states and finally, when the temperature approaches zero, only the minimum-energy states have a non-zero probability of occurrence. However, if the cooling is too rapid, i.e. if the solid is not allowed to reach thermal equilibrium at each temperature value, defects can be 'frozen' into the solid resulting in metastable amorphous structures instead of the low-energy crystalline structure. If the temperature of the heat bath is lowered instantaneously the particles are frozen into one of the metastable amorphous structures. This process is known as *quenching*.

There is some similarity between a solid and a combinatorial optimization problem: in both cases there are many degrees of freedom (the positions of the particles of the solid, the configurations in an optimization problem) and in both cases some global quantity has to be minimized (the energy of the solid, the cost function in combinatorial optimization). The observation of this analogy is the first step in the construction of the simulated annealing algorithm, the next step is to extend this analogy to the *Metropolis algorithm*.

To simulate the evolution to thermal equilibrium of a solid, Metropolis, Rosenbluth, Rosenbluth, Teller & Teller [1953] proposed a *Monte Carlo method*, which generates sequences of states of the solid in the following way. Given the current state of the solid, characterized by the positions of its particles, a small, randomly generated, perturbation is applied, i.e. a small displacement of a randomly chosen particle. If the perturbation results in a lower energy state of the solid, then the process is continued with the new state. If $\Delta E \geq 0$, then the probability of acceptance of the perturbed state is given by $\exp(-\frac{\Delta E}{k_B T})$. This rule for accepting new states is referred to as the *Metropolis criterion*. Guided by this criterion, the solid eventually evolves into thermal equilibrium, i.e. after a large number of perturbations, using the aforementioned acceptance criterion, the probability distribution

of the states approaches the Boltzmann distribution, given by (2.1). In statistical mechanics this Monte Carlo method, which is known as the *Metropolis algorithm*, is a frequently used method to estimate averages or integrals by means of random sampling techniques; see for example Binder's review article [1978].
The Metropolis algorithm can also be used to generate sequences of configurations of a combinatorial optimization problem. In that case, the configurations assume the role of the states of a solid while the cost function C and the *control parameter* c assume the roles of energy and temperature, respectively. The simulated annealing algorithm can be viewed as a sequence of Metropolis algorithms evaluated at decreasing values of the control parameter. It can thus be described as follows. Initially, the control parameter is given a large value and a sequence of trials is generated using the same generation mechanism as in the iterative improvement algorithm. Thus, in each trial, a configuration j is generated by choosing at random an element from the neighbourhood of the current configuration i. This corresponds to the small perturbation in the Metropolis algorithm. Let $\Delta C_{ij} = C(j) - C(i)$, then the probability of configuration j being the next configuration in the sequence equals 1, if $\Delta C_{ij} \leq 0$, and $\exp(-\frac{\Delta C_{ij}}{c})$, if $\Delta C_{ij} > 0$ (the Metropolis criterion). Thus, there is a non-zero probability of continuing with a configuration with higher cost than the current configuration. This sequence of trials is continued until equilibrium is reached, i.e. until the probability distribution of the configurations approaches the Boltzmann distribution, now given by

$$Pr\{\text{configuration} = i\} \stackrel{\text{def}}{=} q_i(c) = \frac{1}{Q(c)} \cdot \exp\left(-\frac{C(i)}{c}\right), \qquad (2.2)$$

where $Q(c)$ is a normalization constant depending on the control parameter c, being the equivalent of the aforementioned partition function.
The control parameter is lowered in steps until it approaches 0, with the system being allowed to approach equilibrium for each step by generating a sequence of trials in the previously described way. After termination, the final 'frozen' configuration is taken as the solution of the problem at hand.
Thus, as with iterative improvement, we have again a generally appli-

cable approximation algorithm: configurations, a cost function and a neighbourhood structure are the only prerequisites to be able to apply simulated annealing.

Comparing iterative improvement and simulated annealing, it is apparent that the situation where the control parameter in the simulated annealing algorithm is set to 0 corresponds to a version of iterative improvement (it is not iterative improvement *per se*, because in an iterative improvement approach the neighbouring configurations are not necessarily examined in random order). In the analogy with condensed matter physics, setting the control parameter to 0 corresponds to the aforementioned quenching process.

On the other hand, simulated annealing is a generalization of iterative improvement in that it accepts, with non-zero but gradually decreasing probability, deteriorations in cost. It is not clear, however, whether it performs better than a repeated application of iterative improvement for a number of different initial configurations: both algorithms converge asymptotically to a globally minimal configuration of the problem at hand. For simulated annealing asymptotic convergence is proved in the next section; for repeated application of iterative improvement it is obvious that convergence is obtained for $N \rightarrow \infty$, where N denotes the number of initial configurations for which the algorithm is applied, if only for the fact that a global minimum is encountered as an initial configuration with probability 1 as $N \rightarrow \infty$. However, Lundy & Mees [1986] construct an example of a combinatorial optimization problem, for which, **in expectation**, both repeated application of iterative improvement and a complete enumeration of all configurations of the problem take an order of magnitude more elementary operations to reach the global minimum than simulated annealing. In chapter 4, extensive comparisons are made between the two algorithms on the basis of a large set of numerical experiments and computational evidence is presented for the assertion that if both algorithms are allowed the same amount of computation time, simulated annealing returns substantially better configurations (in terms of cost) than repeated application of iterative improvement.

2.2 Mathematical model of the algorithm

Simulated annealing can be viewed as an algorithm that continuously attempts to transform a configuration into one of its neighbours. Such an algorithm can mathematically be described by means of a *Markov chain*: a sequence of trials, where the outcome of each trial only depends on the outcome of the previous trial [Feller, 1950]. In the case of simulated annealing, trials correspond to transitions. Since the acceptance of a transition depends only on the cost values of the current and generated configuration, it is clear that the outcome of a transition (the new configuration) only depends on the outcome of the previous transition (the current configuration).
A Markov chain is described by means of a set of *conditional probabilities* $P_{ij}(k)$ for each pair of outcomes (i, j); $P_{ij}(k)$ is the probability that the outcome of the k-th trial is j, given that the outcome of the $(k-1)$-th trial is i. Let $X(k)$ denote the outcome of the k-th trial, then we have:

$$P_{ij}(k) = Pr\{X(k) = j \mid X(k-1) = i\}. \tag{2.3}$$

If the conditional probabilities do not depend on k, we write P_{ij} instead of $P_{ij}(k)$. The corresponding Markov chain is then called *homogeneous*, otherwise it is called *inhomogeneous*.
Returning to simulated annealing, we note that $P_{ij}(k)$ denotes the probability that the k-th transition is a transition from configuration i to configuration j and that $X(k)$ is the configuration obtained after k transitions. In view of this, $P_{ij}(k)$ is called the *transition probability* and the $|\mathcal{R}| \times |\mathcal{R}|$-matrix $P(k)$ the *transition matrix*.
The transition probabilities depend on the value of the control parameter c, the analogue of the temperature in the physical annealing process. Thus, if c is kept constant, the corresponding Markov chain is homogeneous and its transition matrix $P = P(c)$ is given by:

$$P_{ij}(c) = \begin{cases} G_{ij}(c)A_{ij}(c) & \forall j \neq i \\ 1 - \sum_{l=1, l\neq i}^{|\mathcal{R}|} G_{il}(c)A_{il}(c) & j = i, \end{cases} \tag{2.4}$$

where the *generation probability* $G_{ij}(c)$ denotes the conditional probability of generating configuration j, given that the current configuration is i, and the *acceptance probability* $A_{ij}(c)$ denotes the conditional probability of accepting the transition from configuration i to configuration j. The corresponding matrices $G(c)$ and $A(c)$ are called the *generation* and *acceptance matrix*, respectively. As a result of the definition in (2.4), $P(c)$ is a *stochastic matrix*, i.e. a matrix satisfying $\sum_j P_{ij}(c) = 1$ for all i.
We remark that in the initial formulation of the algorithm, the generation matrix is defined by

$$G_{ij}(c) = G_{ij} = \begin{cases} \mid \mathcal{R}_i \mid^{-1} & \text{if } j \in \mathcal{R}_i \\ 0 & \text{elsewhere,} \end{cases} \tag{2.5}$$

i.e. G is independent of c and corresponds to a uniform distribution on the neighbourhoods, while $A_{ij}(c)$ is given by the Metropolis criterion, i.e.

$$A_{ij}(c) = \begin{cases} \exp\left(-\frac{C(j)-C(i)}{c}\right) & \text{if } C(j) > C(i) \\ 1 & \text{if } C(j) \leq C(i). \end{cases} \tag{2.6}$$

As pointed out before, the control parameter c is decreased during the course of the algorithm. We distinguish two mechanisms to carry out this decrement:

- a decrement of c after each transition, resulting in an algorithm, which can be described by a single inhomogeneous Markov chain (the *inhomogeneous algorithm*);

- a decrement of c after a number of transitions, resulting in an algorithm which can be described by a sequence of homogeneous Markov chains, each generated at a fixed value of c (the *homogeneous algorithm*). Note that we do not exclude the possibility that c is decreased after an infinite number of transitions.

The distinction is not as clear-cut as the foregoing suggests: the inhomogeneous algorithm can be considered as a special case of the homogeneous algorithm (the sequence of homogeneous Markov chains collapses into one inhomogeneous Markov chain), but the reverse is

also true if we think of a zero-decrement of c in between the transitions of the homogeneous Markov chains. However, the results with respect to asymptotic convergence of the homogeneous algorithm, which are derived in the next section, presuppose that the length of each homogeneous Markov chain is taken to infinity. Consequently, these results do not pertain to the inhomogeneous algorithm.

2.3 Asymptotic convergence of the algorithm

In this section we consider **arbitrary** generation and acceptance matrices and derive conditions on these matrices to ensure asymptotic convergence of both the homogeneous and the inhomogeneous algorithm to a globally minimal configuration.
For the homogeneous algorithm we derive sufficient conditions on $G(c)$ and $A(c)$ that ensure that if the Markov chains are all of infinite length and if the limit $c \downarrow 0$ is taken, then the algorithm converges in probability to a globally minimal configuration. For the inhomogeneous algorithm we briefly discuss conditions that are both necessary and sufficient. These conditions not only relate to the matrices $A(c)$ and $G(c)$ but also to the way the limit $c \downarrow 0$ is taken.
Essential to the convergence proof for the homogeneous algorithm is the fact that, under certain conditions, the *stationary distribution* of a homogeneous Markov chain exists. The stationary distribution is defined as the vector $\mathbf{q}$ whose i-th component is given by [Feller, 1950]

$$q_i = \lim_{k \to \infty} Pr\{X(k) = i \mid X(0) = j\}, \tag{2.7}$$

for an arbitrary j.
If $\mathbf{q}$ exists, we have that

$$\begin{aligned} &\lim_{k \to \infty} Pr\{X(k) = i\} \\ &= \lim_{k \to \infty} \sum_j Pr\{X(k) = i \mid X(0) = j\} \cdot Pr\{X(0) = j\} \\ &= \sum_j q_i Pr\{X(0) = j\} = q_i. \end{aligned} \tag{2.8}$$

Thus, the stationary distribution is the probability distribution of the configurations after an infinite number of trials (transitions).
The proof of theorem 2.1 is now based on the following arguments. First, it is shown that under certain conditions on the matrices $A(c)$ and $G(c)$ the stationary distribution $\mathbf{q}(c)$ exists for all $c > 0$. Next, it is shown that under additional conditions $\mathbf{q}(c)$ converges to a uniform distribution on the set of globally minimal configurations.

Theorem 2.1
Suppose the following conditions on the matrices $G(c)$ and $A(c)$ are satisfied:

(1) $\forall c > 0,\ \forall i, j \in \mathcal{R}\ \exists p \geq 1, \exists \lambda_0, \lambda_1, \ldots, \lambda_p \in \mathcal{R}\ (i = \lambda_0, j = \lambda_p)$:

$$G_{\lambda_k \lambda_{k+1}}(c) > 0,\ k = 0, 1, \ldots, p-1; \tag{2.9}$$

(2) $$\forall c > 0,\ \forall i, j \in \mathcal{R} : G_{ji}(c) = G_{ij}(c); \tag{2.10}$$

(3) $$\forall c > 0,\ \forall i, j, k \in \mathcal{R} :$$

$$C(i) \leq C(j) \leq C(k) \Rightarrow A_{ik}(c) = A_{ij}(c) A_{jk}(c); \tag{2.11}$$

(4) $$\forall c > 0,\ \forall i, j \in \mathcal{R} : C(i) \geq C(j) \Rightarrow A_{ij}(c) = 1; \tag{2.12}$$

(5) $$\forall c > 0,\ \forall i, j \in \mathcal{R} : C(i) < C(j) \Rightarrow 0 < A_{ij}(c) < 1; \tag{2.13}$$

(6) $$\forall i, j \in \mathcal{R} : C(i) < C(j) \Rightarrow \lim_{c \downarrow 0} A_{ij}(c) = 0. \tag{2.14}$$

Then

$$\lim_{c \downarrow 0} q_i(c) = \begin{cases} \mid \mathcal{R}_{opt} \mid^{-1} & \text{if } i \in \mathcal{R}_{opt} \\ 0 & \text{elsewhere,} \end{cases} \tag{2.15}$$

where

$$q_i(c) = \lim_{k \to \infty} Pr\{X(k) = i \mid c\}, \tag{2.16}$$

and

$$Pr\{X(k) = j \mid X(k-1) = i;\ c\} = G_{ij}(c) A_{ij}(c),\ k = 1, 2, \ldots \tag{2.17}$$

Proof

According to Theorem 2 - Corollary II in chapter 15 of [Feller, 1950], a finite Markov chain[1] has a unique stationary distribution if the Markov chain is:

[1] i.e. a Markov chain defined on a finite set of configurations.

1. *irreducible*, i.e. if for all pairs of configurations (i, j) there is a positive probability of reaching j from i in a finite number of transitions:

$$\forall i, j \; \exists p : 1 \leq p < \infty \; \wedge \; (P^p)_{ij} > 0; \tag{2.18}$$

2. *aperiodic*, i.e. if for all configurations $i \in \mathcal{R}$, the greatest common divisor of all integers $n \geq 1$, such that

$$(P^n)_{ii} > 0 \tag{2.19}$$

is equal to 1.

According to the same theorem, the stationary distribution $\mathbf{q}$ of a finite, irreducible and aperiodic Markov chain is uniquely determined by the following equations:

$$\forall i : q_i > 0, \; \sum_i q_i = 1, \tag{2.20}$$

$$\forall i : q_i = \sum_j q_j P_{ji}. \tag{2.21}$$

We use condition (1) and the fact that $A_{ij}(c) > 0$ for all $i, j \in \mathcal{R}$ (conditions (4) and (5)) to establish irreducibility:

$$\begin{aligned}(P^p)_{ij}(c) &= \sum_{(l_1,\ldots,l_{p-1})} P_{il_1}(c) P_{l_1 l_2}(c) \ldots P_{l_{p-1}j}(c) \\ &= \sum_{(l_1,\ldots,l_{p-1})} G_{il_1}(c) A_{il_1}(c) \ldots G_{l_{p-1}j}(c) A_{l_{p-1}j}(c) \\ &\geq G_{i\lambda_1}(c) A_{i\lambda_1}(c) \ldots G_{\lambda_{p-1}j}(c) A_{\lambda_{p-1}j}(c) > 0.\end{aligned} \tag{2.22}$$

To establish aperiodicity, we use the fact that an irreducible Markov chain is aperiodic if [Romeo & Sangiovanni-Vincentelli, 1985]:

$$\exists i \in \mathcal{R} : P_{ii} > 0. \tag{2.23}$$

Clearly,

$$\forall c > 0 \; \exists i_c \in \mathcal{R}_{opt}, \; j_c \notin \mathcal{R}_{opt} : G_{i_c j_c}(c) > 0, \tag{2.24}$$

otherwise condition (1) would not be satisfied for $i \in \mathcal{R}_{opt}$, $j \notin \mathcal{R}_{opt}$. Using condition (5), we find that (2.24) implies:

$$\forall c > 0 \; \exists i_c, j_c \in \mathcal{R} : A_{i_c j_c}(c) < 1 \wedge G_{i_c j_c}(c) > 0. \tag{2.25}$$

Thus, using the fact that $A_{ij}(c) \leq 1$ for all $i, j \in \mathcal{R}$ (conditions (4) and (5)), we find

$$\sum_{l=1, l \neq i_c}^{|\mathcal{R}|} G_{i_c l}(c) A_{i_c l}(c) = \sum_{l=1, l \neq i_c, j_c}^{|\mathcal{R}|} G_{i_c l}(c) A_{i_c l}(c) + G_{i_c j_c}(c) A_{i_c j_c}(c) <$$

$$\sum_{l=1, l \neq i_c, j_c}^{|\mathcal{R}|} G_{i_c l}(c) + G_{i_c j_c}(c) = \sum_{l=1, l \neq i_c}^{|\mathcal{R}|} G_{i_c l}(c) \leq \sum_{l=1}^{|\mathcal{R}|} G_{i_c l}(c) = 1, \tag{2.26}$$

and, consequently,

$$P_{i_c i_c}(c) = 1 - \sum_{l=1, l \neq i_c}^{|\mathcal{R}|} G_{i_c l}(c) A_{i_c l}(c) > 0. \tag{2.27}$$

Next, we prove that the stationary distribution $\mathbf{q}(c)$ $(c > 0)$ is given by

$$\forall i \in \mathcal{R} : q_i(c) = \frac{A_{i_0 i}(c)}{\sum_{j \in \mathcal{R}} A_{i_0 j}(c)}, \tag{2.28}$$

for an arbitrary $i_0 \in \mathcal{R}_{opt}$.
First, we remark that $\mathbf{q}(c)$, as defined by (2.28), clearly satisfies (2.20). Let N denote the denominator in (2.28). We have that, for all i,

$$\sum_j q_j(c) P_{ji}(c) = \sum_{j \neq i, C(j) \leq C(i)} \frac{1}{N} A_{i_0 j}(c) G_{ji}(c) A_{ji}(c) +$$

$$\sum_{j \neq i, C(j) > C(i)} q_j(c) G_{ji}(c) A_{ji}(c) + q_i(c) P_{ii}(c) =$$

$$\sum_{j \neq i, C(j) \leq C(i)} \frac{1}{N} A_{i_0 i}(c) G_{ij}(c) + \sum_{j \neq i, C(j) > C(i)} q_j(c) G_{ij}(c) + q_i(c) P_{ii}(c) =$$

$$q_i(c) \sum_{j \neq i, C(j) \leq C(i)} G_{ij}(c) + \sum_{j \neq i, C(j) > C(i)} q_j(c) G_{ij}(c) + q_i(c) P_{ii}(c) \tag{2.29}$$

and

$$q_i(c)P_{ii}(c) =$$

$$q_i(c)\left(1 - \sum_{j\neq i, C(j)\leq C(i)} G_{ij}(c)A_{ij}(c) - \sum_{j\neq i, C(j)>C(i)} G_{ij}(c)A_{ij}(c)\right) =$$

$$q_i(c) - q_i(c)\sum_{j\neq i, C(j)\leq C(i)} G_{ij}(c) - \sum_{j\neq i, C(j)>C(i)} \frac{1}{N}A_{i_0 i}(c)G_{ij}(c)A_{ij}(c) =$$

$$q_i(c) - q_i(c)\sum_{j\neq i, C(j)\leq C(i)} G_{ij}(c) - \sum_{j\neq i, C(j)>C(i)} q_j(c)G_{ij}(c). \qquad (2.30)$$

Combining (2.29) and (2.30) yields

$$\forall i \in \mathcal{R} : \sum_j q_j(c)P_{ji}(c) = q_i(c). \qquad (2.31)$$

Using (2.28) and conditions (4) and (6), we have:

$$\lim_{c\downarrow 0} q_i(c) = \lim_{c\downarrow 0} \frac{A_{i_0 i}(c)}{\sum_{j\in\mathcal{R}} A_{i_0 j}(c)} = \frac{\chi_{\mathcal{R}_{opt}}(i)}{\sum_{j\in\mathcal{R}_{opt}} 1} = \mid \mathcal{R}_{opt} \mid^{-1} \chi_{\mathcal{R}_{opt}}(i), \quad (2.32)$$

where $\chi_{\mathcal{R}_{opt}}$ is the *characteristic function* of $\mathcal{R}_{opt}$. In general, the characteristic function χ_A of a set A is defined as follows:

$$\chi_A(i) = \begin{cases} 1 & \text{if } i \in A, \\ 0 & \text{elsewhere.} \end{cases} \qquad (2.33)$$

Finally, combining (2.8) and (2.32) yields

$$\lim_{c\downarrow 0}(\lim_{k\to\infty} Pr\{X(k) = i\}) = \mid \mathcal{R}_{opt} \mid^{-1} \chi_{\mathcal{R}_{opt}}(i), \qquad (2.34)$$

or

$$\lim_{c\downarrow 0}(\lim_{k\to\infty} Pr\{X(k) \in \mathcal{R}_{opt}\}) = 1. \qquad (2.35)$$

□

A few remarks are appropriate at this point.

First of all, we note that condition (2) can be replaced by [Lundy & Mees, 1986]

$$(2)' \quad \forall i \in \mathcal{R} : G_{ij} = \begin{cases} \mid \mathcal{R}_i \mid^{-1} & \text{if } j \in \mathcal{R}_i \\ 0 & \text{elsewhere,} \end{cases} \qquad (2.36)$$

in which case the stationary distributions are given by

$$q_i(c) = \frac{|\mathcal{R}_i| A_{i_0 i}(c)}{\sum_{j \in \mathcal{R}} |\mathcal{R}_j| A_{i_0 j}(c)}. \tag{2.37}$$

In other words, it suffices to demand that G is either symmetric or given by the uniform distribution on the neighbourhoods. If G is both, it can be shown that there exists an integer S satisfying

$$\forall i \in \mathcal{R} : |\mathcal{R}_i| = S \tag{2.38}$$

in the following way. Take an arbitrary $i, j \in \mathcal{R}$, then condition (1) implies that there are $\lambda_1, \ldots, \lambda_{p-1} \in \mathcal{R}$, such that

$$G_{\lambda_k \lambda_{k+1}} = G_{\lambda_{k+1} \lambda_k} > 0,\ k = 0, \ldots, p-1,\ \lambda_0 = 1,\ \lambda_p = j. \tag{2.39}$$

Hence, $|\mathcal{R}_{\lambda_k}| = |\mathcal{R}_{\lambda_{k+1}}|$, $k = 0, 1, \ldots, p-1$ and $|\mathcal{R}_i| = |\mathcal{R}_j|$.

Secondly, it is easily verified that in the initial formulation of the algorithm, conditions (3)-(6) are satisfied. The generation matrix is given by (2.36); thus, according to the previous remark, condition (2)$'$ is also satisfied. Consequently, one should only verify that condition (1) is satisfied. Note that condition (1) states that the Markov chain associated with the matrix $G(c)$ is itself irreducible. If this is not the case, i.e. if it is not possible for an arbitrary pair of configurations (i, j) to construct a finite sequence of transitions leading from i to j, we can still prove asymptotic convergence to a globally minimal configuration if the following condition is satisfied:

$$\begin{aligned}(1)' \quad \forall i \in \mathcal{R}\ \exists i_0 \in \mathcal{R}_{opt},\ p \geq 1, \lambda_0, \lambda_1, \ldots, \lambda_p \in \mathcal{R}\ (i = \lambda_0, i_0 = \lambda_p) :& \\ G_{\lambda_k \lambda_{k+1}}(c) > 0,\ k = 0, 1, \ldots, p-1,& \end{aligned} \tag{2.40}$$

i.e. if for an arbitrary configuration i it is possible to construct a finite sequence of transitions leading from i to a globally minimal configuration i_0 (this situation occurs when we study the job shop scheduling problem in section 4.3). Note that by replacing j by i_0 in (2.22) we find that satisfaction of condition (1)$'$ implies that a similar condition is satisfied for the matrix $P(c)$.

To prove asymptotic convergence in this case we introduce the notions of a closed set and of recurrent and transient configurations

[Feller, 1950]. A *closed set* S is a set of configurations such that $P_{ij} = 0$ whenever $i \in S$ and $j \notin S$ - in the case of an irreducible chain the only closed set is the set of all configurations. A configuration i is called *recurrent* if the probability that the Markov chain ever returns to i is equal to 1, otherwise it is called *transient*. In every chain the recurrent configurations can be uniquely divided into closed sets $S_1, S_2, \ldots, S_K$ such that from any configuration of a given set all configurations of that set and no other can be reached.[2] In addition to the closed sets there is a set T of transient configurations from which configurations in the closed sets can be reached (but not *vice versa*). Now consider the sequence of configurations constituting the Markov chain associated with $P(c)$. There are two possibilities: either the Markov chain starts in a transient configuration or it does not. In the latter case, the configurations constituting the Markov chain all belong to the same closed set S_k ($k \in \{1, \ldots, K\}$) - the Markov chain can then be considered as a Markov chain with transition matrix $P_k(c)$, where $P_k(c)$ is obtained from $P(c)$ by deleting the rows and columns from $P(c)$ corresponding to configurations not belonging to S_k. Note that this Markov chain is aperiodic (this can be shown in the same way as for the Markov chain associated with $P(c)$) and irreducible (because of the properties of S_k); furthermore, S_k **contains at least one globally minimal configuration**. The latter observation is an immediate consequence of (2.40) and the definition of a closed set. In other words, the proof of theorem 2.1 can be repeated with $\mathcal{R}$ replaced by S_k.

If the Markov chain starts in a transient configuration, it will eventually 'land' [Feller, 1950] in a closed set S_k, $k \in \{1, \ldots, K\}$, though it is not *a priori* known which one. The line of reasoning described above can then be applied again.

We can make the preceding arguments more precise by introducing the notion of a *stationary matrix* Q, whose elements q_{ij} are defined by

$$q_{ij} = \lim_{k \to \infty} Pr\{X(k) = j | X(0) = i\}. \tag{2.41}$$

---

[2] We say that a configuration j can be *reached* from a configuration i if $\exists n : 1 \leq n < \infty \wedge (P^n)_{ij} > 0$.

Note that for an irreducible Markov chain, q_{ij} does not depend on i (cf. (2.7)). Because the Markov chain associated with $P(c)$ is aperiodic, we can use the results in [Feller, 1950, chapter 15, sections 6-8] to obtain

$$q_{ij} =$$

$$\begin{cases} 0 & \text{if } j \in T \text{ or } i \in S_k,\ j \notin S_k, \text{ for some } k \in \{1,\ldots,K\}, \\ \frac{A_{i_0 j}(c)}{\sum_{l \in S_k} A_{i_0 l}(c)} & \text{if } i,j \in S_k \text{ for some } k \in \{1,\ldots,K\}, \\ x_{ik} \frac{A_{i_0 j}(c)}{\sum_{l \in S_k} A_{i_0 l}(c)} & \text{if } i \in T,\ j \in S_k \text{ for some } k \in \{1,\ldots,K\}, \end{cases} \tag{2.42}$$

where x_{ik} is the probability that the Markov chain, starting from the transient configuration i, eventually reaches the closed set S_k.
From (2.42) we obtain, for a recurrent configuration $j \in S_k$,

$$0 \leq \lim_{k\to\infty} Pr\{X(k) = j\} = \sum_{i \in \mathcal{R}} Pr\{X(0) = i\} \cdot q_{ij}$$

$$= \left(\sum_{i \in T} Pr\{X(0) = i\} \cdot x_{ik} + \sum_{i \in S_k} Pr\{X(0) = i\} \right) \cdot \frac{A_{i_0 j}(c)}{\sum_{l \in S_k} A_{i_0 l}(c)}$$

$$\leq \frac{A_{i_0 j}(c)}{\sum_{l \in S_k} A_{i_0 l}(c)}. \tag{2.43}$$

Using conditions (4) and (6) we find

$$\lim_{c \downarrow 0} \frac{A_{i_0 j}(c)}{\sum_{l \in S_k} A_{i_0 l}(c)} = 0, \tag{2.44}$$

if $j \in S_k$, $j \notin \mathcal{R}_{opt}$. Consequently, $\lim_{c\downarrow 0}(\lim_{k\to\infty} Pr\{X(k) = j\}) = 0$ for any transient or non-globally recurrent configuration j. In other words,

$$\lim_{c \downarrow 0}(\lim_{k\to\infty} Pr\{X(k) \in \mathcal{R}'_{opt}\}) = 1, \tag{2.45}$$

where $\mathcal{R}'_{opt}$ denotes the non-empty set of globally minimal recurrent configurations.
In chapter 4, where applications of simulated annealing to several

combinatorial optimization problems are discussed, we prove that either condition (1) or condition (1)′ is satisfied for each of these problems.

Finally, we remark that theorem 2.1 does not pertain to the inhomogeneous algorithm, because it is implicitly assumed that an infinite number of transitions is necessary to reach the stationary distribution of each Markov chain. More precisely, it can be shown that the number of transitions necessary to approach the stationary distribution arbitrarily closely is of exponential order (see section 2.4).

We now discuss briefly necessary and sufficient conditions for the inhomogeneous algorithm to converge to a global minimum. The inhomogeneous algorithm is described by an inhomogeneous Markov chain, whose transition matrix $P(k)$ $(k = 1, 2, \ldots)$ is given by

$$P_{ij}(k) = \begin{cases} G_{ij}(c_k)A_{ij}(c_k) & \forall j \neq i \\ 1 - \sum_{l=1, l\neq i}^{|\mathcal{R}|} G_{il}(c_k)A_{il}(c_k) & j = i, \end{cases} \tag{2.46}$$

where the sequence $\{c_k\}$, $k = 1, 2, \ldots$ denotes the sequence of values of the control parameter. A number of authors derive sufficient conditions for asymptotic convergence of the inhomogeneous algorithm to a global minimum, notably Geman & Geman [1984], Anily & Federgruen [1986], Mitra, Romeo & Sangiovanni-Vincentelli [1985] and Gelfand & Mitter [1985]. These derivations are all based on ergodicity theorems for inhomogeneous Markov chains [Seneta, 1981].
Necessary and sufficient conditions are derived by Hajek [1986]. Hajek's result is restricted to the case where the generation matrix is independent of c and the acceptance matrix is given by (2.6) (the Metropolis criterion). In order to formulate this result, we need two definitions:

Definition 2.1 *[Hajek, 1986]*
A configuration j is called reachable at height L *from a configuration i, if the following holds:*

$$\exists p \geq 1,\ \lambda_0, \lambda_1, \ldots, \lambda_p \in \mathcal{R}\ (i = \lambda_0,\ j = \lambda_p):$$

$$G_{\lambda_k \lambda_{k+1}} > 0,\ k = 0, 1, \ldots, p-1 \tag{2.47}$$

and

$$C(\lambda_k) \leq L,\ k = 0, 1, \ldots, p. \tag{2.48}$$

Definition 2.2 *[Hajek, 1986]*
The depth of a local minimum i *is defined as the smallest number* $d(i)$ *such that there is a configuration* j *with* $C(j) < C(i)$ *reachable at height* $C(i) + d(i)$ *from* i. *If* i *is a global minimum, then by definition* $d(i) = +\infty$.

The reader is referred to [Kern, 1986b] for a detailed discussion of the notion depth. In particular, Kern considers the maximum depth D of all local minima (cf. theorem 2.2). For several combinatorial optimization problems, upper bounds on D are derived and it is shown that the computation of D cannot be done in polynomial time, unless $\mathcal{P} = \mathcal{NP}$.
Hajek's result can be formulated as follows:

Theorem 2.2 *[Hajek, 1986]*
Suppose that the transition matrix is given by (2.46), where $A(c_k)$ *is given by (2.6) (the Metropolis criterion) and* $G(c_k) = G$ *satisfies the following two conditions:*

1. *condition (1) of theorem 2.1;*
2. *for any real number* L *and any two configurations* i *and* j, i *is reachable at height* L *from* j *if and only if* j *is reachable at height* L *from* i.

Assume furthermore that the sequence $\{c_k\}$, $k = 1, \ldots$ *satisfies the following two conditions:*

$$1. \quad \lim_{k\to\infty} c_k = 0; \tag{2.49}$$

$$2. \quad c_k \geq c_{k+1},\ k = 1, 2, \ldots. \tag{2.50}$$

Then

$$\lim_{k\to\infty} Pr\{X(k) \in \mathcal{R}_{opt}\} = 1, \tag{2.51}$$

if and only if

$$\sum_{k=1}^{\infty} \exp\left(-\frac{D}{c_k}\right) = \infty, \tag{2.52}$$

where D is given by

$$D = \max \{d(i) \mid i \notin \mathcal{R}_{opt}, i \text{ is a local minimum}\}. \tag{2.53}$$

Again, a few remarks on this result are in order. First of all, we note that the term $\exp(-\frac{D}{c_k})$ relates to the probability of accepting a transition which corresponds to an increase in the cost function of value D.
Secondly, if c_k is of the form

$$c_k = \frac{\Gamma}{\log k}, \; k = 1, 2, \ldots, \tag{2.54}$$

for some constant Γ, theorem 2.2 implies that (2.51) holds if and only if $\Gamma \geq D$. Note that (2.54) is the expression for c_k which results from the necessary conditions derived in [Geman & Geman, 1984; Anily & Federgruen, 1986; Mitra, Romeo & Sangiovanni-Vincentelli, 1985; Gelfand & Mitter, 1985].
Finally, we remark that theorem 2.2 can also be applied to the homogeneous algorithm. If $\{c_k\}$, $k = 1, 2, \ldots$ is the sequence of values of the control parameter and L_k the length of the k-th Markov chain of the homogeneous algorithm, we define the sequence $\{\gamma_l\}$, $l = 1, 2, \ldots$ as follows:

$$\gamma_l = c_1, \; 1 \leq l \leq L_1 \tag{2.55}$$

and

$$\gamma_l = c_k, \; L_{k-1} < l \leq L_k, \; k = 2, \ldots. \tag{2.56}$$

The homogeneous algorithm can now be thought of as an inhomogeneous algorithm with sequence of control-parameter values $\{\gamma_l\}$, $l = 1, 2, \ldots$.

2.4 Asymptoticity revisited

In the previous section conditions are derived for the homogeneous as well as the inhomogeneous algorithm to converge in probability to the set of globally minimal configurations. In this section, we show that the aforementioned conditions imply that both algorithms must be allowed unlimited computation times. In addition, we derive results

indicating that exponential-time behaviour is to be expected if the asymptotic behaviour is to be approximated arbitrarily closely.
Let us first discuss the consequences of the results of the previous section for the computation time of the homogeneous algorithm. In this case, the requirement that the computation time be unlimited is an immediate consequence of (2.7), which implies that each individual Markov chain is of infinite length. At this stage it is appropriate to make two further remarks on the speed of convergence of the probability distribution of the configurations to the stationary distribution. Both remarks relate to the distance between the stationary distribution and the probability distribution of the configurations after a **finite** number of steps.

1. Consider an arbitrary irreducible, aperiodic and finite Markov chain with transition matrix P and stationary distribution $\mathbf{q}$. Let the vector $\mathbf{a}(k)$ denote the probability distribution of the configurations after k transitions. Hence, $\mathbf{a}(0)$ denotes the initial probability distribution and

$$\mathbf{a}(k) = (P^k)^T \mathbf{a}(0), \quad \mathbf{q} = \lim_{k \to \infty} \mathbf{a}(k). \tag{2.57}$$

We are interested in $||\mathbf{a}(k) - \mathbf{q}||_1$ for any finite k and in order to say something about this quantity we use the following two theorems from Seneta [1981] (theorem 2.3 is an abridged version of the Perron-Frobenius theorem for primitive matrices).

Theorem 2.3 *[Seneta, 1981]*
Let P be a non-negative primitive[3] matrix. Then there exists a real-valued eigenvalue r of P, which has the following properties:

(a) r is the largest eigenvalue of P ($r > |\lambda|$, for any eigenvalue $\lambda \neq r$);

(b) with r can be associated strictly positive left and right eigenvectors, which are unique to constant multiples;

(c) r lies between the smallest and largest row sums of P.

[3] A *primitive* matrix P is a matrix for which there exists a positive integer M such that P^M is strictly positive.

Theorem 2.4 *[Seneta, 1981]*
Suppose the distinct eigenvalues of a primitive matrix P are $r, \lambda_2, \ldots, \lambda_t$, where $r > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_t|$. Let $s_2 = m_2 - 1$, where m_2 is the multiplicity of λ_2. Then we have, as $k \to \infty$

$$P^k = r^k \mathbf{w}\mathbf{v}^T + O(k^s \cdot |\lambda_2|^k) \tag{2.58}$$

elementwise, where $\mathbf{w}$ and $\mathbf{v}$ are any positive right and left eigenvectors, respectively, corresponding to r guaranteed by theorem 2.3, providing they are normed so that $\mathbf{v}^T\mathbf{w} = 1$.

Now suppose the matrix P is primitive and stochastic, i.e. satisfies $\sum_j P_{ij} = 1$ for all i. According to theorem 2.3, the largest eigenvalue r is equal to 1. Putting $\mathbf{1}$ for the vector with unity in each position, we find $P\mathbf{1} = \mathbf{1}$ by the stochasticity of P. Hence, $\mathbf{1}$ can be taken as the right eigenvector in theorem 2.4 and we have, as $k \to \infty$

$$P^k = \mathbf{1} \cdot \mathbf{v}^T + O(k^s \cdot |\lambda_2|^k) \tag{2.59}$$

elementwise, where $\mathbf{v}$ is the positive left eigenvector of P satisfying $\mathbf{v}^T \cdot \mathbf{1} = \sum_i v_i = 1$. According to (2.20) and (2.21), $\mathbf{v}$ is the unique stationary distribution $\mathbf{q}$ of the Markov chain associated with P.
Consequently, for $k \to \infty$

$$P^k = \mathbf{1} \cdot \mathbf{q}^T + O(k^s \cdot |\lambda_2|^k) \tag{2.60}$$

elementwise, and

$$||\mathbf{a}^T(k) - \mathbf{q}^T||_1 = ||\mathbf{a}^T(0)P^k - \mathbf{q}^T||_1 = O(k^s \cdot |\lambda_2|^k). \tag{2.61}$$

Equation (2.61) bears on the homogeneous algorithm, because the transition matrix $P(c)$ is both stochastic and primitive. The latter follows from the fact that an irreducible aperiodic matrix is primitive [Seneta, 1981]. From (2.61) it is apparent that the speed of convergence to the stationary distribution is determined by the 'second largest' eigenvalue $\lambda_2(c)$ of the transition matrix $P(c)$. However, for large matrices, such as the ones that occur in problems to which simulated annealing is applied, it is virtually impossible to compute λ_2.

2. The following theorem provides an upper bound to the norm in (2.61).

Theorem 2.5 *[Seneta, 1981]*
Suppose the $n \times n$-matrix P is a stochastic and regular[4] matrix. Let $\mathbf{q}$ be the unique stationary distribution associated with P and let $\mathbf{a}$ be an arbitrary probability vector. Then for $k \geq \varphi(n)$, where $\varphi(n)$ is the number of distinct regular $n \times n$-matrices with elements in $\{0,1\}$,

$$||\mathbf{a}^T P^k - \mathbf{q}^T||_1 \leq 2\beta^{(k/\varphi(n))-1}, \tag{2.62}$$

where β is a constant independent of P and $\mathbf{a}$, $0 \leq \beta < 1$.

To apply this bound to the homogeneous algorithm, we remark that the transition matrix $P(c)$ is stochastic and irreducible and aperiodic and thus regular. Putting $\mathbf{a} = \mathbf{a}(0)$, we find

$$||\mathbf{a}^T(0)P^k(c) - \mathbf{q}^T(c)||_1 \leq 2\beta^{(k/\varphi(|\mathcal{R}|))-1}. \tag{2.63}$$

Thus, the following condition is sufficient to ensure $||\mathbf{a}^T(k) - \mathbf{q}^T(c)||_1 < \epsilon$ for some small positive number ϵ:

$$k > \varphi(|\mathcal{R}|)\left(1 + \frac{\ln\frac{\epsilon}{2}}{\ln\beta}\right). \tag{2.64}$$

According to Isaacson & Madsen [1974], we may take $\varphi(|\mathcal{R}|) = |\mathcal{R}|^2 - 3\cdot|\mathcal{R}| + 3$. Hence, (2.64) implies that to approximate the stationary distribution arbitrarily closely, we need an amount of transitions at least quadratic in the number of configurations. Since this number is usually some exponential function of the size of the problem, (2.64) would typically result in an exponential-time algorithm.

---

[4] A *regular* matrix is a stochastic matrix, whose essential indices form a single aperiodic class [Seneta, 1981]. It would lead us beyond the scope of this tract to explain this notion in more detail. It suffices to remark that a Markov chain with a regular transition matrix has a unique stationary distribution and that an irreducible, aperiodic matrix is regular [Seneta, 1981].

For the inhomogeneous algorithm, unlimited computation time is a consequence of (2.49), (2.50) and (2.52): from (2.49) and (2.50) we conclude that all c_k are non-negative, whereas (2.52) implies that there can be no integer K, such that $c_k = 0$ for $k \geq K$ (unless $D = 0$, in which case there are no non-global local minima, so that even iterative improvement always finds the global minimum). Hence, $c_k > 0$ for all k and the limit in (2.49) is attained only after an infinite number of transitions. Thus, the corresponding inhomogeneous Markov chain is of infinite length.

For the inhomogeneous algorithm, results similar to (2.61) have been obtained independently by a number of authors, notably Anily & Federgruen [1986], Gidas [1985] and Mitra, Romeo & Sangiovanni-Vincentelli [1985]. As an example we discuss the result of Mitra, Romeo & Sangiovanni-Vincentelli [1985] .

