SIMULATED ANNEALING: THEORY AND APPLICATIONS TO LAYOUT PROBLEMS

by

Fabio I. Romeo

Memorandum No. UCB/ERL M89/29

13 March 1989

SIMULATED ANNEALING: THEORY AND APPLICATIONS TO LAYOUT PROBLEMS

by

Fabio I. Romeo

Memorandum No. UCB/ERL M89/29

13 March 1989

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Simulated Annealing: Theory and Applications to Layout Problems

Fabio I. Romeo

Ph.D.

Department of Electrical Engineering
and Computer Science

Abstract

Simulated Annealing (SA) is a probabilistic algorithm to solve combinatorial optimization problems. SA generates new solutions randomly and evaluates the relative cost. If the cost decreases, the solution is always accepted. If the cost increases, the solution is accepted according to a probability distribution whose shape is controlled by a parameter. The number of solutions generated at each value of the temperature, the rule to update the temperature, and the stopping criteria comprise the annealing schedule. The annealing schedule determines the quality of the solution and the computer time necessary to obtain it. A mathematical analysis of SA is essential to understand the features which make it to work well. A model is also necessary to study the effects of different annealing schedules on the quality of the solution and on the required amount of computer time. A model based on Markov chains, has been used to study the behavior of SA . Using this model, it has been proved, under rather general assumptions, that SA produces asymptotically the optimum solution of the combinatorial optimization problems with probability one. Unfortunately, the results require that an infinite number of iterations must be performed for the algorithm to converge to the global optimum. The finite-time behavior of the algorithm has also been investigated. A bound on the departure of the state probability vector from the optimum probability vector after a finite number of iterations has been found. The results are interesting since they guarantee that the algorithm, if given enough time, will provide a solution which is arbitrarily close to the global optimum. However their utility in practical implementations of the algorithm is limited and heuristics which approximate the asymptotic behavior have to be implemented. One such heuristic is presented in this thesis. It consists

of a new problem-independent, adaptive annealing schedule. The parameters of the schedule are determined automatically from measures of statistical quantities collected while the algorithm is being executed. The results obtained applying the new schedule to solve placement problems, show that the CPU time can be significantly reduced with respect to implementations of the algorithm which feature naive annealing schedules.

Prof. Alberto Sangiovanni-Vincentelli
Thesis Committee Chairman

Acknowledgments

I would like to thank my research adviser, Prof. Alberto Sangiovanni Vincentelli. He convinced me to return to Berkeley as a Graduate student, he introduced me to the exciting field of Computer Aided Design and he provided, through the development of my research, an inexhaustible source of ideas, suggestions and advice. He, his wife Annamaria, and his son, Andrea, helped my family and me a lot in the past years and I am honored to consider them among my dearest friends.

I am indebted to the other members of my thesis committee Professors Robert Brayton and Stephen Cohen for their valuable comments and suggestions and to Professor Don Pederson for being the chairman of my Qualifying Examination Committee.

Dr. Donald Huang and I developed together the prototype version of the adaptive annealing schedule described in this thesis during his sojourn at Berkeley as an Industrial Visitor. Working with Dr. Huang was for me very productive and pleasant experience. Later, I had several opportunities to meet with him at conferences and during his periodical visits here at Berkeley and I was more and more impressed with his capability to combine scientific rigor with engineering pragmatism. His untimely death has deprived his family and the CAD community of a very fine and wise man and scientist.

I wish to thank Prof. Carl Sechen for collaboration in obtaining some of the earlier experimental results and for many lively arguments on the practical aspects of Simulated Annealing. Greg Sorkin was very helpful in discussing several of the topics that are reported in this thesis. His brilliant criticisms and counterexamples made me hate him at times but helped improve the accuracy of the results presented in this thesis. Professors Sastry, Varaiya, and Walrand have been very helpful discussing various theoretical aspects of stochastic processes.

I would like to thank Prof. Richard Newton for many stimulating discussions ranging from CAD to macro-economics and management of technology. I thank him also for the suggestions he gave me as my Major Field Adviser especially in my early days as Graduate student and I hope he will forgive me for the innocent

tricks I played on him once while we were skiing.

My colleagues in the CAD group made my staying at Berkeley an unforgettable experience and for this they are greatly acknowledged. Among them Jeff Burns, Ken Kundert, and Karti Mayaram were of great help when I started here at Berkeley, back in 1985. They gave me suggestions and hints on how to get a rusted student back into shape. I am glad to count them among my best friends.

I am also indebted to my financial sponsors, namely Semiconductor Research Corporation under contract SRC-82-11-008, the Italian National Research Council, and Honeywell Bull Italy.

Last, but by no means least, I wish to thank my wife Marialuisa for sharing with me the exciting and not so exciting moments we had in the last four years. My new born daughter Cecilia for giving us only few sleepless nights but innumerable moments of great happiness. My parents for their continuous support and understanding.

Contents

Table of Contents iii

List of Figures vi

List of Tables vii

1 Introduction 1
- 1.1 Classification of Combinatorial Optimization Problems 7
- 1.2 Algorithms for Combinatorial Optimization Problems 9
- 1.3 Questions About Simulated Annealing 12
- 1.4 Outline 14

2 Simulated Annealing Algorithm: Structure and Basic Definitions 18
- 2.1 Algorithm Structure 19
- 2.2 The Mathematical Model 25

3 Asymptotic Behavior: Homogeneous Theory 29
- 3.1 Homogeneous Markov Chains 31
  - 3.1.1 Classification of States 31
  - 3.1.2 Ergodic Theory 34
  - 3.1.3 Rate of Convergence 36
  - 3.1.4 Reversible Markov Chains 38
- 3.2 Asymptotic Properties of Simulated Annealing 40
  - 3.2.1 Ergodic Theory 41
  - 3.2.2 Rate of Convergence 47
  - 3.2.3 General Comments 51
- 3.3 Related Work 53

4 Asymptotic Behavior: Inhomogeneous Theory 56
- 4.1 Inhomogeneous Markov Chains 57
- 4.2 Asymptotic Properties of Simulated Annealing 62
  - 4.2.1 Bounds 62

4.2.2 Estimation of the Coefficient of Ergodicity 63
4.2.3 Weak Ergodicity 64
4.2.4 Properties of the Stationary Probability Distribution 66
4.2.5 Strong Ergodicity 69
4.3 Related Work 70
4.3.1 Sufficient Conditions 70
4.3.2 Necessary and Sufficient Conditions 72
4.4 Concluding Remarks 75

5 Finite Time Behavior **77**
5.1 Components of Finite-Time Behavior 77
5.1.1 Bound for the First Term 78
5.1.2 Bound for the Second Term 80
5.1.3 Bound for the Third Term 82
5.2 Final Results 83
5.3 Concluding Remarks 84

6 The Acceptance and Generate Functions **88**
6.1 Acceptance Function 90
6.1.1 The Optimal Acceptance Function 90
6.1.2 Other Acceptance Functions 95
6.2 Generate Function 96
6.2.1 Range Limiting 96
6.2.2 Adaptive Generate Functions 98
6.3 Errors in the Cost Evaluation 100
6.4 Stationary Probability Distribution 103

7 The Annealing Schedule **110**
7.1 Adaptive Schedules 113
7.1.1 Preliminary Assumptions 118
7.1.2 Initial Temperature 127
7.1.3 The Decrement of T 128
7.1.4 The Equilibrium Condition 129
7.1.5 The Stopping Criterion 134
7.2 Related Work 134
7.2.1 Initial Temperature 134
7.2.2 The Decrement of T 135
7.2.3 The Equilibrium Condition 139
7.2.4 The Stopping Criterion 139

8 Applications of Adaptive Annealing Strategies 141
8.1 The Adaptive Schedule 142
8.1.1 Test Case Characteristics 142
8.1.2 Improvements Over the Simplified Schedule 144
8.1.3 Sensitivity Analysis 151
8.2 The Simplified Schedule 154
8.2.1 Traveling-salesman Problem 155
8.2.2 Placement of Standard Cells 157

9 Conclusions 160

Bibliography 168

List of Figures

1.1 Performance Index. On the x axis it is represented the solution space Ω. On the y axis is reported the value of the performance index. . . 6
1.2 Example of a problem with local minima. 10

2.1 Simulated Annealing Algorithm. 19
2.2 Schematic representation of the solution set Ω with the move set $\mathcal{M}$ 21
2.3 Optimal placement of 24 modules on a 5-by-5 grid. Modules are represented by boxes, connections by solid lines, and grid locations by +. 22
2.4 Acceptance Function Structure 23
2.5 Transition graph. The dashed edges represent transitions dependent both on T and Δc_{ij}24The solid edges represent independent transitions.

3.1 Acceptance function given by (2.3). 47
3.2 Acceptance function given by (3.30). 48

7.1 Example of a non Markov process $Y_n = c(X_n)$ with X_n Markov. The transition probabilities in the figure refer to process X_n. 116
7.2 Gamma density as a function of T. c_* is assumed 0 125

8.1 Hierarchical example with 16 cells. 143
8.2 Logarithm of T/T_0 versus number of iterations. The straight line represents the updating rule of the geometric schedule with $\alpha = .9$. The piecewise line represents the updating rule of the adaptive schedule. 148
8.3 Cost versus logarithm of T. 149
8.4 Cost versus logarithm of T. No prediction adjustments. 150
8.5 Cost density for high value of T. 151
8.6 Cost density for low value of T. 152
8.7 Comparison of performance of the new annealing process with the one reported in literature. Solid line is from literature; circles are from the new annealing process. 156

List of Tables

7.1 Parameters of the linear predictor. 121
7.2 Correlation of the error with the data. 122

8.1 Summary of the characteristics of the test cases. 144
8.2 Parameters used for the geometric schedule. 144
8.3 Parameters used for both the adaptive and the simplified schedule. . 145
8.4 Adaptive Schedule. 145
8.5 Simplified Schedule. 145
8.6 Geometric Schedule. 146
8.7 Variation with respect to the adaptive schedule. 147
8.8 Sensitivity with respect to parameter K 152
8.9 Sensitivity with respect to parameter δ 153
8.10 Sensitivity with respect to parameter λ 153
8.11 Parameter setting for the traveling salesman example. N number of cities. 155
8.12 TimberWolf optimal number of attempts per cell. 158
8.13 Comparisons with TimberWolf Annealing Process. 158
8.14 Comparisons between the two schedules with the same amount of CPU time. 159

Chapter 1

Introduction

Optimization problems are very common in almost every field of science, economy, engineering etc.. Typically an optimization problem arises every time it is necessary to select, among a set of possible solutions, the best one according to a specified cost or reward function. A couple of examples will clarify the concept a little further.