Suppose the sequence $\{c_k\}$ $(k = 1, 2, \ldots)$ is given by (cf. (2.54))

$$c_k = \frac{r \cdot \Delta}{\log k}, \tag{2.65}$$

where Δ is the maximum difference in cost between any two configurations $i \in \mathcal{R}, j \in \mathcal{R}_i$ for which $C(j) > C(i)$ and r is given by

$$r = \min_{i \in \mathcal{R} \setminus \mathcal{R}_{max}} \max_{j \in \mathcal{R}} d(i, j). \tag{2.66}$$

Here, $d(i, j)$ is the minimal number of transitions to reach j from i and $\mathcal{R}_{max}$ is the set of locally maximal configurations. Note that r is an integer such that there is at least one non-locally maximal configuration from which any other configuration can be reached in no more than r transitions. If π is the uniform probability distribution on the set of globally minimal configurations, then

$$\|\mathbf{a}^T(k) - \pi^T\|_1 = O\left(k^{-\min(a,b)}\right), \tag{2.67}$$

where a and b are given by

$$a = \frac{1}{r}(\min_{i \in \mathcal{R}} \min_{j \in \mathcal{R}_i} G_{ij})^r, \tag{2.68}$$

and

$$b = \frac{\delta}{r \cdot \Delta}, \tag{2.69}$$

respectively, and δ is the difference between C_{opt} and the next-to-least cost value. This bound is rather poor in the sense that if one works it out for a particular problem one typically finds that the time required for good accuracy is larger than the number of configurations. For the n-city travelling salesman problem, for example, one cannot do much better than estimating r by n. Under the assumption that the G-matrix is given by a uniform distribution on *2-opt neighbourhoods* (see section 4.2) (2.68) leads to

$$a \simeq \frac{1}{n}\left(\frac{2}{n(n-1)}\right)^n \tag{2.70}$$

and, using $a \ll b$, we find that for some small number $\epsilon > 0$

$$||\mathbf{a}^T(k) - \pi^T||_1 < \epsilon \tag{2.71}$$

implies that $k \simeq O\left(\epsilon^{-n^{2n+1}}\right)$, whereas the number of configurations is $O(n!)$. Thus, an enumeration of all configurations would take less computation time than the inhomogeneous algorithm.

Summarizing we have that both algorithms behave as optimization algorithms if they are allowed an infinite number of transitions. Of course, the latter can never be realized in any practical implementation of such an algorithm. Furthermore, if the asymptotic behaviour is to be approximated arbitrarily closely, we can only derive exponential upper bounds on the number of elementary operations taken by the algorithms. Such operations are the transitions in the Markov chains for the homogeneous algorithm and the steps in the control parameter for the inhomogeneous algorithm.

In chapter 3 we describe how to imitate the asymptotic behaviour of the homogeneous algorithm in polynomial time, with as an inevitable result the loss of any guarantee with respect to the optimality of the configuration returned by the algorithm.

Chapter 3

Finite-time behaviour of simulated annealing

3.1 Introduction

In this chapter, we discuss the behaviour of the homogeneous algorithm in finite time on the basis of the notion of a *cooling schedule*, a set of parameters determining the finite-time behaviour of the algorithm. These parameters are chosen so as to imitate the asymptotic behaviour of the homogeneous algorithm in polynomial time, thereby losing any guarantees with respect to the optimality of the configuration returned by the algorithm. We do not describe any approximations to the asymptotic behaviour of the inhomogeneous algorithm. Such approximations are not reported in the literature, which is probably due to the fact that in practice it is virtually impossible to give accurate approximations to the constant Γ in (2.54). One resorts to conservative estimates, see for example [Kern, 1986b], which lead to unnecessarily slow convergence of the algorithm [Geman & Geman, 1984; Lundy & Mees, 1986].

3.2 An introduction to cooling schedules

We recall from the previous chapter that the asymptotic behaviour of the homogeneous algorithm is characterized by

1. an infinite number of transitions for each value of the control parameter;
2. an (infinite) sequence of values of the control parameter $\{c_k\}$, $k = 1, \ldots$, satisfying $\lim_{k\to\infty} c_k = 0$ and $c_k > 0$ for all k.

Thus, in any finite-time implementation of the algorithm, the following parameters should be specified:

1. a **finite** number of transitions for each value of the control parameter or, in other words, a finite length of each homogeneous Markov chain;
2. a **finite** sequence of values of the control parameter, more specifically:
   (a) an initial value of the control parameter c_1;
   (b) a rule for changing the current value of the control parameter into the next one;
   (c) a final value of the control parameter (a stop criterion).

A choice for these parameters is referred to as a *cooling schedule* or an *annealing schedule*. In this section, we first discuss some general arguments on which the construction of a cooling schedule is usually based and next some approaches from the literature to the problem of determining a cooling schedule. The discussion is confined to generation and acceptance matrices as given by (2.5) and (2.6), respectively, i.e. only the annealing algorithm in its original formulation is considered.

Central in the construction of many cooling schedules is the concept of *quasi-equilibrium*: if L_k is the length of the k-th Markov chain and $\mathrm{a}(l, c_k)$ denotes the probability distribution of the configurations after l transitions of the k-th Markov chain, then the homogeneous algorithm is said to be in quasi-equilibrium at c_k, if "$\mathrm{a}(L_k, c_k)$ **is close to** $\mathrm{q}(c_k)$". The precise statement of this proximity is one of the major points differentiating one cooling schedule from the other - from section 2.4 we know that exponential-time behaviour is to be expected if we specify quasi-equilibrium by $||\mathrm{a}^T(L_k, c_k) - \mathrm{q}^T(c)||_1 < \epsilon$, for some

small real number ϵ.
The actual construction of a cooling schedule is usually based on the following arguments.

- For $c \to \infty$, the stationary distribution is given by the uniform distribution on the set of configurations $\mathcal{R}$, which follows immediately from (2.28), if we substitute the Metropolis criterion (2.6) for the acceptance matrix and take the limit $c \to \infty$. Initially, quasi-equilibrium can therefore be achieved by choosing the initial value of c, c_1, such that virtually all transitions are accepted. For in that case all configurations occur with equal probability, which corresponds to the aforementioned uniform distribution and thus to $\mathrm{q}(\infty)$.

- A *stop criterion* is usually based on the argument that execution of the algorithm can be terminated if the observed improvement in cost over a number of consecutive Markov chains is small.

- The length L_k of the k-th Markov chain and the transformation rule for changing c_k into c_{k+1} are related through the concept of quasi-equilibrium. L_k is usually determined by specifying more precisely the meaning of "$\mathrm{a}(L_k, c_k)$ is close to $\mathrm{q}(c_k)$". Concerning the transformation rule (usually a *decrement rule*), it is intuitively clear that large decrements in c_k will make it necessary to attempt more transitions at the new value of the control parameter, c_{k+1}, to restore quasi-equilibrium at c_{k+1}. For, given quasi-equilibrium at c_k, the larger the decrement in c_k, the larger the difference between $\mathrm{q}(c_k)$ and $\mathrm{q}(c_{k+1})$ and the longer it takes to establish quasi-equilibrium at c_{k+1}. Thus, there is a trade-off between fast decrement of c_k and small values for L_k. Usually, one opts for small decrements in c_k (to avoid extremely long chains), but alternatively, one could use large values for L_k in order to be able to make large decrements in c_k.

The search for adequate cooling schedules has been addressed in many papers during the last few years. In this section we briefly discuss the schedule proposed by Kirkpatrick, Gelatt & Vecchi [1983], and similar schedules. These conceptually simple schedules are all based on

empirical rules rather than on theoretically based choices. A more elaborate and theoretically based schedule is discussed in section 3.3. The discussion of the schedule of Kirkpatrick, Gelatt & Vecchi and similar schedules is gathered around the four aforementioned parameters ((1) and (2a)-(2c)).

- initial value of the control parameter

As already stated, c_1 should be such that virtually all transitions are accepted, i.e. such that

$$\exp(-\Delta C_{ij}/c_1) \simeq 1 \tag{3.1}$$

for almost all cost-increasing transitions. If the *acceptance ratio* χ is defined as the ratio between the number of accepted transitions and the number of proposed transitions, then (3.1) translates into

$$c_1 = \frac{\overline{\Delta C}^{(+)}}{\ln(\chi_0^{-1})}, \tag{3.2}$$

where χ_0 is a required initial acceptance ratio (close to 1, say 0.9) and $\overline{\Delta C}^{(+)}$ the average increase in cost over a number of (arbitrarily chosen) transitions.
Equation (3.2) or a similar one is used in virtually all cooling schedules in the literature; see for instance [Aarts & Van Laarhoven, 1985a, 1985b; Golden & Skiscim, 1986; Johnson, Aragon, McGeoch & Schevon, 1987; Leong & Liu, 1985; Leong, Wong & Liu, 1985; Lundy & Mees, 1986; Morgenstern & Shapiro, 1986; Otten & Van Ginneken, 1984; Skiscim & Golden, 1983].

- final value of the control parameter

Execution of the algorithm is usually terminated if the last configurations of consecutive Markov chains are identical for a number of chains; see for instance [Kirkpatrick, Gelatt & Vecchi, 1983; Morgenstern & Shapiro, 1986; Sechen & Sangiovanni-Vincentelli, 1985].

- length of Markov chains

The length of Markov chains is usually based on the intuitive argument that for each value c_k of the control parameter, a certain amount

of computational effort should be spent to restore quasi-equilibrium. Since accepted transitions are the only ones that contribute to a possible evolution of the algorithm towards quasi-equilibrium, L_k is determined such that the number of accepted transitions is at least some fixed number η_{min}. However, since transitions are accepted with decreasing probability, one would obtain $L_k \to \infty$ for $c_k \downarrow 0$. Consequently, a ceiling is put on L_k by some constant $\overline{L}$ to avoid extremely long Markov chains for low values of c_k; see for instance [Johnson, Aragon, McGeoch & Schevon, 1987; Kirkpatrick, Gelatt & Vecchi, 1983; Leong & Liu, 1985; Leong, Wong & Liu, 1985; Morgenstern & Shapiro, 1986].

- decrement of the control parameter

As mentioned before, one usually opts for small changes in the value of the control parameter and a frequently used decrement rule is given by

$$c_{k+1} = \alpha \cdot c_k, \; k = 1, 2, \ldots, \tag{3.3}$$

where α is a constant smaller than 1, typically between 0.9 and 0.99; see for instance [Bonomi & Lutton, 1984, 1986; Burkard & Rendl, 1984; Johnson, Aragon, McGeoch & Schevon, 1987; Leong & Liu, 1985; Leong, Wong & Liu, 1985; Lundy & Mees, 1986; Morgenstern & Shapiro, 1986; Sechen & Sangiovanni-Vincentelli, 1985].

3.3 A polynomial-time cooling schedule

In this section we discuss the cooling schedule also described in [Aarts & Van Laarhoven, 1985a]. We emphasize once more that this cooling schedule leads to polynomial-time execution of the algorithm on the one hand, but on the other hand precludes any guarantee for the proximity of the final configuration to a globally minimal one[1]. The discussion is again gathered around the four parameters mentioned in the previous section.

[1] To the best of our knowledge, the only reference to such a guarantee in the literature is given by Sasaki & Hajek [1986], who derive an upper bound on the aforementioned proximity for the *maximum matching problem*.

- initial value of the control parameter

The initial value of the control parameter is found by a slight refinement of (3.2) in that we distinguish between cost-increasing and cost-decreasing transitions. Suppose that at a certain value c of the control parameter m_1 transitions have been generated for which $\Delta C_{ij} \leq 0$ and m_2 for which $\Delta C_{ij} > 0$. Let $\overline{\Delta C}^{(+)}$ be the average increase in cost over the m_2 transitions. Then the expected acceptance ratio χ is approximately given by

$$\chi \simeq \frac{m_1 + m_2 \cdot \exp\left(\frac{-\overline{\Delta C}^{(+)}}{c}\right)}{m_1 + m_2}, \tag{3.4}$$

which can be rewritten as

$$c = \frac{\overline{\Delta C}^{(+)}}{\ln\left(\frac{m_2}{m_2 \cdot \chi - m_1(1-\chi)}\right)}. \tag{3.5}$$

We now choose for c_1 the value c_0, which is determined as follows. First, c_0 is given some arbitrary value. Next, the algorithm is executed for a fixed number of transitions, say m_0, and after each transition (3.5), with χ set to χ_0, is used to update the current value of c_0 (m_1 (m_2) now corresponds to the number of cost-decreasing (increasing) transitions obtained so far; eventually, $m_0 = m_1 + m_2$). Numerical experience shows that in this way c_0 reaches a stable value after a small number of transitions. This value is then taken as the initial value of the control parameter.

- decrement rule for the control parameter

We recall from section 3.2 that if the control parameter is decreased in small steps, the stationary distributions of successive Markov chains are close to each other. We may then expect that, after decreasing c_k to c_{k+1}, a small number of transitions suffices to let the probability distribution of the configurations approach the new stationary distribution $\mathbf{q}(c_{k+1})$ (this is, of course, under the assumption that the probability distribution of the configurations is close to $\mathbf{q}(c_k)$ before the decrement of c_k to c_{k+1} and eventually under the assumption that the probability distribution of the configurations is close to $\mathbf{q}(c_1)$ before the decrement of c_1 to c_2; the latter is taken care of by the choice

of c_1).
On the other hand, if we allow large steps in c (and thus fewer Markov chains), we may expect that it will take more transitions to restore quasi-equilibrium after each decrement of c. In other words, there is a trade-off between a large amount of short Markov chains and a small amount of long chains. We readdress this trade-off in section 5.5, where we discuss it from a Bayesian point of view. For the time being, we take the first alternative as a starting-point, i.e. we want the stationary distributions for two successive values of the control parameter to be 'close':

$$\forall i \in \mathcal{R} : \frac{1}{1+\delta} < \frac{q_i(c_k)}{q_i(c_{k+1})} < 1+\delta, \; k = 1, 2, \ldots, \tag{3.6}$$

for some 'small' real number δ, hereinafter referred to as the *distance parameter*. The actual choice of δ is extensively discussed in chapter 4. The following theorem provides a sufficient condition to satisfy (3.6).

Theorem 3.1 *[Aarts & Van Laarhoven, 1985a]*
If

$$\forall i \in \mathcal{R} : \frac{\exp\left(-\frac{\Delta C_{i_0 i}}{c_k}\right)}{\exp\left(-\frac{\Delta C_{i_0 i}}{c_{k+1}}\right)} < 1+\delta, \; k = 1, 2, \ldots, \tag{3.7}$$

for some arbitrary $i_0 \in \mathcal{R}_{opt}$ and for $c_{k+1} < c_k$, then the inequalities of (3.6) are satisfied.

Proof

First, we remark that if the acceptance matrix is given by (2.6) (the Metropolis criterion), the stationary distribution is, according to (2.28), given by

$$\forall i \in \mathcal{R} : \; q_i(c_k) = q_0(c_k) \cdot \exp\left(-\Delta C_{i_0 i}/c_k\right) \tag{3.8}$$

for some arbitrary $i_0 \in \mathcal{R}_{opt}$, where the normalization factor $q_0(c_k)$ is given by

$$q_0(c_k) = \left(\sum_{j \in \mathcal{R}} \exp\left(-\Delta C_{i_0 j}/c_k\right) \right)^{-1}. \tag{3.9}$$

From (3.7) we deduce, using $c_{k+1} < c_k$,

$$\sum_{j \in \mathcal{R}} \exp\left(-\Delta C_{i_0 j}/c_{k+1}\right)$$

$$< \sum_{j \in \mathcal{R}} \exp\left(-\Delta C_{i_0 j}/c_k\right) < (1+\delta) \sum_{j \in \mathcal{R}} \exp\left(-\Delta C_{i_0 j}/c_{k+1}\right), \qquad (3.10)$$

whence

$$q_0(c_{k+1}) > q_0(c_k) > \frac{1}{1+\delta} q_0(c_{k+1}). \qquad (3.11)$$

The second inequality in (3.6) now immediately follows from (3.7), (3.8) and the first inequality in (3.11), whereas the first inequality in (3.6) immediately follows from (3.8), the second inequality in (3.11) and the fact that

$$\forall i \in \mathcal{R} : \exp\left(-\frac{\Delta C_{i_0 i}}{c_{k+1}}\right) < \exp\left(-\frac{\Delta C_{i_0 i}}{c_k}\right). \qquad (3.12)$$

□

Equation (3.7) can be rewritten as

$$\forall i \in \mathcal{R} : c_{k+1} > \frac{c_k}{1 + \frac{c_k \cdot \ln(1+\delta)}{\Delta C_{i_0 i}}}, \; k = 1, 2, \ldots \qquad (3.13)$$

We make a slight simplification by restricting the condition of (3.13) to the set of configurations that 'most probably' occur during the generation of the k-th Markov chain. This set $\mathcal{R}_{c_k}$ is determined by recording the cost values of the configurations $X(1), \ldots, X(L_k) \in \mathcal{R}$ actually occurring during the generation of the k-th Markov chain and assuming that the probability distribution of the cost values of the k-th Markov chain can be approximated by a normal distribution[2], with mean μ_k given by

$$\mu_k = \mu(c_k) = \frac{1}{L_k} \sum_{j=1}^{L_k} C(X(j)) \qquad (3.14)$$

[2]Such an assumption is often made in the literature and supported by computational evidence; see for example [Aarts, Korst & Van Laarhoven, 1988; Hajek, 1985; White, 1984].

and variance σ_k^2 by

$$\sigma_k^2 = \sigma^2(c_k) = \frac{1}{L_k}\sum_{j=1}^{L_k} C^2(X(j)) - \mu_k^2, \tag{3.15}$$

respectively. We now define $\mathcal{R}_{c_k}$ as

$$\mathcal{R}_{c_k} = \{i \in \mathcal{R} | \Delta C_{i0i} \leq \mu_k - C_{opt} + 3\sigma_k\}, \tag{3.16}$$

so that, due to the properties of a normal distribution, a configuration occurring during the generation of the k-th Markov chain has a probability of close to 1 to belong to $\mathcal{R}_{c_k}$. Next, we replace the condition of (3.13) by the following condition

$$c_{k+1} > \frac{c_k}{1 + \frac{c_k \cdot \ln(1+\delta)}{\mu_k - C_{opt} + 3\sigma_k}}. \tag{3.17}$$

Clearly, the condition of (3.17) is a stronger condition for the configurations in $\mathcal{R}_{c_k}$. Finally, since for most optimization problems C_{opt} is not known, we replace $\mu_k - C_{opt} + 3\sigma_k$ by $3\sigma_k$ to obtain the following decrement rule:

$$c_{k+1} = \frac{c_k}{1 + \frac{c_k \cdot \ln(1+\delta)}{3\sigma_k}}. \tag{3.18}$$

We argue that the neglect of $\mu_k - C_{opt}$ can be taken into account by choosing smaller values of δ, since in computational experiments the curves of $(\mu_k - C_{opt})/(\mu_1 - C_{opt})$ and σ_k/σ_1 as a function of c_k are almost identical; see for instance [Aarts, Korst & Van Laarhoven, 1988].

- final value of the control parameter

The stop criterion is based on the decrease of μ_k, as defined by (3.14), during the execution of the algorithm. Figure 3.1 shows a typical example of the behaviour of the average value of the cost function as a function of the control parameter. The behaviour shown in figure 3.1 is observed for many different problem instances and is reported in the literature by a number of authors [Aarts & Van Laarhoven, 1985a; Aarts, Korst & Van Laarhoven, 1988; Hajek, 1985; Kirkpatrick, Gelatt & Vecchi, 1983; White, 1984]. The dashed curve is obtained by smoothing the data points, i.e. by replacing each data

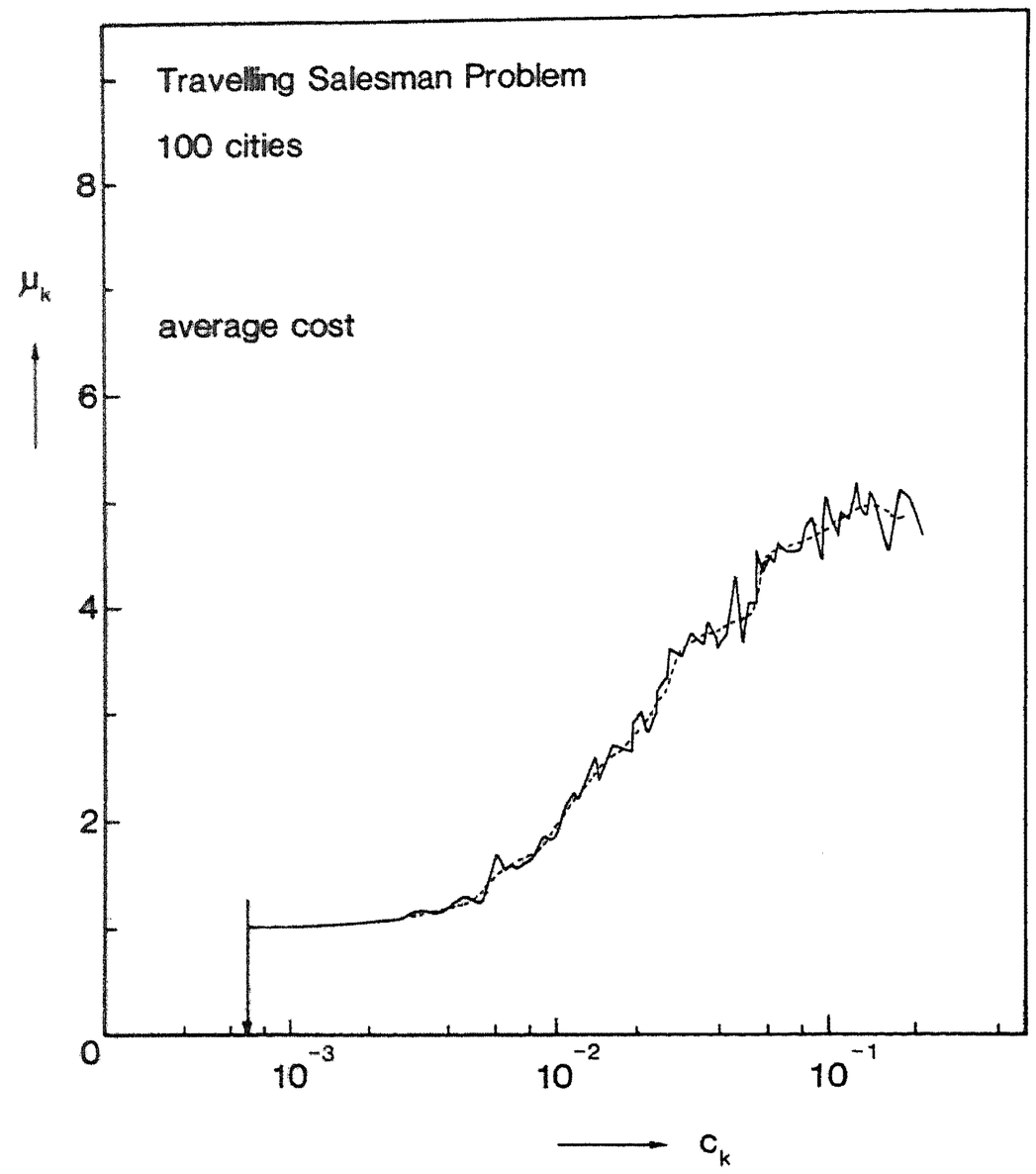


Figure 3.1: Average value of the cost function, μ_k, as a function of the control parameter c_k for a 100-city travelling salesman problem. The dashed curve is obtained by smoothing the data (reproduced from [Aarts & Van Laarhoven, 1985a]).

point $(c, \mu(c))$ by an average over a number of consecutive data points around $(c, \mu(c))$. Let $\overline{\mu} : c \rightarrow \overline{\mu}(c)$ describe the dashed curve and define

$$\Delta\overline{\mu}(c) = \overline{\mu}(c) - C_{opt}. \tag{3.19}$$

It seems reasonable to terminate execution of the algorithm if $\Delta\overline{\mu}(c)$ is small compared to μ_1 (the average cost value of the first Markov chain). For $c \ll 1$ we have

$$\Delta\overline{\mu}(c) \simeq c \cdot \frac{\partial\overline{\mu}(c)}{\partial c}. \tag{3.20}$$

Hence, the algorithm is terminated if

$$\frac{c_k}{\mu_1} \cdot \frac{\partial \overline{\mu}}{\partial c}(c_k) < \epsilon_s, \tag{3.21}$$

where ϵ_s is some small positive number, hereinafter referred to as the stop parameter, and where the derivative $\frac{\partial \overline{\mu}}{\partial c}(c_k)$ is estimated from the dashed curve.

- length of the Markov chains

The decrement rule for the control parameter is such that a 'small' number of transitions should suffice to approach rapidly the stationary distribution for each value of the control parameter (see above). We specify 'small' as the size of the largest neighbourhood, i.e.

$$L_k = L = \max_{i \in \mathcal{R}} |\mathcal{R}_i|, \ k = 1, 2, \ldots, \tag{3.22}$$

primarily because we want the algorithm to have the possibility to visit at least a major part[3] of the neighbourhood of the current configuration before starting the generation of the next Markov chain.

Summarizing, we have derived a cooling schedule which is controlled by three problem-independent parameters: the initial acceptance ratio χ_0, the distance parameter δ and the stop parameter ϵ_s. In chapter 4 we present additional computational evidence for the assertion that near-optimal configurations can be obtained for a wide variety of combinatorial optimization problems with the aforementioned cooling schedule without tuning the three parameters to the problem the algorithm is applied to.

We now prove a theorem stating that the aforementioned decrement rule for the control parameter, together with the final value of the control parameter c determined by (3.21), leads to a total number of steps in the control parameter bounded by $O(\ln |\mathcal{R}|)$. In the proof of the theorem we need three additional quantities:

- the *expected cost* in equilibrium, $\langle C_k \rangle$, given by

$$\langle C_k \rangle = \langle C(c_k) \rangle = \sum_{i \in \mathcal{R}} C(i) q_i(c_k), \tag{3.23}$$

[3] It can be shown that the expected number of **different** elements found when sampling N times with replacement from a set with N elements is approximately $\frac{2}{3}N$ for $N > 5000$ [Aarts, Beenker & Korst, 1985].

where $\mathbf{q}(c_k)$ is again the stationary distribution of the k-th Markov chain;

- the *expected variance* in equilibrium, $\langle \sigma_k^2 \rangle$, given by

$$\langle \sigma_k^2 \rangle = \langle \sigma^2(c_k) \rangle = \sum_{i \in \mathcal{R}} C^2(i) q_i(c_k) - \langle C_k \rangle^2; \tag{3.24}$$

- the *entropy* in equilibrium, S_k, given by

$$S_k = S(c_k) = -\sum_{i \in \mathcal{R}} q_i(c_k) \ln q_i(c_k). \tag{3.25}$$

Theorem 3.2 *[Aarts & Van Laarhoven, 1985a]*
Let the decrement of the control parameter be given by

$$c_{k+1} = \frac{c_k}{1 + \alpha_k c_k}, \; k = 1, 2, \ldots \tag{3.26}$$

with

$$\alpha_k = \frac{\ln(1+\delta)}{3\sigma_k}, \; k = 1, 2, \ldots, \tag{3.27}$$

then, for the first integer K, satisfying

$$\frac{c_K}{\mu_1} \cdot \frac{\partial \overline{\mu}}{\partial c}(c_K) < \epsilon_s, \tag{3.28}$$

we have

$$K = O(\ln |\mathcal{R}|), \tag{3.29}$$

under certain assumptions with respect to the derivatives $\frac{\partial}{\partial c}\overline{\mu}(c)$ and $\frac{\partial}{\partial c}S(c)$.

Proof

The proof of the theorem consists of two parts. First, we express the total number of steps K in the final value of the control parameter c_K. Next, we derive a lower bound on c_K. Equation (3.29) is then obtained by combining the two results.

1. It is easily seen that

$$c_k \leq \frac{c_1}{1+(k-1)\cdot\alpha\cdot c_1},\ k=1,2,\ldots, \tag{3.30}$$

where $\alpha = \min_k \alpha_k$. The proof of (3.30) is by induction. Clearly, (3.30) holds for $k=1$. Suppose, (3.30) holds for k. By using $\alpha_k \geq \alpha$ and the induction hypothesis we obtain

$$c_{k+1} = \frac{1}{c_k{}^{-1}+\alpha_k} \leq \frac{1}{\frac{1+(k-1)\cdot\alpha\cdot c_1}{c_1}+\alpha} = \frac{c_1}{1+k\cdot\alpha\cdot c_1}. \tag{3.31}$$

Hence, (3.30) holds for $k+1$. Consequently, we obtain for K

$$K-1 \leq \frac{c_1-c_K}{\alpha\cdot c_1\cdot c_K} < \frac{1}{\alpha\cdot c_K} = \frac{3\cdot\max_k\sigma_k}{\ln(1+\delta)\cdot c_K} \simeq \frac{3\cdot\sigma_1}{\ln(1+\delta)\cdot c_K}. \tag{3.32}$$

2. By using (3.8) and (3.9), the following relations can be straightforwardly shown to hold:

$$\frac{\partial}{\partial c}\langle C(c)\rangle = c\cdot\frac{\partial}{\partial c}S(c) = \frac{\langle\sigma^2(c)\rangle}{c^2}, \tag{3.33}$$

$$S(\infty) = \lim_{c\to\infty} S(c) = \ln|\mathcal{R}| \tag{3.34}$$

and

$$S(0) = \lim_{c\downarrow 0} S(c) = \ln|\mathcal{R}_{opt}|. \tag{3.35}$$

From (3.33) we conclude that $S(c)$ is increasing in c. Furthermore, we remark that

$$\lim_{L_k\to\infty} \mu_k = \langle C_k\rangle, \tag{3.36}$$

where μ_k is defined by (3.14), since the probability distribution of the configurations approaches the stationary distribution for $L_k \to \infty$.

Using (3.33)-(3.35), we can now express K in terms of $\ln|\mathcal{R}|$ as follows. From (3.21) we conclude

$$\exists\epsilon' \in (0,\epsilon_s]:\ \epsilon' = \frac{c_K}{\mu_1}\frac{\partial}{\partial c}\overline{\mu}(c_K). \tag{3.37}$$

Approximating $\frac{\partial}{\partial c}\overline{\mu}(c)$ by $\frac{\partial}{\partial c}\langle C(c)\rangle$ and using (3.33), we obtain

$$\epsilon' \simeq \frac{c_K^2}{\mu_1}\frac{\partial}{\partial c}S(c_K). \tag{3.38}$$

Next, we approximate $\frac{\partial}{\partial c}S(c_K)$ by $\frac{S(c_K)-S(0)}{c_K-0}$ to obtain

$$\epsilon' \simeq \frac{c_K^2}{\mu_1}\frac{S(c_K)-\ln|\mathcal{R}_{opt}|}{c_K} < \frac{c_K\cdot\ln|\mathcal{R}|}{\mu_1}. \tag{3.39}$$

Consequently,

$$c_K > \frac{\mu_1\cdot\epsilon'}{\ln|\mathcal{R}|} \tag{3.40}$$

and

$$K-1 < \frac{\ln|\mathcal{R}|}{\alpha\cdot\mu_1\cdot\epsilon'} \simeq \frac{3\sigma_1\cdot\ln|\mathcal{R}|}{\ln(1+\delta)\cdot\mu_1\cdot\epsilon'}. \tag{3.41}$$

Thus, the total number of steps in the control parameter is bounded by $\eta\cdot\ln|\mathcal{R}|$, where $\eta = \frac{3\sigma_1}{\ln(1+\delta)\cdot\mu_1\cdot\epsilon'}$.

□

With respect to the approximations in the proof of theorem 3.2, we remark that the first approximation is motivated by (3.36), but is merely supported by computational evidence. The second approximation is motivated by the fact that c_K is very small (note that $\left|\frac{\partial}{\partial c}S(c_K) - \frac{S(c_K)-S(0)}{c_K-0}\right| \downarrow 0$ for $c_K \downarrow 0$).
We finally conclude that, provided μ_1 and σ_1 are independent of the size of the problem and the aforementioned approximations are accurate, the computation time $\mathcal{T}$ of the algorithm with the aforementioned three-parameter schedule satisfies

$$\mathcal{T} = O(\tau\cdot L\cdot\ln|\mathcal{R}|), \tag{3.42}$$

where the term L originates from the length of the Markov chains (3.22), the term $\ln|\mathcal{R}|$ is the upper bound on the number of Markov chains given by theorem 3.2 and τ is the computation time of one transition. For many combinatorial optimization problems, L and τ can be chosen as a polynomial in the problem size, whereas $|\mathcal{R}|$ usually is an exponential or super-exponential function of the problem

size. Consequently, the algorithm runs in polynomial time.
In section 3.4 and in chapter 4, the analytical bound given by (3.42) is compared with numerical calculations found by measuring computation times when applying the algorithm to several combinatorial optimization problems.

3.4 Comparison of several cooling schedules

To compare several of the cooling schedules discussed in sections 3.2 and 3.3, we use simulated annealing with different cooling schedules to solve instances of two well-known combinatorial optimization problems, viz. the graph partitioning problem and the travelling salesman problem.
In an instance of the *graph partitioning problem*, a graph $G = (V, E)$ is given. The set of configurations of an instance is given by the set of partitions (V_1, V_2) of the vertex set V, the cost of a configuration is a weighted sum of the difference in cardinality between V_1 and V_2 (the *imbalance*) and the number of edges with one endpoint in V_1 and one in V_2. The problem is to find a partition with minimal cost.
To be able to apply simulated annealing, we need to define, in addition to the configurations and the cost function, a neighbourhood structure. Let $V = \{v_1, \ldots, v_n\}$ denote the set of vertices, with $n = |V|$, then each configuration of the problem is characterized by a sequence $\mathbf{r}_i = (r_{i1}, r_{i2}, \ldots, r_{in})$, where

$$r_{ij} = \begin{cases} +1 & \text{if } v_j \in V_1, \\ & \qquad j = 1, 2, \ldots, n. \\ -1 & \text{if } v_j \in V_2, \end{cases} \tag{3.43}$$

Consequently, the total number of configurations, $|\mathcal{R}|$, is 2^{n-1}.
The imbalance ξ is given by

$$\xi = \big|\, |V_1| - |V_2| \,\big| = \Big| \sum_{j=1}^{n} r_{ij} \Big|. \tag{3.44}$$

Let $\mathcal{E}$ denote the set of edges with endpoints in different subsets, then

$$|\mathcal{E}| = \frac{1}{4}\sum_{j=1}^{n}\sum_{k=j+1}^{n} a_{jk}(r_{ij} - r_{ik})^2, \tag{3.45}$$

where a_{jk} denotes the number of edges between two vertices $v_j, v_k \in V$. From (3.44) and (3.45) we obtain

$$|\mathcal{E}| + \lambda\xi^2 = \sum_{j=1}^{n}\sum_{k=j+1}^{n}(2\lambda - \frac{1}{2}a_{jk})r_{ij}r_{ik} + \lambda n + \frac{1}{2}\sum_{j=1}^{n}\sum_{k=j+1}^{n} a_{jk}, \tag{3.46}$$

where λ is some real-valued weighting factor. The squared value of ξ is chosen so as to have a smaller cost for increases of ξ at small imbalances than for increases at large imbalances. Omitting the last two terms from (3.46), which are constant and thus irrelevant to the optimization, we obtain the following expression for the cost of a configuration i:

$$C(i) = \sum_{j=1}^{n}\sum_{k=j+1}^{n}(2\lambda - \frac{1}{2}a_{jk})r_{ij}r_{ik}. \tag{3.47}$$

The weighting factor λ is chosen as $\frac{1}{4}\bar{e}$, where $\bar{e}$ is the average *degree* of a vertex, i.e. the average number of edges per vertex. In that way we achieve that the contribution of the two terms in (3.47) to the total cost is approximately equal.
Finally, a transition is generated by changing an (arbitrarily chosen) element r_{ik} from the current sequence $\mathbf{r}_i$ to $-r_{ik}$. Thus, the neighbourhood $\mathcal{R}_i$ of a configuration i is given by

$$\mathcal{R}_i = \left\{ j \in \mathcal{R} \,\middle|\, \sum_{k=1}^{n} |r_{ik} - r_{jk}| = 2 \right\} \tag{3.48}$$

and $|\mathcal{R}_i| = n$ for all i. Furthermore, if i and j are two arbitrary configurations and κ is the number of positions in which $\mathbf{r}_i$ and $\mathbf{r}_j$ differ, then a sequence of κ transitions can be generated such that, starting at configuration i, we end up in configuration j. Consequently, the first condition of theorem 2.1 is satisfied and accordingly the homogeneous algorithm converges asymptotically to a global minimum.
To carry out the comparison, we use the following three cooling schedules:

1. Cooling schedule 1 is the three-parameter schedule described in section 3.3. All results are obtained with the following parameter setting: $\chi_0 = 0.95$, $\delta = 10^{-1}$ and $\epsilon_s = 10^{-3}$.

2. Cooling schedule 2 is the schedule used in the computational study of Johnson, Aragon, McGeoch & Schevon [1987] and is similar to the schedule of Kirkpatrick, Gelatt & Vecchi [1982]; see section 3.2. The initial value of the control parameter is identical to the one of schedule 1, the decrement rule for the control parameter is given by $c_{k+1} = \alpha c_k$ $(k = 1, 2, \ldots)$ and execution of the algorithm is terminated if the cost values of the last configurations of ρ consecutive Markov chains are identical. Here, we use $\chi_0 = 0.95$, $\alpha = 0.95$ and $\rho = 4$.
   Generation of the k-th Markov chain is terminated if either $\eta \cdot (\Lambda \cdot n)$ transitions have been accepted or $\Lambda \cdot n$ transitions have been generated, whichever occurs first. Here, we use $\eta = 0.25$ and choose Λ such that the computation time taken by schedule 2 is at least as large as that of schedule 1.
   Assuming that the stop criterion of this schedule has a similar effect as the stop criterion of the first schedule, we use (3.40) and
   $$K = 1 + \frac{\ln c_K - \ln c_1}{\ln \alpha}, \tag{3.49}$$
   where the K-th Markov chain is again the first Markov chain for which the stop criterion is satisfied, to obtain
   $$K = O(\ln(\ln |\mathcal{R}|)). \tag{3.50}$$

3. Cooling schedule 3 is identical to schedule 2, except for the fact that the length of each Markov chain is fixed to $\Lambda \cdot n$ transitions.

The comparison is carried out by running the algorithm controlled by schedules 1, 2 and 3, respectively, on a set of randomly generated instances of the graph partitioning problem. Each instance is generated in the following way. Given a total number of vertices n and an average vertex degree $\bar{e}$ as well as a maximum number of edges a_{max} between any two vertices, an $n \times n$-matrix $A = (a_{ij})$ is generated such

that its (randomly drawn) elements satisfy two conditions:

$$(1) \quad \forall i,j \in \{1,2,\ldots,n\} : a_{ij} \in \{0,1,\ldots,a_{max}\} \tag{3.51}$$

$$(2) \quad \sum_{j=1}^{n} \sum_{k=j+1}^{n} a_{ij} = \bar{e} \cdot n. \tag{3.52}$$

Here, we use $a_{max} = 5$, whereas n and $\bar{e}$ vary between 50 and 300 and 10 and 30, respectively. The averages in table 3.1 are taken over the 25 final solutions obtained by generating, for given values of n and $\bar{e}$, five problem instances and running the algorithm five times on each instance (with five different seeds for the random number generator). The standard deviations are calculated from five average values, where each average corresponds to an instance and is obtained by averaging over the five runs for that instance. Computation times are CPU-times on a VAX 11/785-computer.
From table 3.1, we conclude that schedules 1 and 3 return final solutions of approximately the same quality, whereas schedule 2 performs slightly better. As for computation times, for fixed $\bar{e}$ the average computation time $\bar{t}$ taken by schedule 1 is approximately given by $\bar{t} = t_1 \cdot n^{1.7}$ for some constant t_1 ($\chi^2 = 0.998$, 0.998 and 0.999 for $\bar{e} = 10$, 20 and 30, respectively). We remark that the bound for the computation time given by theorem 3.2 is $O(n^2)$: $L = n$ and $\ln|\mathcal{R}| = n \ln 2$, whereas τ is independent of n for fixed $\bar{e}$.
For schedules 2 and 3 we have chosen the parameter Λ such that the average computation time is again approximately $O(n^{1.7})$. We note that the bound found by using (3.50) is $O(\Lambda\, n \ln n)$: the Markov chain length L is bounded by $O(\Lambda\, n)$ and $K = O(\ln(n \ln 2)) = O(\ln n)$.

Next, we present some data that have been obtained by applying simulated annealing with different cooling schedules to instances of the *travelling salesman problem*. For a detailed discussion of the application of the algorithm to this problem the reader is referred to section 4.1.

Again, cooling schedules 1, 2 and 3 are used with the following parameter settings: $\chi_0 = 0.95$, $\delta = 10^{-1}$ and $\epsilon_s = 10^{-3}$ (schedule 1), $\chi_0 = 0.95$, $\alpha = 0.95$, $\rho = 4$, $\eta = 0.25$ and $\Lambda = 8$ (schedule 2), $\chi_0 = 0.95$, $\alpha = 0.95$, $\rho = 4$ and $\Lambda = 6$ (schedule 3). The schedules are compared by running the algorithm on the set of five 100-city problems taken from Krolak, Felts & Marble [1971]. The results are

Table 3.1: Average cost of final solution ($\overline{C}_{final}$), average computation time ($\bar{t}$, in seconds on a VAX 11/785) and standard deviations ($\sigma_{\overline{C}}$ and $\sigma_{\bar{t}}$, respectively) for different-sized instances of the graph partitioning problem. The results are obtained with schedules 1, 2 and 3, respectively. For each instance, the best average cost value is marked with an asterisk.

Size		*Schedule 1*				*Schedule 2*					*Schedule 3*				
n	$\bar{e}$	$\overline{C}_{final}$	$\sigma_{\overline{C}}$	$\bar{t}$	$\sigma_{\bar{t}}$	Λ	$\overline{C}_{final}$	$\sigma_{\overline{C}}$	$\bar{t}$	$\sigma_{\bar{t}}$	Λ	$\overline{C}_{final}$	$\sigma_{\overline{C}}$	$\bar{t}$	$\sigma_{\bar{t}}$
50	10	-138.3	8.8	1.7	0.6	2.25	-138.1	8.4	1.8	0.1	1.75	-140.0*	8.2	1.7	0.0
100	10	-287.8	6.7	5.2	0.1	3.00	-286.9	6.2	4.8	0.3	2.50	-288.7*	5.8	5.2	0.3
150	10	-431.7	5.7	10.8	0.4	3.75	-432.2*	9.2	10.2	0.6	3.25	-430.9	6.8	11.4	0.4
200	10	-579.2	10.3	18.9	1.9	4.50	-582.7	12.3	18.4	0.6	4.00	-584.1*	15.7	18.9	0.8
250	10	-730.2	2.3	25.0	4.2	5.25	-733.9*	6.4	24.3	1.9	4.00	-732.9	6.7	24.4	1.6
300	10	-896.8*	9.0	35.9	5.0	6.00	-895.5	6.6	35.9	2.0	4.50	-890.2	5.0	35.2	1.9
50	20	-182.6	5.6	2.1	0.1	2.25	-185.0	5.6	1.9	0.1	1.75	-185.3*	5.4	2.1	0.2
100	20	-405.1	5.8	6.6	0.3	3.00	-404.5	8.5	6.1	0.5	2.50	-406.9*	7.8	7.0	0.3
150	20	-619.4	14.0	13.1	0.6	3.75	-622.4*	16.8	13.1	0.8	3.25	-621.2	15.2	15.1	0.5
200	20	-845.0	9.5	20.6	0.7	4.50	-852.1*	11.7	22.3	1.0	3.25	-849.3	10.4	20.1	0.9
250	20	-1063.3	10.0	29.0	1.6	5.25	-1065.6*	15.0	32.8	2.1	3.50	-1060.4	14.1	29.5	1.1
300	20	-1289.7	23.2	42.8	2.5	6.00	-1291.3*	26.3	46.6	3.1	4.25	-1281.7	24.9	43.3	2.4
50	30	-212.4	8.5	2.5	0.1	2.25	-212.6	6.0	2.6	0.2	1.75	-214.5*	8.6	2.8	0.3
100	30	-482.5	9.9	8.0	0.4	3.00	-484.1*	12.0	8.3	1.0	2.50	-481.0	9.4	8.8	0.2
150	30	-758.1	14.8	15.8	0.4	3.75	-763.0*	17.2	16.0	0.5	2.75	-756.5	15.9	14.8	0.5
200	30	-1017.9	23.4	23.7	0.8	4.50	-1027.3*	18.2	29.6	1.5	3.25	-1021.7	16.3	26.1	1.1
250	30	-1273.7	24.9	36.2	1.7	5.25	-1290.9*	20.8	41.3	2.3	3.50	-1283.8	14.3	36.2	1.4
300	30	-1550.6	12.6	51.2	3.2	6.00	-1561.9*	22.7	60.2	2.7	4.25	-1553.0	12.4	53.2	2.6

displayed in table 3.2 (they are calculated by averaging over the ten final solutions obtained by running the algorithm ten times on each instance). From this table we conclude that the quality of the final solutions returned by the three schedules is again approximately the same.
From the experiments we draw the following conclusions:

1. If 'cooling' is carried out with 'reasonable accuracy', the quality of the final solution returned by the algorithm hardly depends on the particular cooling schedule used.

2. The average time complexity of the schedules can be quite accurately predicted by the worst-case time complexity given by theorem 3.2 and related observations.

3. Schedule 1 is consistently slightly worse than schedule 2, but makes up for this in our opinion by the fact that it needs substantially less parameter tuning: in schedule 1, the only parameter relevant to the quality of the final solution is δ,[4] whereas in schedule 2, we need to find a good choice for α, η and Λ. Therefore, in the applications of simulated annealing discussed in the next chapter, we restrict ourselves to implementations of simulated annealing with schedule 1.

[4] There is substantial computational evidence for the assertion that the quality of the final solution is predominantly determined by the choice of the decrement rule for the control parameter and the length of the Markov chains, see for instance [Johnson, Aragon, McGeoch & Schevon, 1987].

Table 3.2: Average cost of final solution ($\overline{C}_{final}$), average computation time ($\bar{t}$, in seconds on a VAX 11/785), standard deviations (σ_C and σ_t, respectively) and % of average final solution above optimal solution for a set of five travelling salesman problems. The results are obtained with schedules 1, 2 and 3, respectively. For each instance, the best average cost value is provided with an asterisk.

Problem no.	*Optimal solution*	$\overline{C}_{final}$	σ_C	*% above optimal*	$\bar{t}$	σ_t
		Schedule 1				
24	21282	21467.8	102.0	0.87	822.2	16.3
25	22148	22492.8	163.7	1.56	813.2	16.7
26	20749	20928.4	168.8	0.86	807.4	13.1
27	21294	21436.1*	122.5	0.67	823.1	15.5
28	22068	22454.9	190.3	1.75	832.5	18.4
		Schedule 2				
24	21282	21436.3*	69.0	0.73	819.0	20.8
25	22148	22323.4*	131.9	0.79	886.6	63.5
26	20749	20939.4	63.6	0.92	970.6	28.3
27	21294	21442.7	110.9	0.70	861.8	38.9
28	22068	22211.0*	69.6	0.65	894.7	86.4
		Schedule 3				
24	21282	21473.6	134.3	0.90	813.4	40.0
25	22148	22435.0	133.2	1.30	901.6	42.6
26	20749	20900.4*	116.4	0.73	903.5	56.1
27	21294	21515.8	143.2	1.01	825.4	34.3
28	22068	22293.2	129.9	1.02	866.3	54.2

Chapter 4

Empirical analysis of simulated annealing

4.1 Introduction

When analysing the performance of an approximation algorithm, one might consider the following aspects [Ball & Magazine, 1981]:

- the *efficiency* of the algorithm, i.e. the number of elementary operations (the *running time*) required by the algorithm to return a solution;
- the *effectivity* of the algorithm, i.e. the *difference in cost* between the returned solution and a globally minimal solution;
- the *simplicity* of the algorithm;
- the *ease of implementation* of the algorithm;
- the *flexibility* or *robustness* of the algorithm.

With respect to the simulated annealing algorithm, we make some general remarks on the last three aspects. The first two aspects are extensively addressed in the remainder of this chapter; with respect to the difference in cost we remark that we only consider the relative difference, i.e. the ratio between the difference and the globally minimal cost value.

The simplicity of simulated annealing hardly calls for any further comment - it is part of its attraction. It is also an easy-to-implement algorithm - the various simulated annealing algorithms described in this chapter typically consist of a few hundred lines of computer code - though it is not always easy to formulate a problem in a way that lends itself to application of the algorithm, as is shown in the remainder of this chapter. Flexibility refers to the ability of an algorithm to handle problem variations and different problems. The rich variety of problems to which the algorithm is applied, see [Van Laarhoven & Aarts, 1987], strongly supports the claim that simulated annealing is a very flexible optimization technique.
With regard to the first two aspects, one usually distinguishes three ways in which the analysis of an approximation algorithm is actually carried out (see for instance chapters 5, 6 and 7 in [Lawler, Lenstra, Rinnooy Kan & Shmoys, 1985]):

1. *worst-case analysis*, which involves finding bounds on how large the aforementioned difference and running time can be in the worst case. To the best of our knowledge, the only worst-case analysis of the difference in cost between the returned solution and a globally minimal one has been carried out by Sasaki and Hajek [1986] for the *maximum matching problem* - their results combine a worst-case analysis of the difference and an average-case running time analysis (see next approach) for this problem. A worst-case running time result for a specific version of simulated annealing is given by theorem 3.2. Another type of analysis is briefly described on page 27 and leads to an upper bound on the distance between the probability distribution of the configurations after k transitions of the inhomogeneous algorithm and the uniform distribution on the set of globally minimal configurations.

2. *average-case* or *probabilistic analysis*. Underlying such an analysis is a probability distribution over the set of all problem instances. The analysis involves finding analytical expressions for the expected running time and the expected difference between the returned solution and a globally minimal one, where the expectations are calculated according to the aforementioned prob-

ability distribution. Analysing an algorithm such as simulated annealing introduces an additional probabilistic element in an average-case analysis, because the algorithm itself is of a probabilistic nature. Consequently, even for a fixed problem instance, the running time and difference are stochastic variables. It is precisely to this additional element that the previously mentioned result of Sasaki and Hajek [1986] refers: they give an analytical expression for the expected running time of the algorithm (and the worst-case difference between returned solution and a globally minimal one) given an arbitrary instance of the maximum matching problem. We are not aware of any other average-case analysis results in the literature.

3. *empirical analysis*, which involves the analysis of running time and difference, based on a (large) set of computational experiments. A number of such computational studies have been published in recent years, e.g. by Golden & Skiscim [1986], Kirkpatrick [1984], Morgenstern & Shapiro [1986], Nahar, Sahni & Shragowitz [1985,1986] and by Skiscim & Golden [1983]. While not intending to belittle these studies, we feel that they are sometimes carried out in a haphazard way, in particular when it comes to the choice of the parameters of a cooling schedule. Furthermore, the results found by simulated annealing are not always pitted against those found by tailored algorithms for the problem under consideration. Finally, the widely cited computational study of Johnson, Aragon, McGeoch & Schevon [1987] has unfortunately not been published so far.

The empirical analysis described in the remainder of this section has the following characteristics:

- It is based on a carefully derived cooling schedule, which needs a small amount of parameter tuning and whose performance has been shown to be approximately as good as that of the best-known cooling schedules from the literature (see section 3.4).
- It spans a set of representative combinatorial optimization problems, viz.:

– the *travelling salesman problem*, which is probably the best-known problem in combinatorial optimization, and for which many approximation and optimization algorithms were constructed during the past decades (see [Lawler, Lenstra, Rinnooy Kan & Shmoys, 1985]);

– the *job shop scheduling problem*, which is among the hardest combinatorial optimization problems: not only is it $\mathcal{NP}$-hard, but even among the members of the latter class it belongs to the most difficult ones (see for instance [Lawler, Lenstra & Rinnooy Kan, 1982]);

– the *football pool problem*, which is an example of a problem which had never been solved by considering it as an optimization problem before it was tackled by simulated annealing. Through this problem we also illustrate that the choice of a neighbourhood structure can be very critical.

- Where possible, the results found by simulated annealing are not only pitted against those found by other (tailored) approximation algorithms, but also against (upper bounds on) globally minimal solutions of the problems under consideration.

- All results are obtained by averaging over several runs of the algorithm. For a probabilistic algorithm, multiple runs on the same problem are a prerequisite to get meaningful results, but this is not always done in other computational studies.

Each of the following sections is devoted to one of the aforementioned problems. Throughout these sections, computation times are CPU-times on a VAX 11/785-computer, unless explicitly mentioned otherwise. The last section of this chapter is devoted to conclusions based on the computational results.

4.2 The Travelling Salesman Problem

The attraction of the Travelling Salesman Problem (TSP) is probably due to the paradoxical characteristic that it is simple to state and

very difficult to solve to optimality.
Informally, the TSP can be stated as follows. A salesman starts from his home city, visits each city on a given list exactly once and returns to his home city. His problem is to select the order in which he visits the cities such that the total distance travelled in his tour is as small as possible.
More formally, the problem can be described as follows. If a permutation π of the set $\mathcal{N} = \{1, \ldots, n\}$ is interpreted such that $\pi(k)$, $k \in \mathcal{N}$, denotes the successor of city k in the tour of a salesman who is to visit n cities, then each tour can be characterized by a *cyclic permutation* of $\mathcal{N}$, i.e. a permutation π of $\mathcal{N}$ such that for every $k \in \mathcal{N}$ we have

$$\pi^l(k) \neq k,\ l = 1, 2, \ldots, n-1,\ \pi^n(k) = k. \tag{4.1}$$

The n-city TSP can now be stated as follows. Given an $n \times n$-matrix $D = (d_{ij})$, hereinafter referred to as the *distance matrix*, find a cyclic permutation π of $\mathcal{N}$ minimizing

$$\sum_{k \in \mathcal{N}} d_{k\pi(k)}. \tag{4.2}$$

To apply simulated annealing, we recall that we need to define configurations, a cost function and a neighbourhood structure.
Each configuration i of the problem is characterized by a cyclic permutation π_i of $\mathcal{N}$; consequently, for a symmetric distance matrix, the total number of configurations $|\mathcal{R}|$ is $\frac{1}{2}(n-1)!$. The cost of a configuration i, characterized by a permutation π_i, is given by

$$C(i) = \sum_{k \in \mathcal{N}} d_{k\pi_i(k)}. \tag{4.3}$$

Finally, a transition is generated by replacing two edges in the current tour by two non-tour edges (see figure 4.1) (an edge in a tour, characterized by a permutation π, is a pair of cities $(k, \pi(k))$). This transition is called *2-change* and is a special case of *k-change* [Lin, 1965; Lin & Kernighan, 1973], in which k edges are replaced by k other edges. Iterative improvement algorithms based on this type of transition have been shown to be quite effective for the TSP; see for instance [Lin & Kernighan, 1973].

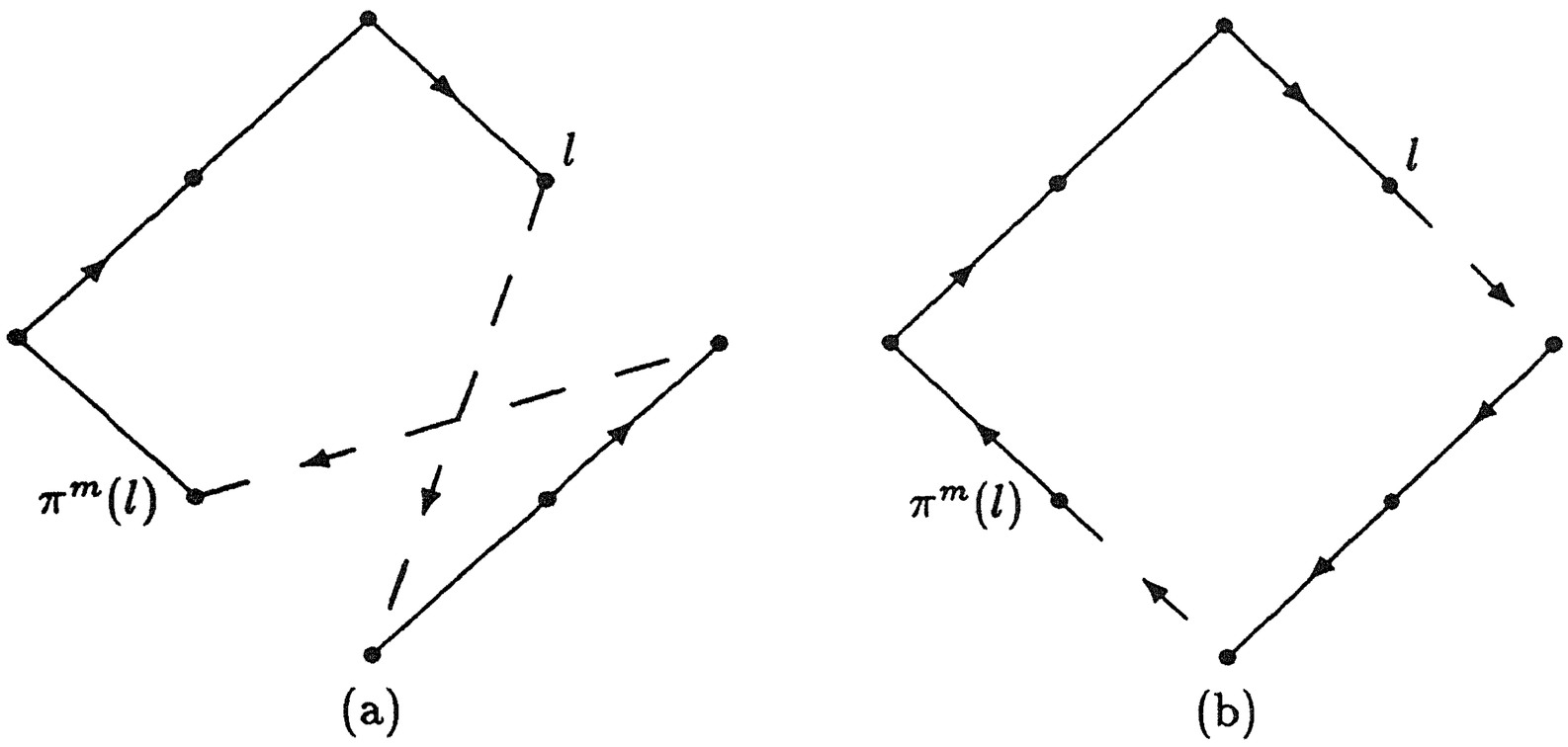


Figure 4.1: 2-change example; (a) current tour (b) tour after reversing the order between cities l and $\pi^m(l)$.

A 2-change can also be seen as a reversal of the order in which the cities in between a selected pair of cities are visited. If π is the cyclic permutation corresponding to the current tour and 2-CH(l,m) denotes a 2-change, where (l,$\pi^m(l)$) ($m \neq n$) is the aforementioned pair of cities, then 2-CH(l,m) results in a cyclic permutation μ, given by

$$\mu(l) = \pi^{m-1}(l); \tag{4.4}$$

$$\mu(\pi(l)) = \pi^m(l); \tag{4.5}$$

$$\mu(\pi^r(l))) = \pi^{r-1}(l),\ r = 2, \ldots, m-1; \tag{4.6}$$

$$\mu(k) = \pi(k),\ k \neq \pi^s(l),\ s = 0, 1, \ldots, m-1. \tag{4.7}$$

Thus, the neighbourhood $\mathcal{R}_i$ of a configuration i, characterized by a permutation π_i, is given by

$$\mathcal{R}_i = \{j |\ \pi_j \text{ is obtained from } \pi_i \text{ by a 2-change}\} \tag{4.8}$$

and $|\mathcal{R}_i| = \binom{n}{2} = \frac{1}{2}n(n-1)$ for all i. Furthermore, suppose i and j are two arbitrary tours. Define the sequence of permutations $\{\pi_{\lambda_t}\}$, $t = 0, 1, \ldots$ by the following two relations:

1. $\pi_{\lambda_0} = \pi_i$;

2. π_{λ_t} is obtained from $\pi_{\lambda_{t-1}}$ by 2-CH(t, m_t), where m_t satisfies

$$\pi_{\lambda_{t-1}}^{m_t-1}(t) = \pi_j(t). \tag{4.9}$$

Note that m_t is chosen such that the successor of t in tour λ_t, given by (4.4), is identical to the successor of t in tour j.

It is possible to show by induction that for $t \geq 1$, the first t cities in the tour λ_t are identical to the first t cities in the tour j, or, more precisely, that for $t \geq 1$

$$\pi_{\lambda_t}(k) = \pi_j(k), \ 1 \leq k \leq t. \tag{4.10}$$

For $t = 1$, we find that π_{λ_1} is obtained from π_i by 2-CH$(1, m_1)$, where m_1 satisfies

$$\pi_i^{m_1-1}(1) = \pi_j(1). \tag{4.11}$$

Substituting $\mu = \pi_{\lambda_1}$, $\pi = \pi_i$, $l = 1$ and $m = m_1$ in (4.4) and combining it with (4.11) yields

$$\pi_{\lambda_1}(1) = \pi_i^{m_1-1}(1) = \pi_j(1). \tag{4.12}$$

Thus, (4.10) holds for $t = 1$. Now suppose (4.10) holds for $t-1$. Using (4.9) and substituting $\mu = \pi_{\lambda_t}$, $\pi = \pi_{\lambda_{t-1}}$, $l = t$ and $m = m_t$ in (4.4) yields

$$\pi_{\lambda_t}(t) = \pi_{\lambda_{t-1}}^{m_t-1}(t) = \pi_j(t). \tag{4.13}$$

Thus, (4.10) also holds for t. For $t = n$, (4.10) implies $\pi_{\lambda_n} = \pi_j$, so that we have a sequence of n transitions leading from π_i to π_j.
We recall from theorem 2.1 that if the generation and acceptance matrix are given by the uniform distribution on the neighbourhoods and the Metropolis criterion, respectively, then the only condition for asymptotic convergence of the homogeneous algorithm is a condition which we have just shown to be satisfied, namely that it is possible, for every two configurations i and j, to construct a finite sequence of transitions leading from i to j.

The empirical analysis of the finite-time behaviour of the algorithm is carried out by running the algorithm on a number of instances of the travelling salesman problem, varying in size from 48 to 442 cities. The instances can be divided into three sets:

1. Nine Euclidean instances, i.e. instances for which the entries of the distance matrix satisfy the triangle inequality: for all $i, j, k \in \mathcal{N}$: $d_{ik} \leq d_{ij} + d_{jk}$. These instances are all taken from the literature. GRO48 is a 48-city problem due to Grötschel [1977]; the 48 cities are 48 major German cities with the distances given in *Shell's Road Atlas of Germany*. TOM57 is a 57-city problem due to Karg & Thompson [1964]; the 57 cities are 57 major US cities with the distances given in the *Rand-McNally Road Atlas of the United States*. EUR100 is a 100-city problem due to Aarts, Korst & Van Laarhoven [1988]; the 100 cities are 100 major European cities with the distances calculated from the degrees of latitude and longitude of the cities. KRO124 through KRO128 are 100-city problems due to Krolak, Felts & Marble [1971]; the original data are randomly generated coordinates in the plane. GRO120 is a 120-city problem due to Grötschel [1977]; the 120 cities are 120 German cities with distances given in the *Deutscher General Atlas*.

2. Two non-Euclidean instances taken from the literature: LIN318 is a 318-city problem due to Lin & Kernighan [1973] and GRO442 is a 442-city problem due to Grötschel [1984]. The data for both problems come from actual problems involving the routing of a numerically controlled drilling machine through a number of points in the plane. Since the drilling time is the same for each point, the problem can be formulated as a TSP, the only exception from a standard TSP being that a particular start and end point are to be used. The latter is enforced by setting the distance between start and end city to a large negative value, as a consequence of which the problems become non-Euclidean. GRO442 is one of the largest instances discussed in the literature for which, without using a *partitioning approach*, a provably globally minimal solution has been found [Grötschel, 1987] (the largest instance solved to optimality is, to our knowledge, a highly structured 2,392-city instance solved by Padberg and Rinaldi [Johnson, 1987]).

3. Randomly generated non-Euclidean instances, i.e. instances for which the entries of the distance matrix are drawn from

a uniform distribution on the interval $[1, 100n]$. We have generated five such problems, RAN50a through RAN50e, all with 50 cities. In general, this type of TSP is more difficult to solve than the Euclidean TSP. Computational evidence for this assertion can be found in, for instance, [Lin & Kernighan, 1973]; in addition, there are theoretical results stating that arbitrary TSP instances are more difficult to solve than instances obeying the triangle inequality; see for instance [Johnson & Papadimitriou, 1985].

The performance of simulated annealing for the Euclidean instances is reported in tables 4.1 and 4.2. The averages in table 4.1 (4.2) are computed from the five (ten) final solutions obtained by running the algorithm, controlled by the cooling schedule described in section 3.3, five (ten) times on each instance. All results are obtained with the parameters χ_0 and ϵ_s set to 0.95 and 10^{-6}, respectively, and for different values of the distance parameter δ.

From tables 4.1 and 4.2 we observe firstly that the quality of the average solution (the average cost value) returned by the algorithm improves when δ is decreased. This is in accordance with the theory underlying the employed cooling schedule: a smaller value of δ stands for a smaller distance between the stationary distributions of two successive Markov chains. Consequently, we may expect that, for a smaller value of δ, after a given number of transitions the probability distribution of the configurations is closer to the stationary distribution. In other words, a smaller value of δ leads to a better approximation of the asymptotic behaviour of the algorithm. Furthermore, we observe that the standard deviation in the final cost values decreases with decreasing δ, which indicates that the reliability of the results increases with decreasing δ. This is especially true for the final cost values in table 4.2, which are a more accurate reflection of the average behaviour of the algorithm than the values in table 4.1, since the averages and standard deviations in the former table are computed from twice as many runs as those in the latter. We also observe that the difference between the average final solution and a globally minimal one does not significantly deteriorate with increasing problem size, and that even for the largest problems the average deviation from the global minimum is less than 2%.

Table 4.1: Average cost of final solution ($\overline{C}_{final}$), average computation time in seconds ($\bar{t}$), standard deviations (σ_C and σ_t, respectively) and % of average final cost value above globally minimal cost value for different-sized instances of the travelling salesman problem. The results are obtained with the simulated annealing algorithm with different values of the distance parameter δ. The averages are obtained from 5 runs.

Problem	δ	$\overline{C}_{final}$	σ_C	%	$\bar{t}$	σ_t
GRO48	10.0	5203.6	65.5	3.12	6.3	0.3
	1.0	5203.0	111.4	3.11	15.9	1.5
	0.1	5094.8	23.9	0.97	93.8	5.6
TOM57	10.0	13340.8	241.4	2.98	10.2	0.5
	1.0	13218.6	153.2	2.03	26.9	2.3
	0.1	13068.0	50.5	0.87	158.2	3.9
EUR100	10.0	21852.6	329.7	3.40	45.8	1.7
	1.0	21711.8	200.9	2.73	121.2	2.5
	0.1	21339.4	171.9	0.97	801.6	8.3
GRO120	10.0	7269.6	81.1	4.72	72.2	3.2
	1.0	7174.8	29.8	3.35	205.6	1.5
	0.1	7057.2	72.0	1.66	1369.4	24.5
LIN318	10.0	43132.8	545.3	4.44	1126.8	29.7
	1.0	42262.6	183.4	2.33	3437.6	147.1
	0.1	41957.4	176.8	1.59	23941.6	967.8
GRO442	10.0	5287.0	28.2	4.30	2749.6	70.0
	1.0	5206.8	25.5	2.72	8458.4	106.9
	0.1	5147.0	20.9	1.54	58674.6	432.4

Table 4.2: Average cost of final solution ($\overline{C}_{final}$), average computation time in seconds ($\bar{t}$), standard deviations (σ_C and σ_t, respectively) and % of average final cost value above globally minimal cost value for the set of five instances of the travelling salesman problem taken from Krolak, Felts & Marble [1971]. The results are obtained with the simulated annealing algorithm with different values of the distance parameter δ. The averages are obtained from 10 runs.

Problem	δ	$\overline{C}_{final}$	σ_C	%	$\bar{t}$	σ_t
KRO124	10.0	22014.5	283.8	3.44	49.5	1.4
	1.0	21632.6	254.0	1.65	129.6	5.6
	0.1	21467.8	102.0	0.87	822.2	16.3
KRO125	10.0	23030.5	440.0	3.98	46.4	1.1
	1.0	22737.0	288.6	2.66	124.1	3.5
	0.1	22492.8	163.7	1.56	813.2	16.7
KRO126	10.0	21465.7	302.9	3.45	46.6	1.4
	1.0	21162.9	278.2	1.99	121.8	2.3
	0.1	20928.4	168.8	0.86	807.4	13.1
KRO127	10.0	22072.4	273.0	3.66	45.8	2.5
	1.0	21757.6	229.4	2.18	123.5	3.9
	0.1	21436.1	122.5	0.67	823.1	15.5
KRO128	10.0	22912.9	259.6	3.83	45.8	1.3
	1.0	22548.4	252.0	2.18	125.1	3.5
	0.1	22454.9	190.3	1.75	832.5	18.4

As for computation times, we observe that the average computation time $\bar{t}$ is approximately given by $\bar{t} = t_0 \cdot n^p \cdot \ln n$, for some constant t_0, where p is 2.53, 2.62 and 2.69 for δ is 10, 1 and 0.1, respectively ($\chi^2 = 1.000$). We remark that the bound for the computation time given by theorem 3.2 is $O(n^3 \cdot \ln n)$ (since $L = \frac{1}{2}n(n-1)$ and $\ln|\mathcal{R}| = \ln(\frac{1}{2}(n-1)!) = O(n \ln n)$), which is only slightly worse than the observed computation times. Finally, we observe that smaller values of δ lead to larger computation times, which is in accordance with the appearance of δ in the denominator of the right-hand side of (3.41).

In table 4.3, simulated annealing is compared with repeated execution of iterative improvement based on 2-changes. The initial configurations to which the iterative improvement algorithm is applied are randomly generated cyclic permutations. The averages for iterative improvement are obtained from five macro-runs (ten for KRO124 through KRO128). Each macro-run consists of repeated execution of the iterative improvement algorithm for a large number of initial configurations and thus yields a large number of local minima. Execution of each macro-run is terminated as soon as the computation time exceeds the computation time of an average run of simulated annealing applied to the same problem instance with the distance parameter δ set to 0.1; $\overline{C}_{best}$ is the average of the best cost value found during each macro-run. The results for simulated annealing are taken from tables 4.1 and 4.2 ($\delta = 0.1$).

We observe that repeated execution of iterative improvement is easily outperformed by simulated annealing for the larger problems. For GRO442 the difference is especially pronounced and the results suggest that the quality of the average best solution returned by (repeated execution of) the iterative improvement algorithm deteriorates significantly with increasing problem size, contrary to simulated annealing.

In table 4.4, a similar comparison is made with repeated execution of the *Lin-Kernighan algorithm* [Lin & Kernighan, 1973], an approximation algorithm tailored to the TSP. The Lin-Kernighan algorithm is a sophisticated iterative improvement algorithm, based on k-changes, where at each stage the algorithm chooses dynamically a 'good' value for k. For each instance, the number of (randomly generated) initial

Table 4.3: Average cost of best solution ($\overline{C}_{best}$) or final solution ($\overline{C}_{final}$), average computation time in seconds ($\overline{t}$), standard deviations (σ_C and σ_t, respectively), average number of local minima per macro-run of iterative improvement ($\overline{lm}$) and % of average final cost value (annealing) above average best cost value (iterative improvement) for different-sized instances of the travelling salesman problem. The results are obtained with repeated execution of the iterative improvement algorithm and with simulated annealing, respectively. The averages are obtained from five (macro-)runs (ten for KRO124 through KRO128).

	Iterative Improvement		*Simulated Annealing*			*Iterative Improvement*			*Simulated Annealing*	
Problem	$\overline{C}_{best}$	σ_C	$\overline{C}_{final}$	σ_C	%	$\overline{lm}$	$\overline{t}$	σ_t	$\overline{t}$	σ_t
GRO48	5056.0	4.6	5094.8	23.9	0.76	399.6	93.9	0.1	93.8	5.6
TOM57	12997.0	46.4	13068.0	50.5	0.55	435.6	158.4	0.2	158.2	3.9
EUR100	21596.6	77.3	21339.4	171.9	-1.19	528.2	802.9	0.3	801.6	8.3
KRO124	21555.1	93.4	21467.8	102.0	-0.41	562.0	823.0	0.5	822.2	16.3
KRO125	22480.7	92.2	22492.8	163.7	0.05	548.9	814.3	0.5	813.2	16.7
KRO126	21056.9	106.3	20928.4	168.8	-0.61	537.5	808.1	0.5	807.4	13.1
KRO127	21752.0	96.0	21436.1	122.5	-1.45	558.8	823.8	0.4	823.1	15.5
KRO128	22551.2	100.3	22454.9	190.3	-0.43	569.2	833.5	0.4	832.5	18.4
GRO120	7146.6	32.0	7057.2	72.0	-1.25	606.8	1370.7	0.4	1369.4	24.5
LIN318	43313.8	159.8	41957.4	176.8	-3.13	851.4	23967.8	3.7	23941.6	967.8
GRO442	5389.6	13.7	5147.0	20.9	-4.50	1011.0	58699.5	17.2	58674.6	432.3

solutions for repeated execution of the Lin-Kernighan algorithm was predetermined such that the average computation time of a macro-run for that instance was approximately the same as an average run of simulated annealing with the distance parameter δ set to 0.1. For computational reasons, it was not possible to calibrate computation times more precisely, for instance in the same way as in the previous comparison. For GRO442, calibration was impossible because of limited memory capacity of the computer; here, the Lin-Kernighan algorithm was allowed substantially less computation time than simulated annealing. The comparison is also biased in favour of simulated annealing by the fact that the cost values in the right-hand part of table 4.4 were obtained by simulated annealing, followed by iterative improvement based on 2-changes and a special type of 3-changes (those where two of the three cities are successors in the current tour).

The results of table 4.4 clearly indicate that repeated execution of the Lin-Kernighan algorithm is superior to the simulated annealing algorithm; in fact, even if the former is allowed only a limited fraction of the computation time taken by the latter, it still finds substantially better solutions. For the Krolak problems, for instance, the Lin-Kernighan algorithm can be executed for ca. 200 initial solutions, when a macro-run is allowed approximately 800 seconds of CPU time - of the returned 200 solutions one third to one half is globally minimal, so that even in a few seconds the Lin-Kernighan algorithm can be expected to find a globally minimal tour.
The difference in performance between simulated annealing and repeated execution of the Lin-Kernighan algorithm is even better illustrated by the results for non-Euclidean instances, as given in table 4.5. The cost value of the average solution returned by simulated annealing (followed by iterative improvement, as in the previous comparison) is compared with a conjectured globally minimum value, found by running the Lin-Kernighan algorithm on 50 (randomly generated) initial solutions; the latter typically takes some 20 seconds CPU time.
We observe that even for extremely small values of the distance parameter, the average solution returned by simulated annealing still deviates substantially in cost from the conjectured global minimum. However, even in this range of δ-values, smaller values for δ still lead

Table 4.4: Average cost of best solution ($\overline{C}_{best}$) or final solution ($\overline{C}_{final}$), average computation time in seconds ($\bar{t}$), standard deviations (σ_C and σ_t, respectively), average number of local minima per macro-run of the Lin-Kernighan algorithm ($\overline{lm}$) and % of average final cost value (annealing) above average best cost value (Lin-Kernighan) for different-sized instances of the travelling salesman problem. The results are obtained with repeated execution of the Lin-Kernighan routine and simulated annealing followed by 2-change and (limited) 3-change, respectively. The averages are obtained from five (macro-)runs (ten for KRO124 through KRO128).

	Lin-Kernighan		*Simulated Annealing*			*Lin-Kernighan*			*Simulated Annealing*	
Problem	$\overline{C}_{best}$	σ_C	$\overline{C}_{final}$	σ_C	%	$\overline{lm}$	$\bar{t}$	σ_t	$\bar{t}$	σ_t
GRO48	5046.0	0.0	5084.2	22.4	0.76	66	82.0	3.3	93.8	5.6
TOM57	12960.0	0.0	13063.4	50.9	0.84	84	166.6	7.6	158.2	3.9
EUR100	21135.0	0.0	21336.2	169.6	0.96	87	741.0	36.5	801.6	8.3
KRO124	21282.0	0.0	21406.3	130.9	0.58	206	699.7	12.7	822.2	16.3
KRO125	22141.0	0.0	22368.9	125.9	1.00	206	917.5	50.8	813.2	16.7
KRO126	20749.0	0.0	20896.7	148.1	0.71	206	618.7	16.9	807.4	13.1
KRO127	21294.0	0.0	21406.8	89.2	0.53	206	781.8	18.1	823.1	15.5
KRO128	22068.0	0.0	22374.1	185.5	1.39	206	774.0	22.1	832.5	18.4
GRO120	6943.8	3.6	7037.6	56.5	1.35	112	1378.4	42.2	1369.4	24.5
LIN318	41433.2	13.9	41862.2	231.6	1.04	214	22984.4	496.9	23941.6	967.8
GRO442	5080.8	5.7	5136.4	13.7	1.09	250	34263.8	558.5	58674.6	432.3

Table 4.5: Average cost of final solution ($\overline{C}_{final}$), average computation time in seconds ($\bar{t}$), standard deviations (σ_C and σ_t, respectively) and % of average final cost value above best cost value found by the Lin-Kernighan algorithm for different-sized instances of the travelling salesman problem. The results are obtained with the simulated annealing algorithm (followed by 2-change and (limited) 3-change) with different values of the distance parameter δ. The averages are obtained from five runs.

Problem	n	δ	$\overline{C}_{final}$	σ_C	%	$\bar{t}$	σ_t
RAN50a	50	10^{-1}	11826.2	490.0	16.10	113.2	2.0
		10^{-2}	10992.2	227.4	7.91	916.4	13.1
		10^{-3}	10757.0	175.4	5.61	9649.3	85.0
RAN50b	50	10^{-1}	11094.8	222.4	14.44	113.1	2.7
		10^{-2}	10721.4	322.4	10.59	932.4	12.8
		10^{-3}	10226.8	166.6	5.49	9594.7	103.1
RAN50c	50	10^{-1}	13103.4	833.1	14.50	111.5	4.3
		10^{-2}	12075.8	338.6	5.52	935.0	7.5
		10^{-3}	11919.2	150.2	4.15	9493.4	54.4
RAN50d	50	10^{-1}	12406.2	582.1	20.19	113.8	1.9
		10^{-2}	11073.4	365.5	7.28	941.5	18.5
		10^{-3}	10827.8	259.6	4.90	9638.6	111.7
RAN50e	50	10^{-1}	13134.8	881.7	22.26	109.4	5.3
		10^{-2}	12243.8	332.3	13.97	906.4	6.2
		10^{-3}	11403.8	263.0	6.15	9252.3	66.1

to better solutions, of course at the cost of a substantial increase in computation time.
From these experiments, we draw the following conclusions:

- The parameter δ serves as a handle in the trade-off between quality of final solution and computation time - smaller values of δ lead to better final solutions and larger computation times.

- If simulated annealing and repeated execution of iterative improvement are based on the same type of transition and if they are allowed the same amount of computation time, the former easily outperforms the latter for large problem instances.

- Simulated annealing cannot compete successfully with the Lin-Kernighan algorithm, a tailored approximation algorithm for the TSP.

4.3 The Job Shop Scheduling Problem

The general job shop scheduling problem can be formulated as follows (see e.g. [Lenstra, 1977]). There are n jobs $J_1, J_2, \ldots, J_n$ that have to be processed on m machines $M_1, M_2, \ldots, M_m$. Each job J_i $(i = 1, \ldots, n)$ consists of n_i operations $O_{i1}, O_{i2}, \ldots, O_{in_i}$ that have to be processed in a given order. Each operation O_{ij} has a processing time t_{ij}, during which it cannot be interrupted. A machine can handle only one operation at a time. Let M_k $(k = 1, \ldots, m)$ denote the set of m_k operations to be processed on machine M_k $(\sum_{k=1}^{m} m_k = \sum_{i=1}^{n} n_i = N)$ and let π_k denote a permutation of the set M_k. π_k is to be interpreted as the order in which the operations in M_k are processed on M_k: if operation O_{ij} is to be processed on machine M_k, $\pi_k(O_{ij})$ denotes the operation following O_{ij} on M_k. We introduce two fictitious operations O_0 and O_{N+1}, such that if O_{ij} and O_{st} are the first and last operations to be processed on machine M_k, respectively, $\pi_k(O_0) = O_{ij}$ and $\pi_k(O_{st}) = O_{N+1}$.
The problem is to find a set of permutations, one for each machine, such that

- the corresponding set of (machine) processing orders is not in conflict with the set of orders in which the operations of each job are to be carried out,
- the *maximum completion time* is minimized, i.e. the time to complete all jobs.

In order to apply simulated annealing, we need to define again the problem in terms of configurations, a cost function and a neighbourhood structure.
Each configuration of the problem corresponds to a set of m permutations. It is convenient to represent a configuration i by means of a weighted digraph G_i. Let $\Pi_i = \{\pi_{i1}, \ldots, \pi_{im}\}$ be the current set of permutations, then the digraph $G_i = (V, A \cup E_i)$ is defined as follows:

- Each vertex in V corresponds to an operation O_v, where it is assumed that the operations O_{ij} are renumbered as O_v, with $v = s_{i-1} + j$, where $s_i = \sum_{k=1}^{i} n_k$. Furthermore, two vertices 0 and $N+1$ are added to V, corresponding to the fictitious initial and final operations O_0 and O_{N+1}; t_0 and t_{N+1} are set to 0. The weight of a vertex v equals the processing time t_v of operation O_v.

- Each edge $e = (v, w)$ in A connects two vertices corresponding to consecutive operations of the same job. Thus, A contains all edges $(v, v+1)$ for which $v \neq s_i$ $(i = 1, \ldots, n)$. Furthermore, A contains edges connecting 0 with $1, s_1 + 1, \ldots, s_{n-1} + 1$ (the vertices corresponding to the first operation of each job) and connecting $s_1, s_2, \ldots, s_n$ (the vertices corresponding to the last operation of each job) with $N + 1$.

- Each edge $e = (v, w)$ in E_i connects two vertices corresponding to consecutive operations on the same machine. Thus, E_i contains all edges $e = (v, w)$ for which $O_w = \pi_{ik}(O_v)$ for some $k \in \{1, \ldots, m\}$.

Note that the set of edges A is common to all configurations. Figure 4.2 illustrates the graph G_i for a 3-job/3-machine instance, in which each job consists of three operations.

For each configuration i, we also define the digraph $\overline{G}_i = (V, A \cup \overline{E}_i)$, such that an edge (v, w) belongs to $\overline{E}_i$ if it belongs to E_i or if operation O_w is processed after operation O_v on some machine M_k and $O_w \neq \pi_{ik}(O_v)$. For example, for the instance shown in figure 4.2, $\overline{E}_i = E_i \cup \{(1,9),(2,8),(7,3)\}$.