Consider a manufacturing plant with several multi-purpose machines. The plant is capable of producing several products. Each one of the products requires a different sequence of operations that can be performed by the plant's machines provided that some re-tooling operations are performed on the machines. Of course the machines are flexible but, for each given set-up of their tools, they can operate only on a limited number of different products. On the other end, the product can be manipulated by only one machine at a time.

The optimization problem consists in scheduling the production such that the plant produces the "best" mix of products in a given time. In this case the possible solutions are all the feasible mixes of products given the type and the number of machines. The value of each one of the solutions, its reward in this case, is simply the monetary value of the corresponding product mix.

Another example: Consider the problem of finding the optimal layout for an integrated circuit realized with the macro-cell design style. The layout of integrated circuits is completely determined once a position is found for each of the

macro-cells and a path is selected for each one of the connections. In practice a number of constraints is imposed on the layout. For example the macro-cells cannot have any overlap, the propagation delay along some critical connections has to be within specified bounds, etc..

The bulk of the problem is to select among the solutions which satisfy the constraints, the *feasible solutions*, the best one according to some *performance index*. A suitable performance index for integrated circuits is proportional to two factors: Circuit area and total length of interconnections. The circuit area is inversely related to the yield of the fabrication process and therefore the smaller the area, the higher the yield, the lower the economic cost of the integrated circuit itself [1,2]. The length of interconnections is inversely related to the speed of operation of the circuit. In practice short connections and hence fast propagation times reduce the need for fast and expensive circuitry therefore reducing the cost of the integrated circuit.

The description of the problems given above is meaningful but it is too vague to be used by an algorithm to compute its solution. It is necessary to describe more precisely what it is meant by constraints and cost function. We will use one of the examples introduced above to see how a description in english is translated into a mathematical model.

Recall the layout problem and concentrate on the placement phase. Assume that the macro-cells can only be placed, for the sake of simplicity, on a one-dimensional grid. Suppose that any of the acceptable placements has to comply with the following constraints:

- Each module can occupy at most one position on the grid;
- Each position on the grid can contain one module at most.

Furthermore, assume that the i-th macro-cell is connected to the j-th one by c_{ij} connections. The length of each connection is proportional to the absolute value of the difference between the positions on the grid occupied by the two modules. The performance index used to rank the solutions is given by the sum of the

length of all the connections. The contribution of the area to the performance index does not appear explicitly because of the simplifying assumptions that module can only be placed on a predefined grid.

The mathematical model of the problem is as follows: Let ω_{ij} be a binary variable such that

$$\omega_{ij} = \begin{cases} 1 & \text{if the } i\text{-th macro-cell occupies the } j\text{-th position on the grid,} \\ 0 & \text{otherwise.} \end{cases}$$

With the above definition of ω_{ij}, the constraints described earlier become

i) Each of the grid locations may be occupied by at most one module, i.e. for all j

$$\sum_i \omega_{ij} \leq 1 \; ; \tag{1.1}$$

ii) Each of the module can be in only one grid location, i.e. for all i

$$\sum_j \omega_{ij} = 1 \; ; \tag{1.2}$$

iii) For all the connections

$$\sum_{k,l} \omega_{ik}\, \omega_{jl}\, |k-l| \leq \overline{l_{ij}} \; ; \tag{1.3}$$

The performance index is given by

$$\sum_{i,j,k,l} c_{ij}\, \omega_{ik}\, \omega_{jl}\, |k-l| \; . \tag{1.4}$$

A feasible solution is represented by a choice of variables ω_{ij} which satisfies the constraints (1.1), (1.2) and (1.3). The best solution is the choice of ω_{ij} which minimize (1.4). Equations (1.1)-(1.4) represent an arbitrary one-dimensional placement problem. The number of slots on the grid, the number of macro-cells, and the number of connections are the *parameters* of the problem. Once the parameters are assigned values, we have an *instance* of the problem.

Of course not any arbitrary combination of the parameters leads to a legal instance of the problem. For example, given the above defined constraints on the

solution, the number of available positions on the grid has to be at least as large as the number of cells to be placed. Apart from this trivial limitation, any reasonable combination of parameters gives rise to a legal instance of the problem.

To find the best solution to the problem, one can think to explore the feasible solution set, check the value of the performance index for each one of the solutions encountered and retain the best one. The procedure followed to explore the solution set is called *algorithm*. An algorithm is said to *solve* a problem $\mathcal{P}$ if the algorithm can be applied to any instance of $\mathcal{P}$ and is always guaranteed to produce, in finite time, a solution.

Summarizing, any optimization problems can be represented by the following generic mathematical model:

$$\min_{\omega \in \Omega} c(\omega)$$

subject to

$$H_i(\omega) \leq 0 \ ,$$

$i = 1, 2, \ldots$. Ω and ω denote the solution set and a generic solution respectively, $c(.)$ the performance index, and $H_i(.)$ the constraints determining the feasibility region.

Optimization problems can be classified in several ways. One such way, perhaps the most natural, classifies them according to the characteristics of the solution space: i.e. continuous or discrete. In continuous problems, the solution space is a sub-set of the euclidean space, namely $\Omega \subseteq \mathbb{R}^n$. In discrete problems, the solution space Ω is a collection, usually finite, of isolated points. Discrete problems with finite solution space are usually termed *combinatorial optimization problems*. In general, in combinatorial optimization problems the distinction between solution set and feasible solution set is absent in the sense that constraints are taken into account by the definition of Ω. Consequently, an instance of a combinatorial optimization problem is completely defined by the pair (Ω, c).

The structure of Ω together with the characteristics of c and H determine the type of algorithm necessary to solve the problem.

In general, the more is known about the problem, the more efficient is the algorithm that can be used to solve it. For example if both c and H are known

together with their derivatives as analytic functions of ω and $\Omega \subset \mathbb{R}^n$, then efficient algorithms based on Kuhn-Tucker conditions can be used [3,4,5]. Moreover, if, in addition, both c and H are linear functions of ω, the "famous and efficient" simplex algorithm can be used [6]. The power of these methods derives from the fact that the knowledge of c and H as functions of ω gives the algorithm a global information about the solution space at once and hence enables it to find the optimal solution exactly.

If c and/or H are known only locally and not as a function of the solution or they are non differentiable or the derivatives can only be computed numerically, the situation becomes more complex and one has to resort to more sophisticated techniques [7]. Algorithms to solve these type of problems have only a local understanding of the solution space and as such, unless the cost function and the feasibility region are both convex, they find a solution which is only locally optimal. The following example will make the the difference between local and global more clear.

Example 1.0.1 Suppose the optimization problem to be solved has no constraints and that the performance index is the one reported in Figure 1.1. Consider two cases. In the first one, the algorithm "knows" the analytic expression of the performance index which is given by

$$c(\omega) = 5\omega^2 + 10\sin^3(5\omega) \ .$$

Furthermore the derivative of $c(\omega)$ with respect to ω

$$\frac{dc(\omega)}{dx} = 10\omega + 150\sin^2(5\omega)\cos(5\omega) \ ,$$

is also known analytically. The value of ω which maximizes $c(\omega)$ is determined exactly by finding, among the ω_* that satisfy

$$\frac{dc(\omega)}{dx} = 0 \ , \tag{1.5}$$

the one for which $c(\omega_*)$ is minimum.

Consider now the case in which the algorithm can only evaluate the cost function and its derivative once it is given a value of ω, and suppose it uses a

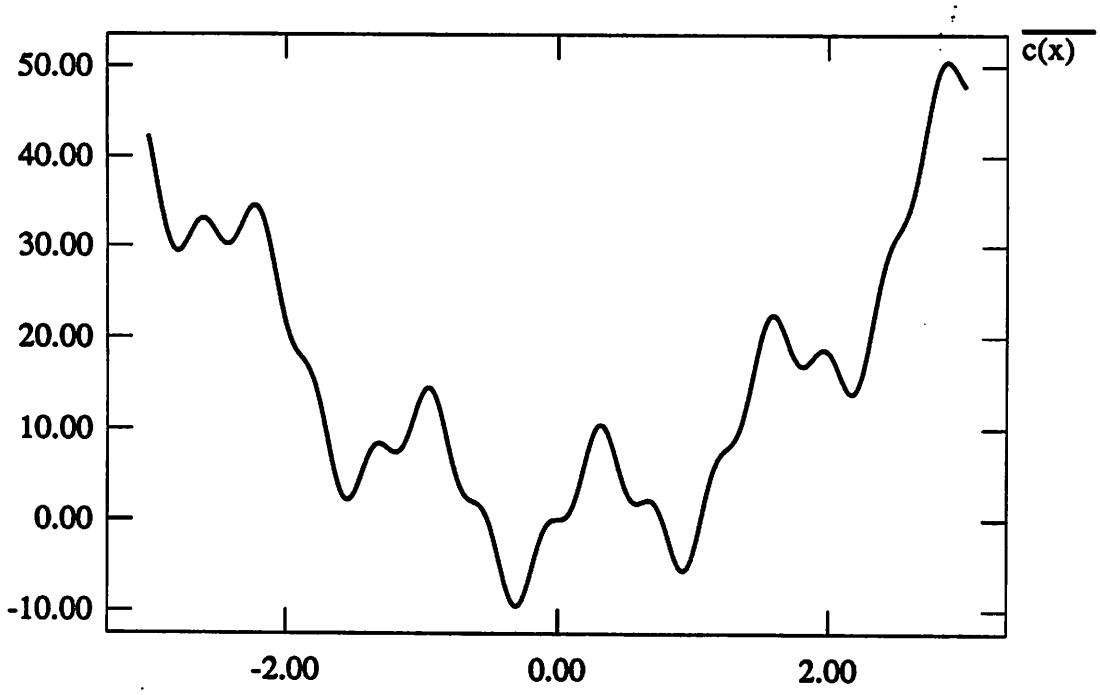

Figure 1.1: Performance Index. On the x axis it is represented the solution space Ω. On the y axis is reported the value of the performance index.

steepest descent method to find the optimum. It is clear from Figure 1.1 that the algorithm will terminate with a solution which is a minimum for the function but it will find the global minimum only if the starting solution is "close enough" to it.

In conclusion, the first algorithm always finds the global optimum because it has a global information about the performance index. The second one has only local information on the performance index and this translates into the fact that the minimum it finds depends on the value of ω_0 from which it started.

As a final remark, notice that the occasions in which both the performance index and its derivatives are known analytically are rather unusual. Furthermore, even if they are known, it is very unlikely that the solution to equation (1.5) is known in closed form. □

In combinatorial optimization problems, the function c is defined only on isolated points and, as a consequence of the characteristics of Ω, any algorithm can only make use of the values attained by the cost function on the elements of Ω.