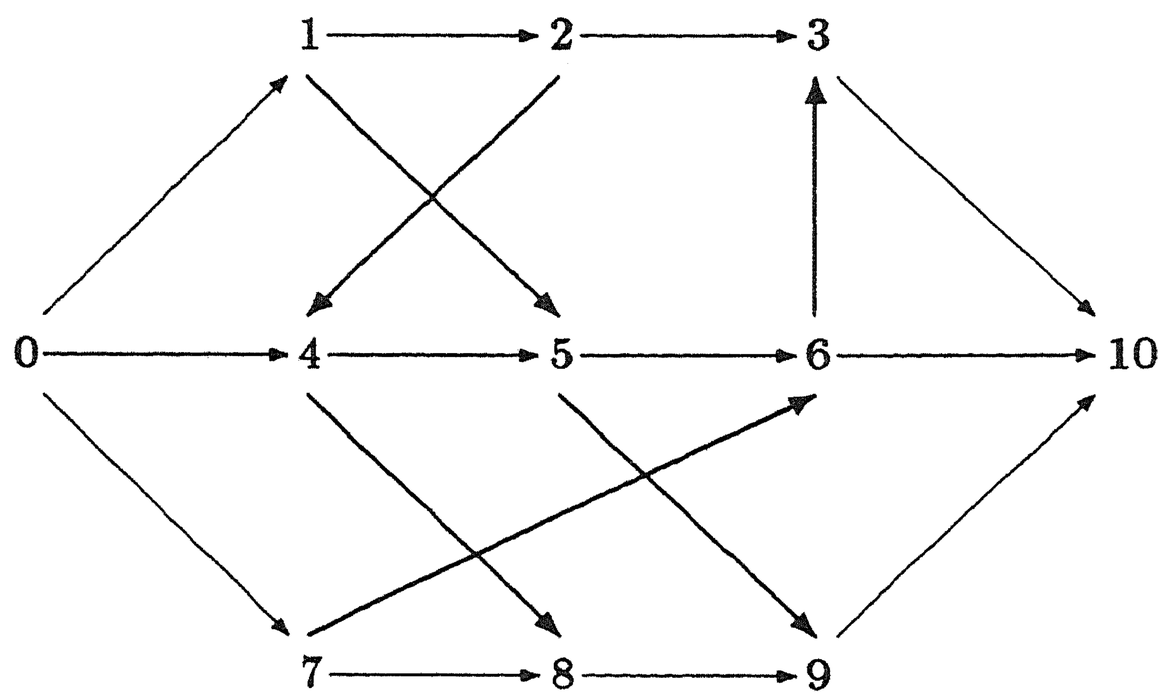


Figure 4.2: A configuration i of a 3-job/3-machine instance. Operations O_1, O_5 and O_9 are processed by machine M_1 in the aforementioned order, so are O_2, O_4 and O_8 by machine M_2, whereas O_3, O_6 and O_7 are processed by machine M_3 in the order (O_7, O_6, O_3). O_0 and O_{10} are the fictitious initial and final operations, respectively. Thin arrows denote edges in A, thick arrows edges in E_i.

We now use the following two observations [Lenstra, 1977]:

- A set Π_i of m permutations corresponds to a configuration i if and only if the corresponding digraph G_i is acyclic.
- If the digraph G_i is acyclic, the time to complete all jobs is given by the length of the *longest directed path* from vertex 0 to vertex $N+1$ in G_i. Here, the length of a path is given by the sum of the weights of the vertices on the path.

From the first observation we learn that the number of configurations is **no larger** than the number of sets Π_i, given by $\prod_{k=1}^{m} m_k!$.
From the second observation we learn that the cost of a configuration i is given by the longest directed path from 0 to $N+1$ in G_i. To compute such a cost, we use a simple *labelling algorithm*, based on Bellman's equations [Bellman, 1958], for solving the longest-path problem in a digraph $G = (V, E)$. The algorithm is based on processing the edges of G one by one and can be described as follows.

Step 0: Label all vertices to 0 ($label(v) := 0,\ \forall v \in V$). Set $k := 1$ and $E_k := E$.

Step 1: Seek an edge $e = (v, w)$ in the graph $G_k = (V, E_k)$, whose start vertex v has no predecessors. If such an edge cannot be found then terminate, else go to step 2.

Step 2: Relabel w to $\max(label(w), label(v) + weight(v))$. Set $E_{k+1} := E_k \setminus e$ and $k := k + 1$. Go to step 1.

If G is acyclic, the algorithm terminates after $|E|$ steps, when $E_{|E|+1} = \emptyset$. The label of each vertex then equals the length of the longest path from the source (the vertex in G without predecessors) to that vertex.
In our case, the sets A and E_i contain $N + n$ and $N + m$ edges, respectively. Consequently, the number of edges in the digraph G_i is $2N + n + m$ and accordingly the labelling algorithm takes $O(N)$ time to compute the length of the longest path from 0 to $N+1$. The longest path itself is constructed by building a path from 0 to $N + 1$, consisting of edges $e = (v, w)$, for which $label(v) + weight(v) = label(w)$. Usually, there are several longest paths from 0 to $N + 1$; we say that an edge is *critical* with respect to G_i if it is on some longest path in G_i.
Finally, a transition is generated by choosing vertices v and w, such that

1. O_v and O_w are successive operations on the same machine M_k,
2. (v, w) is a critical edge,

and reversing the order in which O_v and O_w are processed on machine M_k. Thus, in the digraph G_i such a transition results in reversing the edge connecting v and w and changing the edges (u, v) and (w, x) to (u, w) and (v, x), respectively, where $O_u = \pi_{ik}^{-1}(O_v)$ and $O_x = \pi_{ik}(O_w)$. Note that for each configuration i, the size of the neighbourhood $\mathcal{R}_i$ is bounded by

$$|\mathcal{R}_i| < \sum_{k=1}^{m}(m_k - 1) = N - m. \tag{4.14}$$

Our choice of transition is motivated by two facts:

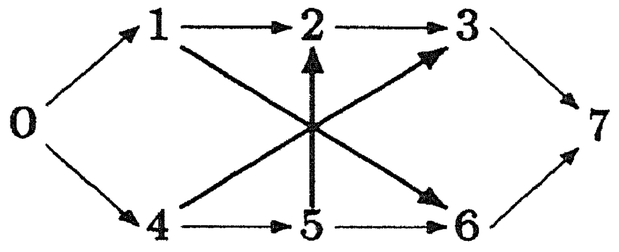
- Reversing a critical edge in a digraph G_i can never lead to a cyclic digraph G_j (see lemma 4.2).

- If the reversal of a non-critical edge in G_i leads to an acyclic graph G_j, the longest path q in G_j cannot be shorter than the longest path p in G_i (because G_j still contains the path p).

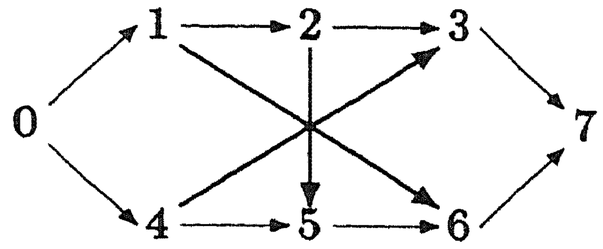
Thus, we exclude beforehand some cost-increasing transitions and, in addition, all transitions that might result in a cyclic digraph.

Summarizing, we associate to each configuration i an acyclic digraph G_i, the cost of a configuration i is given by the longest path in G_i and calculated by the labelling algorithm, and a transition corresponds to the reversal of a critical edge in G_i.

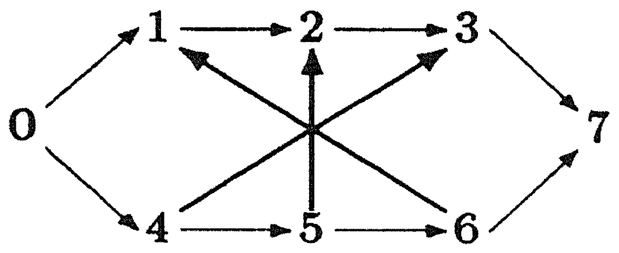
To study the asymptotic behaviour of the homogeneous algorithm applied to an instance of this problem, we recall that we need to check whether condition (1) of theorem 2.1 is satisfied. However, this condition cannot always be satisfied: in figure 4.3 an instance is shown in which there are pairs of configurations (i, j) for which it is not possible to construct a sequence of transitions leading from i to j. However, the subsequent theorem (theorem 4.1) states that for an arbitrary configuration i there is always a sequence of transitions leading from i to a globally minimal configuration. Thus, condition $(1)'$ is satisfied and, in accordance with the remarks on page 18, the homogeneous algorithm still converges asymptotically to a globally minimal configuration.
To prove theorem 4.1, we need two lemmas.



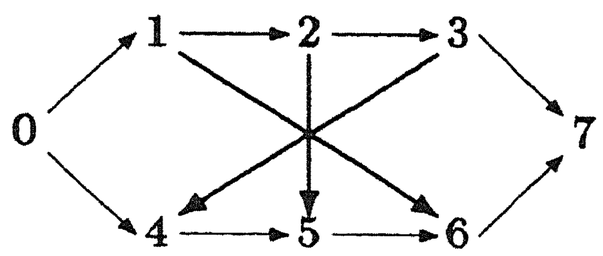
G_1 - longest path $= \{0, 4, 5, 2, 3, 7\}$



G_2 - longest path $= \{0, 1, 2, 5, 6, 7\}$



G_3 - longest path $= \{0, 4, 5, 6, 1, 2, 3, 7\}$



G_4 - longest path $= \{0, 1, 2, 3, 4, 5, 6, 7\}$

Figure 4.3: The four configurations of a 2-job/3-machine instance. Operations O_1 and O_6 are processed by machine M_1, O_2 and O_5 by machine M_2 and O_3 and O_4 by machine M_3. All operations have a processing time of one time unit; O_0 and O_7 are the fictitious initial and final operations, respectively. Thin arrows denote edges in A, thick arrows edges in E_i. Configurations 1 and 2 are globally minimal; it can be easily checked that 1 and 2 are reachable from 3 and 4, respectively, and from each other. However, 3 and 4 are not reachable from any other configuration.

Lemma 4.1
Consider an arbitrary configuration i and an arbitrary global minimum $i_0 \in \mathcal{R}_{opt}$. If $i \notin \mathcal{R}_{opt}$, then the set $K_i^{i_0}$ defined by

$$K_i^{i_0} = \{e = (v, w) \in E_i |\ e \text{ is critical } \wedge (w, v) \in \overline{E}_{i_0}\} \tag{4.15}$$

is not empty.

Proof

The proof consists of two parts: first, we show that E_i always contains critical edges, unless $i \in \mathcal{R}_{opt}$, next that there are always critical edges in E_i that do not belong to $\overline{E}_{i_0}$ unless again $i \in \mathcal{R}_{opt}$.

1. Suppose E_i contains no critical edges, then all critical edges belong to A. Consequently, the longest path consists of edges connecting vertices corresponding to operations of the same job; accordingly, its length is given by the total processing time of that job. But this is a lower bound to the longest path in any graph G_j, hence $i \in \mathcal{R}_{opt}$.

2. Suppose that for all critical edges e in E_i, we have that $e \in \overline{E}_{i_0}$. We then know that the longest path p in G_i is also a path q in $\overline{G}_{i_0}$. The **longest** path r in $\overline{G}_{i_0}$ is also the longest path in G_{i_0} and because $i_0 \in \mathcal{R}_{opt}$, we have $length(r) \leq length(p)$. But by definition $length(r) \geq length(q) = length(p)$. Consequently, $length(p) = length(r)$ and $i \in \mathcal{R}_{opt}$.

□

Lemma 4.2
Suppose $e = (v, w) \in E_i$ is a critical edge. Let G_j be the graph obtained from the acyclic graph G_i by reversing the edge e in E_i. Then G_j is also acyclic.

Proof

Suppose G_j is cyclic. Because G_i is acyclic, the edge (w, v) is part of the cycle in G_j. Consequently, there is a path $(v, x, y, \ldots, w)$ in G_j. But this path can also be found in G_i and is clearly a longer path from v to w than the edge (v, w). This contradicts the assumption that (v, w) is on a longest path in G_i. Hence, G_j is acyclic.

□

Given a configuration $i_0 \in \mathcal{R}_{opt}$, we now define the following two sets for an arbitrary configuration i:

$$M_i^{i_0} = \{e = (v, w) \in E_i | (w, v) \in \overline{E}_{i_0}\} \tag{4.16}$$

$$\overline{M}_i^{i_0} = \{e = (v, w) \in \overline{E}_i | (w, v) \in \overline{E}_{i_0}\}. \tag{4.17}$$

eorem 4.1
· each configuration $i \notin \mathcal{R}_{opt}$ it is possible to construct a sequence transitions leading from i to a globally minimal configuration.

Proof

We choose an arbitrary configuration $i_0 \in \mathcal{R}_{opt}$ and construct a sequence of configurations $\{\lambda_0, \lambda_1, \ldots\}$ as follows:

1. $\lambda_0 = i$
2. λ_{k+1} is obtained from λ_k by reversing an edge $e \in K^{i_0}_{\lambda_k}$ in E_{λ_k}. According to lemma 4.2, this can be done without creating a cycle in $G_{\lambda_{k+1}}$. Furthermore, this operation is of the aforementioned type of transition.

It can easily be seen that for all k,

$$|\overline{M}^{i_0}_{\lambda_{k+1}}| = |\overline{M}^{i_0}_{\lambda_k}| - 1. \tag{4.18}$$

Hence, for $k = |\overline{M}^{i_0}_i|$, $|\overline{M}^{i_0}_{\lambda_k}| = 0$. Using $K^{i_0}_i \subseteq M^{i_0}_i \subseteq \overline{M}^{i_0}_i$, we find $K^{i_0}_{\lambda_k} = \emptyset$ for $k = |\overline{M}^{i_0}_i|$. According to lemma 4.1, this implies $\lambda_k \in \mathcal{R}_{opt}$.

□

The empirical analysis of the finite-time behaviour is carried out by running the homogeneous algorithm on a number of instances, varying in size from 6 jobs on 6 machines to 30 jobs on 10 machines. For all instances, the number of operations of each job equals the number of machines. In that case, the number of configurations of each instance is bounded by $(n!)^m$, the labelling algorithm takes $O(n \cdot m)$ time to compute the cost of a configuration and the size of the neighbourhood of a configuration is bounded by $m(n-1)$.
FIS1, FIS2 and FIS3 are three instances due to Fisher & Thompson [1963], the forty instances in tables 4.7 and 4.8 are due to Lawrence [1984]. FIS2 is a notorious 10-job/10-machine instance that has defied solution for more than 20 years. A couple of years ago, a solution with cost 930 was found after a combination of manual operations and several hours of computation time. This solution was only recently proved to be globally minimal [Carlier & Pinson, 1986]. For FIS1, FIS2 and FIS3, the processing times of the operations are randomly drawn and range from 1 to 10 (FIS1) or 99 (FIS2 and FIS3) units of time. The sequence of machines for each job is such that

lower-numbered machines tend to be used for earlier operations. For the Lawrence instances, processing times are drawn from a uniform distribution on the interval [5,99]; the sequence of machines for each job is random.

The performance of simulated annealing on these instances is reported in table 4.6 for the Fisher-Thompson instances, and in tables 4.7 and 4.8 for the Lawrence instances. The averages in these tables are computed from five solutions, obtained by running the algorithm, controlled by the cooling schedule described in section 3.3, five times on each instance and recording the best configuration encountered during each run (this need not necessarily be the final configuration). All results are obtained with the parameters χ_0 and ϵ_s set to 0.95 and 10^{-6}, respectively, and for different values of the distance parameter δ. For the Lawrence instances, δ is chosen such that the average best solution is comparable in cost to the best known solution for each instance (as reported by Adams, Balas & Zawack [1988]).

From tables 4.6, 4.7 and 4.8 we can make similar observations as for the travelling salesman problem (see section 4.2):

- From table 4.6 we observe that both the quality and the reliability of the average best solution increase with decreasing δ; the difference between the average best solution and a globally minimal one does not deteriorate significantly with increasing problem size. For the notorious second instance, the five best solutions returned have cost values of 937 (twice), 945 (twice) and 948, respectively.

- As for computation times, we remark that the bound for the computation time given by theorem 7 is $O((nm)^3 \ln n)$ ($L = O(nm)$, $|\mathcal{R}| = O((n!)^m)$ and $\tau = O(nm)$). Thus, for fixed m the bound is $O(n^3 \ln n)$, for fixed n it is $O(m^3)$. For the A, B and C instances in table 4.7, for which m is constant and for which the displayed results are obtained with the same value of δ, the average computation time $\bar{t}$ is approximately given by $\bar{t} = t_0 \cdot n^{2.215} \cdot \ln n$, for some constant t_0 ($\chi^2 = 1.00$); for instances G2, B2 and I2, for which n is constant and for which the displayed results are again obtained with the same value of δ, the average computation time is approximately given by

Table 4.6: Average cost of best solution ($\overline{C}_{best}$), average computation time in seconds ($\bar{t}$), standard deviations (σ_C and σ_t, respectively), cost of best solution (C_{best}), number of local minima per macro-run of iterative improvement ($\overline{lm}$) and % of average best cost value above globally minimal cost value for different-sized instances of the job shop scheduling problem. The results are obtained with the simulated annealing algorithm with different values of the distance parameter δ (upper part) and with repeated execution of the iterative improvement algorithm (lower part), respectively. The averages are obtained from five (macro-)runs.

			Simulated annealing						
Prob	m	n	δ	$\overline{C}_{best}$	σ_C	%	C_{best}	$\bar{t}$	σ_t
FIS1	6	6	10^{-1}	56.0	1.3	1.82	55	8	1
			10^{-2}	55.0	0.0	0.00	55	52	8
FIS2	10	10	10^{-1}	1039.6	15.1	11.78	1028	113	13
			10^{-2}	985.8	22.1	6.00	951	779	61
			10^{-3}	942.4	4.5	1.33	937	5945	180
FIS3	20	5	10^{-1}	1354.2	26.5	16.24	1325	123	13
			10^{-2}	1229.0	33.6	5.49	1184	848	93
			10^{-3}	1187.0	18.7	1.89	1173	6840	389

			Iterative Improvement						
Prob	m	n	$\overline{lm}$	$\overline{C}_{best}$	σ_C	%	C_{best}	$\bar{t}$	σ_t
FIS1	6	6	803.2	55.4	0.8	0.73	55	52	0
FIS2	10	10	9441.2	1018.2	9.1	9.48	1006	5945	0
FIS3	20	5	5221.0	1331.4	9.5	14.28	1319	6841	0

$\bar{t} = t_1 \cdot m^{2.406}$, for some constant t_1 ($\chi^2 = 1.00$). Thus, the observed computation times are in good accordance with the bound given by theorem 7.

Table 4.6 also contains results obtained by repeated execution of the iterative improvement algorithm based on the same neighbourhood structure as simulated annealing. The averages are again obtained from 5 macro-runs, where each macro-run is terminated as soon as the computation time exceeds the computation time of an average run of simulated annealing with the distance parameter δ set to 10^{-3}.
We observe that repeated execution of iterative improvement is easily outperformed by simulated annealing for the two larger problems. In comparison with the travelling salesman problem, the difference between simulated annealing and iterative improvement is even more pronounced: for FIS3, for instance, the average best solution obtained by simulated annealing is almost 11% better in cost than the one obtained by repeated execution of iterative improvement.
Tables 4.7 and 4.8 also contain for each instance the cost value of the best solution obtained by the algorithm of Adams, Balas & Zawack [1988]. This is a sophisticated approximation algorithm tailored to the job shop scheduling problem; it is based on a partial enumeration of configurations using branch-and-bound techniques. In [Adams, Balas & Zawack, 1988], the algorithm is shown to be superior to approaches based on *priority dispatching* rules (rules for choosing an operation from a specified subset of as yet unscheduled operations): the typical improvement is reported to be between 4% and 10%. The computation time taken by this algorithm is significantly less than that of simulated annealing: for FIS2, for instance, the algorithm takes approximately 15 minutes on a VAX 11/780 computer (which amounts to ca. 7 minutes on a VAX 11/785).
From tables 4.7 and 4.8 we observe that for all instances where Adams, Balas & Zawack find a globally minimal solution, the average best solution obtained by simulated annealing is also globally minimal, except for instances C1, D3 and D5, where it is slightly higher in cost than a global minimum. For all instances where Adams, Balas & Zawack do not find a globally minimal solution, the best solution obtained in 5 runs of the simulated annealing algorithm is significantly

Table 4.7: Average cost of best solution ($\overline{C}_{best}$), computation time ($\bar{t}$, in seconds on a VAX 11/785), standard deviations (σ_C and σ_t, respectively) and best cost of 5 runs (C_{best}) for different-sized instances of the job shop scheduling problem. The averages are obtained from 5 runs of the simulated annealing algorithm. The table also includes for each instance the cost of the best solution found by Adams, Balas & Zawack [1987] (C_A). Provably globally minimal solutions are marked with an asterisk.

	m	n	δ	$\overline{C}_{best}$	σ_C	C_{best}	C_A	$\bar{t}$	σ_t
A1	10	10	10^{-2}	966.2	10.1	956	978	685.7	83.3
A2	10	10	10^{-2}	787.8	1.6	785	787	719.7	109.0
A3	10	10	10^{-2}	861.2	0.4	861	859	672.6	69.0
A4	10	10	10^{-2}	853.4	4.6	848	860	830.1	85.4
A5	10	10	10^{-2}	908.4	4.2	902	914	667.4	126.9
B1	10	15	10^{-2}	1067.6	3.7	1063	1084	1991.0	341.1
B2	10	15	10^{-2}	944.2	4.7	938	944	2163.2	154.6
B3	10	15	10^{-2}	1032.0	0.0	1032*	1032*	2092.9	89.7
B4	10	15	10^{-2}	966.6	8.7	952	976	2097.7	406.0
B5	10	15	10^{-2}	1004.4	14.4	992	1017	2133.4	374.5
C1	10	20	10^{-2}	1219.0	2.0	1218*	1218*	4342.4	597.8
C2	10	20	10^{-2}	1273.6	5.2	1269	1291	4535.3	392.0
C3	10	20	10^{-2}	1244.8	15.4	1224	1250	4354.0	349.8
C4	10	20	10^{-2}	1226.0	6.5	1218	1239	4408.4	450.9
C5	10	20	10^{-2}	1355.0	0.0	1355*	1355*	3956.4	428.2
D1	10	30	10^{-1}	1784.0	0.0	1784*	1784*	1517.4	58.1
D2	10	30	10^{-1}	1850.0	0.0	1850*	1850*	1752.2	124.6
D3	10	30	10^{-1}	1726.6	15.2	1719*	1719*	1880.1	130.8
D4	10	30	10^{-2}	1721.0	0.0	1721*	1721*	11075.4	402.9
D5	10	30	10^{-1}	1890.0	4.0	1888*	1888*	1667.6	107.9

Table 4.8: Average cost of best solution ($\overline{C}_{best}$), computation time ($\bar{t}$, in seconds on a VAX 11/785), standard deviations (σ_C and σ_t, respectively) and best cost of 5 runs (C_{best}) for different-sized instances of the job shop scheduling problem. The averages are obtained from 5 runs of the simulated annealing algorithm. The table also includes for each instance the cost of the best solution found by Adams, Balas & Zawack [1987] (C_A). Provably globally minimal solutions are marked with an asterisk.

	m	n	δ	$\overline{C}_{best}$	σ_C	C_{best}	C_A	$\bar{t}$	σ_t
F1	5	10	10^{-1}	666.0	0.0	666*	666*	20.2	3.5
F2	5	10	10^{-3}	655.0	0.0	655*	655*	1066.7	45.0
F3	5	10	10^{-3}	601.6	5.4	597	605	1041.8	63.0
F4	5	10	10^{-3}	590.0	0.0	590	593	975.1	191.8
F5	5	10	10^{-1}	593.0	0.0	593	593	19.0	4.2
G1	5	15	10^{-1}	926.0	0.0	926*	926*	51.9	5.8
G2	5	15	10^{-2}	890.0	0.0	890*	890*	376.2	48.3
G3	5	15	10^{-1}	863.0	0.0	863*	863*	54.9	7.3
G4	5	15	10^{-1}	951.0	0.0	951*	951*	47.1	5.9
G5	5	15	10^{-1}	958.0	0.0	958*	959	44.6	2.0
H1	5	20	10^{-1}	1222.0	0.0	1222*	1222*	107.8	17.2
H2	5	20	10^{-2}	1039.0	0.0	1039*	1039*	655.1	30.7
H3	5	20	10^{-1}	1150.0	0.0	1150*	1150*	117.6	18.0
H4	5	20	10^{-1}	1292.0	0.0	1292*	1292*	93.1	20.6
H5	5	20	10^{-2}	1207.0	0.0	1207*	1207*	735.6	26.3
I1	15	15	10^{-2}	1300.0	7.8	1293	1305	5345.6	399.8
I2	15	15	10^{-2}	1442.4	5.7	1433	1423	5287.4	688.5
I3	15	15	10^{-2}	1227.2	8.2	1215	1215	5479.8	614.8
I4	15	15	10^{-2}	1258.2	5.2	1248	1248	5765.6	800.3
I5	15	15	10^{-2}	1247.4	9.9	1234	1234	5373.4	1066.4

lower in cost than the best solution found by Adams, Balas & Zawack (the typical improvement is between 1 and 3 %); the average best solution is only marginally better.
Admittedly, the performance of the tailored algorithm is likely to improve if it is allowed more computation time. Nevertheless, the results in tables 4.7 and 4.8 indicate that simulated annealing is a promising approach to job shop scheduling and certainly superior to traditional approaches to this problem, such as procedures based on priority dispatching rules.

4.4 The Football Pool Problem

Consider the set V_k^n of all n-tuples $\mathbf{x} = (x_1, \ldots, x_n)$ with elements x_i in $Z_k = \{0, \ldots, k-1\}$ and define the *Hamming distance* $d_H(\mathbf{x}, \mathbf{y})$ between two tuples $\mathbf{x}$ and $\mathbf{y}$ in V_k^n as the number of positions in which $\mathbf{x}$ and $\mathbf{y}$ differ, i.e.

$$d_H(\mathbf{x}, \mathbf{y}) = |\{i \in \{1, 2, \ldots, n\} | x_i \neq y_i\}|. \tag{4.19}$$

The *rook domain* $R(\mathbf{x})$ of a tuple $\mathbf{x}$ in V_k^n is defined as

$$R(\mathbf{x}) = \{\mathbf{y} \in V_k^n | d_H(\mathbf{x}, \mathbf{y}) \leq 1\}. \tag{4.20}$$

A subset W of V_k^n is called a *covering by rook domains* of V_k^n if

$$V_k^n = \bigcup_{\mathbf{x} \in W} R(\mathbf{x}). \tag{4.21}$$

Equation (4.21) implies that each tuple in V_k^n is on a Hamming distance not larger than one from at least one tuple in W. Hereinafter, the tuples in W are called *rooks*.
We are now interested in determining the size of the smallest subset W of V_k^n satisfying (4.21), i.e. the size $\sigma(n, k)$ of a *minimal covering* of V_k^n. For $k = 3$, this problem is known as the *football pool problem*: in a football pool of n matches, $\sigma(n, 3)$ is the minimal number of forecasts containing at least one forecast with at most one incorrect result. Hereinafter, we only consider the case $k = 3$ and write σ_n for $\sigma(n, 3)$.

Table 4.9: Some results for σ_n, n between 2 and 8.

n	2	3	4	5	6	7	8
σ_n	3	5	9	27	≤ 79	≤ 216	≤ 567

Wille's paper [Wille, 1987] is the first attempt to solve the football pool problem by simulated annealing: table 4.9 is taken from this paper and contains the values of σ_2 through σ_5 and upper bounds for σ_6 through σ_8. All entries in the table are the result of work of a combinatorial nature;[1] indeed, Wille's paper is the first attempt to solve the problem by stating it as an optimization problem and following an algorithmic approach. The main result is a new upper bound to σ_6: 74 instead of the previously known value of 79.
Here, we discuss this problem for two reasons:

- It is challenging, because globally minimal solutions of its instances are not known for $n > 5$.

- It illustrates clearly the care one has to take in formulating a problem in terms of configurations, a cost function and a neighbourhood structure. Because it is mainly discussed to illustrate this implementation aspect and because of the limited amount of computational experiments, we do not address asymptotic convergence in this section nor do we extensively analyse the numerical data obtained.

Both aspects are also characteristics of many problems in for instance *computer-aided circuit design* to which simulated annealing has been successfully applied; see [Van Laarhoven & Aarts, 1987].

Each configuration i of the problem corresponds to a covering W_i of V_k^n. Thus, the total number of configurations $|\mathcal{R}|$ is bounded by the total number of subsets of V_k^n, given by $2^{|V_k^n|}$. Since $|V_k^n| = 3^n$, we have $|\mathcal{R}| < 2^{3^n}$.

[1] We refer to [Wille, 1987] for references to the papers in which the entries of table 4.9 were originally published.

If a subset W_i of V_k^n is not a covering of V_k^n, the corresponding configuration or solution i is said to be *partial*, otherwise it is called *feasible*. Thus, in the case of a partial solution i, V_k^n contains tuples not *covered* by any rook in W_i. A tuple $\mathbf{x}$ in V_k^n is said to be covered by a tuple $\mathbf{y}$ in V_k^n if $d_H(\mathbf{x},\mathbf{y}) \leq 1$. A partial solution j is easily transformed into a feasible solution k by adding those tuples to W_j that are not covered by any rook in W_j, i.e. by putting

$$W_k = W_j \cup \{\mathbf{y} \in V_k^n | \mathbf{y} \notin \bigcup_{\mathbf{x} \in W_j} R(\mathbf{x})\}. \tag{4.22}$$

Since the cost function is related to the generation mechanism employed for transitions, we discuss the two items together. For each mechanism we also derive values or bounds for $|\mathcal{R}_i|$, the size of the neighbourhood of a configuration i (we recall that in the cooling schedule employed, $|\mathcal{R}_i|$ determines the length of the Markov chains; see section 3.3). We have carried out numerical experiments with three generation mechanisms:

- *mechanism A*
  A transition is generated by either deleting a rook from the current covering W_i or adding a tuple $\mathbf{x} \notin W_i$ to W_i (the two alternatives are each chosen with a probability of 50%). If the first alternative results in a subset W_j which is not a covering, (4.22) is used to transform the partial configuration j into a feasible configuration k. In this way, we achieve that the algorithm generates feasible configurations only. The cost of a configuration i is given by the number of rooks in the covering W_i. The average value of $|\mathcal{R}_i|$ is given by

$$\overline{|\mathcal{R}_i|} = 0.5 \cdot \rho_i + 0.5 \cdot (3^n - \rho_i) = 0.5 \cdot 3^n, \tag{4.23}$$

  where ρ_i is the number of rooks in the covering W_i.

- *mechanism B*
  A transition is generated by either deleting a rook from the current covering W_i or adding a tuple $\mathbf{x} \notin W_i$ to W_i (the two alternatives are each chosen with a probability of 50%). Note that the first type of transition may result in a partial configuration. The cost of a (partial) configuration i is now given by the

sum of the number of rooks in W_i and the number of uncovered tuples in V_k^n. The second term penalizes the extent to which a partial configuration is not feasible.
If the algorithm eventually returns a partial configuration j, it is transformed into a feasible configuration k by a simple algorithm, based on a complete enumeration of all pairs of tuples not yet covered by a rook in W_j. If $d_H(\mathbf{x},\mathbf{y}) \leq 2$ for such a pair $(\mathbf{x},\mathbf{y})$, it is straightforward to show that

$$\exists \mathbf{z} \in V_k^n : \; d_H(\mathbf{x},\mathbf{z}) \leq 1 \wedge d_H(\mathbf{y},\mathbf{z}) \leq 1. \qquad (4.24)$$

The tuple $\mathbf{z}$ is then added to W_j. After the enumeration, the tuples still uncovered are also added to W_j to obtain a covering W_k. Note that $C(k) \leq C(j)$.
The average value of $|\mathcal{R}_i|$ is again given by (4.23).

- *mechanism C*
  A transition is generated by replacing a rook in the current covering W_i by a tuple $\mathbf{x} \notin W_i$. In this case all configurations, partial or feasible, correspond to subsets of V_k^n of the same cardinality s. In other words, the algorithm attempts to find a covering of V_k^n with s rooks. The cost of a (partial) configuration is now given by the number of uncovered tuples in V_k^n.
  If the algorithm eventually returns a partial configuration j, the aforementioned enumeration is used to obtain a feasible configuration k. Finally, $|\mathcal{R}_i|$ is given by

$$|\mathcal{R}_i| = s(3^n - s). \qquad (4.25)$$

Mechanism C is the same as the one used by Wille [1987], except that in the latter mechanism a rook $\mathbf{y}$ is replaced by a tuple $\mathbf{x} \in R(\mathbf{y}) \setminus W_i$.

The performance of simulated annealing on the football pool problem for 6 matches is given in table 4.10. The averages are computed from five solutions, obtained by running the algorithm five times with each of the aforementioned mechanisms. If necessary, the final solution found by using mechanisms B or C is transformed into a feasible solution by means of the enumeration described above. All results are

Table 4.10: Average number of rooks in final configuration of the 6-match problem ($\bar{\rho}_{final}$), average computation time in seconds ($\bar{t}$) and standard deviations (σ_ρ and σ_t, respectively). The results are obtained with the simulated annealing algorithm with different generation mechanisms for transitions and different values of the distance parameter δ. The averages are obtained from 5 runs.

Mechanism	δ	$\bar{\rho}_{final}$	σ_ρ	$\bar{t}$	σ_t
A	0.10	126.4	2.4	1400.8	79.6
B	0.01	86.0	0.6	410.9	4.0
C	10.00	83.6	1.9	668.6	28.1
C	0.10	79.0	2.5	8175.8	254.5

obtained with the parameters χ_0 and ϵ_s of the cooling schedule set to 0.95 and 10^{-6}, respectively. Since different Markov chain lengths are used in each of the three alternatives and, moreover, the computation time of one transition is different for each alternative, identical values of the distance parameter δ would result in widely different computation times. To avoid this, the value of δ is chosen such that the resulting computation times are somewhat comparable. The results for mechanism C are obtained with $s = 70$.

From table 4.10 we observe that the results obtained by the algorithm with mechanism A are inferior to those obtained by the other implementations, in terms of both cost and computation time. This seems to indicate the necessity to allow the algorithm to generate partial solutions in order to obtain near-optimal solutions. Note that the results obtained by the algorithm with mechanisms B or C are comparable: C is slightly better, but takes slightly more computation time as well.

The best solutions we obtained during a large number of runs consisted of 73 rooks. These results were typically found in ca. 8000 seconds of CPU time. Consequently, we conclude that $\sigma_6 \leq 73$, which is a slight improvement over Wille's result and a significant one over

the bound found by combinatorial analysis (as displayed in table 4.9). Using simulated annealing, we have also improved the upper bounds for σ_7 and σ_8, to 186 and 486, respectively. For a detailed description of these results the reader is referred to [Van Laarhoven, Aarts, Van Lint & Wille, 1988].

4.5 Concluding remarks

The preceding sections illustrate that simulated annealing is indeed a widely applicable optimization technique. Moreover, the results described in various sections are consistent in the conclusions which can be drawn from them:

- The algorithm has a potential for finding high-quality solutions; a sufficient amount of computation time to realize this potential is usually quite large.

- The probabilistic element of the algorithm (the acceptance of cost-increasing transitions with a non-zero probability) makes simulated annealing a significantly better technique than the iterative improvement algorithm on which it is based. The difference between the two algorithms is especially pronounced for large problem instances.

- Simulated annealing is not a panacea: if a sophisticated tailored algorithm is available from the literature, it is usually competitive with and often superior to simulated annealing.

- Formulating a problem in a way that lends itself to application of simulated annealing is not a trivial task; once a formulation is found, the algorithm is easy to implement.

With respect to the last conclusion, it is appropriate to make two further remarks:

- Although we have tried to equalize the influence of computation times in the comparisons between simulated annealing and tailored algorithms, we should add that computation times 'may

depend as much on the effort and skill applied to the programming as on the algorithms themselves' [Johnson, Aragon, McGeoch & Schevon, 1987]. Thus, even if algorithms are allowed the same amount of computation time, it is difficult to guarantee equality of treatment in this respect. This leaves the first conclusion unaffected, however.

- We did not investigate the possibility of improving the performance of simulated annealing by introducing more problem-specific elements into the algorithm, besides the three basic ones (configurations, a cost function and a neighbourhood structure). For large instances of the TSP, for example, one could think of partitioning the set of cities into subsets, each of which contains a number of cities situated close to each other, and restricting transitions to pairs of cities belonging to the same subset. Such an approach is followed by Bonomi & Lutton [1984] - they report satisfactory solutions of TSPs with up to 10 000 cities.

Chapter 5

A Bayesian approach to simulated annealing

5.1 Introduction

In chapters 3 and 4, the finite-time behaviour of the simulated annealing algorithm is discussed in terms of the running time required by the algorithm and the difference in cost between the returned solution and a globally minimal one. A worst-case bound for the running time required by a particular implementation of the algorithm is derived in chapter 3, and in chapter 4 both running time and difference are analysed in an empirical way. In this chapter we analyse the finite-time behaviour of the algorithm by considering the algorithm from a *Bayesian* point of view: the configurations constituting the Markov chains are seen as the outcome of a *random experiment*, with unknown parameters characterizing the probability distribution from which this outcome is generated. Given the outcome, i.e. the Markov chain, and the probability distribution, we would like to draw conclusions about the parameters, for instance by computing their expected values. This *inference problem* and its Bayesian solution are the main subjects of this chapter. Since we assume that the configurations are sampled from the stationary distribution of the corresponding Markov chain, the Bayesian solution amounts to the development of a Bayesian mechanism to estimate the ever-changing stationary distribution.
After a brief digression on Bayes's method in section 5.2, where we

closely follow [Boender, 1984], we return to simulated annealing in section 5.3, where we use Bayes's method to analyse the finite-time behaviour of simulated annealing. Numerical experiments are described in section 5.4, and in section 5.5 we show how the Bayesian information can be used to derive 'optimal' rules for choosing some of the parameters of a cooling schedule. We end this chapter with some concluding remarks.

5.2 Bayes's method

Consider a random experiment whose outcome is generated from a probability distribution with unknown parameters. An example of such an experiment is the sampling of configurations of a combinatorial optimization problem $(\mathcal{R}, C)$; if we assume that the cost function C takes its values from the set of integers $\{l, \ldots, u\}$, the outcome of the experiment can be described by the values taken by the random variables $N_l, N_{l+1}, \ldots, N_u$. Here, the *frequency count* N_i $(i = l, \ldots, u)$ denotes the number of configurations with cost value i in the outcome of the experiment. If the fraction of configurations with cost value i $(i = l, \ldots, u)$ in the population is equal to θ_i and if the configurations are sampled from a uniform distribution on $\mathcal{R}$, the frequency counts are well known to follow a *multinomial distribution* with parameters $\theta_l, \ldots, \theta_u$ [Feller, 1950], i.e.

$$Pr\{(N_l, \ldots, N_u) = (n_l, \ldots, n_u)\} = \begin{cases} \dfrac{n!}{\prod_{i=l}^{u} n_i!} \prod_{i=l}^{u} \theta_i^{n_i} & \text{if } \sum_{i=l}^{u} n_i = n, \\ 0 & \text{elsewhere,} \end{cases} \tag{5.1}$$

where n is the *sample size*, i.e. the number of configurations sampled in the experiment.

In this chapter, we consider the case where the parameters characterizing the distribution from which the outcome is generated are unknown. Hereinafter, we refer to this distribution as the *likelihood function*. In our example, these parameters are the minimum and maximum cost values, l and u, respectively, and the values θ_i of the

fractions of configurations with cost value i. The problem we are interested in is to say something about the values of the parameters, given the outcome of the experiment and the likelihood function.
The Bayesian solution to this inference problem is due to the Reverend Thomas Bayes [1763] and can be described as follows. It is assumed that some information is available on the values of the unknown parameters in the form of a probability distribution. This distribution is known as the *prior distribution.* Given the prior distribution, the outcome of the experiment and the likelihood function, Bayes's formula is used to derive another probability distribution, known as the *posterior distribution.* The latter is the Bayesian answer to the aforementioned inference problem.
Bayes's formula is given by

$$Pr\{X = x|Y = y\} = \frac{Pr\{X = x\} \cdot Pr\{Y = y|X = x\}}{\sum_x Pr\{X = x\} \cdot Pr\{Y = y|X = x\}} \qquad (5.2)$$

and is immediately obtained from the definition of a conditional probability and the equality

$$Pr\{Y = y\} = \sum_x Pr\{Y = y|X = x\} \cdot Pr\{X = x\}. \qquad (5.3)$$

In the inference problem, the random variables X and Y denote the set of unknown parameters and the outcome of the experiment, respectively. Thus, $Pr\{X = x\}$ and $Pr\{X = x|Y = y\}$ are the aforementioned prior and posterior distributions, respectively, and $Pr\{Y = y|X = x\}$ is the likelihood function (given values of the parameters). Given the outcome of the experiment, Bayes's formula converts the prior distribution on the values of the parameters into a posterior distribution.
The use of a prior distribution introduces a subjective element in Bayesian statistics, because the choice of a prior distribution is usually left to the user. Consequently, different assumptions about the prior distribution will not necessarily lead to identical posterior distributions, as a result of which 'Bayesians' are sometimes accused of fabricating data. However, this criticism is only valid in situations where the prior distribution is not generally agreed upon and where the posterior distribution is predominantly determined by the prior

distribution and not by the outcome of the experiment. In the experiments we study in this chapter the latter is certainly not the case, because of the large amounts of data constituting the outcomes of the experiments - the length of the Markov chains varies between 10^3 and 10^6 in the TSP-instances to which the Bayesian approach is applied in section 5.5. We can therefore call upon the observation that, although Bayesian solutions obtained under distinct prior distributions are different, they converge to the same answer for $n \rightarrow \infty$ under certain mild conditions on the prior distributions. We would also like to point out that 'non-Bayesians' can be considered to be merely unsophisticated Bayesians, since they (the non-Bayesians) recognize only the trivial prior distributions which assign to every possible value of the unknown parameter a probability of either 0 or 1.
In addition, it is often thought that the use of a prior distribution entails the controversial *subjective interpretation* of probabilities.[1] However, Von Mises [1964], a chief exponent of the relative frequency approach, argues that this view is definitely wrong. For a discussion of Von Mises's arguments the reader is referred to [Boender, 1984]; here, it suffices to quote Boender's conclusion that 'the arguments put forward by Von Mises (...) also suffice to justify our assumptions on the existence of probability distributions on relevant problem parameters' [Boender, 1984, p. 13].
Before returning to simulated annealing, we consider once again our example and apply Bayes's method to it. First, we need to assume a prior distribution on the random variables L, U and $\Theta_L, \ldots, \Theta_U$, corresponding to the values of the unknown parameters l, u, and $\theta_l, \ldots, \theta_u$, respectively. For the random variables corresponding to the minimum and maximum values of the cost function, L and U, respectively, we assume arbitrary prior distributions, i.e.

$$Pr\{L = l\} = \lambda_l,\ l = m_m, \ldots, m_M \tag{5.4}$$

and

$$Pr\{U = u\} = \upsilon_u,\ u = x_m, \ldots, x_M, \tag{5.5}$$

[1] This is the approach in which the statement 'event A occurs with a probability $\frac{2}{5}$' is believed to express someone's personal or subjective belief, as opposed to the *relative frequency approach*, in which such a statement expresses that the outcome of a certain experiment is A two out of five times [Hogg & Craig, 1978].

where the *hyperparameters* $m_m, m_M, x_m, x_M, \lambda_{m_m}, .., \lambda_{m_M}, \upsilon_{x_m}, .., \upsilon_{x_M}$ have to be provided by the user. Next, given $L = l$ and $U = u$, the random variables $\Theta_l, \ldots, \Theta_u$ are assumed to follow a *Dirichlet distribution*, which is well known to be the *natural conjugate prior*[2] for the parameters $\theta_l, \ldots, \theta_u$ of the multinomial distribution given by (5.1) [Lindley, 1978]. The Dirichlet distribution with parameters $\alpha_l, \ldots, \alpha_u$, which is denoted by $D(\alpha_l, \ldots, \alpha_u)$, is given by the following joint probability density function [Wilks, 1962]:

$$f(\theta_l, \ldots, \theta_u) = \begin{cases} \dfrac{\Gamma\left(\sum_{i=l}^{u} \alpha_i\right)}{\prod_{i=l}^{u} \Gamma(\alpha_i)} \prod_{i=l}^{u} \theta_i^{\alpha_i - 1} & \text{if } \sum_{i=l}^{u} \theta_i = 1, \\ 0 & \text{elsewhere.} \end{cases} \tag{5.6}$$

The choice of a Dirichlet distribution is only a minor restriction, because its parameters can be chosen such that the expected values $E[\Theta_i]$, $i = l, \ldots, u$, with respect to the Dirichlet distribution are equal to those with respect to any other distribution.

By multiplying the distributions given by (5.4), (5.5) and (5.6) we obtain the following *joint prior distribution* for $(L, U, \Theta_L, \ldots, \Theta_U)$:

$$f(l, u, \theta_l, \ldots, \theta_u) = Pr\{L = l\} \cdot Pr\{U = u\} \cdot f(\theta_l, \ldots, \theta_u | l, u)$$

$$= \begin{cases} \lambda_l \cdot \upsilon_u \cdot \dfrac{\Gamma\left(\sum_{i=l}^{u} \alpha_{i|lu}\right)}{\prod_{i=l}^{u} \Gamma(\alpha_{i|lu})} \prod_{i=l}^{u} \theta_i^{\alpha_{i|lu} - 1} & \text{if } m_m \le l \le m_M,\ x_m \le u \le x_M \\ & \text{and } \sum_{i=l}^{u} \theta_i = 1, \\ 0 & \text{elsewhere,} \end{cases} \tag{5.7}$$

where the $\alpha_{i|lu}$ are also hyperparameters to be provided by the user. We write $\alpha_{i|lu}$, because the prior distribution for $(\Theta_l, \ldots, \Theta_u)$ is conditional on l and u, i.e. conditional on given values of the random

[2] If a prior distribution belongs to a class $\mathcal{L}$ of distributions, then it is said to be the natural conjugate prior if the posterior distribution also belongs to $\mathcal{L}$. Natural conjugate priors are usually chosen because they lead to a relatively simple computation of the posterior distribution.

variables L and U.
In addition to the frequency counts N_i $(i = l, \ldots, u)$, we define W and Z to be the random variables corresponding to the minimum and maximum sampled cost values. Following [Boender 1984], we use the notation C_{nwz} for the event (or outcome of the experiment) $\{(N_W, \ldots, N_Z) = (n_w, \ldots, n_z)\}$. The probability of this event, given values of the parameters, is given by (cf. (5.1)):

$$Pr\{C_{nwz}|(L,U,\Theta) = (l,u,\theta)\} = \begin{cases} \dfrac{n!}{\prod_{i=w}^{z} n_i!} \prod_{i=w}^{z} \theta_i^{n_i} & \text{if } l \le w,\ z \le u \\ & \text{and } \sum_{i=w}^{z} n_i = n, \\ 0 & \text{elsewhere,} \end{cases} \tag{5.8}$$

where Θ and θ denote the vectors $(\Theta_L, \ldots, \Theta_U)$ and $(\theta_l, \ldots, \theta_u)$, respectively.
Using (5.7), (5.8), Bayes's formula and the fact that $n_i = 0$ for $i < w$ or $i > z$, we obtain the following expression for the joint posterior distribution for $(L, U, \Theta_L, \ldots, \Theta_U)$:

$$f(l,u,\theta_l,\ldots,\theta_u|C_{nwz}) = \frac{f(l,u,\theta_l,\ldots,\theta_u)Pr\{C_{nwz}|(L,U,\Theta) = (l,u,\theta)\}}{\sum_{s=m_m}^{w}\sum_{t=x}^{x_M} \int\cdots\int_{\{\psi|\sum_{i=s}^{t}\psi_i=1\}} f(s,t,\psi_s,\ldots,\psi_t)Pr\{C_{nwz}|(L,U,\Theta) = (s,t,\psi)\}\prod_{i=s}^{t} d\psi_i}$$

$$= \frac{\lambda_l \cdot \upsilon_u \cdot \dfrac{\Gamma\left(\sum_{i=l}^{u}\alpha_{i|lu}\right)}{\prod_{i=l}^{u}\Gamma(\alpha_{i|lu})} \prod_{i=l}^{u} \theta_i^{\alpha_{i|lu}+n_i-1}}{\sum_{s=m_m}^{w}\sum_{t=x}^{x_M} \int\cdots\int_{\{\psi|\sum_{i=s}^{t}\psi_i=1\}} \lambda_s \cdot \upsilon_t \cdot \dfrac{\Gamma\left(\sum_{i=s}^{t}\alpha_{i|st}\right)}{\prod_{i=s}^{t}\Gamma(\alpha_{i|st})} \prod_{i=s}^{t} \psi_i^{\alpha_{i|st}+n_i-1} \prod_{i=s}^{t} d\psi_i}, \tag{5.9}$$

if $m_m \le l \le w,\ z \le u \le x_M,\ \sum_{i=l}^{u}\theta_i = 1$ and 0 elsewhere. Note that since the prior distribution in our example is the product of

two probability density functions of the discrete type and one of the continuous type, we have to replace the summation in Bayes's formula by a composite of two summations and one multifold integral.

Equation (5.9) can be simplified by using the identity

$$\int \cdots \int_{\{\psi | \sum_{i=s}^{t} \psi_i = 1\}} \prod_{i=s}^{t} \psi_i^{\alpha_{i|st}+n_i-1} \prod_{i=s}^{t} d\psi_i = \frac{\prod_{i=s}^{t} \Gamma(\alpha_{i|st}+n_i)}{\Gamma\left(\sum_{i=s}^{t} \alpha_{i|st}+n\right)} \tag{5.10}$$

to obtain

$$f(l,u,\theta_l,\ldots,\theta_u|C_{nwz}) = \frac{\lambda_l \cdot \upsilon_u \cdot \frac{\Gamma\left(\sum_{i=l}^{u} \alpha_{i|lu}\right)}{\prod_{i=l}^{u} \Gamma(\alpha_{i|lu})} \prod_{i=l}^{u} \theta_i^{\alpha_{i|lu}+n_i-1}}{\sum_{s=m_m}^{w} \sum_{t=z}^{x_M} \lambda_s \cdot \upsilon_t \cdot \frac{\Gamma\left(\sum_{i=s}^{t} \alpha_{i|st}\right)}{\Gamma\left(\sum_{i=s}^{t} \alpha_{i|st}+n\right)} \prod_{i=s}^{t} \frac{\Gamma(\alpha_{i|st}+n_i)}{\Gamma(\alpha_{i|st})}}. \tag{5.11}$$

By integrating the joint posterior distribution given by (5.11) over all $\theta_l,\ldots,\theta_u$, we obtain the marginal joint probability density function for (L,U), given by (cf. [Boender, 1984]):

$$Pr\{L=l,\ U=u|C_{nwz}\} = \begin{cases} \frac{b_{lun}}{B_{nwz}} & \text{if } m_m \le l \le w,\ z \le u \le x_M, \\ 0 & \text{elsewhere,} \end{cases} \tag{5.12}$$

where the terms b_{lun} and B_{nwz} are defined as

$$b_{lun} \stackrel{\text{def}}{=} \lambda_l \upsilon_u \frac{\Gamma\left(\sum_{i=l}^{u} \alpha_{i|lu}\right)}{\Gamma\left(\sum_{i=l}^{u} \alpha_{i|lu}+n\right)} \prod_{i=l}^{u} \frac{\Gamma(\alpha_{i|lu}+n_i)}{\Gamma(\alpha_{i|lu})} \tag{5.13}$$

and

$$B_{nwz} \stackrel{\text{def}}{=} \sum_{s=m_m}^{w} \sum_{t=z}^{x_M} b_{stn}, \tag{5.14}$$

respectively.
Finally, using

$$f(\theta_l, \ldots, \theta_u | l, u; C_{nwz}) = \frac{f(l, u, \theta_l, \ldots, \theta_u | C_{nwz})}{Pr\{L = l,\ U = u | C_{nwz}\}}, \tag{5.15}$$

we obtain

$$f(\theta_l, \ldots, \theta_u | l, u; C_{nwz}) =$$

$$\begin{cases} \dfrac{\Gamma\left(n + \sum_{i=l}^{u} \alpha_{i|lu}\right)}{\prod_{i=l}^{u} \Gamma(\alpha_{i|lu} + n_i)} \prod_{i=l}^{u} \theta_i^{\alpha_{i|lu} + n_i - 1} & \text{if } \sum_{i=l}^{u} \theta_i = 1, \\ 0 & \text{elsewhere.} \end{cases} \tag{5.16}$$

Hence, given $L = l$ and $U = u$, the posterior distribution for $(\Theta_l, \ldots, \Theta_u)$ is again a Dirichlet distribution with parameters $\alpha_{l|lu} + n_l, \ldots, \alpha_{u|lu} + n_u$.
Equations (5.12) and (5.16) are the Bayesian solutions to the inference problem and provide the posterior information on the values of the unknown parameters l, u and $\theta_l, \ldots, \theta_u$.

5.3 Bayes's method and simulated annealing

5.3.1 Introduction

In our Bayesian approach to simulated annealing, we consider the generation of each homogeneous Markov chain as an instance of the experiment described in the previous section. Thus, in contrast to the usual approach, in which Bayes's method is used to make inferences from the outcome of one experiment, we have a sequence of experiments and we use the posterior information of each experiment to choose the prior distribution of the next experiment. In addition, there are three modifications for each experiment:

1. Since we consider minimization problems, we are not much interested in the (unknown) maximum cost value u. Thus, with respect to this parameter, we simply assume that u is an upper bound to all cost values.

2. The sampling of configurations is not carried out from a uniform distribution on the set of configurations. Instead, we assume that the configurations are drawn from the stationary distribution of the Markov chain under consideration. This assumption is motivated by two facts:

   (a) Asymptotically, the configurations are sampled from the stationary distribution.

   (b) Our choice of the parameters of a cooling schedule is such that the probability distribution of the configurations is always 'close' to the stationary distribution (see the discussion in section 3.3).

3. The frequency counts do not follow a multinomial distribution, because the random variables $X(k)$ are not mutually stochastically independent ($X(k)$ denotes the k-th sampled configuration). However, in appendix A we show that asymptotically, i.e. for $n \to \infty$, there is only a slight difference between the situation where the random variables $X(k)$ are independent and the situation we are dealing with. In both cases, the frequency counts follow a multivariate normal distribution; the mean vectors of both distributions are identical, but the covariance matrices are slightly different. Moreover, in the appendix we show that the difference between the two covariance matrices can be made arbitrarily small by considering only the sequence of configurations $X(t)$, $X(2t)$, etc., for some properly chosen integer $t > 1$. Hereinafter, we nevertheless consider the original sequence of configurations $X(1)$, $X(2)$, etc., and we assume that for the k-th Markov chain, the frequency counts do follow a multinomial distribution with parameters $\theta_{kl}, \ldots, \theta_{ku}$, where θ_{ki}, $i = l, \ldots, u$, now denotes the (unknown) probability of sampling a configuration with cost value i, under the assumption that the

configurations are sampled from the stationary distribution of the k-th Markov chain.

The natural way to connect the Bayesian approach to Markov chain k with the one to Markov chain $k+1$ is to take the posterior distributions found for Markov chain k as prior distributions for Markov chain $k+1$. However, the posterior results for the k-th Markov chain concern the random variables $\Theta_{kL}, \ldots, \Theta_{ku}$ and L, whereas for the $(k+1)$-th Markov chain we need to assume a prior distribution on the random variables $\Theta_{k+1,L}, \ldots, \Theta_{k+1,u}$ and, again, L. Under the aforementioned assumption with respect to the distribution from which the configurations are sampled, we can derive a simple relation between the parameters θ_{ki} and $\theta_{k+1,i}$, $i = l, \ldots, u$, which can then be used to transform the conditional posterior distribution for $(\Theta_{kl}, \ldots, \Theta_{ku})$ (cf. (5.16)) into a conditional prior distribution for $(\Theta_{k+1,l}, \ldots, \Theta_{k+1,u})$. This relation is obtained in the following way.

Let X denote the random variable corresponding to a sampled configuration, then, using the expression for the stationary distribution of a Markov chain given by (3.8) and (3.9), we find that θ_{ki}, $i = l, \ldots, u$, is given by

$$\theta_{ki} = Pr\left\{ C(X) = i \;\middle|\; \forall x \in \mathcal{R} : \; Pr\{X = x\} = \frac{\exp\left(\frac{-C(x)}{c_k}\right)}{\sum_{y \in \mathcal{R}} \exp\left(\frac{-C(y)}{c_k}\right)} \right\}, \tag{5.17}$$

where c_k is the value of the control parameter for the k-th Markov chain. From (5.17), we obtain

$$\theta_{ki} = \xi_i \cdot |\mathcal{R}| \cdot \frac{\exp\left(\frac{-i}{c_k}\right)}{\sum_{y \in \mathcal{R}} \exp\left(\frac{-C(y)}{c_k}\right)}, \; i = l, \ldots, u, \tag{5.18}$$

where ξ_i, $i = l, \ldots, u$, denotes the fraction of configurations with cost value i. Furthermore,

$$\sum_{y \in \mathcal{R}} \exp\left(\frac{-C(y)}{c_k}\right) = \sum_{j=l}^{u} \xi_j \cdot |\mathcal{R}| \cdot \exp\left(\frac{-j}{c_k}\right). \tag{5.19}$$

Combining (5.18) and (5.19) yields

$$\theta_{ki} = \frac{\xi_i \exp\left(\frac{-i}{c_k}\right)}{\sum_{j=l}^{u} \xi_j \exp\left(\frac{-j}{c_k}\right)}, \; i = l, \ldots, u. \tag{5.20}$$

From (5.20), we find, using $\sum_{i=l}^{u} \xi_i = \sum_{i=l}^{u} \theta_{ki} = 1$,

$$\xi_i = \frac{\theta_{ki} \exp\left(\frac{i}{c_k}\right)}{\sum_{j=l}^{u} \theta_{kj} \exp\left(\frac{j}{c_k}\right)}, \; i = l, \ldots, u \tag{5.21}$$

and, consequently,

$$\theta_{k+1,i} = \frac{\xi_i \exp\left(\frac{-i}{c_{k+1}}\right)}{\sum_{j=l}^{u} \xi_j \exp\left(\frac{-j}{c_{k+1}}\right)} = \frac{a_{ki}\theta_{ki}}{\sum_{j=l}^{u} a_{kj}\theta_{kj}}, \; i = l, \ldots, u, \tag{5.22}$$

where a_{ki}, $i = l, \ldots, u$, is given by

$$a_{ki} = \exp\left(\frac{i(c_{k+1} - c_k)}{c_k \cdot c_{k+1}}\right). \tag{5.23}$$

Given the conditional posterior distribution $f(\theta_{kl}, \ldots, \theta_{ku}|l; C_{nw_k})$ for $(\Theta_{kl}, \ldots, \Theta_{ku})$ (cf. (5.16)), we can now obtain a joint probability density function $g(\theta_{k+1,l}, \ldots, \theta_{k+1,u}|l; C_{nw_k})$ for $(\Theta_{k+1,l}, \ldots, \Theta_{k+1,u})$ by putting

$$\Theta_{k+1,i} = \frac{a_{ki}\Theta_{ki}}{\sum_{j=l}^{u} a_{kj}\Theta_{kj}}, \; i = l, \ldots, u \tag{5.24}$$

and considering the transformation given by (5.22) and (5.23). This transformation is the subject of the next subsection. Unfortunately, $g(\theta_{k+1,l}, \ldots, \theta_{k+1,u}|l; C_{nw_k})$ is not a Dirichlet distribution so that it is not a natural conjugate prior and as such not an attractive candidate for the conditional prior distribution for $(\Theta_{k+1,l}, \ldots, \Theta_{k+1,u})$. Instead, in subsection 5.3.3, we compute approximations of the expected values $E[\Theta_{k+1,i}|l; C_{nw_k}]$, $i = l, \ldots, u$, with respect to the distribution $g(\theta_{k+1,l}, \ldots, \theta_{k+1,u}|l; C_{nw_k})$.
Next, in subsection 5.3.4, we construct a Dirichlet distribution whose parameters $\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l}$ are such that the $E[\Theta_{k+1,i}|l; C_{nw_k}]$'s with respect to the Dirichlet distribution are identical to the approximations of the expected values as they are computed in subsection 5.3.3. Furthermore, since the latter requirement yields only $u - l$ equations for the $u - l + 1$ parameters $\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l}$, we impose

one additional requirement on the parameters $\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l}$. Finally, the $\mathcal{D}(\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l})$ distribution is taken as the conditional prior distribution for $(\Theta_{k+1,l}, \ldots, \Theta_{k+1,u})$.
In subsection 5.3.4, we also summarize the results obtained by giving the full expressions for the prior and posterior distributions of each Markov chain.

5.3.2 Transformation of the posterior distribution

In this subsection we consider the transformation given by (5.22) and (5.23). According to section 5.2, the conditional posterior distribution $f(\theta_{kl}, \ldots, \theta_{ku}|l; C_{nw_k})$ for $(\Theta_{kl}, \ldots, \Theta_{ku})$ is a $\mathcal{D}(\alpha_{kl|l} + n_{kl}, \ldots, \alpha_{ku|l} + n_{ku})$ distribution ((5.16)), where n_{ki}, $i = l, \ldots, u$, denotes the number of configurations in Markov chain k with cost value i; $n_{ki} = 0$ for $i < w_k$, where w_k is the smallest cost value sampled during the generation of Markov chain k. We are thus interested in solving the following problem: given that the joint probability density function of the random variables $\Theta_{kl}, \ldots, \Theta_{ku}$ is a $\mathcal{D}(\alpha_{kl|l} + n_{kl}, \ldots, \alpha_{ku|l} + n_{ku})$ distribution, find the joint probability density function of the random variables $\Theta_{k+1,l}, \ldots, \Theta_{k+1,u}$, defined by (5.24).
For convenience sake, we reformulate this problem as follows. Let $X_0, \ldots, X_{m-1}$ be random variables following a $\mathcal{D}(\beta_0, \ldots, \beta_m)$ distribution, i.e.[3]

$$f(x_0, \ldots, x_{m-1}) = \begin{cases} \dfrac{\Gamma\left(\sum_{i=0}^{m} \beta_i\right)}{\prod_{i=0}^{m} \Gamma(\beta_i)} \prod_{i=0}^{m-1} x_i^{\beta_i - 1} \left(1 - \sum_{i=0}^{m-1} x_i\right)^{\beta_m - 1} \\ \qquad \text{if } 0 \leq \sum_{i=0}^{m-1} x_i \leq 1, \\ 0 \qquad \text{elsewhere} \end{cases} \tag{5.25}$$

[3] We have explicitly taken into account the fact that f is non-zero only for those values of $x_0, \ldots, x_m$ for which $\sum_{i=0}^{m} x_i = 1$ by the substitution $x_m = 1 - \sum_{i=0}^{m-1} x_i$.

and let the random variables $Y_0, \ldots, Y_{m-1}$ be defined by

$$Y_i = \frac{a_i X_i}{a_m \left(1 - \sum_{j=0}^{m-1} X_j\right) + \sum_{j=0}^{m-1} a_j X_j}, \; i = 0, \ldots, m-1, \tag{5.26}$$

where the a_i's are positive, real-valued parameters, satisfying (cf. (5.23))

$$a_0 > a_1 > \ldots > a_m. \tag{5.27}$$

We express X_m in terms of $X_0, \ldots, X_{m-1}$, because for convergence of the series involved in the computations of subsection 5.3.3, it is essential that the parameter a_j associated with the dependent variable X_j be the smallest of all parameters a_i, $i = 0, \ldots, m$. In our case, the latter can be achieved by choosing X_m as the dependent variable. For ease of notation, we renumber $X_0, \ldots, X_{m-1}$ and $Y_0, \ldots, Y_{m-1}$ to $X_m, \ldots, X_1$ and $Y_m, \ldots, Y_1$, respectively, and the parameters a_i, $i = 0, \ldots, m$ accordingly. We are thus interested in the following problem. Let $X_1, \ldots, X_m$ be random variables following a $\mathcal{D}(\beta_0, \ldots, \beta_m)$ distribution, i.e.

$$f(x_1, \ldots, x_m) = \begin{cases} \dfrac{\Gamma\left(\sum_{i=0}^{m} \beta_i\right)}{\prod_{i=0}^{m} \Gamma(\beta_i)} \prod_{i=1}^{m} x_i^{\beta_i - 1} \left(1 - \sum_{i=1}^{m} x_i\right)^{\beta_0 - 1} \\ \qquad\qquad \text{if } 0 \leq \sum_{i=1}^{m} x_i \leq 1, \\ 0 \qquad\qquad \text{elsewhere} \end{cases} \tag{5.28}$$

and let the random variables $Y_1, \ldots, Y_m$ be defined by

$$Y_i = \frac{a_i X_i}{a_0 \left(1 - \sum_{j=1}^{m} X_j\right) + \sum_{j=1}^{m} a_j X_j}, \; i = 1, \ldots, m, \tag{5.29}$$

where the parameters a_i satisfy

$$a_0 < a_1 < \ldots < a_m, \tag{5.30}$$

then the problem is to find the joint probability density function of $(Y_1, \ldots, Y_m)$. We solve this problem by considering the transformation

$$y_i = \frac{a_i x_i}{a_0 \left(1 - \sum_{j=1}^{m} x_j\right) + \sum_{j=1}^{m} a_j x_j}, \; i = 1, \ldots, m, \tag{5.31}$$

from which we obtain

$$x_i = \frac{\frac{y_i}{a_i}}{\frac{1}{a_0}\left(1 - \sum_{j=1}^{m} y_j\right) + \sum_{j=1}^{m} \frac{y_j}{a_j}} = \frac{a_0 y_i}{a_i S}, \; i = 1, \ldots, m, \tag{5.32}$$

where the term S is defined by

$$S \stackrel{\text{def}}{=} 1 + \sum_{j=1}^{m} (a_0 - a_j) \frac{y_j}{a_j}. \tag{5.33}$$

The joint probability density function of $(Y_1, \ldots, Y_m)$ can now be found by substituting the expression of (5.32) in $f(x_1, \ldots, x_m)$, given by (5.28), and multiplying the result by the *Jacobian*, defined as the determinant of the matrix $J = (J_{ij}) = \left(\frac{\partial x_i}{\partial y_j}\right)$.
From (5.32) and (5.33) we obtain, using $\frac{\partial S}{\partial y_j} = \frac{a_0 - a_j}{a_j}$,

$$\frac{\partial x_i}{\partial y_j} = \frac{a_0}{a_i S} \delta_{ij} - \frac{a_0 y_i}{a_i S^2} \cdot \frac{a_0 - a_j}{a_j}, \tag{5.34}$$

where δ_{ij} is the Kronecker symbol.
From (5.34) we deduce

$$J = S^{-2} \left(D - \mathbf{p} \cdot \mathbf{q}^T\right), \tag{5.35}$$

where the diagonal $m \times m$-matrix D and the m-vectors $\mathbf{p}$ and $\mathbf{q}$ are defined as

$$D \stackrel{\text{def}}{=} \text{diag}\left(\frac{a_0 S}{a_1}, \ldots, \frac{a_0 S}{a_m}\right), \tag{5.36}$$

$$\mathbf{p}^T \stackrel{\text{def}}{=} \left(a_0 \frac{y_1}{a_1}, \ldots, a_0 \frac{y_m}{a_m}\right) \tag{5.37}$$

and

$$\mathbf{q}^T \stackrel{\text{def}}{=} \left(\frac{a_0 - a_1}{a_1}, \ldots, \frac{a_0 - a_m}{a_m}\right), \tag{5.38}$$

respectively.
The Jacobian can now be found by using the following lemma.

Lemma 5.1
Suppose $M = D + \mathbf{s} \cdot \mathbf{t}^T$, *where* M *and* D *are* $m \times m$*-matrices,* $\mathbf{s}$ *and* $\mathbf{t}$ m*-vectors and* $D = diag(d_1, \ldots, d_m)$. *Then*

$$|M| = \left(1 + \sum_{i=1}^{m} \frac{s_i t_i}{d_i}\right) \prod_{j=1}^{m} d_j. \tag{5.39}$$

Proof

$$|M| = \left| \begin{pmatrix} d_1 + s_1 t_1 & s_1 t_2 & \cdots & s_1 t_m \\ s_2 t_1 & d_2 + s_2 t_2 & \cdots & s_2 t_m \\ \vdots & \vdots & & \vdots \\ s_m t_1 & s_m t_2 & \cdots & d_m + s_m t_m \end{pmatrix} \right|$$

$$= \prod_{k=1}^{m} s_k t_k \cdot \left| \begin{pmatrix} 1 + \frac{d_1}{s_1 t_1} & 1 & \cdots & 1 \\ 1 & 1 + \frac{d_2}{s_2 t_2} & \cdots & 1 \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \cdots & 1 + \frac{d_m}{s_m t_m} \end{pmatrix} \right|. \tag{5.40}$$

By subtracting the first column from all other columns and adding a multiple of each row to the first row, we find that in general

$$\left| \begin{pmatrix} 1 + u_1 & 1 & \cdots & 1 \\ 1 & 1 + u_2 & \cdots & 1 \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \cdots & 1 + u_m \end{pmatrix} \right| = \left(u_1 + \sum_{i=1}^{m} \frac{u_1}{u_i}\right) \prod_{j=2}^{m} u_j$$

$$= \left(1 + \sum_{i=1}^{m} \frac{1}{u_i}\right) \prod_{j=1}^{m} u_j. \tag{5.41}$$

Substituting $\frac{d_i}{s_i t_i}$ for u_i, $i = 1, \ldots, m$, in (5.41) we obtain

$$|M| = \prod_{k=1}^{m} s_k t_k \left(1 + \sum_{i=1}^{m} \frac{s_i t_i}{d_i}\right) \prod_{j=1}^{m} \frac{d_j}{s_j t_j} = \left(1 + \sum_{i=1}^{m} \frac{s_i t_i}{d_i}\right) \prod_{j=1}^{m} d_j. \quad (5.42)$$

□

Using lemma 5.1 we obtain

$$|J| = S^{-2m} \cdot |D - \mathbf{p}\mathbf{q}^T|$$

$$= S^{-2m} \left(1 - \sum_{i=1}^{m} \frac{a_0 \frac{y_i}{a_i} \cdot \frac{a_0 - a_i}{a_i}}{\frac{a_0 S}{a_i}}\right) \prod_{j=1}^{m} \left(\frac{a_0 S}{a_j}\right) = S^{-1} \prod_{j=1}^{m} \left(\frac{a_0}{a_j S}\right) \quad (5.43)$$

and thus

$$g(y_1, \ldots, y_m) = f\left(\frac{a_0 y_1}{a_1 S}, \ldots, \frac{a_0 y_m}{a_m S}\right) \cdot S^{-1} \prod_{j=1}^{m} \left(\frac{a_0}{a_j S}\right). \quad (5.44)$$

Equation (5.44) yields, by substituting for f a $D(\beta_0, \ldots, \beta_m)$ distribution and using the equality

$$1 - \sum_{j=1}^{m} \frac{a_0 y_j}{a_j S} = \frac{1}{S}\left(S - a_0 \sum_{j=1}^{m} \frac{y_j}{a_j}\right) = \frac{1}{S}\left(1 - \sum_{j=1}^{m} y_j\right), \quad (5.45)$$

the following joint probability density function for $(Y_1, \ldots, Y_m)$:

$$g(y_1, \ldots, y_m) = \begin{cases} \frac{\Gamma(\beta)}{\prod_{i=0}^{m} \Gamma(\beta_i)} \prod_{i=1}^{m} \left(\frac{a_0}{a_i}\right)^{\beta_i} \prod_{i=1}^{m} y_i^{\beta_i - 1} \left(1 - \sum_{i=1}^{m} y_i\right)^{\beta_0 - 1} \\ \quad \times \left(1 - \sum_{i=1}^{m} \frac{a_i - a_0}{a_i} y_i\right)^{-\beta} & \text{if } 0 \le \sum_{i=1}^{m} y_i \le 1, \quad (5.46) \\ 0 & \text{elsewhere,} \end{cases}$$

where $\beta = \sum_{i=0}^{m} \beta_i$. From (5.46) we obtain, after renumbering $(Y_1, \ldots, Y_m)$ to $(Y_{m-1}, \ldots, Y_0)$ and resubstituting $Y_m = 1 - \sum_{i=0}^{m-1} Y_i$,

the following joint probability density function for $(Y_0, \ldots, Y_m)$:

$$g(y_0, \ldots, y_m) = \begin{cases} \dfrac{\Gamma(\beta)}{\prod_{i=0}^{m} \Gamma(\beta_i)} \prod_{i=0}^{m} \left(\frac{a_m}{a_i}\right)^{\beta_i} \prod_{i=0}^{m} y_i^{\beta_i - 1} \left(1 - \sum_{i=0}^{m} \frac{a_i - a_m}{a_i} y_i\right)^{-\beta} \\ \qquad \text{if } \sum_{i=0}^{m} y_i = 1, \\ 0 \qquad \text{elsewhere,} \end{cases} \tag{5.47}$$

Returning to the original problem, we use (5.47) to obtain the following joint probability density function for $(\Theta_{k+1,l}, \ldots, \Theta_{k+1,u})$:

$$g(\theta_{k+1,l}, \ldots, \theta_{k+1,u} | l; C_{nw_k})$$

$$= \begin{cases} \dfrac{\Gamma(\alpha_{k|l}+n)}{\prod_{i=l}^{u} \Gamma(\alpha_{ki|l}+n_{ki})} \prod_{i=l}^{u} \left(\frac{a_{ku}}{a_{ki}}\right)^{\alpha_{ki|l}+n_{ki}} \prod_{i=l}^{u} \theta_{k+1,i}^{\alpha_{ki|l}+n_{ki}-1} \\ \qquad \times \left(1 - \sum_{i=l}^{u} \frac{a_{ki} - a_{ku}}{a_{ki}} \theta_{k+1,i}\right)^{-(\alpha_{k|l}+n)} \\ \qquad \text{if } \sum_{i=l}^{u} \theta_{k+1,i} = 1, \\ 0 \qquad \text{elsewhere,} \end{cases} \tag{5.48}$$

where $\alpha_{k|l} = \sum_{i=l}^{u} \alpha_{ki|l}$. Clearly, (5.48) is not a Dirichlet distribution.

5.3.3 Computation of $E[\Theta_{k+1,i}|l; C_{nw_k}]$

In this subsection we consider the following problem: given that the joint probability density function of the random variables $\Theta_{k+1,l}, \ldots, \Theta_{k+1,u}$ is given by (5.48), find the expected values $E[\Theta_{k+1,i}|l; C_{nw_k}]$, $i = l, \ldots, u$.
For convenience sake, we reformulate the problem as follows. Let the joint probability density function of the random variables $Y_1, \ldots, Y_m$ be given by (5.46), then the problem is to determine $E[Y_i]$,

$i = 1, \ldots, m$, i.e. to compute the expression

$$E[Y_i] = \frac{\Gamma(\beta)}{\prod_{j=0}^{m} \Gamma(\beta_j)} \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \int \cdots \int_{\{0 \le \sum_{j=1}^{m} y_j \le 1\}} y_i \prod_{j=1}^{m} {y_j}^{\beta_j - 1}$$

$$\times \left(1 - \sum_{j=1}^{m} y_j\right)^{\beta_0 - 1} \left(1 - \sum_{j=1}^{m} \frac{a_j - a_0}{a_j} y_j\right)^{-\beta} dy_1 \ldots dy_m. \qquad (5.49)$$

A straightforward approach to the computation of the expression given by (5.49) is due to Boender [1986] and is based on rewriting $E[Y_i]$ in terms of the stochastic variables $X_1, \ldots, X_m$ and using the Taylor expansion of the integrand. This approach, which is discussed in more detail in appendix B, is not as elaborate as the one discussed in the remainder of this section, but it leads to an approximation of $E[Y_i]$ with unknown accuracy. We therefore resort to a more intricate approach, which consists of three steps:

- Firstly, we show that the expected value $E[Y_i]$ is the integral representation of a *Lauricella function* [Lauricella, 1893];
- Secondly, we show that the Lauricella function, which is defined as a multiple series, can be written as a simple series of *cycle indicator functions* [Riordan, 1978];
- Thirdly, we derive an approximation of the sum of this series and discuss its accuracy.

We now discuss these steps in more detail.

STEP 1

We need the following two lemmas.

Lemma 5.2

For $m \ge 1$, $\kappa > 0$ and $|\sum_{i=1}^{m} z_i| \le 1$ the following identity holds:

$$\left(1 - \sum_{i=1}^{m} z_i\right)^{-\kappa} = \sum_{i_1=0}^{\infty} \cdots \sum_{i_m=0}^{\infty} \frac{(\kappa)_{i_1 + \cdots + i_m}}{i_1! i_2! \ldots i_m!} {z_1}^{i_1} \ldots {z_m}^{i_m}, \qquad (5.50)$$

where $(\kappa)_n$ *denotes the* Pochhammer symbol*:*

$$(\kappa)_n = \frac{\Gamma(\kappa+n)}{\Gamma(\kappa)}. \tag{5.51}$$

Proof

We use the serial expansion of $(1-z)^{-a}$, which, for $(|z|<1)$, is given by

$$(1-z)^{-a} = \sum_{i=0}^{\infty} \frac{(a)_i}{i!} z^i. \tag{5.52}$$

The proof of (5.50) is by induction on m. For $m=1$, the equality in (5.50) is immediately obtained from (5.52) by the substitution $z=z_1$, $i=i_1$ and $a=\kappa$. Suppose that (5.50) holds for some $m>1$. We write

$$\begin{aligned}\left(1-\sum_{i=1}^{m+1} z_i\right)^{-\kappa} &= \left(1-\sum_{i=1}^{m} z_i\right)^{-\kappa} \cdot \left(\frac{1-\sum_{i=1}^{m+1} z_i}{1-\sum_{i=1}^{m} z_i}\right)^{-\kappa} \\ &= \left(1-\sum_{i=1}^{m} z_i\right)^{-\kappa} \cdot \left(1-\frac{z_{m+1}}{1-\sum_{i=1}^{m} z_i}\right)^{-\kappa}. \end{aligned} \tag{5.53}$$

By substituting $z = z_{m+1}\cdot(1-\sum_{i=1}^{m} z_i)^{-1}$, $i=i_{m+1}$ and $a=\kappa$ in (5.52), we can rewrite (5.53) as

$$\begin{aligned}\left(1-\sum_{i=1}^{m+1} z_i\right)^{-\kappa} &= \left(1-\sum_{i=1}^{m} z_i\right)^{-\kappa} \sum_{i_{m+1}=0}^{\infty} \frac{(\kappa)_{i_{m+1}}}{i_{m+1}!} {z_{m+1}}^{i_{m+1}} \left(1-\sum_{i=1}^{m} z_i\right)^{-i_{m+1}} \\ &= \sum_{i_{m+1}=0}^{\infty} \frac{(\kappa)_{i_{m+1}}}{i_{m+1}!} {z_{m+1}}^{i_{m+1}} \left(1-\sum_{i=1}^{m} z_i\right)^{-(\kappa+i_{m+1})} \end{aligned} \tag{5.54}$$

and by using the induction hypothesis as

$$\left(1-\sum_{i=1}^{m+1} z_i\right)^{-\kappa}$$

$$= \sum_{i_1=0}^{\infty} \cdots \sum_{i_{m+1}=0}^{\infty} \frac{(\kappa+i_{m+1})_{i_1+\cdots+i_m} (\kappa)_{i_{m+1}}}{i_1! i_2! \ldots i_{m+1}!} {z_1}^{i_1} \ldots {z_{m+1}}^{i_{m+1}}. \tag{5.55}$$

Finally, by using the equality $(\kappa+m)_i(\kappa)_m = (\kappa)_{i+m}$ in (5.55), we obtain (5.50) for $m+1$.

□

Lemma 5.3
The integral representation of the Lauricella function $F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m)$ *defined by*

$$F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m)$$
$$= \sum_{i_1=0}^{\infty} \cdots \sum_{i_m=0}^{\infty} \frac{(a)_{i_1+\cdots+i_m} (b_1)_{i_1} \cdots (b_m)_{i_m}}{(c)_{i_1+\cdots+i_m}} \frac{{\gamma_1}^{i_1}}{i_1!} \cdots \frac{{\gamma_m}^{i_m}}{i_m!}, \tag{5.56}$$

where, for convergence, $|\gamma_1| < 1, \ldots, |\gamma_m| < 1$, *is given by*

$$F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m) = \frac{\Gamma(c)}{\Gamma(c - \sum_{i=1}^{m} b_i) \prod_{i=1}^{m} \Gamma(b_i)}$$

$$\times \int \cdots \int_{\{0 \le \sum_{i=1}^{m} \mu_i \le 1\}} \prod_{i=1}^{m} {\mu_i}^{b_i - 1} \left(1 - \sum_{i=1}^{m} \mu_i\right)^{c - 1 - \sum_{i=1}^{m} b_i} \left(1 - \sum_{i=1}^{m} \gamma_i \mu_i\right)^{-a} d\mu_1 \ldots d\mu_m. \tag{5.57}$$

Proof

By using lemma 5.2, with $z_i = \gamma_i \mu_i$, $i = 1, \ldots, m$ and $\kappa = a$, we obtain

$$\int \cdots \int_{\{0 \le \sum_{i=1}^{m} \mu_i \le 1\}} \prod_{i=1}^{m} {\mu_i}^{b_i - 1} \left(1 - \sum_{i=1}^{m} \mu_i\right)^{c - 1 - \sum_{i=1}^{m} b_i} \left(1 - \sum_{i=1}^{m} \gamma_i \mu_i\right)^{-a} d\mu_1 \ldots d\mu_m$$

$$= \sum_{i_1=0}^{\infty} \cdots \sum_{i_m=0}^{\infty} \frac{(a)_{i_1+\cdots+i_m}}{i_1! i_2! \ldots i_m!} {\gamma_1}^{i_1} \ldots {\gamma_m}^{i_m}$$

$$\times \int \cdots \int_{\{0 \le \sum_{i=1}^{m} \mu_i \le 1\}} \prod_{i=1}^{m} {\mu_i}^{b_i + i_i - 1} \left(1 - \sum_{i=1}^{m} \mu_i\right)^{c - 1 - \sum_{i=1}^{m} b_i} d\mu_1 \ldots d\mu_m$$

$$= \sum_{i_1=0}^{\infty} \cdots \sum_{i_m=0}^{\infty} \frac{(a)_{i_1+\cdots+i_m}}{i_1! i_2! \ldots i_m!} {\gamma_1}^{i_1} \ldots {\gamma_m}^{i_m}$$

$$\times \frac{\Gamma(b_1 + i_1) \ldots \Gamma(b_m + i_m)}{\Gamma(c + i_1 + \cdots + i_m)} \Gamma\left(c - \sum_{i=1}^{m} b_i\right)$$

$$= \frac{\Gamma(b_1)\dots\Gamma(b_m)\Gamma\left(c-\sum_{i=1}^{m} b_i\right)}{\Gamma(c)}$$

$$\times \sum_{i_1=0}^{\infty}\cdots\sum_{i_m=0}^{\infty} \frac{(a)_{i_1+\cdots+i_m}}{i_1!i_2!\dots i_m!}{\gamma_1}^{i_1}\dots{\gamma_m}^{i_m}\frac{(b_1)_{i_1}\dots(b_m)_{i_m}}{(c)_{i_1+\cdots i_m}}$$

$$= \frac{\Gamma(b_1)\dots\Gamma(b_m)\Gamma\left(c-\sum_{i=1}^{m} b_i\right)}{\Gamma(c)} \cdot F_D(a,b_1,\dots,b_m;c;\gamma_1,\dots,\gamma_m). \tag{5.58}$$

N.B. For $m=1$, (5.56) defines the well-known *hypergeometric function* ${}_2F_1(a,b_1;c;\gamma_1)$; for $m=2$, we obtain the *Appell function* $F_1(a,b_1,b_2;c;\gamma_1,\gamma_2)$ (see [Appell & Kampé de Fériet, 1926; Slater, 1966]). The Lauricella functions are known as hypergeometric functions of m variables.