It is clear from what precedes that, combinatorial optimization for which the only available information comes from evaluations of the cost function are in general difficult to be solved "exactly" [1]. Aim of the next section is to make the notion of "problem difficult to solve" more precise.

1.1 Classification of Combinatorial Optimization Problems

Combinatorial optimization problems are classified according to the computational complexity of the "best" known algorithm used to determine an optimal solution. A first classification partitions the totality of combinatorial optimization problems into *decidable* and *undecidable*. A combinatorial optimization problem is said to be undecidable, if it is impossible to find an algorithm that can solve it.

Example 1.1.1 Here is an example of an undecidable problem. Given an arbitrary computer program and an arbitrary input to it, is it possible to specify any algorithm that can decide whether or not the program will terminate when applied to that input? Other examples of undecidable problems can be found in [8]. □

Among decidable problems there are two classes of problems that are of particular interest: The first class consists of the problems for which there exist a deterministic algorithm whose worst case complexity is polynomial in the size of the input. The class is called P.

The second class contains the problems for which it is possible to verify in polynomial time the correctness of a proposed solution. For the problems in the second class, the algorithm comprises a guessing stage in which a solution is proposed and a checking stage in which it is verified. Because of the existence of the guessing stage, algorithms such as the one described above are called non-deterministic. Problems for which there exist a non-deterministic algorithm whose worst case complexity in the size of the input is polynomial belong to the NP class. Problems in the NP class are often referred to as *intractable*.

[1] An exact solution is a solution which corresponds to the global optimum of c

While it is clear that $P \subseteq NP$, no formal proof of $P \subset NP$ has been found. However the conjecture $P \subset NP$ is commonly accepted.

If it is possible to find a problem $\mathcal{P}$ which is not P, if it is assumed that $P \subset NP$, and if all the other problems in NP can be polynomially reduced to $\mathcal{P}$, then to solve $\mathcal{P}$ in polynomial time is equivalent to solve in polynomial time any problem which is in NP. In other words, a problem with the characteristics of problem $\mathcal{P}$ is the "hardest" problem in NP. In 1971, Cook proved that the "satisfiability" problem is in fact a problem with the characteristics of $\mathcal{P}$ [9]. In 1972, Karp [10] proved that there are many other combinatorial optimization problems that share with the satisfiability problem the burden of being the hardest ones in NP. The class of these problem has been named the $NP-$ *complete* class.

Summarizing, $NP-$ *complete* is defined as the subclass of NP with the property that every element in the NP class can be reduced to any one of the members of $NP-$ *complete* by means of a polynomial time transformation.

The existence of the $NP-$ *complete* class is of remarkable importance since the discovery of a deterministic algorithm with polynomial worst-case complexity that solves one of the problem in the class will solve in polynomial time all the other problems in the class and hence all the other problems in NP.

The theory of $NP-$ *completeness* has been developed for a subset of combinatorial optimization problems, the decision problems, i.e. problems in which the answer is simply yes or no. For example recalling the placement problem introduced above, its decision version would be: Given a number of macro-cells to be placed on a grid and a bound $\hat{C}$, is there a placement with cost less or equal to $\hat{C}$?

It is clear that in general the decision version of the problem is simpler since it does not require to compute the actual solution but only to answer an "existence" question. For this reason, a further class has been introduced to complete the classification of decidable problems: All the problems whose decision version belongs to the $NP-$ *complete* class are said to be $NP-$ *hard*.

Unfortunately the great majority of problems that are of interest in practical applications belongs to the $NP-$ *complete* class (See Garey and Johnson for an extensive survey of the theory of $NP-$ *completeness* [8]).

1.2 Algorithms for Combinatorial Optimization Problems

Because of the *NP* nature of many of the interesting problems, the computation time necessary to solve them exactly may grow exponentially fast and the size of problems of practical interest is usually large enough to require prohibitive computing times. For this reason, to use exact methods such as branch and bound or integer programming is not feasible in practice for several problems.

The solution is to replace exact algorithms with approximate ones or with heuristics. These methods are not guaranteed to produce the optimal solution but require an execution time which is faster, usually polynomial in the size of the problem. Heuristics, in essence, trade speed for quality and, in fact, the difference in cost between the obtained solution and the best one establishes the quality of the heuristic method.

Heuristics are of two different types: Methods that compute the solution constructively starting from raw data and methods that iteratively improve an existing solution [2].

Constructive methods are tailored to the characteristics of the problem to be solved, and therefore are difficult to export to different applications.

Iterative improvement algorithms are more general. They all exhibit the following common structure: Starting from an initial solution, $\omega_0 \in \Omega$ with Ω the set of solutions, a sequence of solutions is generated until a satisfactory one is found. The rules according to which a new solution is generated and the algorithm terminates, specify the algorithm.

Iterative algorithms have a number of drawbacks:

- The search terminates with a local minimum, i.e. with a configuration ω_{j^*} such that if we denote by $c(\omega_j)$ the cost of ω_j and by Ω_j the set of configurations that can be generated from ω_j by the algorithm in one step, $c(\omega_{j^*}) \leq c(\omega_j)$,

[2]The distinction is conceptual more that factual since it is uncommon to find a practical implementation of an heuristic method which features one only of the two strategies

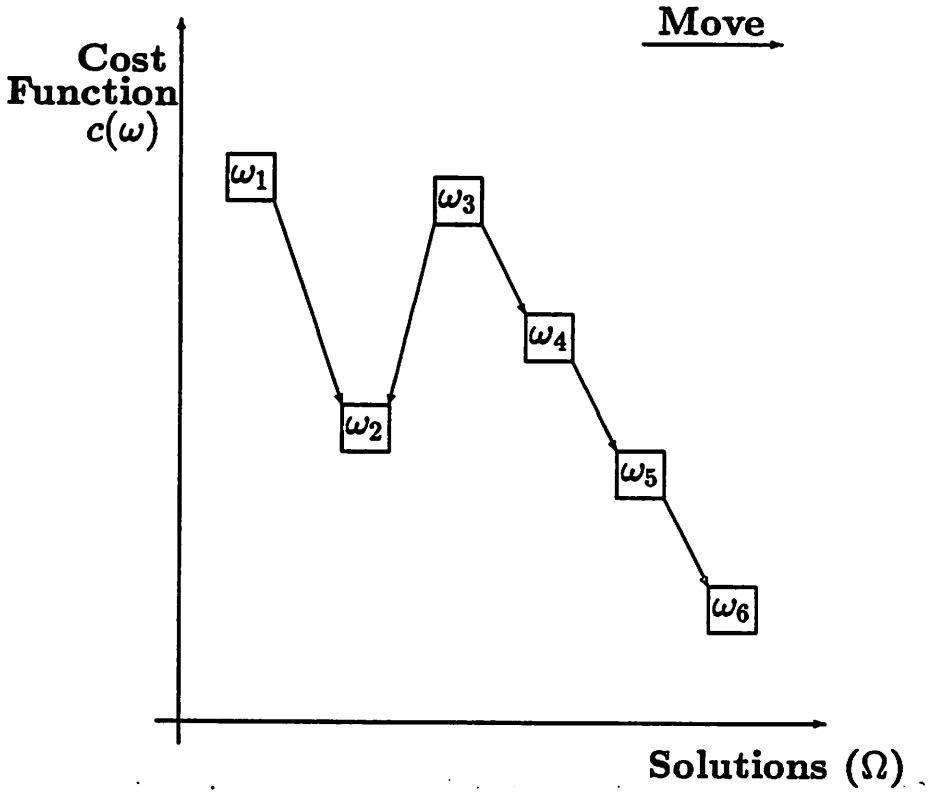

Figure 1.2: Example of a problem with local minima.

for all $\omega_j \in \Omega_{j^*}$.

- The final solution is dependent on the starting solution and on the rule used to generate the next solution.

Often heuristic algorithms have only a "local" view of the problem and the strategy they implement is to select moves which reduce "maximally" the cost. These algorithms implement the so called *greedy* strategy. An example of such behavior is reported in Fig. 1.2 where the cost of each solution is symbolically represented by the ordinate of the corresponding box and an arrow is present between two solution if the algorithm can generate one from the other. Assume the algorithm starts from solution ω_3 and selects the best solution from $\Omega_3 = \{ \omega_2, \omega_4 \}$ namely ω_2. Solution ω_2 is a local minimum and the algorithm terminates giving ω_2

as final solution, while the best solution is clearly ω_6.

Several strategies can be used to improve the behavior of iterative improvement heuristics. For example, repeated executions of the algorithm starting from different initial conditions would reduce the possibility of being trapped in a local minimum. The introduction of more general "moves" to explore the state space modifies the topology of the solution space such that the number of local minima is, in general, reduced. Notice however that more general moves require a more detailed knowledge of the particular problem and, as a consequence, reduce the portability of the algorithm to other applications.

A completely different approach to avoid the above mentioned behavior, is to resort to randomizing algorithms, i.e. algorithms which generate the next configuration randomly (See e.g. [11]). The configuration is recorded as a new temporary solution if its cost is lower than the present temporary solution. The algorithm terminates after a certain number of moves has been carried out. Since the probability of stopping at an optimum is proportional to the ratio between the number of optimal configurations and the number of total configurations, randomizing algorithms perform well if the number of optimal solutions is fairly high. Note that randomizing algorithms can *climb hills*, i.e., moves that generate solutions with cost higher than the present one are accepted.

Simulated Annealing (SA) as proposed independently by Kirkpatrick et al. [12] and Černy [13], allows *hill climbing* moves, but moves are accepted according to a criterion which takes the cost into consideration and not blindly as randomizing algorithms. The mechanism which controls the acceptance of new solutions is based on the observation that combinatorial optimization problems with a large solution space exhibit properties similar to mechanical systems with many degrees of freedom.

In particular, bringing a mechanical system into a low energy state such as growing a crystal from a melted substance, has been considered in [12] similar to the process of finding an optimum solution of a combinatorial optimization problem. Annealing is the well-known process to grow crystals and the Metropolis Monte Carlo method [14,15] has been proposed to simulate the annealing process

in physical sciences. Similar procedures has been proposed as an effective method for finding global minima of optimization problems [16,17,18].

SA when applied to combinatorial optimization generates moves randomly and checks whether the cost of the new configuration satisfies an acceptance criterion based on a parameter, T, sometimes called "temperature" in analogy with the physical case. If the cost decreases, the move is accepted. If the cost increases, then a random number r, uniformly distributed in the interval $[0,1]$, is generated and compared with the value of a function

$$0 < f_T(\Delta c_{ij}) \leq 1 ,$$

where Δc_{ij} is the variation in cost obtained by moving from solution ω_i to ω_j. T, the temperature, is the controlling parameter. If the random number is larger than $f_T(\Delta c_{ij})$, the move is accepted, otherwise the move is discarded. f is directly proportional to T and inversely proportional to Δc_{ij}. Hence, fixed Δc_{ij}, the higher the value of T, the more likely an hill-climbing move is accepted. Conversely, fixed T, the higher Δc_{ij}, the less likely it is that the move will be accepted.

For each given value of T, a certain number of moves are generated and checked before T is decreased. The initial value of T, the number of moves generated at each fixed value of T and the rate of decrease of T and the criterion used to decide when to terminate the algorithm comprise the *annealing schedule.*

1.3 Questions About Simulated Annealing

Since its introduction in 1983, SA has been applied to solve all kinds of optimization problems arising in engineering (See e.g. [19,20,21,22,23,24,25,26,27, 28,29]), computer sciences (See e.g. [30,31,32,33,34,35,36,37,38]), and image recognition (See e.g. [39,40,41,42,43,44]) besides physics (See e.g. [15,45]) where the SA ancestor, the Metropolis algorithm has been used quite extensively since the mid fifties [14].

Together with a great deal of popularity, SA has also spurred discussions in the scientific community. Among SA's advantages are its capability to deal

with a large number of problems quite naturally and effectively. Being an iterative improvement algorithm, the dependence of SA on the problem is limited to the selection of the move set. From this follows that it is relatively easy to build a SA based algorithm to solve a new problem.

The flip side of the coin is the limited efficiency of the algorithm. If *naive* implementations of SA are compared with heuristics especially tailored on the particular problem, given a required level of quality for the solution, the time required by SA exceeds by a significant amount the time required by *ad hoc* heuristics [30]. This limitation is severe when the problem to be solved is relatively simple or is a problem for which a polynomial algorithm is known. For example to solve graph partitioning the time required by the most efficient algorithm [46,47] is one order of magnitude smaller than the best SA implementations. Nevertheless, SA regains a lot of its attractiveness when it is applied to solve "real world" problems. Experimental data (See e.g. [21,27,29,48]) show that SA produces very good results when compared to other techniques for the solution of complex combinatorial optimization problems such as those arising from the layout of integrated circuits. For these problems, even if the required amount of computer time is large [3], no other known algorithm is capable of producing results of similar quality.

The structure of SA raises a number of questions about its features and the characteristics of the solution produced. For example:

- Does the algorithm ever converge?
- Does the initial solution have any influence on the final one?
- What is the probability that the final solution produced by SA is the optimal one?
- What is the influence of the annealing schedule on the quality of the final solution?

It is clear that to find an answer to the preceding questions is important in view of the application of the algorithm to solve practical problems.

[3] A 1,500 standard cell placement problem can take as much as 24 hours of a VAX 11/780 [19]

If the algorithm converges, it becomes important to establish the behavior of the expected value of the cost of the final solution as a function of the amount of time spent by the algorithm to find the solution. If the expected value decreases as the amount of time increases, we know that the more we can afford to spend in terms of running time, the better will be the quality of the solution found.

Independence guarantees that the algorithm does not need an initial solution which is close enough to the optimal solution and this rules out the often difficult problem of determining a good starting point for an iterative algorithm.

Convergence and independence of the final solution from the initial one guarantee that the algorithm is "robust".

Finally, given the fact that the longer the time spent, the better the quality of the solution, the fourth question is more properly restated as follows: Given a fixed amount of time, is there an optimal way to select the annealing schedule so that the expected value of the final cost is minimized? To provide an adequate answer to this question is very important for practical implementations of SA because of the large amount of time required to run the algorithm.

The goal of this thesis is twofold: The first one is to analyze the asymptotic behavior (large number of iterations) of SA and establish the conditions on the generate, accept, and update functions for the algorithm to converge to the optimal solution with probability one. The second one, is to derive, from the asymptotic results, viable strategies to design SA algorithms for practical implementations. In particular an annealing schedule, whose parameters are determined adaptively from the data collected during the execution of the algorithm, is presented. The annealing schedule is problem independent and uses efficiently the allotted computer time. The next section contains a detailed outline of the thesis.

1.4 Outline

The first part of Chapter 2 contains a detailed description of the structure of SA and of its parameters. To properly answer the questions raised in the previous section, it is necessary to derive a mathematical model for the operation of SA. The

natural model for SA is a discrete inhomogeneous Markov chain [49]. The Markov chain is a stochastic process defined on a countable state space. A probability measure, synthetically represented by a matrix, is defined on the set of transitions between states. In SA the entries of the matrix represent the probability that a new solution is generated and accepted. The entries are either constant or dependent upon T and Δc. The details of the model are presented in the second part of Chapter 2.

Given the model, It is possible to prove convergence to the global optimum following two different approaches.

The first, the simpler one, assumes that the algorithm performs a number of iterations at each value of T which is large enough for the probability distribution to reach a stationary value. With this assumption, it is proved, under mild assumptions on the mechanism to generate and accept new solutions, that SA converges asymptotically to a solution which is independent of the initial condition. Moreover the final solution of SA asymptotically approaches the optimum solution of combinatorial optimization problems with probability one [50,51]. Chapter 3 is dedicated to present these results together with the necessary background on the theory of the homogeneous Markov chains. The details of the proofs presented in Chapter 3, underline the essential properties of the algorithm, so that it is possible to derive a class of "Probabilistic Hill-Climbing Algorithms" that share the same asymptotic properties of SA [50,51].

The theory developed in Chapter 3 proves convergence independently of the particular annealing schedule as long as the sequence $\{T_m\}$ approaches zero as m approaches infinity. The flip side of the result is that stationarity has to be achieved and this, in general, requires an infinite number of iterations at each value of T. A new question can then be raised: Is it necessary to achieve stationarity at each value of T to prove convergence to the global optimum? The answer is no. In fact, if a finite number of iterations is taken at each value of T, it can be proved that a sufficient condition for the convergence with probability one to the optimal

solution, requires that the sequence of temperatures T_m be

$$T_m = \frac{k}{\log(m + m_0)},$$

where m is the iteration counter and m_0 and k are two problem dependent constants. The details of the convergence proof are contained in Chapter 4 together with a short review of the results on inhomogeneous Markov chains which are needed [52,53].

Results similar to those presented in Chapters 3 and 4 were proved independently by a number of authors in the same time frame (See e.g. [39,54,55,56,57, 58]).

The results presented in Chapter 4 are stronger that those presented in Chapter 3 but are still asymptotic in the sense that they require the algorithm to perform an infinite number of iterations. As such their utility in practical implementations of the algorithm is limited.

In Chapter 5 the finite time behavior of the Markov chain is studied. The results obtained show that after m iterations have been performed, the distance of the state probability distribution $\mathbf{p}(m)$ from the optimal one [4] represented by $\mathbf{e}_*$ is given by

$$\|\mathbf{p}(m) - \mathbf{e}_*\| = O((r/m)^{\min(a,b)})$$

where r, a, and b are quantities characteristic of the problem to be solved. Unfortunately to estimate r, a, and b in real problems is not always feasible and this limits the applicability of the bound.

Designing a SA algorithm for a given combinatorial optimization problem amounts to selecting a generation mechanism, an acceptance strategy, and an annealing schedule. The characteristics of all these parameters influence the performances of the SA algorithm. It turns out that the efficiency of the acceptance strategy is independent of the particular instance of the combinatorial optimization problem to be solved. However, the performances of the algorithm are strongly related to generation mechanism and annealing schedule.

[4] SA converges with probability one to the global optimum of the combinatorial optimization problem if the probability distribution of the states converges toward a probability distribution which is greater than zero only for states which are global optima. Such a probability distribution is sometimes referred to as the optimal probability distribution.

The criteria to design an efficient generation mechanism and an efficient acceptance function are presented in Chapter 6.

The annealing schedule is, among the three parameters mentioned above, the one which affects the quality of the solution the most. Unfortunately the results from Chapters 3, 4, and 5 cannot be turned immediately into feasible annealing schedules. The designer of an SA algorithm for a practical application has to resort to heuristic criteria to obtain a good trade-off between speed and quality of the solution.

The annealing schedules that have been proposed, often rely on a number of parameters that are to be fixed before the execution of the algorithm. This is usually puzzling. First of all a good choice of the parameters is related to the particular instance of the problem. As a consequence, to determine a good parameter setting often requires the execution of a large number of time consuming experiments with the aggravation that the optimal parameter setting for one problem is of limited utility for any other problem. Second these parameters are properly interpreted only if the operation of the algorithm is properly understood. Hence the user, who has usually only limited knowledge of the behavior of SA, finds it difficult to determine a good parameter setting.

The "ideal" annealing schedule has to require only a limited number of parameters. These parameters must be common to a large class of problems and the annealing schedule has to be able to self tune its characteristics on the basis of the data that are collected during the operation of the algorithm. The details of the adaptive annealing schedule are presented in Chapter 7. The adaptive annealing schedule has been applied to SA with very good results as reported in Chapter 8 of this thesis.

The concluding remarks are collected in Chapter 9.

Chapter 2

Simulated Annealing Algorithm: Structure and Basic Definitions

Consider an instance of a combinatorial optimization problem (Ω, c). The function, $c : \Omega \mapsto \mathbb{R}^+$, assigns to each element of Ω its cost or reward. Let us consider from now on problems where the minimum over Ω of c is sought [1] .

Once the cost function c is given, we can define the set of the global minima for (Ω, c) as follows:

$$\Omega_* = \{ \omega_i : c(\omega_i) \leq c(\omega_j) \text{ for all } \omega_j \in \Omega \} . \tag{2.1}$$

For the sake of simplicity, we avoid to mention explicitly the dependence on ω anytime the context will not lead to any confusion. In practice $c(\omega_i)$ will be replaced by c_i and ω_i by i. Furthermore, for a reason that will be clearer at the end of this chapter, we will indicate a solution also as a *state*.

[1] The choice of minimum problems as opposed to maximum problems is a matter of taste and does not affect the generality of the conclusions. In fact it is immediate to see that the minimum of a function is coincident with the maximum of the same function with all the entries with the sign changed.

2.1 Algorithm Structure

The strategy implemented by SA consists of exploring the solution space starting from an arbitrary selected solution, or state, and generating a new one by perturbing it. Every time a new solution is generated, its cost is evaluated and the new solution is either accepted or rejected according to an acceptance rule.

```
Simulated Annealing ( j0, T0 ) {
    /* Given an initial state j0 and an initial value for the parameter T, T0 */
    T = T0;
    X = j0;
    while ( Outer loop criterion is not satisfied ) {
        while ( Inner loop criterion is not satisfied ) {
            j = generate ( X );
            if ( accept ( j, X, T ) ) {
                X = j;
            }
        }
        T = update (T);
    }
}
```

Figure 2.1: Simulated Annealing Algorithm.