□

We can now express the expected value $E[Y_i]$, given by (5.49), as a Lauricella function.

Theorem 5.1
Let the joint probability density function of the m random variables $Y_1,\dots,Y_m$ be given by (5.46), then for $i=1,\dots,m$

$$E[Y_i] = \prod_{j=1}^{m}\left(\frac{a_0}{a_j}\right)^{\beta_j}\cdot\beta_i\sum_{i_1=0}^{\infty}\cdots\sum_{i_m=0}^{\infty}\frac{1}{\beta+\sum_{j=1}^{m} i_j}$$

$$\times\frac{(\beta_1)_{i_1}\dots(\beta_{i-1})_{i_{i-1}}(\beta_i+1)_{i_i}(\beta_{i+1})_{i_{i+1}}\dots(\beta_m)_{i_m}}{i_1!i_2!\dots i_m!}\prod_{j=1}^{m}\left(1-\frac{a_0}{a_j}\right)^{i_j}. \tag{5.59}$$

Proof

By substituting $a=\beta$, $b_j=\beta_j$, $j=1,\dots,m$, $j\neq i$, $b_i=\beta_i+1$, $c=\beta+1$, $\mu_j=y_j$, $j=1,\dots,m$, $\gamma_j=1-\frac{a_0}{a_j}$, $j=1,\dots,m$ (which implies $|\gamma_j|<1$, $j=1,\dots,m$ because of (5.30)) in (5.57) and

combining the resulting equality with (5.49), we obtain

$$E[Y_i] = \frac{\Gamma(\beta)}{\prod_{j=0}^{m}\Gamma(\beta_j)}\prod_{j=1}^{m}\left(\frac{a_0}{a_j}\right)^{\beta_j}\ \frac{\Gamma(\beta_0)\prod_{j=1,j\neq i}^{m}\Gamma(\beta_j)\,\Gamma(\beta_i+1)}{\Gamma(\beta+1)}\times$$

$$F_D\left(\beta,\beta_1,..,\beta_{i-1},\beta_i+1,\beta_{i+1},..,\beta_m;\beta+1;\left(1-\frac{a_0}{a_1}\right),..,\left(1-\frac{a_0}{a_m}\right)\right)$$

$$=\frac{\beta_i}{\beta}\prod_{j=1}^{m}\left(\frac{a_0}{a_j}\right)^{\beta_j}\times$$

$$F_D\left(\beta,\beta_1,..,\beta_{i-1},\beta_i+1,\beta_{i+1},..,\beta_m;\beta+1;\left(1-\frac{a_0}{a_1}\right),..,\left(1-\frac{a_0}{a_m}\right)\right). \tag{5.60}$$

Substituting the expression for the Lauricella function, given by (5.56), in (5.60) yields the desired result.

□

STEP 2

In this step we show that the multiple series of (5.59) can be written as a simple series of *cycle indicator functions*. Again, we need two lemmas.

Lemma 5.4
Consider the function $G(b_1,\ldots,b_m;\gamma_1,\ldots,\gamma_m)$, *given by*

$$G(b_1,\ldots,b_m;\gamma_1,\ldots,\gamma_m)$$

$$=\sum_{i_1=0}^{\infty}\cdots\sum_{i_m=0}^{\infty}(b_1)_{i_1}\ldots(b_m)_{i_m}\frac{{\gamma_1}^{i_1}}{i_1!}\ldots\frac{{\gamma_m}^{i_m}}{i_m!} \tag{5.61}$$

(note that G is the Lauricella function $F_D(a,b_1,\ldots,b_m;a;\gamma_1,\ldots,\gamma_m)$*). Let* Λ_k *be the k-th degree term of* $G(b_1,\ldots,b_m;\gamma_1,\ldots,\gamma_m)$, *denoted by* $\Lambda_k=[G]_k$, *i.e.* Λ_k *is found by summing those terms in (5.61), for which* $i_1+\cdots+i_m=k$ $(k=0,1,\ldots)$. *Then* $\Lambda_0=1$ *and for* $k>0$, *we have*

$$\Lambda_k=\sum_{\{(k_1,\ldots,k_k)|\sum_{j=1}^{k}jk_j=k\}}\frac{1}{k_1!\ldots k_k!}\left(\frac{t_1}{1}\right)^{k_1}\ldots\left(\frac{t_k}{k}\right)^{k_k}, \tag{5.62}$$

where the t_j's ($j = 1, \ldots, k$) are given by

$$t_j = \sum_{l=1}^{m} b_l {\gamma_l}^j. \tag{5.63}$$

Proof

$\Lambda_0 = 1$ is immediately obtained by observing that $i_1 + \cdots + i_m = 0$ implies $i_j = 0$ $(j = 1, \ldots, m)$.
Next, we use (5.52) to rewrite (5.61) as

$$G(b_1, \ldots, b_m; \gamma_1, \ldots, \gamma_m) = \prod_{l=1}^{m} (1 - \gamma_l)^{-b_l}. \tag{5.64}$$

Consequently, we have to prove that for $k > 0$

$$\left[\prod_{l=1}^{m} (1 - \gamma_l)^{-b_l}\right]_k = \sum_{\{(k_1,\ldots,k_k) | \sum_{j=1}^{k} j k_j = k\}} \frac{1}{k_1! \ldots k_k!} \left(\frac{t_1}{1}\right)^{k_1} \cdots \left(\frac{t_k}{k}\right)^{k_k}. \tag{5.65}$$

Using

$$(1 - z)^{-b} = \exp(-b \log(1 - z)) = \exp\left(b \sum_{j=1}^{\infty} \frac{z^j}{j}\right), \tag{5.66}$$

we find

$$\prod_{l=1}^{m} (1 - \gamma_l)^{-b_l} = \exp\left(\sum_{l=1}^{m} b_l \left(\sum_{j=1}^{\infty} \frac{{\gamma_l}^j}{j}\right)\right) = \exp\left(\sum_{j=1}^{\infty} \frac{1}{j} \left(\sum_{l=1}^{m} b_l {\gamma_l}^j\right)\right)$$

$$= \exp\left(\sum_{j=1}^{\infty} \frac{1}{j} t_j\right) = \prod_{j=1}^{\infty} \exp\left(\frac{t_j}{j}\right) = \prod_{j=1}^{\infty} \left(\sum_{k_j=0}^{\infty} \frac{1}{k_j!} \left(\frac{t_j}{j}\right)^{k_j}\right). \tag{5.67}$$

Observing that all product terms ${\gamma_1}^{p_1} \ldots {\gamma_m}^{p_m}$ in $\left(\frac{t_j}{j}\right)^{k_j}$ satisfy

$$p_1 + \cdots + p_m = j k_j, \tag{5.68}$$

we find the k-th degree term of the right-hand side of (5.67) by choosing $k_1, k_2, \ldots$ such that

$$\sum_{j=1}^{\infty} j k_j = k. \tag{5.69}$$

Equation (5.69) implies that $k_j = 0$ for $j > k$. Hence, for $k > 0$, we find

$$\left[\prod_{l=1}^{m}(1-\gamma_l)^{-b_l}\right]_k = \sum_{\{(k_1,\ldots,k_k)|\sum_{j=1}^{k} jk_j=k\}} \prod_{j=1}^{k}\left(\frac{1}{k_j!}\left(\frac{t_j}{j}\right)^{k_j}\right). \qquad (5.70)$$

□

Lemma 5.5
The Lauricella function $F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m)$ *can be written as*

$$F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m)$$
$$= 1 + \sum_{k=1}^{\infty} \frac{(a)_k}{(c)_k} \sum_{\{(k_1,\ldots,k_k)|\sum_{j=1}^{k} jk_j=k\}} \prod_{j=1}^{k} \frac{1}{k_j!}\left(\frac{t_j}{j}\right)^{k_j}, \qquad (5.71)$$

where the t_j*'s* $(j = 1, \ldots, k)$ *are again given by (5.68).*

Proof

$[F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m)]_0 = 1$ is again obtained by setting $i_1 = \cdots = i_m = 0$. Furthermore, (5.56) can be rewritten as

$$F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m) = [F_D(a, b_1, \ldots, b_m; c; \gamma_1, \ldots, \gamma_m)]_0$$
$$+ \sum_{k=1}^{\infty} \sum_{\{(i_1,\ldots,i_m)|\sum_{j=1}^{m} i_j=k\}} \frac{(a)_{i_1+\cdots+i_m}}{(c)_{i_1+\cdots+i_m}} \prod_{j=1}^{m} (b_j)_{i_j} \frac{{\gamma_j}^{i_j}}{i_j!}$$
$$= 1 + \sum_{k=1}^{\infty} \frac{(a)_k}{(c)_k} \sum_{\{(i_1,\ldots,i_m)|\sum_{j=1}^{m} i_j=k\}} \prod_{j=1}^{m} (b_j)_{i_j} \frac{{\gamma_j}^{i_j}}{i_j!} = 1 + \sum_{k=1}^{\infty} \frac{(a)_k}{(c)_k} \Lambda_k, \qquad (5.72)$$

where Λ_k denotes again the k-th degree term of the function $G(b_1, \ldots, b_m; \gamma_1, \ldots, \gamma_m)$, given by (5.61). Equation (5.71) now immediately follows from lemma 5.4.
N.B. The terms Λ_k are proportional to the *cycle indicator functions* $C_k(t_1, \ldots, t_k)$ given by

$$C_k(t_1, \ldots, t_k)$$
$$= \sum_{\{(k_1,\ldots,k_k)|\sum_{j=1}^{k} jk_j=k\}} \frac{k!}{1^{k_1}k_1!2^{k_2}k_2!\ldots k^{k_k}k_k!} {t_1}^{k_1}\ldots{t_k}^{k_k}. \qquad (5.73)$$

$C_k(t_1, \ldots, t_k)$ is the *generating function* for the number of permutations of k elements with k_1 unit cycles, k_2 2-cycles and so on; see [Riordan, 1978].

□

We are now able to express $E[Y_i]$ as a simple series of cycle indicator functions:

Theorem 5.2
Let the joint probability density function of the m random variables $Y_1, \ldots, Y_m$ be given by (5.46), then for $i = 1, \ldots, m$

$$E[Y_i] = \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \times \left\{\frac{1}{\beta} + \sum_{k=1}^{\infty} \frac{1}{\beta + k} \cdot \frac{1}{k!} C_k(t_{i1}, \ldots, t_{ik})\right\}, \quad (5.74)$$

where the t_{ij}'s $(j = 1, \ldots, k)$ are given by

$$t_{ij} = \sum_{l=1}^{m} \beta_l \left(1 - \frac{a_0}{a_l}\right)^j + \left(1 - \frac{a_0}{a_i}\right)^j. \quad (5.75)$$

Proof

Equation (5.74) is immediately obtained by combining (5.60), (5.71) and (5.73).

□

STEP 3

In this step we derive an accurate approximation of the sum of the simple series given by (5.74). We need some properties of cycle indicator functions, which are expressed by the following three lemmas.

Lemma 5.6 *[Riordan, 1978]*
The cycle indicator functions satisfy the following recursive relation:

$$C_{k+1}(s_1, \ldots, s_{k+1}) = \sum_{j=0}^{k} \frac{k!}{(k-j)!} s_{j+1} C_{k-j}(s_1, \ldots, s_{k-j}), \quad (5.76)$$

where $C_0 \stackrel{\text{def}}{=} 1$.

Lemma 5.7
Let $C_k(s)$ *and* $C_k(s+t)$ *denote* $C_k(s_1,\ldots,s_k)$ *and* $C_k(s_1+t_1,\ldots,s_k+t_k)$, *respectively, then*

$$C_k\,(s+t) = \sum_{j=0}^{k} \left(\begin{array}{c} k \\ j \end{array} \right) C_j\,(s)\, C_{k-j}\,(t)\,. \tag{5.77}$$

Proof

The proof is by induction on k. For $k = 1$ we have

$$C_1(s_1+t_1) = s_1 + t_1 = \left(\begin{array}{c} 1 \\ 1 \end{array} \right) C_1\,(s_1) + \left(\begin{array}{c} 1 \\ 0 \end{array} \right) C_1\,(t_1)\,, \tag{5.78}$$

so that (5.77) holds for $k = 1$.
Suppose (5.77) holds for some $k > 1$. Using lemma 5.6 we find

$$C_{k+1}\,(s+t) = \sum_{l=0}^{k} \frac{k!}{(k-l)!}(s_{l+1}+t_{l+1})C_{k-l}\,(s+t)$$

$$= \sum_{l=0}^{k} \frac{k!}{(k-l)!}s_{l+1}C_{k-l}\,(s+t) + \sum_{l=0}^{k} \frac{k!}{(k-l)!}t_{l+1}C_{k-l}\,(s+t)\,. \tag{5.79}$$

By using the induction hypothesis and lemma 5.6, the first sum in the right-hand side of (5.79) can be rewritten as follows:

$$\sum_{l=0}^{k} \frac{k!}{(k-l)!}s_{l+1}C_{k-l}\,(s+t)$$

$$= \sum_{l=0}^{k} \frac{k!}{(k-l)!}s_{l+1} \sum_{j=0}^{k-l} \left(\begin{array}{c} k-l \\ j \end{array} \right) C_j\,(s)\, C_{k-l-j}\,(t)$$

$$= \sum_{l=0}^{k} \sum_{j=l}^{k} \frac{k!}{(j-l)!(k-j)!}s_{l+1}C_{j-l}\,(s)\, C_{k-j}\,(t)$$

$$= \sum_{j=0}^{k} \left(\begin{array}{c} k \\ j \end{array} \right) C_{k-j}\,(t) \sum_{l=0}^{j} \frac{j!}{(j-l)!}s_{l+1}C_{j-l}\,(s)$$

$$= \sum_{j=0}^{k} \left(\begin{array}{c} k \\ j \end{array} \right) C_{k-j}(t)\, C_{j+1}(s) = \sum_{j=1}^{k+1} \left(\begin{array}{c} k \\ j-1 \end{array} \right) C_j(s)\, C_{k-j+1}(t)\,. \tag{5.80}$$

Analogously, we find the following identity for the second sum in the right-hand side of (5.79):

$$\sum_{l=0}^{k} \frac{k!}{(k-l)!} t_{l+1} C_{k-l}(s+t) = \sum_{j=0}^{k} \left(\begin{array}{c} k \\ j \end{array} \right) C_j(s)\, C_{k-j+1}(t)\,. \tag{5.81}$$

By adding the right-hand sides of (5.80) and (5.81) and using the identity $\left(\begin{array}{c} k \\ j-1 \end{array} \right) + \left(\begin{array}{c} k \\ j \end{array} \right) = \left(\begin{array}{c} k+1 \\ j \end{array} \right)$ we obtain (5.77) for $k+1$.

□

Before stating lemma 5.8, we redefine the terms t_{ij}, given by (5.75), as

$$t_{ij} = \sigma_j + {\tau_i}^j,\ i = 1, \ldots, m,\ j = 1, \ldots, k, \tag{5.82}$$

where

$$\sigma_j \stackrel{\text{def}}{=} \sum_{l=1}^{m} \beta_l \left(1 - \frac{a_0}{a_l}\right)^j,\ j = 1, \ldots, k \tag{5.83}$$

and

$${\tau_i} \stackrel{\text{def}}{=} \left(1 - \frac{a_0}{a_i}\right),\ i = 1, \ldots, m. \tag{5.84}$$

Lemma 5.8
Let the variables t_{ij} be given by (5.82) through (5.84), then the cycle indicator function $C_k(t_{i1}, \ldots, t_{ik})$ is given by:

$$C_k(t_{i1}, \ldots, t_{ik}) = k! \sum_{j=0}^{k} \frac{C_j(\sigma_1, \ldots, \sigma_j)}{j!} {\tau_i}^{k-j}. \tag{5.85}$$

Proof

Equation (5.85) is immediately obtained from lemma 5.7 and the identity $C_j\left(x, x^2, \ldots, x^j\right) = j! \cdot x^j$ [Riordan, 1978].

□

Theorem 5.3
The expected value $E[Y_i]$, $i = 1, \ldots, m$, *given by (5.74), can be rewritten as*

$$E[Y_i] = \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \times \left\{ \sum_{p=0}^{\infty} {\tau_i}^p \cdot \left\{ \sum_{k=0}^{\infty} \frac{1}{(\beta + k + p)k!} C_k(\sigma_1, \ldots, \sigma_k) \right\} \right\}, \tag{5.86}$$

where the variables σ_j *and* τ_i *are given by (5.83) and (5.84), respectively. Furthermore, for* $i = 1, \ldots, m$, $E[Y_i]$ *satisfies the following inequality:*

$$\left| E[Y_i] - \frac{\beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \Psi(\vec{\beta})}{1 - \tau_i} \right| < \frac{\beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \Psi(\vec{\beta})}{\beta} \cdot \frac{\tau_i}{(1 - \tau_i)^2}, \tag{5.87}$$

where $\Psi(\vec{\beta})$ *denotes the sum of the series* $\sum_{k=0}^{\infty} \frac{1}{(\beta+k)k!} C_k(\sigma_1, \ldots, \sigma_k)$ *and* $\vec{\beta}$ *denotes the vector* $(\beta_0, \ldots, \beta_m)$.

Proof

Equation (5.86) is immediately obtained from (5.74) and (5.85).
To prove the inequality in (5.87) we introduce the notation ϱ_{pk} for $\frac{1}{(\beta+k+p)k!} C_k(\sigma_1, \ldots, \sigma_k)$; consequently, $\Psi(\vec{\beta}) = \sum_{k=0}^{\infty} \varrho_{0k}$ and

$$E[Y_i] = \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \times \left\{ \sum_{p=0}^{\infty} {\tau_i}^p \cdot \left\{ \Psi(\vec{\beta}) - \sum_{k=0}^{\infty} (\varrho_{0k} - \varrho_{pk}) \right\} \right\}, \tag{5.88}$$

where

$$\varrho_{0k} - \varrho_{pk} = \frac{p}{(\beta + k)(\beta + k + p)} \frac{1}{k!} C_k(\sigma_1, \ldots, \sigma_k). \tag{5.89}$$

Hence,

$$\left| E[Y_i] - \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \cdot \sum_{p=0}^{\infty} {\tau_i}^p \cdot \Psi(\vec{\beta}) \right|$$

$$< \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \cdot \sum_{p=0}^{\infty} {\tau_i}^p \cdot \sum_{k=0}^{\infty} \frac{p}{(\beta + k)(\beta + k + p)} \frac{1}{k!} C_k(\sigma_1, \ldots, \sigma_k)$$

$$= \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \cdot \sum_{k=0}^{\infty} \frac{1}{(\beta+k)k!} C_k(\sigma_1, \ldots, \sigma_k) \cdot \sum_{p=0}^{\infty} \frac{p{\tau_i}^p}{\beta+k+p}$$

$$< \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \cdot \Psi(\vec{\beta}) \cdot \sum_{p=0}^{\infty} \frac{p{\tau_i}^p}{\beta} = \beta_i \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \cdot \Psi(\vec{\beta}) \cdot \frac{\tau_i}{\beta(1-\tau_i)^2}. \tag{5.90}$$

□

Theorem 5.4
The expected value of the random variable Y_0, defined by

$$Y_0 = 1 - \sum_{i=1}^{m} Y_i, \tag{5.91}$$

is given by

$$E[Y_0] = \beta_0 \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \Psi(\vec{\beta}). \tag{5.92}$$

Proof

The derivation of (5.92) consists of the following set of equalities, which is obtained by using several of the above lemmas:

$$E[Y_0] = \frac{\Gamma(\beta)}{\prod_{j=0}^{m} \Gamma(\beta_j)} \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \int \cdots \int_{\{0 \le \sum_{j=1}^{m} y_j \le 1\}} \prod_{j=1}^{m} {y_j}^{\beta_j - 1}$$

$$\times \left(1 - \sum_{j=1}^{m} y_j\right)^{\beta_0} \left(1 - \sum_{j=1}^{m} \frac{a_j - a_0}{a_j} y_j\right)^{-\beta} dy_1 \ldots dy_m$$

lemma 5.3 $$\frac{\Gamma(\beta)}{\prod_{j=0}^{m} \Gamma(\beta_j)} \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \frac{\Gamma(\beta_0+1) \prod_{j=1}^{m} \Gamma(\beta_j)}{\Gamma(\beta+1)}$$

$$\times F_D\left(\beta, \beta_1, \ldots, \beta_m; \beta+1; \left(1 - \frac{a_0}{a_1}\right), \ldots, \left(1 - \frac{a_0}{a_m}\right)\right)$$

$$= \frac{\beta_0}{\beta} \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j}$$

$$\times F_D\left(\beta, \beta_1, \ldots, \beta_m; \beta+1; \left(1-\frac{a_0}{a_1}\right), \ldots, \left(1-\frac{a_0}{a_m}\right)\right)$$

$$\underset{\text{lemma 5.5}}{=} \frac{\beta_0}{\beta} \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \times \left\{\sum_{k=0}^{\infty} \frac{(\beta)_k}{(\beta+1)_k} \cdot \frac{1}{k!} C_k(\sigma_1, \ldots, \sigma_k)\right\}$$

$$= \beta_0 \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \Psi(\vec{\beta}) \tag{5.93}$$

□

Using the fact that $(1-\tau_i)^{-1} = a_i/a_0,\ i = 1, \ldots, m$, theorems 5.3 and 5.4 yield

$$E[Y_i] = \Upsilon(\vec{\beta}) \cdot a_i \beta_i (1+\epsilon_i),\ i = 1, \ldots, m \tag{5.94}$$

and

$$E[Y_0] = \Upsilon(\vec{\beta}) \cdot a_0 \beta_0, \tag{5.95}$$

where $\Upsilon(\vec{\beta})$ is independent of i and is given by

$$\Upsilon(\vec{\beta}) = \frac{1}{a_0} \prod_{j=1}^{m} \left(\frac{a_0}{a_j}\right)^{\beta_j} \Psi(\vec{\beta}), \tag{5.96}$$

and where, according to (5.87), the absolute value of the relative error ϵ_i is bounded by $(a_i - a_0)/a_0\beta$. Consequently,

$$\lim_{\beta \to \infty} \epsilon_i = 0,\ i = 0, \ldots, m. \tag{5.97}$$

From (5.94) and (5.97) we conclude that we can make the difference between $E[Y_i]$ and $\Upsilon(\vec{\beta}) a_i \beta_i$ arbitrary small by choosing β sufficiently large, provided $\lim_{\beta\to\infty} E[Y_i]$ and $\lim_{\beta\to\infty} \Upsilon(\vec{\beta}) a_i \beta_i$ exist. Furthermore, if $E[Y_i]$ would equal $\Upsilon(\vec{\beta}) a_i \beta_i$ we would have $\Upsilon(\vec{\beta}) = \left(\sum_{j=0}^{m} a_j \beta_j\right)^{-1}$, because (5.91) implies that $\sum_{i=0}^{m} E[Y_i] = 1$. We therefore conjecture that we can also make the difference between $E[Y_i]$ and $a_i\beta_i / \left(\sum_{j=0}^{m} a_j \beta_j\right)^{-1}$ arbitrary small by choosing β sufficiently large, provided the limits exist. These arguments are made more precise in the following theorem.

Theorem 5.5
If we restrict ourselves to limits $\beta \to \infty$, for which

$$\lim_{\beta\to\infty} \frac{a_i\beta_i}{\sum_{j=0}^{m} a_j\beta_j},\ i = 0,\ldots,m,\ \textit{and}\ \lim_{\beta\to\infty} a_j\beta_j \cdot \Upsilon(\vec{\beta}),\ j = 0,\ldots,m, \tag{5.98}$$

exist, we have

$$\lim_{\beta\to\infty} E[Y_i] = \lim_{\beta\to\infty} \frac{a_i\beta_i}{\sum_{j=0}^{m} a_j\beta_j},\ i = 0,\ldots,m. \tag{5.99}$$

Proof

First, we remark that (5.94) and (5.98) imply the existence of the limits

$$\lim_{\beta\to\infty} E[Y_i],\ \ i = 0,\ldots,m, \text{ and } \lim_{\beta\to\infty} \sum_{j=0}^{m} a_j\beta_j \cdot \Upsilon(\vec{\beta}). \tag{5.100}$$

By using (5.94) and (5.97) and the existence of the limits in (5.98) and (5.100) we can write:

$$\begin{aligned}
\lim_{\beta\to\infty} \left(E[Y_i] - \frac{a_i\beta_i}{\sum_{j=0}^{m} a_j\beta_j} \right) &= \lim_{\beta\to\infty} \left(\frac{a_i\beta_i}{\sum_{j=0}^{m} a_j\beta_j} \left\{ \sum_{j=0}^{m} a_j\beta_j \Upsilon(\vec{\beta})(1+\epsilon_i) - 1 \right\} \right) \\
&= \lim_{\beta\to\infty} \left(\frac{a_i\beta_i}{\sum_{j=0}^{m} a_j\beta_j} \right) \lim_{\beta\to\infty} \left(\sum_{j=0}^{m} a_j\beta_j \Upsilon(\vec{\beta})(1+\epsilon_i) - 1 \right) \\
&= \lim_{\beta\to\infty} \left(\frac{a_i\beta_i}{\sum_{j=0}^{m} a_j\beta_j} \right) \lim_{\beta\to\infty} \left(\sum_{j=0}^{m} a_j\beta_j \Upsilon(\vec{\beta}) - 1 \right).
\end{aligned} \tag{5.101}$$

Summing the leftmost and rightmost sides of (5.101) for $i = 0,\ldots,m$ and using $\sum_{i=0}^{m} E[Y_i] = 1$ yields

$$0 = 1 \cdot \lim_{\beta\to\infty} \left(\sum_{j=0}^{m} a_j\beta_j \Upsilon(\vec{\beta}) - 1 \right), \tag{5.102}$$

which implies

$$\lim_{\beta \to \infty} \sum_{j=0}^{m} a_j \beta_j \Upsilon(\vec{\beta}) = 1. \tag{5.103}$$

Substitution of (5.103) in (5.101) yields the desired result. □

Finally, returning to the original problem - the computation of the expected values $E[\Theta_{k+1,i}|l; C_{nw_k}]$ for $i = l, \ldots, u$ - we obtain the following result:

$$\lim_{\alpha_{k|l} \to \infty} E[\Theta_{k+1,i}|l; C_{nw_k}] = \lim_{\alpha_{k|l} \to \infty} \frac{a_{ki}(\alpha_{ki|l} + n_{ki})}{\sum_{j=l}^{u} a_{kj}(\alpha_{kj|l} + n_{kj})}, \; i = l, \ldots, u, \tag{5.104}$$

provided the path $\alpha_{k|l} \to \infty$ is such that the limits in (5.104) exist. In the next subsection we show that, as the algorithm proceeds, $\alpha_{k|l}$ indeed becomes very large (cf. (5.113)), so that we conclude that the expected values $E[\Theta_{k+1,i}|l; C_{nw_k}]$, $i = l, \ldots, u$, are approximately given by $a_{ki}(\alpha_{ki|l} + n_{ki}) / \sum_{j=l}^{u} a_{kj}(\alpha_{kj|l} + n_{kj})$ and that the approximation becomes better as the algorithm proceeds.

5.3.4 Construction of the prior distribution and summary

We recall from the previous subsection that the expected values $E[\Theta_{k+1,i}|l; C_{nw_k}]$, $i = l, \ldots, u$, with respect to the joint probability density function given by (5.48) are approximately given by (5.104). Furthermore, it is well known that the expected values with respect to a $D(\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l})$ distribution are given by $\alpha_{k+1,i|l}/\alpha_{k+1|l}$, $i = l, \ldots, u$, where $\alpha_{k+1|l} = \sum_{j=l}^{u} \alpha_{k+1,j|l}$ [Wilks, 1962]. The problem now is to choose the parameters $\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l}$ such that the expected values with respect to the two distributions are the same.
In a simpler form, the problem can be stated as follows. Given random variables $Z_1, \ldots, Z_m$ following a $D(\eta_0, \ldots, \eta_m)$ distribution, choose the parameters $\eta_0, \ldots, \eta_m$ such that

$$(i) \quad E[Z_i] \stackrel{\text{def}}{=} \frac{\eta_i}{\sum_{j=0}^{m} \eta_j} = \frac{\zeta_i}{\sum_{j=0}^{m} \zeta_j}, \; i = 1, \ldots, m, \tag{5.105}$$

$$(ii) \quad \sum_{j=0}^{m} \eta_j = \xi, \tag{5.106}$$

where $\varsigma_0, \varsigma_1, \ldots, \varsigma_m$ and ξ are given parameters. From (5.105)-(5.106) we obtain immediately:

$$\eta_i = \xi \frac{\varsigma_i}{\sum_{j=0}^{m} \varsigma_j}, \; i = 0, \ldots, m. \tag{5.107}$$

In our original problem we put $\xi = \alpha_{k|l} + n$;[4] consequently, we obtain

$$\alpha_{k+1,i|l} = (\alpha_{k|l} + n) \frac{a_{ki}(\alpha_{ki|l} + n_{ki})}{\sum_{j=l}^{u} a_{kj}(\alpha_{kj|l} + n_{kj})}, \; i = l, \ldots, u. \tag{5.108}$$

We can now summarize the results obtained as follows.
At the beginning of Markov chain $k+1$ we have two prior distributions:

1. The prior distribution for the random variable L is given by (cf. (5.4))

$$Pr\{L = l\} = \lambda_{k+1,l}, \; l = m_m, \ldots, v_k, \tag{5.109}$$

where v_k denotes the minimum of m_M and the smallest cost value sampled during the generation of Markov chains $1, \ldots, k$, i.e.

$$v_k = \min \left\{ m_M, \min_{1 \le \kappa \le k} w_\kappa \right\} \tag{5.110}$$

and the parameters $\lambda_{k+1,l}$ are obtained from the posterior distribution for L of Markov chain k (cf. (5.13)):

$$\lambda_{k+1,l} = \frac{\lambda_{kl} \dfrac{\Gamma\left(\sum_{i=l}^{u} \alpha_{ki|l}\right)}{\Gamma\left(\sum_{i=l}^{u} \alpha_{ki|l} + n\right)} \prod_{i=l}^{u} \dfrac{\Gamma(\alpha_{ki|l} + n_{ki})}{\Gamma(\alpha_{ki|l})}}{\sum_{s=m_m}^{v_k} \lambda_{ks} \dfrac{\Gamma\left(\sum_{i=s}^{u} \alpha_{ki|s}\right)}{\Gamma\left(\sum_{i=s}^{u} \alpha_{ki|s} + n\right)} \prod_{i=s}^{u} \dfrac{\Gamma(\alpha_{ki|s} + n_{ki})}{\Gamma(\alpha_{ki|s})}}, \tag{5.111}$$

[4] According to Silver [1965], the additive property expressed by (5.108) ($\alpha_{k+1.i|l} \propto \alpha_{ki|l} + n_{ki}$) implies that we can think of the prior parameter $\alpha_{k+1.i|l}$ as an equivalent number of occurrences of a configuration with cost value i. Therefore, the sum of the posterior parameters and the sum of the prior parameters, $\sum_{i=l}^{u}(\alpha_{ki|l} + n_{ki})$ and $\sum_{i=l}^{u} \alpha_{k+1.i|l}$, respectively, can be interpreted as the number of equivalent occurrences representing the posterior knowledge of chain k and the prior knowledge of chain $k+1$, respectively. It seems reasonable to assume that these amounts of knowledge are the same; therefore we choose $\alpha_{k+1.i|l}$ such that $\sum_{i=l}^{u} \alpha_{k+1.i|l} = \sum_{i=l}^{u}(\alpha_{ki|l} + n_{ki}) = \alpha_{k|l} + n$.

for $l = m_m, .., v_k$. Equation (5.111) can be simplified by using (5.108), from which we obtain

$$\sum_{i=l}^{u} \alpha_{ki|l} = \sum_{i=l}^{u} (\alpha_{k-1|l} + n) \frac{a_{k-1,i}(\alpha_{k-1,i|l} + n_{k-1,i})}{\sum_{j=l}^{u} a_{k-1,j}(\alpha_{k-1,j|l} + n_{k-1,j})} = \alpha_{k-1|l} + n. \tag{5.112}$$

Consequently,

$$\alpha_{k|l} = \sum_{i=l}^{u} \alpha_{ki|l} = \alpha_{1|l} + (k-1)n = \sum_{i=l}^{u} \alpha_{1i|l} + (k-1)n, \tag{5.113}$$

where the hyperparameters $\alpha_{1l|l}, \ldots, \alpha_{1u|l}$ are the parameters of the conditional prior distribution for $(\Theta_{1l+1}, \ldots, \Theta_{1u})$ of the first Markov chain.
Substituting (5.113) in (5.111) yields

$$\begin{aligned}
\lambda_{k+1,l} &= \frac{\lambda_{kl} \dfrac{\Gamma\left(\alpha_{1|l} + (k-1)n\right)}{\Gamma\left(\alpha_{1|l} + kn\right)} \prod_{i=l}^{u} \dfrac{\Gamma(\alpha_{ki|l} + n_{ki})}{\Gamma(\alpha_{ki|l})}}{\sum_{s=m_m}^{v_k} \lambda_{ks} \dfrac{\Gamma\left(\alpha_{1|s} + (k-1)n\right)}{\Gamma\left(\alpha_{1|s} + kn\right)} \prod_{i=s}^{u} \dfrac{\Gamma(\alpha_{ki|s} + n_{ki})}{\Gamma(\alpha_{ki|s})}} \\
&= \frac{\lambda_{kl} \prod_{i=l}^{u} \prod_{j=0}^{n_{ki}-1} (\alpha_{ki|l} + j) \Big/ \prod_{j=0}^{n-1} ((k-1)n + \alpha_{1|l} + j)}{\sum_{s=m_m}^{v_k} \lambda_{ks} \prod_{i=s}^{u} \prod_{j=0}^{n_{ki}-1} (\alpha_{ki|s} + j) \Big/ \prod_{j=0}^{n-1} ((k-1)n + \alpha_{1|s} + j)},
\end{aligned} \tag{5.114}$$

for $l = m_m, \ldots, v_k$.
We remark that if the parameters $\alpha_{ki|s}$ do not depend on s, for instance in the case of a symmetric Dirichlet distribution, then (5.114) simplifies to

$$\lambda_{k+1,l} = \frac{\lambda_{kl} \Big/ \prod_{j=0}^{n-1} ((k-1)n + \alpha_{1|l} + j)}{\sum_{s=m_m}^{v_k} \lambda_{ks} \Big/ \prod_{j=0}^{n-1} ((k-1)n + \alpha_{1|s} + j)}, \quad l = m_m, \ldots, v_k. \tag{5.115}$$

As we can observe from (5.115), the expression for the parameter $\lambda_{k+1,l}$ is now independent of the frequency counts, and depends only on the sample size n. The cost values of the configurations sampled during the generation of Markov chain k enter (5.115) only through its denominator by means of the parameter v_k. Equation (5.114) can be used in the computation of the expected value of the random variable L at the beginning of the $(k+1)$-th Markov chain, given by

$$E_{k+1}[L] = \sum_{l=m_m}^{v_k} l \cdot \lambda_{k+1,l}. \tag{5.116}$$

2. The conditional prior distribution for the random variables $\Theta_{k+1,l+1}, \ldots, \Theta_{k+1,u}$ is a $D(\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l})$ distribution, whose parameters are given by (cf. (5.108))

$$\alpha_{k+1,i|l} = (\alpha_{k|l} + n) \frac{a_{ki}(\alpha_{ki|l} + n_{ki})}{\sum_{j=l}^{u} a_{kj}(\alpha_{kj|l} + n_{kj})}, \quad i = l, \ldots, u. \tag{5.117}$$

Consequently, the expected values of the random variables $\Theta_{k+1,i}$, $i = m_m, \ldots, u$, are given by

$$E[\Theta_{k+1,i}] = \sum_{l=m_m}^{\min(v_k,i)} E[\Theta_{k+1,i}|L=l] \cdot \lambda_{k+1,l} = \sum_{l=m_m}^{\min(v_k,i)} \frac{\alpha_{k+1,i|l}}{\alpha_{k+1|l}} \cdot \lambda_{k+1,l}. \tag{5.118}$$

At the end of Markov chain $k+1$ we have two posterior distributions.

1. The posterior distribution for the random variable L is the prior distribution for L of Markov chain $k+2$, so it is obtained from (5.109) and (5.114) by substituting $k+2$ and $k+1$ for $k+1$ and k, respectively.

2. The conditional posterior distribution for the random variables $\Theta_{k+1,l+1}, \ldots, \Theta_{k+1,u}$ is a $D(\alpha_{k+1,l|l} + n_{k+1,l}, \ldots, \alpha_{k+1,u|l} + n_{k+1,u})$ distribution.

5.4 Computational results

The computational results with Bayes's method are illustrated in this section by results obtained for the 48-city instance of the travelling salesman problem (GRO48) introduced by Grötschel [1977]. We have also applied the Bayesian analysis to larger instances of the TSP, but the results have been more or less the same. Therefore, rather than presenting many similar data for different instances, we extensively discuss the results obtained for the aforementioned single instance.

In order to be able to apply Bayes's method, we have to specify prior distributions of the unknown parameters. In our case, we have to provide the hyperparameters of the prior distributions of the first Markov chain, viz. m_m, m_M and λ_{1l}, $l = m_m, \ldots, m_M$ for the prior distribution of L, and u and $\alpha_{1i|l}$, $i = l, \ldots, u$, $l = m_m, \ldots, m_M$ for the conditional prior distribution of $(\Theta_{1l}, \ldots, \Theta_{1u})$. The computational results are discussed by considering the influence of the aforementioned hyperparameters on two quantities:

- the *expected minimum cost value* at the beginning of each Markov chain, given by (5.116);
- the *expected average cost value* at the beginning of each Markov chain, given by

$$E[\overline{C}_{k+1}] = \sum_{i=m_m}^{u} i \cdot E[\Theta_{k+1,i}] = \sum_{i=m_m}^{u} \sum_{l=m_m}^{\min(v_k,i)} \frac{\alpha_{k+1,i|l}}{\alpha_{k+1|l}} \cdot \lambda_{k+1,l} \cdot i$$

$$= \sum_{l=m_m}^{v_k} \sum_{i=l}^{u} \frac{\alpha_{k+1,i|l}}{\alpha_{k+1|l}} \cdot \lambda_{k+1,l} \cdot i. \qquad (5.119)$$

The first quantity is compared with the actual minimum cost value, which is 5048 for this instance, the second quantity with $\mu(c_{k+1})$, the average cost value sampled during the generation of a Markov chain, as given by (3.14).

We also revisit the accuracy of the approximation of the expected values $E[\Theta_{k+1,i}|l; C_{nw_k}]$ as derived in section 5.3.3, in particular the influence of the distance parameter δ, determining the decrement of the control parameter c in the simulated annealing algorithm.

Since the cost value of a solution of GRO48 can take thousands of different values, which requires the computation of α_i for thousands of values of i, we simplify the computations by dividing the range of cost values $[m_m, u]$ into 500 equally sized intervals. Thus, expressions like (5.119) are replaced by similar expressions in which the indices refer to intervals and cost values are replaced by the medians of the corresponding intervals.

(a) The expected minimum cost values

We recall that the expected minimum cost value $E_{k+1}[L]$ at the beginning of Markov chain $k+1$ is given by (5.116), and that the quantities $\lambda_{k+1,l}$ in (5.116) are recursively given by (5.114). In figure 5.1, $E_{k+1}[L]$ and v_k are plotted against c_k on a logarithmic scale. The curves in this figure are all calculated from the same sequence of Markov chains, generated by running the simulated annealing algorithm on GRO48 with the parameters χ_0, δ and ϵ_s set to 0.95, 1 and 10^{-6}, respectively, but with different hyperparameters for the prior distributions. The first prior distribution of L is chosen as a uniform distribution on the interval $[m_m, m_M]$, where m_m and m_M are set to 5000 and 5999, respectively. We choose a uniform distribution, because we assume that generally the only *a priori* knowledge we have about the minimum cost value is that it is somewhere between m_m and m_M. The hyperparameters $\alpha_{1i|l}$, $i = l, \ldots, u$, $l = m_m, \ldots, m_M$ are chosen such that $\sum_{i=l}^{u} \alpha_{1i|l} = \alpha_{1|l} = (u - l + 1) \cdot \alpha_{init}$, where α_{init} is a constant set to 10^3, 1 and 10^{-3} in the case of the dashed, the dotted and the thick solid curve in figure 5.1. We discuss the choice of the parameters $\alpha_{1i|l}$ in more detail in part (b) of this section.