The general structure of SA, as described in Figure 2.1, actually represents a class of algorithms. To obtain a particular instance of the class, it is necessary to

specify the values of five parameters:

1. **generate** function;
2. **accept** function;
3. **update** function;
4. **outer loop** criterion;
5. **inner loop** criterion.

From now on, **SA** will be used to indicate the whole class of algorithms reported in Figure 2.1. Any further assumptions on any of the parameters of the algorithm will be explicitly stated whenever necessary.

The following simple example will help to understand the nomenclature used to describe the algorithm.

Example 2.1.1 Recall the placement problem introduced in Chapter 1. Suppose three interconnected modules, $\{a, b, c\}$, have to be placed on a one-dimensional grid so that the total length of the interconnection is minimized. The state space Ω consists of 6 states, namely all the possible placements of the three modules, i.e. $\Omega = \{\omega_1, \omega_2, \cdots, \omega_6\}$ with

$$\begin{aligned} \omega_1 &= a, b, c \\ &\vdots \quad\quad \vdots \\ \omega_6 &= b, a, c\,. \end{aligned}$$

□

There are, in general, several ways in which the current state can be perturbed to generate the next one. The action taken to perturb a state is called *move*. The collection of all legal moves determines the move set $\mathcal{M}$. The set of moves $\mathcal{M}$ induces a metric on Ω. In fact once $\mathcal{M}$ is specified, it automatically determines for each state i its neighbors. More formally, let $\Omega_i \subset \Omega$, denote the set of all the states $j \in \Omega$ such that there exist a move $m \in \mathcal{M}$ that applied to i produces j. With the topology induced by $\mathcal{M}$ we can now define a characteristic sub-set of Ω as follows:

$$\Omega_m = \{\, i : c_i \leq c_j \text{ for all } j \in \Omega_i \,\}\,, \tag{2.2}$$

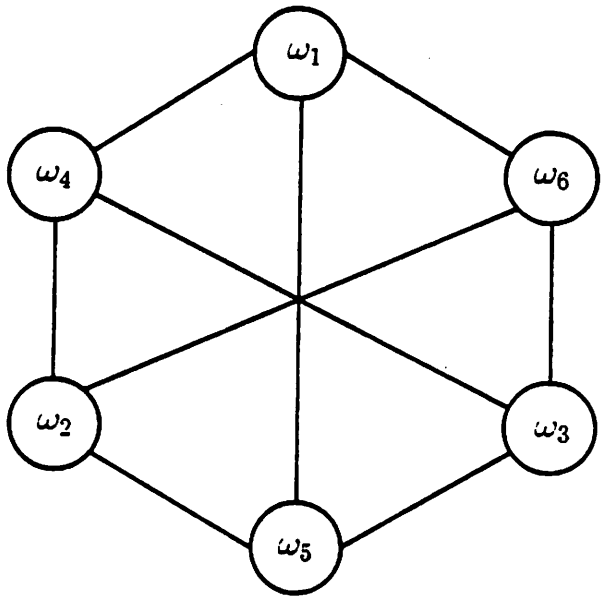

Figure 2.2: Schematic representation of the solution set Ω with the move set $\mathcal{M}$

Ω_m is the set of local minima . Notice how, unlike Ω_*, Ω_m depends on the selected move set.

The pair of sets Ω and $\mathcal{M}$ can be represented with an undirected graph $\mathcal{G}$ in which each solution corresponds to a vertex on the graph and there exist an edge between vertex i and j if $j \in \Omega_i$.

Example 2.1.1 [Continuation] Back to the example. Assume that in this case $\mathcal{M}$ consist of moves of one type only, i.e. pairwise interchange of any two elements of Ω. If Ω is equipped with the topology induced by $\mathcal{M}$ we obtain the graph $\mathcal{G}$ depicted in Figure 2.2. □

To select an appropriate set of moves $\mathcal{M}$ is crucial. $\mathcal{M}$ in fact has to be such that for each couple of states, say i and j, there must exist a finite sequence of moves that allows the algorithm to visit one starting from the other. This characteristic of $\mathcal{M}$ may seem trivial but in general it is not. The following example will illustrate the point.

Example 2.1.2 Consider a placement problem in which 24 modules have to be placed on a 5-by-5 grid. Imagine that the connections among modules are such that the optimal solution is the one represented in Figure 2.3. In the selected move set consist of pairwise interchanges of modules and the initial solution has any of the modules in the location at the center of the grid, it is clear that the optimal solution of Figure 2.3 will never be reached. However, if the original move set is

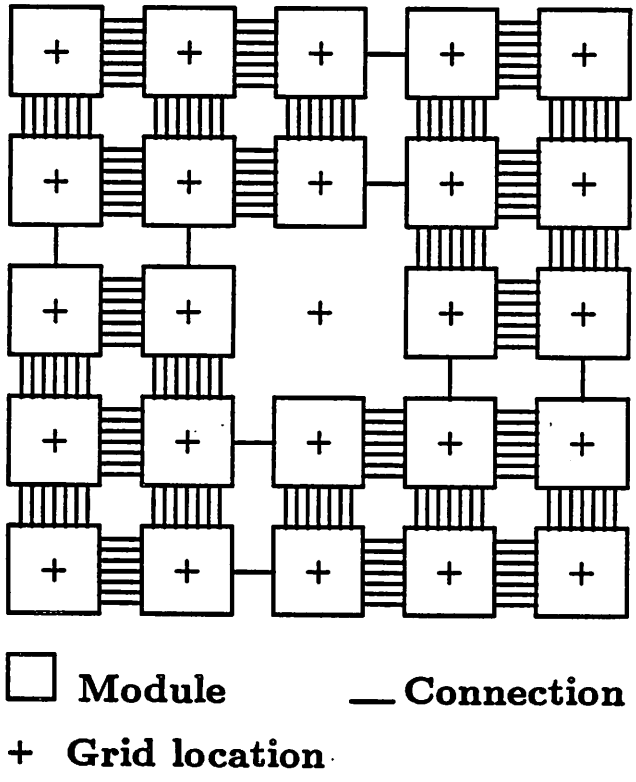

Figure 2.3: Optimal placement of 24 modules on a 5-by-5 grid. Modules are represented by boxes, connections by solid lines, and grid locations by +.

replaced with another one in which the contents of two locations on the grid is swapped, it is easy to see that all the space is reachable. The example is trivial but shows how, even in a simple case, it is possible to select a set $\mathcal{M}$ which makes $\mathcal{G}$ not strongly connected. □

The **generate** function selects, according to a given criterion, a state $i \in \Omega$

and a move $m \in \mathcal{M}$ to generate the next state $j \in \Omega_i$.

The **acceptance** function determines whether the new generated state j is accepted. Its structure is shown in Figure 2.4.

```
accept ( j, i, T ) {
    /* the function returns a 1 if the cost variation passes a test
    T is the control parameter */
    Δc_ij = c_j − c_i;
    y = f_T(Δc_ij);
    r = random (0,1);
    /* random is a function which returns a pseudo-random number
    uniformly distriduted on the interval [0,1] */
    if (r < y)
        return (1);
    else
        return (0);
}
```

Figure 2.4: Acceptance Function Structure

The acceptance strategy of Figure 2.4 is implemented by a family of functions indexed by a parameter T. For each value of $T \in (0, \infty)$ $f_T(c) : \mathbb{R}^+ \mapsto (0, 1]$. T controls the shape of $f_T(c)$.

The introduction of the generation and acceptance function modifies the undirected graph $\mathcal{G}$. In particular each original edge is replaced now by a pair of new edges. Each new edge has a weight appended to it. The weight is given by the value of the generation function times the value of the acceptance function for the perturbation represented by the edge. Self-loops are introduced to account for new states that have been generated but rejected.

Example 2.1.1 [Continuation] Assume that the connections between the modules

are such that for all $j, i = 1, 2, \cdots, 6$, if $i \leq j$ the corresponding costs satisfy

$$c_i \leq c_j \, ,$$

and assume that the acceptance function is given by

$$f_T(\Delta c_{ij}) = \min\Big[1, \exp\Big(-\frac{\Delta c_{ij}}{T}\Big)\Big] \, , \tag{2.3}$$

with

$$\Delta c_{ij} = c_j \, - \, c_i \, .$$

The graph of Figure 2.2 becomes the one shown in Figure 2.5.

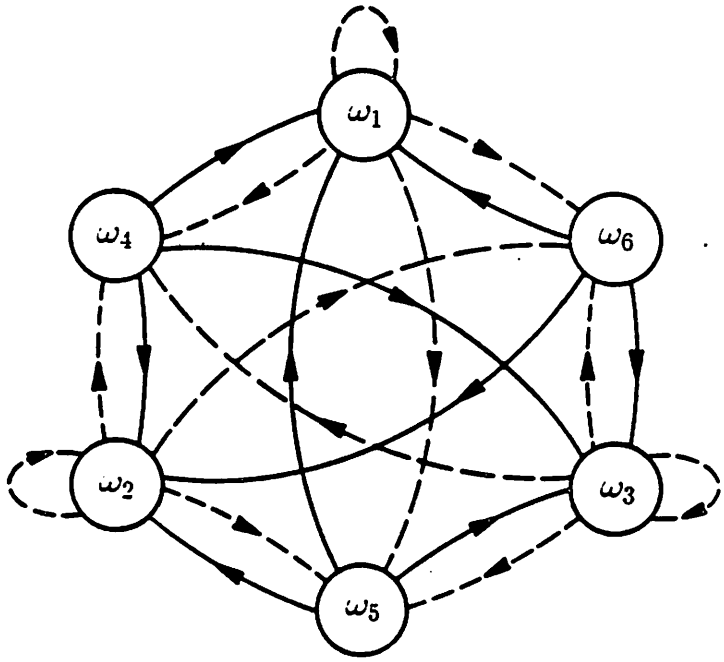

Figure 2.5: Transition graph. The dashed edges represent transitions dependent both on T and Δc_{ij}. The solid edges represent independent transitions.

There are now two types of edges: Edges whose weight depends upon T and the difference of cost Δc_{ij}, dashed in the figure. Edges whose weight are independent of T and Δc_{ij}, solid in the figure. Edges of the first type are of two

different type: Self-loops and edges related to "up-hill" moves. The weight of self-loops does not decrease as T decreases, to signify that as T is reduced the probability to generate a move and accept it does not increase. On the contrary the weight of "up-hill" moves decreases as T decreases.

Finally note that the presence of edges whose weights are independent of T and Δc_{ij} is a consequence of the choice of (2.3) as acceptance function. In general the weight of "up-hill" edges is non increasing as T decreases while for "down-hill" edges weights are non decreasing. □

The remaining three parameters, namely the **update function**, the **outer loop** and the **inner loop** criterion are usually referred to as the *annealing schedule.* A careful choice of these three parameters is important to achieve the best trade-off between the speed of the algorithm and the quality of the final solution.

In Kirkpatrick's instance of SA [12], new states are generated by selecting uniformly at random one of the moves. The acceptance function is given by (2.3). The control parameter T, is updated by means of the following geometric law

$$T_{m+1} = \alpha\, T_m\,, \tag{2.4}$$

with $0 < \alpha < 1$. The inner-loop criterion is satisfied when a minimum number of moves, proportional to the size of the problem (e.g. the number of modules to be placed in Example 2.1.1) is attempted at each fixed value of T. The outer loop criterion is verified when a stationary point in the cost function is reached.

2.2 The Mathematical Model

In this section a mathematical model that describes the behavior of the algorithm is introduced. The model will be used in the following chapters to discuss the convergence properties of the algorithm and to develop the adaptive annealing schedule. In the remaining part of the thesis it will be assumed that the generation function selects at random, according to a specified probability distribution, one of the admissible moves.

Given the random nature of both the generate and acceptance function, the state visited by SA after k moves is a random variable, X_k. Therefore each execution of SA is a realization of a stochastic process and is represented by the values taken on by the sequence of random variables $\{ X_n \}$ indexed by n (number of moves or discrete time).

The probability that the state visited by SA at the $(n+1)$-st iteration be j given that at the n-th iteration it was i, consists of two independent contributions: The probability of generating j from i at the n-th iteration, represented by $G_{ij}(T_n)$, and the probability that j is actually accepted as the new configuration $f_{T_n}(\Delta c_{ij})$. If we assume that the two contributions are independent, the expression for the transition probability is then given by:

$$\Pr\{ X_{n+1} = j \mid X_n = i \} = P_{ij}(T_n) = \begin{cases} G_{ij}(T_n)\, f_{T_n}(\Delta c_{ij}) & \text{for all } j \in \Omega_i\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{2.5}$$

$G_{ij}(T_n)$ is defined as follows

$$G_{ij}(T_n) = \begin{cases} \dfrac{\hat{G}_{ij}(T_n)}{G_i(T_n)} & \text{for all } j \in \Omega_i\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{2.6}$$

where the expression for $G_i(T_n)$

$$G_i(T_n) = \sum_{j \in \Omega_i} \hat{G}_{ij}(T_n)\,, \tag{2.7}$$

guarantees

$$\sum_{j \in \Omega_i} G_{ij}(T_n) = 1$$

which is necessary since $G_{ij}(T_n)$ has to be a probability measure on Ω_i.

Equations (2.5)-(2.7) define the transition probability only if $i \neq j$. However, since f is not, in general, identically equal to one, there is a finite probability that the algorithm will remain in configuration i. The following equation determines this probability:

$$P_{ii}(T_n) = 1 - \sum_{j \in \Omega_i} P_{ij}(T_n)\,. \tag{2.8}$$

The transition probabilities can be represented in a more compact form with a square matrix $\mathbf{P}(T_n)$ whose entries are defined by (2.5), (2.8). For every value of T_n, $\mathbf{P}(T_n)$ is a stochastic matrix according to the following

Definition 2.2.1 *A m-by-m matrix* $\mathbf{P}$ *such that:*

i) For all i, j

$$P_{ij} \geq 0 \, ;$$

ii) For all i

$$\sum_{j=1}^{m} P_{ij} = 1 \, ;$$

is said to be stochastic. Similarly, a vector $\mathbf{p}$ *such that:*

i) For all i

$$p_i \geq 0 \, ;$$

ii)

$$\sum_{i=1}^{m} p_i = 1 \, ;$$

is called a probability vector. □

The stochastic process represented by the sequence of random variables $\{ X_n \}$ produced by SA algorithms is a *Markov process.* In fact, (2.5) implies that the value of X_{n+1} depends only on the value of X_n, i.e., the probability of any particular future behavior of the process, when its present state is known exactly, is not altered by additional knowledge concerning its past behavior. Furthermore it is easily seen that (2.5) can be generalized to describe an $k - step$ transition of the SA algorithm as follows

$$\Pr\{ X_{n+k} = j \mid X_n = i \} = P_{ij}(T_n, T_{n+k}) \, , \tag{2.9}$$

where $P_{ij}(T_n, T_{n+k})$ is the (i, j) element of matrix $\mathbf{P}(T_n, T_{n+k})$ defined by

$$\mathbf{P}(T_n, T_{n+k}) = \begin{cases} \prod_{i=n}^{n+k-1} \mathbf{P}(T_i) & \text{if } k \geq 1 \, , \\ \mathbf{I} & \text{if } k = 0 \, , \end{cases} \tag{2.10}$$

where **I** is the identity matrix. To make the notation less cumbersome, the explicit dependence on T will be omitted whenever possible and $\mathbf{P}(m)$ and $\mathbf{P}(m,n)$ will replace $\mathbf{P}(T_m)$ and $\mathbf{P}(T_m, T_n)$ respectively.

With definition of the multistep transition probability matrix as in (2.9) and (2.10), the semi-group property of Markov processes [59]

$$P_{ij}(q,m) = \sum_{l=1}^{n} P_{il}(q,k) P_{lj}(k,m) \,, \tag{2.11}$$

is automatically satisfied by the matrix multiplication rules for all q, k, m such that $q \leq k \leq m$ [2].

Finally since the configuration space of combinatorial optimization problems is a countable and, in general, finite set, the Markov process becomes a *Markov chain.*

[2] Equation (2.11) is also known as the discrete version of the Chapman-Kolmogorov equation.

Chapter 3

Asymptotic Behavior: Homogeneous Theory

The aim of this chapter is to show that, given an arbitrary combinatorial optimization problem, SA can produce a solution which is arbitrarily close to the global optimum. Given the stochastic nature of the algorithm, convergence to the global optimum is proved if it is possible to show that for any given ϵ, there exists a corresponding k_ϵ such that for all $k \geq k_\epsilon$

$$\Pr\{ X_k \in \Omega_* \} \geq 1 - \epsilon \,, \tag{3.1}$$

where X_k is the random variable which describes the solution produced by SA at the k-th iteration. Given that the mathematical model for SA is a finite Markov chain, the probability distribution over Ω is represented by a finite real vector. Accordingly (3.1) can be rephrased in terms of distance of the probability distribution of the states after k transitions $\mathbf{p}(k)$ from the optimal probability distribution $\mathbf{e}_*$, i.e. a probability vector whose entries are different from zero only if the corresponding state belongs to Ω_*.

Given the finiteness of the state space, all the L^p norms are equivalent and the choice of one norm over the others does not affect the generality of the results

obtained [60]. However, the L^1 vector norm

$$\|\mathbf{x}\| = \sum_{i=1}^{n} |x_i|$$

and the corresponding induced matrix norm

$$\|\mathbf{A}\| = \sup_i \sum_{j=1}^{n} |a_{ij}|$$

have the advantage to extend easily the results proved for Markov chain with finite state space, to the case in which the state space is only countable [61].

Two approaches have been followed to prove convergence in the form (3.1). The first one, simpler, assumes that T is fixed for a time long enough for the chain to reach its stationary probability distribution $\pi(T)$. With this assumption, the behavior of SA as T approaches zero is determined completely by the corresponding behavior of $\pi(T)$. With this assumption, the appropriate mathematical model for SA is an *homogeneous* Markov chain. The converge results proved using this model are presented in this chapter.

The second approach, instead, assumes that only a small number of iterations is performed at each value of the T. In this case, obviously, the Markov chain does not reach the stationary probability distribution. The dependence of the transition probability matrix on T cannot be disregarded and the theory of *inhomogeneous* Markov chains has to be used. The details of this second approach are presented in the following chapter.

The remaining part of the present chapter is subdivided into three sections. First the basic results related to the existence of stationarity probability distribution for homogeneous Markov chains are presented. Then, the conditions on the acceptance and generation function are derived for SA to converge in the sense of (3.1). Finally, Section 3.3 contains a review of similar results proved independently by other authors.

3.1 Homogeneous Markov Chains

In this section, a few basic definitions and theorems on homogeneous Markov chains, relevant to the following discussion, are reviewed. All the major results are presented without proofs and a reference to a basic text where the proof can be found is given.

3.1.1 Classification of States

We will start the discussion by introducing the classical classification of the states of the Markov chain.

Definition 3.1.1 *State j is said to be accessible from state i if for some integer $m \geq 0$, $P_{ij}^{(m)} > 0$. Two states i and j, accessible to each other, are said to communicate.* □

The relation induced by this definition is an equivalence relation. The equivalence classes induced by this relation consist of all those states for which there exist a probability greater than zero, to go from one state to the other in both directions in a finite number of steps.

Definition 3.1.2 *A Markov chain is said to be irreducible if the equivalence relation induces a unique class.*

Example 3.1.1 Consider a Markov chain with n states. If there is an ordering of the states such that the transition probability matrix $\mathbf{P}$ has the following block representation

$$\begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{pmatrix} 0 & \mathbf{P}_2 \\ \mathbf{P}_1 & 0 \end{pmatrix} \end{array}, \tag{3.2}$$

where $\mathbf{P}_i$, $i = 1, 2$ are n_i-by-n_i matrices that cannot be put into a block representation with the same structure as in (3.2), then the Markov chain is irreducible. If

$\mathbf{P}$ has the following block representation

$$\mathbf{P} = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{pmatrix} \mathbf{P}_1 & 0 \\ \mathbf{T}_{21} & \mathbf{P}_2 \end{pmatrix} \end{array},$$

where $\mathbf{P}_i$, $i = 1, 2$ are two matrices with elements not all zero, the corresponding Markov chain is not irreducible. Furthermore states $n_1 + 1, \cdots, n_1 + n_2$ are called *transient.* □

Definition 3.1.3 *The period of state i is said to be the greatest common divisor of all integers $m \geq 1$ such that*

$$P_{ii}^{(m)} > 0 .$$

Definition 3.1.4 *A Markov chain in which each state has period one is said to be aperiodic.* □

It is easy to show that periodicity is a *class property* i.e., all the states in an equivalence class have the same period [62].

Theorem 3.1.1 *If the Markov chain is irreducible and there exists a state, say i, such that*

$$P_{ii} > 0 ,$$

then the Markov chain is aperiodic.