From figure 5.1 we observe that the values $E_{k+1}[L]$ are highly influenced by the choice of α_{init}. If $\alpha_{init} \gg 1$, $E_{k+1}[L]$ quickly approaches v_k,[5] but if we choose $\alpha_{init} \leq 1$, there is a substantial difference between $E_{k+1}[L]$ and v_k, as indicated by the dotted and thick solid curves in figure 5.1. Unfortunately, we have no guidelines for the choice of α_{init}. Here, we briefly discuss a Bayesian solution to the problem of choosing α_{init}. The Bayesian solution is a two-stage approach due to

[5] This observation can be explained more precisely in the case where the hyperparameters $\alpha_{1i|l}$ are all equal to 1 (as opposed to our case, where only their average $\alpha_{1|l}$ equals 1). From appendix C we learn that in that case $\lim_{n\to\infty} E_2[L] = v_1$.

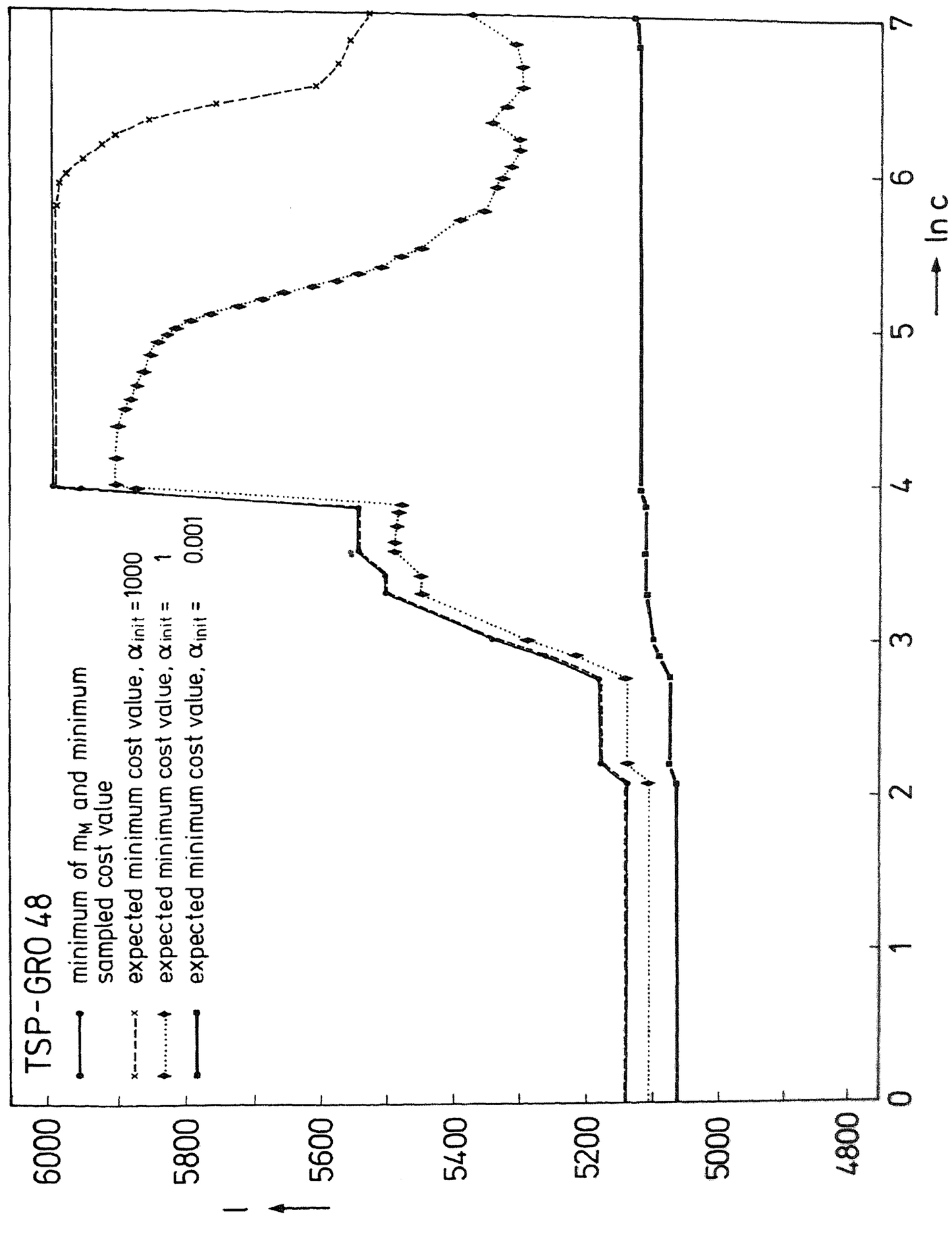


Figure 5.1: Smallest cost value sampled during the generation of a Markov chain and expected minimum cost values (for different values of α_{init}) plotted against the value of the control parameter c on a logarithmic scale.

Boender & Rinnooy Kan [1987] and incorporates a prior distribution for α_{init} in the model. Boender & Rinnooy Kan apply this approach successfully to similar problems.

Let A denote the random variable corresponding to the value of the unknown parameter α_{init}. If the prior distribution of A is given by a probability density function $\pi : \mathbb{R}^+ \rightarrow [0,1]$, then, following Boender & Rinnooy Kan [1987], it is possible to show that at the beginning of Markov chain $k+1$ the joint prior distribution for (L, A) is given by (cf. (5.111)):

$$f(l, \alpha_{init}) = \frac{\lambda_{kl}\pi(\alpha_{init}) \dfrac{\Gamma\left(\sum_{i=1}^{u}\alpha_{ki|l}\right)}{\Gamma\left(\sum_{i=1}^{u}\alpha_{ki|l}+n\right)} \dfrac{\prod_{i=l}^{u}\Gamma(\alpha_{ki|l}+n_{ki})}{\prod_{i=l}^{u}\Gamma(\alpha_{ki|l})}}{\sum_{s=m_m}^{v_k}\int_{\beta=0}^{\infty}\lambda_{ks}\pi(\beta) \dfrac{\Gamma\left(\sum_{i=s}^{u}\alpha_{ki|s}\right)}{\Gamma\left(\sum_{i=s}^{u}\alpha_{ki|s}+n\right)} \dfrac{\prod_{i=s}^{u}\Gamma(\alpha_{ki|s}+n_{ki})}{\prod_{i=s}^{u}\Gamma(\alpha_{ki|s})} d\beta}, \quad (5.120)$$

where the hyperparameters $\alpha_{ki|l}$ are completely determined by α_{init} (see part (b) of this section).

By integrating the distribution, given by (5.120), over all α_{init}, we obtain the marginal probability density function of L, yielding a new set of equations for the quantities $\lambda_{k+1,l}$, and a new expression for the expected value $E_{k+1}[L]$, which is, one might hope, a better approximation of the true minimum cost value than the expected value computed from (5.111).

Unfortunately, it does not seem possible to find an analytical expression for the integrals involved, so that one has to resort to numerical integration.

Finally, we mention that for $\alpha_{init} \gg 1$, the influence of the hyperparameters m_m and m_M on $E_{k+1}[L]$ is negligible, because, as we have seen, in that case $E_{k+1}[L]$ and v_k are virtually identical for sufficiently large k, whereas, for such k, v_k does not depend on m_m or m_M (only for small k we have $v_k = m_M$). However, if $\alpha_{init} \leq 1$, the analysis of the results with respect to the expected minimum cost values is further complicated by the influence of the hyperparameter m_m.

(b) The expected average cost values

First, we analyse the influence of the parameters for the prior distribution of $(\Theta_{1l}, \ldots, \Theta_{1u})$ on the expected average cost values.
In our experiments, we set $u = 25000$, which, from computational results, is known to be an upper bound to the cost values of virtually each configuration of this instance.
If no information is available on the parameters Θ_{1i}, the unknown probabilities of sampling a configuration with cost value i during the generation of the first Markov chain, then it is standard Bayesian practice to put

$$\alpha_{1i|l} = \alpha_{init}, \; i = l, \ldots, u, \; l = m_m, \ldots, m_M, \tag{5.121}$$

for some constant α_{init}.
However, since the configurations are drawn from a uniform distribution on $\mathcal{R}$ during the generation of the first Markov chain, we know that θ_{1i} equals the fraction of configurations with cost value i, denoted by $\omega(i)$. In the literature, it is usually assumed (and confirmed by computational results) that $\omega(i)$ follows a normal distribution, see for instance [Aarts, Korst & Van Laarhoven, 1988], i.e.

$$\omega(i) \propto \exp\left(-\frac{(i - \langle C \rangle)^2}{2\sigma_C^2}\right), \tag{5.122}$$

where $\langle C \rangle$ and σ_C are the average and the standard deviation of the cost values of the configurations, respectively. In our example, we estimate $\langle C \rangle$ and σ_C by the average and standard deviation of the cost values of a random walk through the configuration space, $\overline{C}$ and $\tilde{\sigma}_C$, respectively. By putting

$$\alpha_{1i|l} = \rho_l \cdot \exp\left(-\frac{(i - \overline{C})^2}{2\tilde{\sigma}_C^2}\right), \tag{5.123}$$

where ρ_l is given by

$$\rho_l = \frac{\alpha_{init} \cdot (u - l + 1)}{\sum_{j=l}^{u} \exp\left(-\frac{(j-\overline{C})^2}{2\tilde{\sigma}_C^2}\right)}, \tag{5.124}$$

we achieve that $E[\Theta_{1i}|L=l] \stackrel{\text{def}}{=} \frac{\alpha_{1i|l}}{\alpha_{1|l}} \simeq \omega(i)$, $l = m_m, \ldots, m_M$, and that

$$\overline{\alpha}_{1|l} \stackrel{\text{def}}{=} \frac{\alpha_{1|l}}{u-l+1} = \alpha_{init},\ l = m_m, \ldots, m_M. \tag{5.125}$$

In other words, if the hyperparameters $\alpha_{1i|l}$ are chosen according to (5.123) and (5.124), the distribution of the random variable Θ_{1i}, $i = l, \ldots, u$, is such that its expected value is approximately equal to the value of the parameter θ_{1i}, $i = l, \ldots, u$. Thus, (5.123) and (5.124) seem a better choice for the hyperparameters $\alpha_{1i|l}$ than (5.121). This is confirmed by figure 5.2, where the average cost value sampled during the generation of a Markov chain is plotted against the value of the control parameter c on a logarithmic scale. In the same figure the expected average cost values, as given by (5.119), are also plotted against c on a logarithmic scale; the dashed curve is obtained when the hyperparameters $\alpha_{1i|l}$ are given by (5.121), the dotted curve when they are given by (5.123) and (5.124). The three curves are calculated from the same sequence of Markov chains, obtained by running the simulated annealing algorithm with the parameters χ_0, δ, ϵ_s, m_m, m_M and α_{init} set to 0.9, 10, 10^{-6}, 5000, 5999 and 1, respectively. Clearly, when the $\alpha_{1i|l}$'s are given by (5.121) the expected values for the average cost are much too low. This can be explained by using the following two observations:

- In our example, the parameters $\overline{C}$ and $\tilde{\sigma}_C$ in (5.123) are given by 20300 and 1100, respectively. Consequently, (5.123) sets $\alpha_{1i|l}$ to 0 for i smaller than ca. 17000, whereas for the same range of i-values $\alpha_{1i|l}$ is set to $\alpha_{init} = 1$ through (5.121).

- The transformation of the $D(\alpha_{kl|l}+n_{kl}, \ldots, \alpha_{ku|l}+n_{ku})$ posterior distribution into a $D(\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l})$ prior distribution, as given by (5.117), satisfies the following condition:

$$\exists i_k \in \mathbb{N}: \quad \begin{array}{ll} \alpha_{k+1,i|l} \geq \alpha_{ki|l} + n_{ki}, & i = l, \ldots, i_k, \\ \alpha_{k+1,i|l} < \alpha_{ki|l} + n_{ki}, & i = i_k+1, \ldots, u. \end{array} \tag{5.126}$$

Equation (5.126) follows from the fact that $a_{kl} > a_{kl+1} > \cdots > a_{ku}$ and that $\sum_{i=l}^{u} \alpha_{k+1,i|l} = \sum_{i=l}^{u} (\alpha_{ki|l} + n_{ki})$.

Combining the two observations, we have that (5.121) yields a positive *bias* of the random variables Θ_{1i} for small values of i, and that the bias is further increased by the transformation of the posterior distribution of Θ_{1i} into a prior distribution of Θ_{2i} (and by all subsequent transformations).

Hereinafter, we only consider the case where the hyperparameters $\alpha_{ki|l}$ are given by (5.123) and (5.124), so that α_{init} is the only hyperparameter to be provided by the user.

The influence of the value of this parameter is shown in figure 5.3, where the relative difference between the expected and the sampled average cost value is plotted on a logarithmic scale against the value of the control parameter c for different values of α_{init}. The relative difference Δ_k is given by

$$\Delta_k = \frac{E[\overline{C}_k] - \mu(c_k)}{\mu(c_k)}. \tag{5.127}$$

The three curves are calculated from the same sequence of Markov chains, obtained by running the simulated annealing algorithm with the parameters χ_0, δ, ϵ_s, m_m and m_M set to 0.9, 1, 10^{-6}, 5000 and 5999, respectively. The solid curve is obtained for $\alpha_{init} = 0.001$, the dashed and dotted curves for $\alpha_{init} = 1$ and $\alpha_{init} = 1000$, respectively.

Before analysing figure 5.3, we recall that the conditional posterior distribution is a $D(\alpha_{kl|l} + n_{kl}, \ldots, \alpha_{ku|l} + n_{ku})$ distribution, so that if $\alpha_{init} \ll 1$, the prior $D(\alpha_{k+1,l|l}, \ldots, \alpha_{k+1,u|l})$ distribution is approximately obtained from a posterior $D(n_{kl}, \ldots, n_{ku})$ distribution, because the $\alpha_{ki|l}$ can be neglected. Conversely, if $\alpha_{init} \gg 1$, we can neglect the n_{ki}, so that the prior distribution is then approximately obtained from a posterior $D(\alpha_{kl|l}, \ldots, \alpha_{ku|l})$ distribution. In other words, if $\alpha_{init} \ll 1$, the $(k+1)$-th prior distribution is determined by the frequency counts, which are assumed to follow the stationary distribution of the $(k+1)$-th Markov chain, whereas if $\alpha_{init} \gg 1$, the $(k+1)$-th prior distribution is obtained from a transformation of the k-th prior distribution and thus from k transformations of the first prior distribution, which is assumed to be an accurate approximation of the stationary distribution of the first Markov chain. Thus, in both cases the prior distributions are assumed to follow the stationary distributions of the Markov chains. Therefore, we conclude that both small

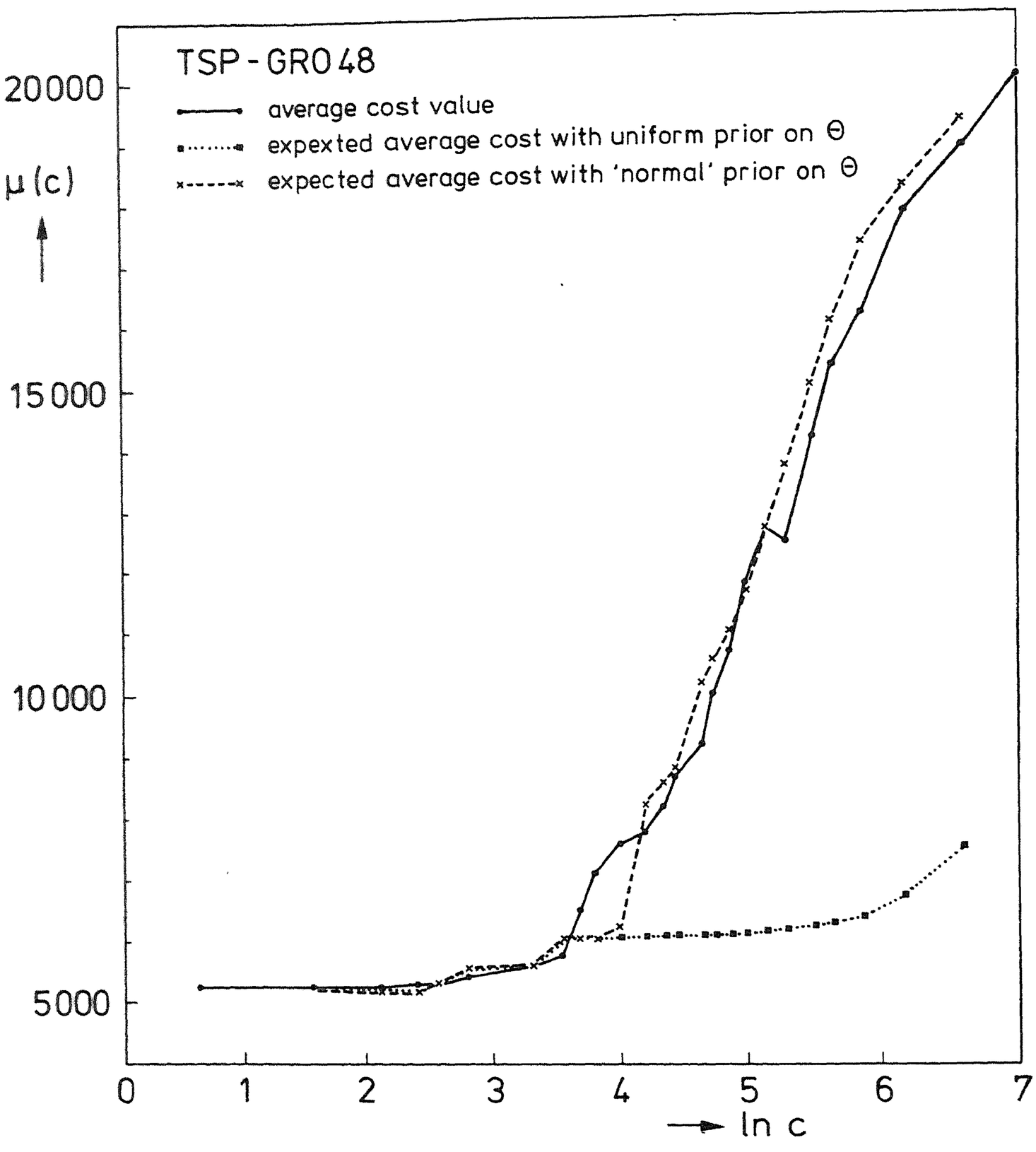


Figure 5.2: Average cost value sampled during the generation of a Markov chain and expected average cost values (for two prior distributions of $(\Theta_{1l}, \ldots, \Theta_{1u})$) plotted against the value of the control parameter c on a logarithmic scale.

and large values of α_{init} should yield approximately the same results for the expected average cost values.

Indeed, figure 5.3 confirms the robustness of the expected average cost values with respect to the choice of α_{init}: the curves obtained for $\alpha_{init} = 1$ and $\alpha_{init} = 0.001$, respectively, almost coincide, whereas the curve for $\alpha_{init} = 1000$ roughly follows the same pattern as the other two. Moreover, if we average the relative difference over all Markov chains, we obtain similar values for the three curves: 0.727%, 0.743% and 1.756% for $\alpha_{init} = 0.001$, 1 and 1000, respectively. Note that Δ_k varies between -20% and $+30\%$ and approaches 0 for $c \downarrow 0$. For larger instances of the travelling salesman problem we observed the same behaviour, but with smaller fluctuations of Δ_k (for instance, for the 120-city instance introduced by Grötschel [1977], Δ_k varies between -8% and $+18\%$).

Apart from being determined by the hyperparameters $\alpha_{k+1,i|l}$, the expected average cost value $E[\overline{C}_{k+1}]$ is also determined by the hyperparameters m_m and m_M (cf. (5.119)). We recall from part (a) of this section that, for $\alpha_{init} \gg 1$ and sufficiently large k, the parameters $\lambda_{k+1,l}$ are approximately given by

$$\lambda_{k+1,l} \simeq \begin{cases} 1 & \text{if } l = v_k, \\ 0 & \text{if } m_m \leq l < v_k. \end{cases} \tag{5.128}$$

Consequently, we can write

$$E[\overline{C}_{k+1}] \simeq \sum_{i=v_k}^{u} \frac{\alpha_{k+1,i|v_k}}{\alpha_{k+1|v_k}} \cdot i. \tag{5.129}$$

Using the definition of v_k, given by (5.110), we thus observe that $E[\overline{C}_{k+1}]$ is only influenced by m_M and then only for those values of k for which $\min_{1 \leq \kappa \leq k} w_\kappa > m_M$, since in that case $v_k = m_M$. We argue that this influence disappears as the algorithm proceeds, since eventually the smallest sampled cost value becomes smaller than m_M ($\lim_{k \to \infty} \min_{1 \leq \kappa \leq k} w_\kappa < m_M$).

If $\alpha_{init} \leq 1$, we recall from part (a) that $E_{k+1}[L]$ is somewhere between m_m and v_k, so that we may expect that in this case smaller values of m_m lead to smaller values of $E[\overline{C}_{k+1}]$. Indeed, figure 5.4 shows

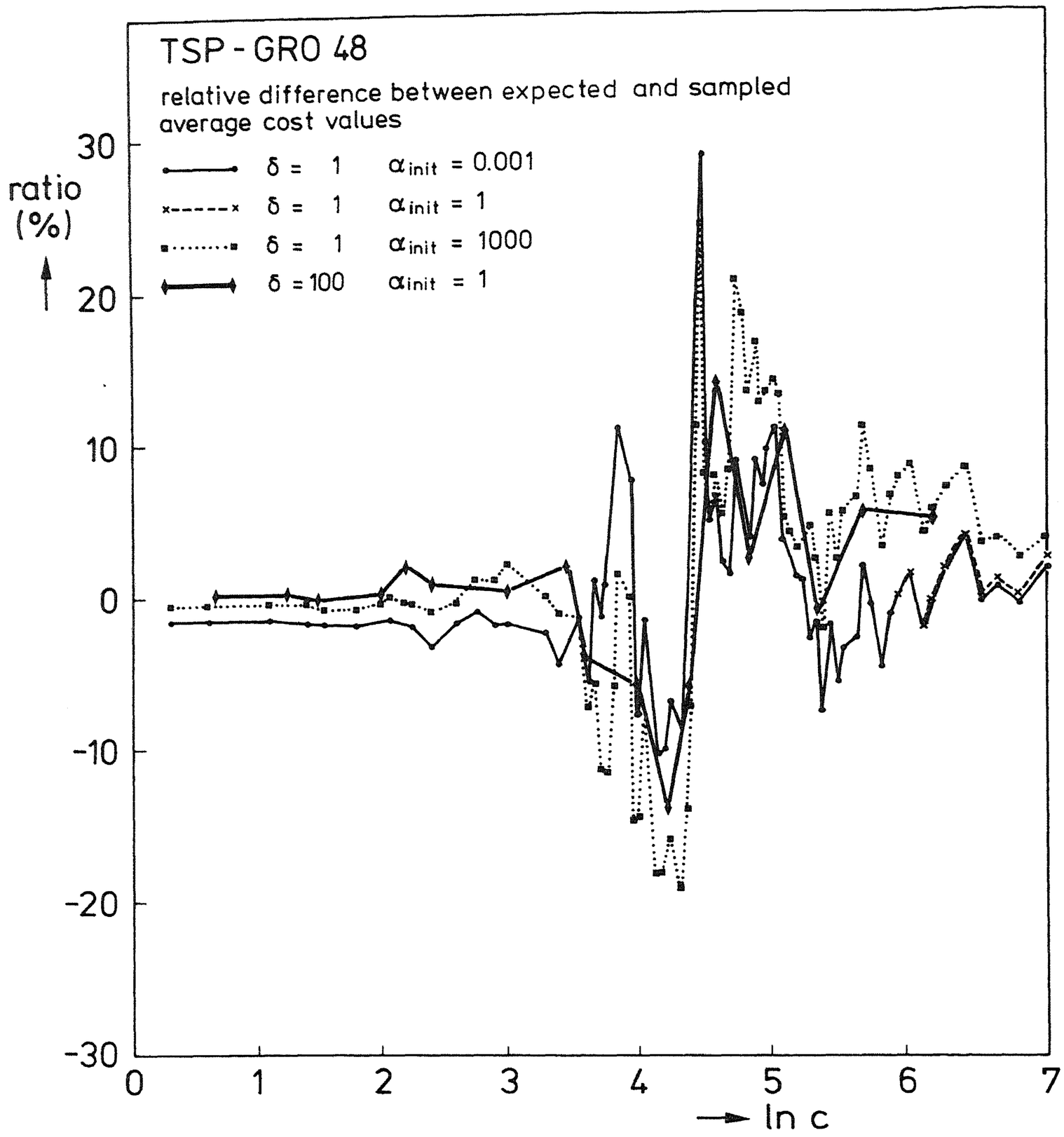


Figure 5.3: Relative difference between expected and sampled average cost values plotted on a logarithmic scale against the value of the control parameter c for different values of the hyperparameter α_{init}.

that if m_m is set to 4000, which is substantially smaller than the minimum cost value, the predicted average cost values are, on average, substantially smaller than the sampled ones. Moreover, as $c \downarrow 0$, the relative difference between expected and sampled cost values does not converge anymore to 0.

(c) Accuracy of the approximation of $E[\Theta_{k+1,i}|l; C_{nwk}]$

We recall from subsection 5.3.3 that the relative error ϵ_i in (5.94) is bounded by $(a_{ki} - a_{kl})/(a_{kl}(\alpha_{k|l} + n))$. Using (5.113) we can rewrite the relative error as

$$\left(\frac{a_{ki}}{a_{kl}} - 1\right) / (\alpha_{1|l} + k \cdot n). \tag{5.130}$$

Since $a_{kl} > a_{kl+1} > \cdots > a_{ku}$, the relative error is the largest for $i = u$; its absolute value is bounded by a multiple of

$$1 - \exp\left((l-u)\left(\frac{c_k - c_{k+1}}{c_k \cdot c_{k+1}}\right)\right). \tag{5.131}$$

Consequently, the accuracy of the approximation deteriorates for increasing values of $c_k - c_{k+1}$ or, equivalently, for increasing values of the distance parameter δ of the simulated annealing algorithm. However, even for $\delta = 100$, the expected average cost values roughly follow the same pattern as those obtained for $\delta = 1$, as can be seen by comparing the thick solid curve and the dashed curve in figure 5.3.

The computational results with Bayes's method are summarized in the following concluding remarks:

- We are able to choose the prior distribution of $(\Theta_{1l}, \ldots, \Theta_{1u})$ such that the Θ_{ki}'s are in expectation approximately equal to the parameters θ_{1i}, by taking into account the fact that θ_{1i} is equal to the fraction of configurations with cost value i, which is assumed to follow a normal distribution. The only hyperparameter of the prior distribution of $(\Theta_{1l}, \ldots, \Theta_{1u})$, α_{init}, does not significantly influence the expected average cost value $E[\overline{C}_{k+1}]$, $k = 1, 2, \ldots$. Furthermore, $E[\overline{C}_{k+1}]$, computed from the prior distribution at the beginning of the $(k+1)$-th Markov chain, predicts the average behaviour of the algorithm during that chain quite accurately.

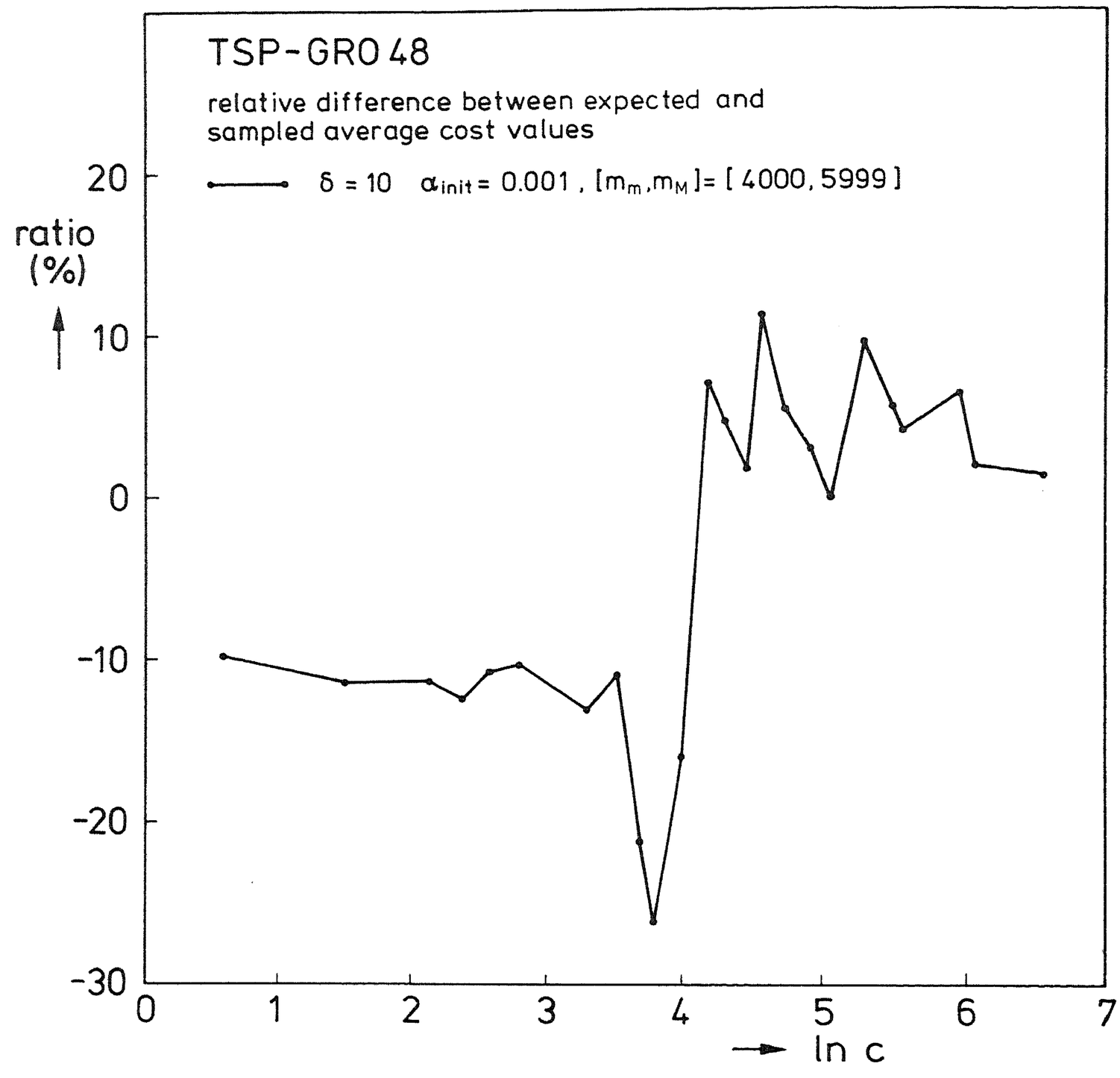


Figure 5.4: Relative difference between expected and sampled average cost values plotted against the value of the control parameter c on a logarithmic scale. The first prior distribution for L is given by a uniform distribution on the interval $[4000, 5999]$.

- We choose the first prior distribution of L as a uniform distribution on the interval $[m_m, m_M]$. We then find that the expected value of L is highly affected by the choice of α_{init}. For example, for $\alpha_{init} \gg 1$, we find that the expected minimum cost value $E_{k+1}[L]$, $k = 1, 2, \ldots$, is approximately given by v_k, the minimum of m_M and the minimum sampled cost value. To find an appropriate value for α_{init} is a difficult problem, but we conjecture that it can be solved by a two-stage approach, where a prior distribution on α_{init} is assumed and where the algorithm thus uses a probability distribution on values of α_{init} rather than a fixed value.

In the next section we are only interested in the expected average cost values. For that reason, we no longer assume the existence of a prior distribution on the random variable L, but carry out all computations with $L = l$, where l is a lower bound to all cost values. We then only have to compute the parameters $\alpha_{k+1,i}$, $i = l, \ldots, u$ at the beginning of Markov chain $k + 1$ (instead of computing the parameters $\alpha_{k+1,i|l}$, $i = l, \ldots, u$, $l = m_m, \ldots, v_k$).

5.5 Cooling schedules and Bayesian statistics

We recall from chapter 3 that the parameters of a cooling schedule are:

- the length of the Markov chains;
- the initial value of the control parameter;
- the decrement rule for the control parameter;
- the final value of the control parameter.

In this section we address the decision problem which arises when the decrement rule and the length of the Markov chains are considered. The decision problem is connected with the trade-off between fast decrement of the control parameter and short Markov chains and can

informally be described as the problem of choosing optimally from the range of alternatives between a small number of long Markov chains (i.e. large steps in the control parameter) and a large number of short Markov chains (i.e. small steps in the control parameter). In this section we use the results of section 5.3 to give a Bayesian solution to this problem.

Our approach to the decision problem can be outlined as follows. As a starting-point we take the decrement rule of the control parameter derived in section 3.3, which is given by

$$c_{k+1} = \frac{c_k}{1 + \frac{c_k \ln(1+\delta)}{3\sigma_k}}, \tag{5.132}$$

so that we can associate the size of the decrement of the control parameter with the size of the distance parameter δ. Next, we compute the expected average cost value of Markov chain $k+1$ for several values of δ. Suppose $D = \{\delta_1, \ldots, \delta_d\}$ is the set of δ-values for which the aforementioned computation is carried out and suppose the elements of D are such that $\delta_1 < \cdots < \delta_d$. Let $c_{k+1,j}$ and $E[\overline{C}_{k+1,j}]$, $j = 1, \ldots, d$, denote the value of the control parameter and the expected average cost value corresponding to the choice $\delta = \delta_j$, $j = 1, \ldots, d$, respectively. Using (5.108) and (5.129) we obtain

$$E[\overline{C}_{k+1,j}] = \frac{1}{\sum_{i=v_k}^{u} a_{kij}(\alpha_{ki|v_k} + n_{ki})} \cdot \sum_{i=v_k}^{u} a_{kij}(\alpha_{ki|v_k} + n_{ki}) \cdot i, \; j = 1, \ldots, d, \tag{5.133}$$

where

$$a_{kij} = \exp\left(\frac{i(c_{k+1,j} - c_k)}{c_k \cdot c_{k+1,j}}\right), \; j = 1, \ldots, d, \; i = v_k, \ldots, u. \tag{5.134}$$

It is straightforward to show that $E[\overline{C}_{k+1,j}]$ is a monotonously increasing function of $c_{k+1,j}$ and thus a monotonously decreasing function of δ_j, $j = 1, \ldots, d$. This is in accordance with our intuition that smaller values of the control parameter correspond to smaller average cost values.

We define the *yield* Y_j associated with the choice $\delta = \delta_j$ as $\overline{C} - E[\overline{C}_{k+1,j}]$, where the constant $\overline{C}$, denoting again the estimated

average cost value of all configurations (cf. (5.123)), is included so as to always have positive yields.[6] Thus, Y_j denotes the expected improvement of the algorithm over the average cost when c_{k+1} is set to $c_{k+1,j}$. If we are also able to find a suitable expression for the *cost* T_j associated with the choice $\delta = \delta_j$, then the optimal choice $j = j_0$ in the aforementioned decision problem follows from a simple *multicriteria analysis*: since we are interested in maximal yield and minimal cost, we choose j_0 such that

$$Y_{j_0} - \lambda \cdot T_{j_0} = \max\{Y_j - \lambda \cdot T_j | j = 1, \ldots, d\}, \tag{5.136}$$

where λ is a suitably chosen positive weighting factor.
To find an expression for T_j, we define T_j as the number of transitions sufficient to re-establish quasi-equilibrium after a decrement of c_k to $c_{k+1,j}$ (cf. section 3.3). If s_j is the number of times we have to apply (5.132) with $\delta = \delta_1$ to decrease c_k to $c_{k+1,j}$, then we assume $T_j = s_j \cdot T_1$. In other words, we assume that the number of transitions to restore quasi-equilibrium after one (large) step from c_k to $c_{k+1,j}$ is the same as the total number of transitions sufficient to restore quasi-equilibrium when c_k is decreased to $c_{k+1,j}$ in s_j small steps.
By applying the same line of reasoning as in the first part of the proof of theorem 3.2, we find

$$s_j \leq \frac{\ln(1+\delta_j)}{\ln(1+\delta_1)}, \; j = 1, \ldots, d, \tag{5.137}$$

where the equality in (5.137) holds if $\sigma(c)$ is constant for values of c i the interval $[c_{k+1,j}, c_k]$. Hereinafter, we approximate s_j by the boun of (5.137), in which case our decision problem is solved by choosin j_0 at the end of Markov chain k from $\{1, \ldots, d\}$ such that

$$E[\overline{C}_{k+1,j}] - \overline{C} + \lambda \cdot \frac{\ln(1+\delta_j)}{\ln(1+\delta_1)} \cdot T_1 \tag{5.138}$$

[6] There are several alternatives to this definition of yield; for example, we ma also define the yield as some function of the $100p$-th *percentile* y_j corresponding the choice $\delta = \delta_j$, i.e. as a function of the value y_j satisfying

$$Pr\{C(X) \leq y_j | \text{Markov chain } k+1 \text{ is generated at } c_{k+1} = c_{k+1,j}\} = p. \tag{5.13}$$

Note that this is also a monotonously increasing function of $c_{k+1,j}$.

is minimal for $j = j_0$. We discuss the choice of T_1 and λ later.
We can now define a 'Bayesian' cooling schedule, whose decrement rule and length of the $(k+1)$-th Markov chain L_{k+1} are given by

$$c_{k+1} = \frac{c_k}{1 + \frac{c_k \ln(1+\delta_{j_0})}{3\sigma_k}} \tag{5.139}$$

and

$$L_{k+1} = s_{j_0} \cdot T_1, \tag{5.140}$$

respectively, where j_0 is the optimal solution to the aforementioned decision problem.
We recall that in the cooling schedule described in section 3.3 the length of the $(k+1)$-th Markov chain is given by

$$L_{k+1} = L = \max_{i \in \mathcal{R}} |\mathcal{R}_i|, \; k = 0, 1, 2, \ldots \tag{5.141}$$

Note that the underlying idea of this cooling schedule is that small steps in the control parameter enable the use of short Markov chains. This corresponds to choosing 'a large amount of short Markov chains' in the trade-off discussed at the beginning of this section. Therefore, to compare this cooling schedule with the Bayesian schedule, we put $T_1 = L$ and $\delta_1 = \delta$, where δ is the distance parameter used in the non-Bayesian schedule. The comparison then allows us to check whether there is Bayesian support for this choice. We remark that the choice $T_1 = L$ implicitly defines a 'short' Markov chain as one with length L and that we therefore only investigate whether it is worthwhile to use Markov chains consisting of more than L transitions. There is also a computational reason for this restriction: if we consider Markov chains consisting of fewer than L transitions, the total number of Markov chains generated by the algorithm with the Bayesian schedule increases, which entails a further increase in computation time, because the computations involved in the Bayesian approach are carried out at the end of each Markov chain.
If K is the total number of Markov chains generated by the algorithm with the non-Bayesian schedule, then the total number of transitions is $K \cdot L$. During the execution of the algorithm, the sampled average cost value gradually decreases from approximately $\overline{C}$ to approximately

C_{opt}. Thus,

$$\lambda = \frac{\overline{C} - C_{opt}}{K \cdot L} \tag{5.142}$$

seems a good choice to weigh transitions and cost values.
The comparison between the two cooling schedules is carried out by running the algorithm on the 11 TSP-instances from the literature, discussed in section 4.2. The results are displayed in table 5.1. For the non-Bayesian schedule the results are obtained with the distance parameter δ set to 0.1, for the Bayesian schedule δ is chosen optimally from the set $D = \{0.1, 0.25, 0.5, 1, 2.5, 5, 10, 25, 50, 100\}$ at the end of each Markov chain. The averages are again obtained by running the algorithm five times on each instance.

Table 5.1: Average cost of final solution ($\overline{C}_{final}$), average number of transitions in thousands ($\overline{tr}$) and standard deviations in thousands (σ_C and σ_{tr}, respectively) for different-sized instances of the travelling salesman problem. The averages are obtained from 5 runs.

	non-Bayesian schedule				*Bayesian schedule*			
Problem	$\overline{C}_{final}$	σ_C	$\overline{tr}$	σ_{tr}	$\overline{C}_{final}$	σ_C	$\overline{tr}$	σ_{tr}
GRO48	5095	24	533	15	5100	49	537	10
TOM57	13068	51	875	67	13040	59	873	15
EUR100	21339	172	4320	55	21353	161	4325	111
KRO124	21468	102	4369	95	21529	144	4262	146
KRO125	22493	163	4317	100	22393	168	4117	129
KRO126	20928	169	4294	77	20876	71	4258	156
KRO127	21436	123	4405	107	21444	128	4396	121
KRO128	22455	190	4387	89	22306	123	4255	129
GRO120	7057	72	7241	138	7073	35	7013	175
LIN318	41957	177	103608	1913	41870	116	101143	3226
GRO442	5147	21	236655	1710	5147	37	226045	2451

From table 5.1 we observe that the results for both schedules are

very similar with respect to the average final cost value, whereas the average number of transitions is slightly less for the Bayesian schedule (up to 4.5% less for the largest instance). If the Bayesian approach is consistent, the small difference between the two schedules should either mean that the choice $\delta = \delta_1 = 0.1$ is nearly always the optimal choice or that (5.138) yields almost similar values for $j = 1, \ldots, d$ if $\delta \neq \delta_1$. A closer analysis of the data shows that both conclusions are valid; the first one is confirmed by table 5.2, where we display for each instance the average number of times $\bar{t}_j$ the choice $\delta = \delta_j$ is made, and from which we conclude that, on average, the choice $\delta = \delta_1$ is made in 93 out of 100 cases. We mention two more observations, which can be made from the data:

- A choice other than $\delta = \delta_1$ tends to be made only for the first Markov chains.