Proof. From the assumption $P_{ii} \geq 0$, it follows that state i has period 1. Therefore every state in the same equivalence class as state i has period 1. Since the chain is irreducible, all the states are members of the same class and hence the chain is aperiodic. □

Example 3.1.2 The Markov chain represented by the following probability transition matrix

$$\mathbf{P} = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{pmatrix} 0 & \mathbf{P}_2 \\ \mathbf{P}_1 & 0 \end{pmatrix} \end{array},$$

where $\mathbf{P}_1$ and $\mathbf{P}_2$ are two matrices which elements are not all zero, is periodic with period at least two. □

Definition 3.1.5 *A Markov chain which is aperiodic and irreducible is said to be regular .*

Theorem 3.1.2 *A Markov chain is regular if there exists an integer m such that the corresponding transition probability matrix $\mathbf{P}^m$ has no zero entries.*

Proof. Assume that the Markov chain is regular and has n states. Suppose that there exists a pair of indexes (i,j) such that $P_{ij}^{(m)} = 0$ with $m \geq n$. If $m \geq n$ and $P_{ij}^{(m)} = 0$ only two cases are possible: First there is no connection between i and j but in this case the Markov chain would not be irreducible. Second there is no path of length m between i and j but there is a shorter path with length $k < n$. In this case i and j are members of a periodic class with period k. In both cases we are contradicting the regularity assumption. □

Definition 3.1.6 *Let $h_{ii}^{(k)}$ be the probability that starting from state i, the first return to state i occurs at the k-th transition, i.e.,*

$$h_{ii}^{(k)} = \Pr\{ X_k = i, X_j \neq i, \ \ j = 1, 2, ..., k-1, \mid X_0 = i \} .$$

h_{ii} is defined by means of the following recursion

$$P_{ii}^{(k)} - \sum_{l=0}^{k} h_{ii}^{(k-l)} P_{ii}^{(l)} = \begin{cases} 1 & \text{if } k = 0 , \\ 0 & \text{if } k > 0 . \end{cases} \tag{3.3}$$

A state i is recurrent if $\sum_{k=1}^{\infty} h_{ii}^{(k)} = 1$.

□

This definition says that a state i is recurrent if , starting from state i, the probability of returning to state i after some finite length of time is one. Recurrence is a relevant characteristic only when the Markov chain has a countable state space as shown in the following example

Example 3.1.3 The following probability transition matrix

$$P_{ij} = \begin{cases} q & \text{if } i = j+1\,, \\ p & \text{if } i = j-1\,, \\ 0 & \text{otherwise,} \end{cases} \tag{3.4}$$

represents the Markov process known as one-dimensional random walk on the positive and negative integers. At each transition, a particle moves with probability q one unit to the right and with probability p one unit to the left ($p+q=1$). If the process starts from the origin, and if $p \neq q$, there is a non zero probability that a particle initially at origin will drift to $+\infty$ if $q > p$ ($-\infty$ in the other case) without ever coming back to the origin. Hence the origin is not recurrent. The following condition

$$p = q = 1/2\,, \tag{3.5}$$

is necessary to ensure the recurrence of the Markov chain determined by (3.4). In [62] a formal proof of condition (3.5) is given. □

If the state space is finite, recurrence is implied by irreducibility. Recurrence is introduced here because ergodic results are proved for countable chains, i.e. a more general class that contains finite chains as those arising in the study of SA as a special case.

Recurrence as periodicity is a class property, i.e., all the states in an equivalence class are either recurrent or non recurrent.

3.1.2 Ergodic Theory

The properties that have been introduced above define a large class of Markov chains for which an ergodic theory has been developed. The two main results of this theory are recalled below.

Theorem 3.1.3 *Let i be the initial state and let $P_{ii}^{(0)} = 1$. If a Markov chain is regular and recurrent, then*

i) The following limit exists

$$\lim_{k\to\infty} P_{ii}^{(k)} = \frac{1}{\sum_{k=0}^{\infty} k h_{ii}^{(k)}},$$

where $h_{ii}^{(k)}$ is defined recursively by equation (3.3).

ii) For every j,

$$\lim_{k\to\infty} P_{ji}^{(k)} = \lim_{k\to\infty} P_{ii}^{(k)} = \pi_i = \Pi_{ji} . \tag{3.6}$$

Matrix Π is called the constant matrix of the chain.

Proof. The formal proof of the theorem is rather cumbersome and can be found in [62]. □

If n is large enough, from (3.6) follows that $P_{ji}^{(k)}$, the probability of being at the k-th iteration in state i, starting from state j, depends only on the state itself and is totally independent on the initial state j. In other words, the memory of the initial condition is completely lost. Note that π_i defined in Theorem 3.1.3 is always larger than or equal to zero. If it is strictly larger than zero, then the following important result holds.

Theorem 3.1.4 *If π_i's of equation (3.6) are greater than zero for all i and the Markov chain is regular and recurrent, then π_i's are uniquely determined by the following set of equations*

$$\sum_{i=1}^{|\Omega|} \pi_i = 1 , \tag{3.7}$$

and for all j

$$\sum_{i=1}^{|\Omega|} \pi_i P_{ij} = \pi_j , \tag{3.8}$$

$$\pi_j \geq 0 . \tag{3.9}$$

Proof. See [62]. □

The row vector $\pi = [\pi_1, \pi_2, \ldots, \pi_{|\Omega|}]$ is called the *stationary probability distribution vector* of the Markov chain.

The stationary probability distribution is very important since it characterizes completely the asymptotic behavior of a Markov chain.

3.1.3 Rate of Convergence

The results from the previous section allow us to state, under some mild assumptions on the transition matrix $\mathbf{P}$, that the Markov chain converges to its stationary probability distribution vector $\boldsymbol{\pi}$. The next point we want to address is related to the rate of convergence. In other words, we are interested in establishing how fast an arbitrary initial probability distribution vector $\mathbf{p}$ converges to the stationary probability distribution vector $\boldsymbol{\pi}$.

To state results on the rate of convergence we use two general theorems from the theory of non-negative matrices and we recall that a Markov chain has a transition probability matrix which is non-negative and stochastic according to Definition 2.2.1.

Theorem 3.1.5 (Frobenius-Perron) *Let* $\mathbf{P}$ *be a stochastic matrix. Let* $\boldsymbol{\pi}$ *the vector that satisfies (3.8) then:*

i) $\lambda_0 = 1$ *is an eigenvalue of* $\mathbf{P}$ *with algebraic multiplicity 1.*

ii) *Let* λ_i, $i = 0, 1, 2, \cdots, n-1$ *be the eigenvalues of* $\mathbf{P}$. *For all* $i = 1, 2, \cdots, n-1$

$$|\lambda_i| < \lambda_0 .$$

iii) $\boldsymbol{\pi}$ *and* $\mathbf{1}$ *are the left and right eigenvectors of* $\mathbf{P}$ *corresponding to* λ_0. $\mathbf{1}$ *is a column vector with all the entries equal to 1.*

Proof. Many different proofs of the Frobenius-Perron Theorem are available in the literature on Markov chains and non-negative matrices. One of these can be found for example on [63, page 2]. □

A direct consequence of the Frobenius-Perron theorem is:

Theorem 3.1.6 *Let* $\mathbf{P}$ *be a stochastic matrix. If the eigenvalues of* $\mathbf{P}$ *are ordered such that for all* $i = 0, 1, \ldots, n-1$

$$|\lambda_i| \leq |\lambda_{i+1}| \tag{3.10}$$

and

$$\lambda_0 = 1 \,,$$

then as $k \to \infty$

$$\mathbf{P}^k = \mathbf{1}\boldsymbol{\pi} + k^{m_1-1}|\lambda_1|^k \mathbf{B}(k, m_1) \,,$$

where m_1 *is the geometric multiplicity of* λ_1 *and* $\mathbf{B}(k, m_1)$ *is a matrix such that, for all* k

$$\|\mathbf{B}(k, m_1)\| \leq b_k \leq \bar{b} < \infty \,.$$

$\mathbf{B}(k, m_1)$ *is the residual matrix evaluated at the eigenvalue* λ_1 [64].

Proof. For the proof of this theorem see [63, page 9]. □

The combination of the Frobenius-Perron Theorem and of Theorem 3.1.6 yields:

Theorem 3.1.7 *If* $\mathbf{p}$ *and* $\mathbf{q}$ *are two probability vectors and* $\mathbf{P}$ *is a stochastic matrix, then as* $k \to \infty$

$$\frac{\|\mathbf{p}\mathbf{P}^k - \mathbf{q}\mathbf{P}^k\|}{\|\mathbf{p} - \mathbf{q}\|} \leq O(k^{m_1-1}|\lambda_1|^k)$$

Proof. The proof follows from the fact that $\mathbf{P}$ is a stochastic matrix. In fact

$$\begin{aligned} \|\mathbf{p}\mathbf{P}^k - \mathbf{q}\mathbf{P}^k\| &= \|(\mathbf{p} - \mathbf{q})\mathbf{P}^k\| \leq \\ &\leq \bar{b}\, k^{m_1-1}\, |\lambda_1|^k\, \|\mathbf{p} - \mathbf{q}\| \,. \end{aligned}$$

□

As a corollary to Theorem 3.1.7 is it easy to see how, regardless of the initial probability distribution, further transitions generate a probability distribution which approaches geometrically fast the stationary probability distribution. The above conclusion is stated more precisely in the following

Corollary 3.1.1 *Let* $\mathbf{p}$ *be the initial probability distribution vector of the states in the Markov chain and let* $\mathbf{p}^{(k)}$ *be the probability distribution vector after* k *iterations have been performed. If* $\boldsymbol{\pi}$ *is the asymptotic probability distribution vector then* $\|\boldsymbol{\pi} - \mathbf{p}^{(k)}\|$ *goes to zero at least as fast as* $O(k^{m_1-1}|\lambda_1|^k)$ *as* $k \to \infty$.

3.1.4 Reversible Markov Chains

In this section we will show how a particular structure of the transition probability matrix allows us to be more specific about the rate of convergence of the Markov chain to its stationarity distribution.

Definition 3.1.7 *A Markov chain is said to be reversible if its transition probability matrix* $\mathbf{P}$ *is such that for all pairs* (i, j)

$$\pi_i P_{ij} = \pi_j P_{ji} \,, \tag{3.11}$$

where π *is the stationary probability distribution of the chain.* □

Equation (3.11) it is often referred to as *detailed balance equation* as opposed to the more general *balance equation* (3.8). Note that while (3.11) implies (3.8), the reverse is not in general true [1].