- The ratio between the CPU-times for the Bayesian and for the non-Bayesian schedule is typically between 1.2 and 1.5. The difference is due to the additional computational effort involved in the Bayesian approach and can be reduced by dividing the range of cost values into a smaller number of intervals than the 500 into which it is divided in order to obtain the results in table 5.1. Preliminary investigations indicate that this does not significantly influence the results obtained.

From the comparison we draw the following conclusions:

- The choice for small steps in the control parameter and short Markov chains, underlying the cooling schedule of section 3.3, is a good one; there is no Bayesian support for the assumption that larger steps and longer Markov chains would lead to a more efficient implementation of the algorithm.

- The observation that the values of the expression of (5.138) for different values of j are almost identical implies that there is not much difference between 'slow cooling' with short Markov chains and 'fast cooling' with long Markov chains. This confirms the first conclusion from section 3.4, namely that as long as 'cooling' is carried out reasonably accurately, the average final cost value is near-optimal.

Table 5.2: Average number of choices $\bar{t}_j$ for $\delta = \delta_j$, $j = 1, \ldots, 10$, for different-sized instances of the travelling salesman problem. The averages are obtained from 5 runs.

Problem	$\bar{t}_1$	$\bar{t}_2$	$\bar{t}_3$	$\bar{t}_4$	$\bar{t}_5$	$\bar{t}_6$	$\bar{t}_7$	$\bar{t}_8$	$\bar{t}_9$	$\bar{t}_{10}$
GRO48	133.6	1.4	3.2	2.0	0.6	1.6	1.6	0.6	1.4	3.2
TOM57	185.4	2.6	1.4	2.6	2.2	1.0	0.6	0.8	1.8	3.6
EUR100	284.0	2.4	3.0	2.2	1.8	1.0	0.4	0.4	0.8	9.4
KRO124	274.2	3.0	1.8	3.6	2.4	0.6	0.2	1.8	1.2	8.0
KRO125	263.8	2.2	1.8	2.8	2.0	2.4	1.4	1.2	1.4	6.8
KRO126	280.2	2.8	3.0	1.6	2.4	1.8	1.0	0.8	1.0	8.0
KRO127	289.2	2.4	1.4	4.0	1.8	2.0	1.2	1.2	1.4	7.6
KRO128	273.8	1.6	2.0	3.6	1.8	1.8	0.8	1.4	0.6	8.1
GRO120	331.6	3.0	3.0	3.4	1.2	2.6	1.2	1.0	1.0	9.0
LIN318	782.2	4.4	3.2	2.8	1.4	2.2	1.0	2.0	1.6	20.0
GRO442	849.2	5.8	7.4	4.8	3.0	1.6	2.4	2.0	2.6	22.4

5.6 Concluding remarks

The Bayesian approach to the inference problem formulated in section 5.1 has been successful in so far that the Bayesian results allow us to predict quite accurately the average behaviour of the simulated annealing algorithm in terms of the average cost value of the configurations constituting the Markov chains. Furthermore, the numerical experiments indicate that these predictions are quite robust with respect to the choice of the parameters of the Bayesian approach.

As yet, the Bayesian approach does not allow us to draw conclusions about the unknown minimum cost value, because such conclusions depend to a great extent on the value of the parameter α_{init}, for the choice of which we have no guidelines available. Here, a Bayesian approach to the problem of finding an appropriate value of α_{init} might be the answer.

We have revisited the problem of choosing the parameters of a cooling schedule and have used the Bayesian results to make an optimal choice

in the trade-off between small steps in the control parameter and short Markov chains on the one hand and large steps and long Markov chains on the other. This has led to the formulation of a Bayesian cooling schedule. From numerical experiments we conclude that the choice underlying the cooling schedule described in section 3.3, viz. the choice for small steps in c and short Markov chains, is usually the best one. Furthermore, despite the fact that the Bayesian schedule is based on some bold assumptions and approximations, the results obtained with both the Bayesian and the non-Bayesian schedule are approximately the same. Further research is necessary to investigate whether a refinement of the Bayesian schedule might eventually lead to a more effective and efficient algorithm than the current algorithm with the cooling schedule described in section 3.3.

Chapter 6

Conclusion

In this final chapter we return to some previous conclusions and briefly indicate some areas for future research.

The description of the simulated annealing algorithm in terms of the generation of Markov chains makes it possible to analyse the asymptotic convergence of the algorithm. Under certain mild conditions, the algorithm can be shown to find asymptotically a globally minimal solution to every combinatorial optimization problem. This is not a very satisfactory result - asymptotic convergence can be shown to hold unconditionally for the simplest algorithm imaginable, namely a complete enumeration of all configurations of the problem. Furthermore, the aforementioned conditions imply that the algorithm is allowed an unlimited amount of computation time. In practice, one is of course mainly interested in the finite-time behaviour of the algorithm, i.e. in the behaviour of simulated annealing as an approximation algorithm. However, it does not seem possible to show that simulated annealing is a *polynomial-time approximation scheme*: if the asymptotic behaviour is to be approximated arbitrarily closely in finite time, we can derive no other than exponential upper bounds on the computation time taken by the algorithm.

In this connection, we mention the area of probabilistic analysis, which still presents a host of challenging questions, as for instance:

1. Given a problem (or a probability distribution over the set of all problem instances) and a specific version of the simulated an-

nealing algorithm, is it possible to find an analytical expression for the probability distribution of the difference in cost between the returned solution and a globally minimal one?

2. Is it possible to show that, on average, the solution found by simulated annealing has a lower cost than the best solution found by simply enumerating as many solutions as is possible in the same time span? In other words, is simulated annealing a more effective approximation algorithm than complete enumeration? Thus far we have only discussed some results comparing the efficiency of simulated annealing and complete enumeration as optimization algorithms. The problem constructed by Lundy & Mees [1986] (see chapter 2) is an example where simulated annealing is more efficient than complete enumeration. On the other hand, conditions like (2.64) suggest that only if the number of elementary operations of the simulated annealing algorithm exceeds the total number of configurations will the returned solution be arbitrarily close to a globally minimal one.

In chapter 4, we have considered the performance of the algorithm from a computational point of view. The results obtained in this chapter support the widely held opinion that simulated annealing is a generally applicable, high-quality, but time-consuming approximation algorithm. However, we conjecture that the computation times taken by the algorithm can be considerably reduced through execution on a multi-processor architecture, without affecting the quality of the solutions found by the algorithm. See for example [Aarts, De Bont, Habers & Van Laarhoven, 1986], where it is shown that an almost linear speed-up can be achieved on an 8-processor architecture when applying simulated annealing to instances of the travelling salesman problem. Another promising area is that of the *Boltzmann machine* [Aarts & Korst, 1987], which can be seen as a massively parallel hardware implementation of the simulated annealing algorithm.

In chapter 5, we have considered the algorithm from a Bayesian point of view. The Bayesian approach allows us to analyse accurately the average behaviour of the simulated annealing algorithm with respect to cost values and gives some further guidance on the choice of implementation of the algorithm.

We remark that it would be interesting to investigate further applications of the Bayesian results. It would be worth investigating, for instance, whether our results support the assumption that the configuration density $\omega(i)$ (see section 5.4) is exponentially distributed in the region close to the globally minimal cost value [Hajek, 1985; Aarts, Korst, Van Laarhoven, 1988]. A Bayesian approach to simulated annealing different from the one described in this tract would be to use the set of solutions, each solution found by applying simulated annealing with a different initial configuration to the same problem instance, to make inferences about the true structure of the cost function of that instance.

Finally, we express the hope that we have been able to show that randomization is an important and promising concept in combinatorial optimization.

Appendix A

In this appendix we compare the asymptotic characteristics of the following two situations:

1. a sampling of configurations from a set $\mathcal{R}$, such that the random variables $X(k)$ are mutually stochastically independent ($X(k)$ denotes the k-th sampled configuration);

2. a sampling of configurations from a set $\mathcal{R}$, such that the random variables $X(k)$ constitute an irreducible and aperiodic Markov chain, with transition matrix $P = (p_{ij})$.

We assume that the sampling distribution in both situations is identical and given by the stationary distribution $\mathbf{q}$ of the Markov chain considered in the second situation. For convenience sake, we assume that $\mathcal{R} = \{1, \ldots, m\} \subset \mathbb{N}$. Thus, the frequency counts $N_i(n)$ are given by

$$N_i(n) = |\{X(k) | X(k) = i, 1 \leq k \leq n\}|, \; i = 1, \ldots, m. \qquad \text{(A.1)}$$

It is well known [Feller, 1950] that in the first situation the probability density function of $(N_1(n), \ldots, N_m(n))$ is a multinomial distribution with parameters $q_1, \ldots, q_m$. Furthermore, according to Serfling [1980], the probability density function of $\sqrt{n}\left(\frac{N_1(n)}{n} - q_1, \ldots, \frac{N_m(n)}{n} - q_m\right)$ is asymptotically multivariate normal with mean vector $\vec{\mu} = (0, \ldots, 0)$ and covariance matrix $\Sigma_1 = (\sigma_{ij})$, where

$$\sigma_{ij} = \begin{cases} q_i(1 - q_i) & \text{if } j = i, \\ -q_i q_j & \text{if } j \neq i. \end{cases} \qquad \text{(A.2)}$$

In the second situation it is possible to show that the probability density function of $\sqrt{n}\left(\frac{N_1(n)}{n} - q_1, \ldots, \frac{N_m(n)}{n} - q_m\right)$ is also asymptotically multivariate normal with mean vector $\vec{\mu} = (0, \ldots, 0)$ and covariance matrix Σ_2, but $\Sigma_2 \neq \Sigma_1$. In order to do so, we use the following results, which can be found in [Billingsley, 1978]:

- For a function f on $\mathcal{R}$, $\sqrt{n}\left(\frac{\sum_{k=1}^{n} f(X(k))}{n} - e\right)$ is asymptotically normal with parameters $\mu = 0$ and

$$\sigma^2 = \sum_{i=1}^{m}\sum_{j=1}^{m} \beta_{ij}(f(i) - e)(f(j) - e), \tag{A.3}$$

where

$$e = \sum_{i=1}^{m} q_i f(i), \tag{A.4}$$

$$\beta_{ij} = \delta_{ij} q_i - q_i q_j + 2q_i \sum_{l=1}^{\infty}(p_{ij}^{(l)} - q_j),\ i, j = 1, \ldots, m \tag{A.5}$$

and

$$p_{ij}^{(l)} = Pr\{X(l) = j | X(0) = i\},\ i = 1, \ldots, m, \tag{A.6}$$

i.e. $p_{ij}^{(l)}$ is the probability of reaching configuration j from configuration i after l transitions [Billingsley, 1978, p. 318-319].

- For random vectors $Y(n) = (Y_1(n), \ldots, Y_m(n))$ and $Z = (Z_1, \ldots, Z_m)$, a necessary and sufficient condition for $Y(n) \Rightarrow Z$ is that $\sum_{i=1}^{m} a_i Y_i(n) \Rightarrow \sum_{i=1}^{m} a_i Z_i$ for each vector $(a_1, \ldots, a_m) \in \mathbb{R}^m$. Here, $Y(n) \Rightarrow Z$ denotes that for $n \to \infty$ the distribution of $Y(n)$ converges to the distribution of Z. In the literature, this result is known as the *Cramèr-Wold device* [Billingsley, 1978, p. 335].

We remark that, by using (A.4), we can rewrite (A.3) as

$$\sigma^2 = \sum_{i=1}^{m}\sum_{j=1}^{m} f(i) f(j) \left(\beta_{ij} - q_j \sum_{l=1}^{m} \beta_{il} - q_i \sum_{k=1}^{m} \beta_{kj} + q_i q_j \sum_{k=1}^{m}\sum_{l=1}^{m} \beta_{kl}\right). \tag{A.7}$$

The following lemma is an immediate consequence of the first of the aforementioned results.

Lemma A.1
For each vector $\vec{a} \in \mathbb{R}^m$, $\sum_{i=1}^m a_i \sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right)$ *is asymptotically normal with parameters* $\mu = 0$ *and*

$$\sigma^2 = \sum_{i=1}^m \sum_{j=1}^m a_i a_j \left(\beta_{ij} - q_j \sum_{l=1}^m \beta_{il} - q_i \sum_{k=1}^m \beta_{kj} + q_i q_j \sum_{k=1}^m \sum_{l=1}^m \beta_{kl}\right), \quad \text{(A.8)}$$

where β_{ij} *is given by (A.5).*

Proof

For an arbitrary vector $\vec{a} \in \mathbb{R}^m$, we define the function $f_{\vec{a}}$ on $\mathcal{R}$ by:

$$f_{\vec{a}}(i) = a_i, \; i = 1, \ldots, m. \quad \text{(A.9)}$$

Consequently, $\sum_{k=1}^n f_{\vec{a}}(X(k)) = \sum_{i=1}^m a_i N_i(n)$, and since $\sqrt{n}\left(\frac{\sum_{i=1}^m a_i N_i(n)}{n} - \sum_{i=1}^m a_i q_i\right) = \sum_{i=1}^m a_i \sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right)$ the lemma immediately follows from the first observation.

□

We now apply the Cramèr-Wold device with $Y_i(n) = \sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right)$, $i = 1, \ldots, m$ and Z a random vector with a multivariate normal distribution with mean vector $\vec{\mu} = (0, \ldots, 0)$ and covariance matrix $\Sigma_2 = (\tau_{ij})$, where

$$\tau_{ij} = \beta_{ij} - q_j \sum_{l=1}^m \beta_{il} - q_i \sum_{k=1}^m \beta_{kj} + q_i q_j \sum_{k=1}^m \sum_{l=1}^m \beta_{kl}, \; i, j = 1, \ldots, m, \quad \text{(A.10)}$$

where β_{ij} is again given by (A.5). To prove $Y(n) \Rightarrow Z$ it is, according to the Cramèr-Wold device, sufficient to show that for each vector $(a_1, \ldots, a_m) \in \mathbb{R}^m$, $\sum_{i=1}^m a_i Y_i(n) = \sum_{i=1}^m a_i \sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right) \Rightarrow \sum_{i=1}^m a_i Z_i$. But the latter immediately follows from lemma A.1, because the fact that Z has a multivariate normal distribution with parameters $\vec{\mu} = (0, \ldots, 0)$ and $\Sigma_2 = (\tau_{ij})$ implies that $\sum_{i=1}^m a_i Z_i$ has a normal distribution with parameters $\mu = 0$ and $\sigma^2 = \sum_{i=1}^m \sum_{j=1}^m a_i a_j \tau_{ij}$.

To see that $\Sigma_1 \neq \Sigma_2$, we recall that we have just shown that for each vector $(a_1, \ldots, a_m) \in \mathbb{R}^m$, the distribution of $\sum_{i=1}^m a_i \sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right)$ is asymptotically normal with parameters $\mu = 0$ and

$\sigma^2 = \sum_{i=1}^{m} \sum_{j=1}^{m} a_i a_j \tau_{ij}$. Thus, if we put $a_j = 0$ for $j \neq i$ and $a_i = 1$, we have that the distribution of $\sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right)$ is asymptotically normal with parameters $\mu = 0$ and $\sigma^2 = \tau_{ii}$. Comparing this result to the following one [Billingsley, 1978, p. 319]:

- for $i = 1, \ldots, m$, $\sqrt{n}\left(\frac{N_i(n)}{n} - q_i\right)$ is asymptotically normal with parameters $\mu = 0$ and

$$\sigma^2 = q_i(1 - q_i) + 2q_i \sum_{l=1}^{\infty} (p_{ii}^{(l)} - q_i), \; i = 1, \ldots, m, \qquad \text{(A.11)}$$

we conclude that $\tau_{ii} = q_i(1 - q_i) + 2q_i \sum_{l=1}^{\infty}(p_{ii}^{(l)} - q_i)$. Since $\sigma_{ii} = q_i(1 - q_i)$, we have $\Sigma_1 \neq \Sigma_2$. In fact, it is possible to show that $\Sigma_1 = \Sigma_2$ if the term $q_i \sum_{l=1}^{\infty}(p_{ij}^{(l)} - q_j)$ is deleted from (A.5) (note that $\lim_{l \to \infty} p_{ij}^{(l)} = q_j$).

Summarizing, we have that in both situations the distribution of $\sqrt{n}\left(\frac{N_1(n)}{n} - q_1, \ldots, \frac{N_m(n)}{n} - q_m\right)$ is asymptotically multivariate normal with identical mean vectors and different covariance matrices $\Sigma_1 = (\sigma_{ij})$ and $\Sigma_2 = (\tau_{ij})$, where the difference $\sigma_{ij} - \tau_{ij}$ is related to

$$q_i \sum_{l=1}^{\infty}(p_{ij}^{(l)} - q_j). \qquad \text{(A.12)}$$

Using the following result on the speed of convergence of $p_{ij}^{(l)}$ to q [Feller, 1950]:

$$\forall j \in \mathcal{R} \; \exists A_j \in \mathbb{R}, \; \rho_j \in (0,1) : |p_{ij}^{(l)} - q_j| < A_j \rho_j^l, \qquad \text{(A.1}$$

it is straightforward to see that the difference between the two covariance matrices can be made arbitrarily small by considering only the sequence of configurations $X(t)$, $X(2t)$, etc., for some integer $t >$ in that case, the difference $\sigma_{ij} - \tau_{ij}$ is related to

$$q_i \sum_{k=1}^{\infty}(p_{ij}^{(kt)} - q_j), \qquad \text{(A.1}$$

and by using (A.13) this difference can be bounded by

$$q_i \sum_{k=1}^{\infty} A_j \rho_j^{kt} = \frac{q_i A_j \rho_j^t}{1 - \rho_j^t}. \qquad \text{(A.15)}$$

Since $\rho_j \in (0,1)$, the right-hand side of (A.15) can be made arbitrarily small by choosing t sufficiently large.

Appendix B

In this appendix we describe the alternative approach to the computation of the expected values $E[Y_i]$, $i = 1, \ldots, m$, which is due to Boender [1986].
We recall from section 5.3.2 that the joint probability density function of the random variables $Y_1, \ldots, Y_m$ is obtained by considering a transformation of the random variables $X_1, \ldots, X_m$, following a $\mathcal{D}(\beta_0, \ldots, \beta_m)$-distribution. Consequently, we can also write $E[Y_i]$ as (cf. (5.29))

$$E[Y_i] = E\left[\frac{a_i X_i}{a_0 + \sum_{j=1}^{m}(a_j - a_0)X_j}\right]$$

$$= \int_{\{0 \le \sum_{j=1}^{m} x_j \le 1\}} \cdots \int \frac{a_i x_i}{a_0 + \sum_{j=1}^{m}(a_j - a_0)x_j} f(x_1, \ldots, x_m)\, dx_1 \ldots dx_m, \quad \text{(B.1)}$$

for $i = 1, \ldots, m$, where $f(x_1, \ldots, x_m)$ is given by (5.28). Using the Taylor expansion of

$$\phi(x_1, \ldots, x_m) \stackrel{\text{def}}{=} \frac{a_i x_i}{a_0 + \sum_{j=1}^{m}(a_j - a_0)x_j} \quad \text{(B.2)}$$

at the point $(x_1, \ldots, x_m) = (\frac{\beta_1}{\beta}, \ldots, \frac{\beta_m}{\beta})$, which is given by

$$\phi(x_1, \ldots, x_m) = \sum_{n=1}^{\infty} \frac{1}{n!} \sum_{i_1=1}^{m} \cdots \sum_{i_n=1}^{m} (x_{i_1} - \frac{\beta_{i_1}}{\beta}) \ldots (x_{i_n} - \frac{\beta_{i_n}}{\beta})$$

$$\times \frac{\partial^n}{\partial x_{i_1} \ldots \partial x_{i_n}} \phi(\frac{\beta_1}{\beta}, \ldots, \frac{\beta_m}{\beta}), \tag{B.3}$$

we find that we can rewrite (B.1) as

$$E[Y_i] = \sum_{n=1}^{\infty} \frac{1}{n!} \left\{ \sum_{i_1=1}^{m} \cdots \sum_{i_n=1}^{m} \frac{\partial^n}{\partial x_{i_1} \ldots \partial x_{i_n}} \phi(\frac{\beta_1}{\beta}, \ldots, \frac{\beta_m}{\beta}) \right.$$

$$\left. \times \left(\int \cdots \int_{\{0 \le \sum_{j=1}^{m} x_j \le 1\}} (x_{i_1} - \frac{\beta_{i_1}}{\beta}) \ldots (x_{i_n} - \frac{\beta_{i_n}}{\beta}) \cdot f(x_1, \ldots, x_m) dx_1 \ldots dx_m \right) \right\}, \tag{B.4}$$

for $i = 1, \ldots, m$.

By considering only the first two terms in (B.4), the following approximation of $E[Y_i]$ is found [Boender, 1986]:

$$E[Y_i] \simeq \frac{a_i \beta_i}{\sum_{j=0}^{m} a_j \beta_j} \left\{ 1 + \frac{\beta}{\beta + 1} \left(\frac{\sum_{j=0}^{m} {a_j}^2 \beta_j}{\left(\sum_{j=0}^{m} a_j \beta_j \right)^2} - \frac{a_i}{\sum_{j=0}^{m} a_j \beta_j} \right) \right\}, \tag{B.5}$$

for $i = 1, \ldots, m$.

Unfortunately, it does not seem possible to find an analytical expression for the error in the approximation of (B.5) - it requires the evaluation of integrals of the form

$$\int \cdots \int_{\{0 \le \sum_{j=1}^{m} x_j \le 1\}} \frac{\partial^3}{\partial x_{i_1} \partial x_{i_2} \partial x_{i_3}} \phi(\psi_1, \ldots, \psi_m)$$

$$\times (x_{i_1} - \frac{\beta_{i_1}}{\beta}) \ldots (x_{i_n} - \frac{\beta_{i_n}}{\beta}) f(x_1, \ldots, x_m) dx_1 \ldots dx_m, \tag{B.6}$$

where $\psi_i = \frac{\beta_i}{\beta} + \varepsilon_i \left(x_i - \frac{\beta_i}{\beta} \right)$, $i = 1, \ldots, m$, $\varepsilon_i \in (0,1)$, $i = 1, \ldots, m$ and $i_1, i_2, i_3 \in \{1, \ldots, m\}$.

In addition, extending the approximation of (B.5) by higher order terms does not seem to be a promising way to go either, because of the irregularity of these terms (the third-order term, for example, is given by

$$\frac{a_i \beta_i}{\left(\sum_{j=0}^{m} a_j \beta_j \right)^4} \cdot \frac{2\beta}{(\beta+2)(\beta+1)} \left\{ \beta \left(\sum_{j=0}^{m} {a_j}^3 \beta_j \right) \right.$$

$$\left. - 2 \left(\sum_{j=0}^{m} a_j \beta_j \right) \left(\sum_{j=0}^{m} {a_j}^2 \beta_j \right) + 2 a_i \left(\sum_{j=0}^{m} a_j \beta_j \right)^2 - {a_i}^2 \beta \left(\sum_{j=0}^{m} a_j \beta_j \right) \right\} \tag{B.7}$$

(4 subterms), the fourth-order term consists of 17 subterms etc.).

Appendix C

In this appendix we compute $E_2[L]$ for uniform distributions, i.e. for the case where the parameters λ_{1l} and $\alpha_{1i|l}$ are given by

$$\lambda_{1l} = \frac{1}{m_M - m_m + 1},\ l = m_m, \ldots, m_M \tag{C.1}$$

and

$$\alpha_{1i|l} = 1,\ i = l, \ldots, u,\ l = m_m, \ldots, m_M. \tag{C.2}$$

Substituting (C.1) and (C.2) in (5.111) yields

$$\lambda_{2l} = \frac{(u-l)!/(u-l+n)!}{\sum_{s=m_m}^{m_M} (u-s)!/(u-s+n)!},\ l = m_m, \ldots, m_M. \tag{C.3}$$

Consequently,

$$E_2[L] = \sum_{l=m_m}^{m_M} l\lambda_{2l} = \frac{\sum_{l=m_m}^{m_M} l(u-l)!/(u-l+n)!}{\sum_{s=m_m}^{m_M} (u-s)!/(u-s+n)!}. \tag{C.4}$$

Putting $a = u - m_m$ and $b = u - m_M$ and using the identity

$$\sum_{t=a}^{\infty} \frac{t!}{(t+n)!} = \frac{1}{n-1} \frac{a!}{(a+n-1)!} \tag{C.5}$$

(see [Boender, 1984]), we find for the denominator in (C.4)

$$\sum_{s=m_m}^{m_M} \frac{(u-s)!}{(u-s+n)!} = \sum_{t=a}^{b} \frac{t!}{(t+n)!} = \sum_{t=b}^{\infty} \frac{t!}{(t+n)!} - \sum_{t=a+1}^{\infty} \frac{t!}{(t+n)!}$$

$$= \frac{1}{n-1}\left\{\frac{b!}{(b+n-1)!} - \frac{(a+1)!}{(a+n)!}\right\}. \qquad \text{(C.6)}$$

For the nominator in (C.4) we find

$$\sum_{l=m_m}^{m_M} \frac{l(u-l)!}{(u-l+n)!} = \sum_{t=a}^{b} \frac{(u-t)t!}{(t+n)!} = (u+1)\sum_{t=b}^{a} \frac{t!}{(t+n)!} - \sum_{t=b}^{a} \frac{(t+1)!}{(t+n)!} =$$

$$\frac{u+1}{n-1}\left\{\frac{b!}{(b+n-1)!} - \frac{(a+1)!}{(a+n)!}\right\} - \frac{1}{n-2}\left\{\frac{(b+1)!}{(b+n-1)!} - \frac{(a+2)!}{(a+n)!}\right\}. \qquad \text{(C.7)}$$

Combining (C.4), (C.6) and (C.7) yields

$$E_2[L] = (u+1) - \frac{n-1}{n-2}\left\{\frac{(b+1)!(a+n)! - (b+n-1)!(a+2)!}{b!(a+n)! - (b+n-1)!(a+1)!}\right\}$$

$$= (u+1) - \frac{n-1}{n-2}\left\{\frac{(b+1)p(n) - q(n)(a+2)}{p(n) - q(n)}\right\}, \qquad \text{(C.8)}$$

where $p(n) = (a+n)\ldots(a+2)$ and $q(n) = (b+n-1)\ldots(b+1)$.
Equation (C.8) can be simplified to

$$E_2[L] = (u+1) - \frac{n-1}{n-2}\left\{(b+1) + (b-a+1)\frac{q(n)/p(n)}{1-q(n)/p(n)}\right\}. \qquad \text{(C.9)}$$

Finally, using

$$\lim_{n\to\infty} \frac{q(n)}{p(n)} = \lim_{n\to\infty} \frac{(a+1)\ldots(b+1)}{(a+n)\ldots(b+n)} = 0, \qquad \text{(C.10)}$$

we find

$$\lim_{n\to\infty} E_2[L] = (u+1) - (b+1) = m_M. \qquad \text{(C.11)}$$

Bibliography

AARTS, E.H.L., G.F.M. BEENKER AND J.H.M. KORST (1985), Asymptotic Probability Calculations for Aselect Sampling, *Philips Research Report No. 6073.*

AARTS, E.H.L, F.M.J. DE BONT, J.H.A. HABERS AND P.J.M. VAN LAARHOVEN (1986), Parallel Implementations of the Statistical Cooling Algorithm, *Integration 4*, 209-238.

AARTS, E.H.L. AND J.H.M. KORST (1987), Boltzmann Machines and Their Applications, *Proc. PARLE, Springer Lecture Notes in Computer Science 258*, 34-50.

AARTS, E.H.L., J.H.M. KORST AND P.J.M. VAN LAARHOVEN (1988), Solving Travelling Salesman Problems by Simulated Annealing: A Quantitative Analysis, *J. of Statys. Phys. 50*, 187-206.

AARTS, E.H.L AND P.J.M. VAN LAARHOVEN (1985a), Statistical Cooling: A General Approach to Combinatorial Optimization Problems, *Philips J. of Research 40*, 193-226.

AARTS, E.H.L AND P.J.M. VAN LAARHOVEN (1985b), A New Polynomial Time Cooling Schedule, *Proc. IEEE Int. Conference on Computer-Aided Design*, Santa Clara, November 1985, pp. 206-208.

ADAMS, J., E. BALAS AND D. ZAWACK (1988), The Shifting Bottleneck Procedure for Job Shop Scheduling, *Management Sci. 34*, 391-401.

AHO, A.V., J.E. HOPCROFT AND J.D. ULLMAN (1974), *The Design and Analysis of Computer Algorithms*, (Addison Wesley, Reading).

ANILY, S. AND A. FEDERGRUEN (1986), Simulated Annealing Methods with General Acceptance probabilities, Graduate School

of Business, Columbia University, New York, to appear in: *Applied Probability.*

APPELL, P. AND J. KAMPÉ DE FÉRIÉT (1926), *Fonctions hypergeometriques et hypersphériques*, (Gauthier Villars, Paris).

BALL, M. AND M. MAGAZINE (1981), The Design and Analysis of Heuristics, *Networks 11*, 215-219.

BAYES, TH.R. (1763), An Essay Towards Solving a Problem in the Doctrine of Chances, *Philosophical Transactions of the Royal Society London*, 370-418, reprinted in: *Biometrika 45 (1958)*, 293-315.

BELLMAN, R.E. (1958), On a Routing Problem, *Quart. Appl. Math. 16*, 87-90.

BILLINGSLEY, P. (1978), *Probability and Measure*, (Wiley, New York).

BINDER, K. (1978), *Monte Carlo Methods in Statistical Physics*, (Springer, New York).

BOENDER, C.G.E. (1984), The Generalized Multinomial Distribution: A Bayesian Analysis and Applications, *Ph.D. Thesis*, Erasmus University Rotterdam, Rotterdam.

BOENDER, C.G.E. (1986), *private communication.*

BOENDER, C.G.E. AND A.H.G. RINNOOY KAN (1987), A multinomial Bayesian approach to the estimation of population an vocabulary size, *Biometrika 74*, 849-856.

BONOMI, E. AND J.-L. LUTTON (1984), The N-city Travelling Salesman Problem: Statistical Mechanics and the Metropolis Algorithm, *SIAM Rev. 26*, 551-568.

BONOMI, E. AND J.-L. LUTTON (1986), The Asymptotic Behaviour of Quadratic Sum Assignment Problems: A Statistical Mechanics Approach, *European J. of Oper. Res. 26*, 295-300.

BREUER, M., ED. (1972), *Design Automation of Digital Systems*, (Prentice Hall, New York).

BURKARD, R.E. AND F. RENDL (1984), A thermodynamically motivated simulation procedure for combinatorial optimization problems, *European J. of Oper. Res. 17*, 169-174.

CARLIER, J. AND E. PINSON (1986), An Algorithm for Solving the Job-shop Problem, to appear in: *Management Sci.*

ČERNY, V. (1985), Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm, *J. Opt. Theory Appl. 45*, 41-51.

COOK, S.A. (1983), An Overview of Computational Complexity, *Comm. ACM 26*, 400-408.

FELLER, W. (1950), *An Introduction to Probability Theory and Applications, vol. 1*, (Wiley, New York).

FISHER, H. AND G.L. THOMPSON (1963), Probabilistic Learning Combinations of Local Job-shop Scheduling Rules, in: J.F. MUTH AND G.L. THOMPSON, EDS., *Industrial Scheduling*, (Prentice Hall, Englewood Cliffs), pp. 225-251.

GAREY, M.R. AND D.S. JOHNSON (1979), *Computers and Intractability: A Guide to the Theory of NP-Completeness*, (W.H. Freeman and Co., San Francisco).

GELFAND, S.B. AND S.K. MITTER (1985), Analysis of Simulated Annealing for Optimization, *Proc. 24th Conf. on Decision and Control*, Ft. Lauderdale, December 1985, pp. 779-786.

GEMAN, S. AND D. GEMAN (1984), Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images, *IEEE Proc. Pattern Analysis and Machine Intelligence PAMI-6*, 721-741.

GIDAS, B. (1985), Nonstationary Markov Chains and Convergence of the Annealing Algorithm, *J. Statis. Phys. 39*, 73-131.

GOLDEN, B.L. AND C.C. SKISCIM (1986), Using Simulated Annealing to Solve Routing and Location Problems, *Naval Logistics Research Quarterly 33*, 261-279.

GRÖTSCHEL, M. (1977), *Polyedrische Charakterisierungen Kombinatorischer Optimierungsprobleme* (in German), (Hain, Meisenheim am Glan).

GRÖTSCHEL, M. (1984), Polyedrische Kombinatorik und Schnittebenenverfahren (in German), Universität Augsburg, *Preprint No. 38.*

GRÖTSCHEL, M. (1987), *private communication.*

HAJEK, B. (1985), A Tutorial Survey of Theory and Applications of Simulated Annealing, *Proc. 24th Conf. on Decision and Control*, Ft. Lauderdale, December 1985, pp. 755-760.

HAJEK, B. (1986), Cooling Schedules for Optimal Annealing, submitted to *Mathematics of Operations Research, revised version.*

HOGG, R.V. AND A.T. CRAIG (1978), *Introduction to Mathematical Statistics*, (Collier Macmillan, London, 4th ed.).

ISAACSON, D. AND R. MADSEN (1974), Positive Columns for Stochastic Matrices, *J. Appl. Prob. 11*, 829-834.

JOHNSON, D.S. AND C.H. PAPADIMITRIOU (1985), Computational Complexity, in: E.L. LAWLER, J.K. LENSTRA, A.H.G. RINNOOY KAN AND D.B. SHMOYS, EDS., *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, (Wiley, Chichester), pp. 37-85.

JOHNSON, D.S., C.H. PAPADIMITRIOU AND M. YANNAKAKIS (1985), How easy is local search?, *Proc. Annual Symp. of Foundations of Computer Science*, Los Angeles, pp. 39-42.

JOHNSON, D.S., C.R. ARAGON, L.A. MCGEOCH AND C. SCHEVON (1987), Optimization by Simulated Annealing: an Experimental Evaluation (Part I), *preprint.*

JOHNSON, D.S. (1987), *private communication.*

KARG, R.L. AND G.L. THOMPSON (1964), A Heuristic Approach to Solving Travelling Salesman Problems, *Management Sci. 10*, 225-247.

KERN, W. (1986a), A Probabilistic Analysis of the Switching Algorithm for the Euclidean TSP, Universität zu Köln, *Report No. 86.28.*

KERN, W. (1986b), On the Depth of Combinatorial Optimization Problems, Universität zu Köln, *Report No. 86.33.*

KIRKPATRICK, S., C.D. GELATT JR. AND M.P. VECCHI (1983), Optimization by Simulated Annealing, *Science 220*, 671-680.

KIRKPATRICK, S. (1984), Optimization by Simulated annealing: Quantitative Studies, *J. Statis. Phys. 34*, 975-986.

KROLAK, P.D., W. FELTS AND G. MARBLE (1971), A Man-Machine Approach Toward Solving the Traveling Salesman Problem, *Comm. ACM 14*, 327-334.

LAARHOVEN, P.J.M. VAN AND E.H.L. AARTS (1987), *Simulated Annealing: Theory and Applications*, (Reidel, Dordrecht).

LAARHOVEN, P.J.M. VAN, E.H.L. AARTS, J.H. VAN LINT AND L.T. WILLE (1988), New Upper Bounds for The Football Pool Problem for 6, 7 and 8 Matches, to appear in: *J. Comb. Theory A.*

LAURICELLA, G., (1893), Sulle funzioni ipergeometriche a piu variabili, *Rendiconti del Circolo Matematico di Palermo 7*, 111-113.

LAWLER, E.L. (1976), *Combinatorial Optimization: Networks and Matroids*, (Holt, Rinehart and Winston, New York).

LAWLER, E.L., J.K. LENSTRA AND A.H.G. RINNOOY KAN (1982), Recent Developments in Deterministic Sequencing and Scheduling: A Survey, in: M.A.H. DEMPSTER, J.K. LENSTRA AND A.H.G. RINNOOY KAN, EDS., *Deterministic and Stochastic Scheduling*, (Reidel, Dordrecht), pp. 35-73.

LAWLER, E.L., J.K. LENSTRA, A.H.G. RINNOOY KAN AND D.B. SHMOYS (1985), *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, (Wiley, Chichester).

LAWRENCE, S. (1984), Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques (supplement), Graduate School of Industrial Administration, Carnegie Mellon University.

LENSTRA, J.K. (1977), *Sequencing by Enumerative Methods*, Mathematical Centre Tract 69, Centre for Mathematics and Computer Science, Amsterdam.

LEONG, H.W. AND C.L. LIU (1985), Permutation Channel Routing, *Proc. IEEE Int. Conference on Computer Design*, Port Chester, October 1985, pp. 579-584.

LEONG, H.W., D.F. WONG AND C.L. LIU (1985), A Simulated-Annealing Channel Router, *Proc. IEEE Int. Conference on Computer-Aided Design*, Santa Clara, November 1985, pp. 226-229.

LIN, S. (1965), Computer Solutions of the Traveling Salesman Problem, *Bell System Tech. J. 44*, 2245-2269.

LIN, S. AND B.W. KERNIGHAN (1973), An Effective Heuristic Algorithm for the Traveling Salesman Problem, *Oper. Res. 21*, 498-516.

LINDLEY, D.V. (1978), *Bayesian Statistics, A Review*, (SIAM, Philadelphia).

LUNDY, M. AND A. MEES (1986), Convergence of an Annealing Algorithm, *Math. Prog. 34*, 111-124.

LUTTON, J.-L. AND E. BONOMI (1986), Simulated Annealing Algorithm for the Minimum Weighted Perfect Euclidean Matching Problem, *R.A.I.R.O. Recherche opérationnelle 20*, 177-197.

METROPOLIS, N., A. ROSENBLUTH, M. ROSENBLUTH, A. TELLER AND E. TELLER (1953), Equation of State Calculations by Fast Computing Machines, *J. of Chem. Physics 21*, 1087-1092.

MISES, R. VON (1964), *Mathematical Theory of Probability and Statistics*, (Academic Press, New York).

MITRA, D., ROMEO, F. AND A.L. SANGIOVANNI-VINCENTELLI (1985), Convergence and Finite-Time Behavior of Simulated Annealing, *Proc. 24th Conf. on Decision and Control*, Ft. Lauderdale December 1985, pp. 761-767.

MORGENSTERN, C.A. AND H.D. SHAPIRO (1986), Chromati Number Approximation Using Simulated Annealing, Departmen of Computer Science, The University of New Mexico, Albuquerque, *Technical Report No. CS86-1.*

NAHAR, S., S. SAHNI AND E. SHRAGOWITZ (1985), Experiments with Simulated Annealing, *Proc. 22nd Des. Automation Conf.*, Las Vegas, June 1985, pp. 748-752.

NAHAR, S., S. SAHNI AND E. SHRAGOWITZ (1986), Simulated Annealing and Combinatorial Optimization, *Proc. 23rd Des. Automation Conf.*, Las Vegas, June 1986, pp. 293-299.

OTTEN, R.H.J.M. AND L.P.P.P. VAN GINNEKEN (1984), Floorplan Design using Simulated Annealing, *Proc. IEEE Int. Conference on Computer-Aided Design*, Santa Clara, November 1984, pp. 96-98.

PAPADIMITRIOU, C.H. AND K. STEIGLITZ (1982), *Combinatorial Optimization: Algorithms and Complexity*, (Prentice-Hall, New York).

RIORDAN, J. (1978), *An Introduction to Combinatorial Analysis*, (Princeton University Press, Princeton).

ROMEO, F. AND A.L. SANGIOVANNI-VINCENTELLI (1985), Probabilistic Hill Climbing Algorithms: Properties and Applications, *Proc. 1985 Chapel Hill Conference on VLSI*, May 1985, pp. 393-417.

SASAKI, G.H. AND B. HAJEK (1986), The Time Complexity of Maximum Matching by Simulated Annealing, to appear in: *Journal of the ACM.*

SECHEN, C. AND A.L. SANGIOVANNI-VINCENTELLI (1985), The TimberWolf Placement and Routing Package, *IEEE J. Solid State Circuits SC-20*, 510-522.

SENETA, E. (1981), *Non-negative Matrices and Markov Chains*, (Springer Verlag, New York, 2nd ed.).

SERFLING, R.J. (1980), *Approximation Theorems of Mathematical Statistics*, (Wiley, New York).

SILVER, E.A. (1965), Bayesian Determination of the Reorder Point of a Slow Moving Item, *Operations Research 13*, 989-997.

SKISCIM, C.C. AND B.L. GOLDEN (1983), Optimization by Simulated Annealing: A Preliminary Computational Study for the TSP, presented at the *N.I.H.E. Summer School on Combinatorial Optimization*, Dublin.

SLATER, L.J. (1966), *Generalized Hypergeometric Functions*, (Cambridge University Press, Cambridge).

WAGNER, H.M. (1975), *Principles of Operations Research*, (Prentice-Hall, Englewood Cliffs).

WHITE, S.R. (1984), Concepts of Scale in Simulated Annealing, *Proc. IEEE Int. Conference on Computer Design*, Port Chester, November 1984, pp. 646-651.

WILKS, S.S. (1962), *Mathematical Statistics*, (Wiley, New York).

WILLE, L.T. (1987), The Football Pool Problem for 6 Matches: A New Upper Bound Obtained by Simulated Annealing, *J. Comb. Theory A45*, 171-177.

Author Index

Subject Index