Theorem 3.1.8 *If the Markov chain is reversible, then the eigenvalues of its transition probability matrix are real, and all have geometric multiplicity 1.*

Proof. Let us define the matrix $\mathbf{D}$

$$\mathbf{D} = \mathrm{diag}\{\, \pi_0, \pi_1, \ldots, \pi_{n-1} \,\} \,, \tag{3.12}$$

with the indices ordered as in (3.10). The matrix

$$\mathbf{L} = \mathbf{D}^{1/2}\mathbf{P}\mathbf{D}^{-1/2} \tag{3.13}$$

[1] The origin of the name reversible Markov chain comes from the study of the stochastic process obtained by reversing the time flow on a Markov chain [65]. The process is again a Markov chain and its transition probability matrix $\tilde{\mathbf{P}}$ is given by

$$\begin{aligned}
\tilde{P}_{ij} &= \Pr\{ X_m = j \mid X_{m+1} = i, X_{m+2} = i_2, \ldots, X_{m+k} = i_k \} = \\
&= \frac{\Pr\{ X_m = j, X_{m+1} = i, X_{m+2} = i_2, \ldots, X_{m+k} = i_k \}}{\Pr\{ X_{m+1} = i, X_{m+2} = i_2, \ldots, X_{m+k} = i_k \}} = \\
&= \frac{\Pr\{ X_m = j \}\Pr\{X_{m+1} = i \mid X_m = j\}}{\Pr\{ X_{m+1} = i \}} \\
&\quad \frac{\Pr\{ X_{m+2} = i_2, \ldots, X_{m+k} = i_k \mid X_{m+1} = i \mid X_m = j \}}{\Pr\{ X_{m+2} = i_2, \ldots, X_{m+k} = i_k \mid X_{m+1} = i \}} = \frac{\pi_j P_{ji}}{\pi_i} \,.
\end{aligned}$$

is symmetric, in fact

$$\begin{aligned} L_{ij} &= \frac{\pi_i^{1/2} P_{ij}}{\pi_j^{1/2}} = \\ &= \frac{\pi_i^{1/2} P_{ji} \pi_j}{\pi_j^{1/2} \pi_i} = \\ &= \frac{\pi_j^{1/2} P_{ji}}{\pi_i^{1/2}} = L_{ji} \end{aligned}$$

From the symmetry of $\mathbf{L}$, $\mathbf{P}$ is similar to a diagonal matrix, i.e. there exists a non singular transformation $\mathbf{S}$ such that

$$\mathbf{SPS}^{-1} = \mathrm{diag}\{\, \lambda_0, \lambda_1, \ldots, \lambda_{n-1} \,\} \,,$$

with

$$\mathbf{S} = \mathbf{TD}^{1/2}$$

and $\mathbf{T}$ is an orthogonal matrix. Since $\mathbf{P}$ is a real matrix, it follows that the eigenvalues are real and their geometric multiplicity is 1. □

Theorem 3.1.8 yields a different version of Corollary 3.1.1.

Corollary 3.1.2 *Let* $\mathbf{p}$ *be the initial probability distribution vector of a reversible Markov chain and let* $\mathbf{p}^{(k)}$ *be the probability distribution vector after* k *iterations have been performed. If* $\boldsymbol{\pi}$ *is the asymptotic probability distribution vector then*

$$\|\boldsymbol{\pi} - \mathbf{p}^{(k)}\| \leq |\lambda_1|^k \sum_{i=1}^{n-1} \|\mathbf{D}^{-1/2} \mathbf{r}^{(i)} \mathbf{l}^{(i)} \mathbf{D}^{1/2}\| \,, \tag{3.14}$$

where $\mathbf{r}^{(i)}$ *and* $\mathbf{l}^{(i)}$ *are the right and left eigenvectors associated to the* i*-th eigenvalue of* $\mathbf{L}$ *respectively with* $\mathbf{L}$ *as in (3.13).*

Proof. The proof follows by direct computation using the dyadic decomposition of $\mathbf{P}$ [64]. In fact, since $\mathbf{P}$ is similar to the diagonal matrix $\mathrm{diag}\{\lambda_0, \lambda_1, \ldots, \lambda_{n-1}\}$, we have for any k

$$\mathbf{P}^k = \mathbf{1}\boldsymbol{\pi} + \sum_{i=1}^{n-1} \lambda_i^k \mathbf{D}^{-1/2} \mathbf{r}^{(i)} \mathbf{l}^{(i)} \mathbf{D}^{1/2} \,. \tag{3.15}$$

Substituting (3.15) into (3.14)

$$\begin{aligned}\|\boldsymbol{\pi} - \mathbf{p}^{(k)}\| &= \|\sum_{i=1}^{n-1} \lambda_i^k \mathbf{p}\mathbf{D}^{-1/2}\mathbf{r}^{(i)}\mathbf{l}^{(i)}\mathbf{D}^{1/2}\| \leq \\ &\leq |\lambda_1|^k \sum_{i=1}^{n-1} \|\mathbf{p}\mathbf{D}^{-1/2}\mathbf{r}^{(i)}\mathbf{l}^{(i)}\mathbf{D}^{1/2}\| \leq \\ &\leq |\lambda_1|^k \sum_{i=1}^{n-1} \|\mathbf{D}^{-1/2}\mathbf{r}^{(i)}\mathbf{l}^{(i)}\mathbf{D}^{1/2}\| .\end{aligned}$$

□

Corollary 3.1.2 has another interesting formulation which gives a better insight on the behavior of the Markov chain. If (3.15) is rewritten componentwise we have [66]:

$$P_{ij}^{(k)} = \pi_j + \sqrt{\frac{\pi_j}{\pi_i}} \sum_{h=1}^{n-1} \lambda_h^k \, r_i^{(h)} l_j^{(h)} . \tag{3.16}$$

Equation (3.16) conveniently breaks $P_{ij}^{(k)}$ into two components. The first one is the limit to which $P_{ij}^{(k)}$ converges as k approaches infinity while the second one measures the distance of $\mathbf{P}$ from the *constant matrix* $\boldsymbol{\Pi}$ defined in (3.6). From (3.16) it is clear that loss of memory is dependent not only on the magnitude of λ_1 but also depends, at least on the short time horizon, on the ratio of the stationary probability of state j versus the stationary probability of state i.

3.2 Asymptotic Properties of Simulated Annealing

Results quoted in the previous section cannot be applied directly to the Markov chain representing the stochastic process generated by SA. In fact, these results are valid for transition probability matrices whose entries are constant. In SA, some of the transition probabilities (See Chapter 2) depend on the parameter T which is updated during the evolution of the algorithm and hence are dependent on time. However, if T is fixed, i.e. in the inner loop of SA (See Figure 2.1), the transition probabilities are constant and the results from the theory of homogeneous Markov chain can be applied.

3.2.1 Ergodic Theory

The strategy to prove the ergodic properties of SA is as follow: First determine which conditions SA must satisfy so that a stationary probability distribution exists for each given value of the parameter T. Then introduce a function $\pi_i(T)$, defined on the set of configurations, with the property that, as T approaches zero, $\pi_i(T)$ is different from zero only for those configurations that are global optima for the combinatorial optimization problem. Finally select an acceptance function f such that $\pi_i(T)$ is the stationary probability distribution of the Markov chain describing SA.

If we can prove that, given a function $\pi_i(T)$ with the properties mentioned above and a suitable generation function $G_{ij}(T)$, it is possible to determine an acceptance function f with the features outlined in Chapter 2, then an implementation of an SA algorithm is obtained which converges with probability one to the global optimal solution of the combinatorial optimization problem.

Before we begin to present the details of the convergence proof, it is necessary to make the following assumption on the characteristics of the sequence of control parameters $\{T_k\}$.

Assumption 3.2.1 *The sequence of control parameters $\{T_k\}$ has to be such that:*

$$\lim_{k\to\infty} T_k = 0$$

and $T_m > T_k$ for all $k > m$.

□

The first condition on SA is related to the irreducibility of the underlying Markov chain. In SA context, irreducibility is tantamount to require that for all $i \in \Omega$ and $j \in \Omega_i$

$$G_{ij}(T) > 0\,, \tag{3.17}$$

for any T. Equation (3.17) imposes a constraint on the generation mechanism used. However, (3.17) by itself is not sufficient to ensure irreducibility. In fact we have to require that the acceptance function f assigns a non zero probability to any of the

edges of $\mathcal{G}$. This is guaranteed if f belongs to the class of admissible functions $\mathcal{F}$ defined as follows:

Definition 3.2.1 *f_T is an admissible function if for all c and for all $T \neq 0$*

$$0 < f_T(c) \leq 1 .$$

□

The next condition is related to the aperiodicity of the Markov chain.

Proposition 3.2.1 *Let the Markov chain corresponding to* **SA** *be irreducible for all $T > 0$. If the acceptance function $f \in \mathcal{F}$ is such that there exists at least a pair of states i and j for which*

$$0 < f_T(\Delta c_{ij}) < 1, \tag{3.18}$$

for all $T > 0$, then the Markov chain is aperiodic for all $T > 0$.

Proof. If (3.18) holds, then according to (2.5), $P_{ii}(T) > 0$, for all $T > 0$ and the proof follows from Theorem 3.1.1 and the irreducibility of the Markov chain. □

The condition of Proposition 3.2.1 is always satisfied since there is at least one state for which (3.18) holds: The global optimum.

Proposition 3.2.2 *Let Ω be finite and the Markov chain associated to* **SA** *be irreducible for all $T > 0$, then the Markov chain is recurrent for all $T > 0$.*

Proof. Since the state space of the Markov chain is finite, then after a number of steps greater than $|\Omega|$ at least a state of the Markov chain has been visited twice. By definition, that state is recurrent. Since recurrence is a class property and the Markov chain is irreducible by hypothesis, the Markov chain is recurrent. □

According to Theorem 3.1.4, the Markov chain associated with an SA algorithm which satisfies the conditions of Propositions 3.2.1 and 3.2.2, has a stationary probability distribution.

Now we look for a form of the stationary probability distribution which, as T goes to zero, is different from zero only in global minima of c.

To this end, we have to find under which conditions a stationary probability distribution $\pi_i(T)$ is different from zero, as T goes to zero, only if the i-th configuration is the global optimal solution.

Theorem 3.2.1 *Let $\pi_i(T)$, be a function that for all $i \in \Omega$ maps $\mathbb{R}^+$ into $(0,1]$. $\pi_i(T)$ is defined by*

$$\pi_i(T) = \frac{g(c(i),T)}{Z(T)}\,, \tag{3.19}$$

where $Z(T)$ is a normalizing factor such that

$$\sum_{i\in\Omega} \pi_i(T) = 1 \tag{3.20}$$

and g is such that for all i, j, and $T > 0$

$$g(c,T) > 0 \tag{3.21}$$

$$\lim_{T\leftarrow 0} \frac{g(c_i,T)}{g(c_j,T)} = \begin{cases} 0 & \text{if } c_i > c_j\,, \\ \dfrac{g(c_i)}{g(c_j)} & \text{if } c_i = c_j\,, \\ \infty & \text{if } c_i < c_j\,. \end{cases} \tag{3.22}$$

Then

$$\lim_{T\to 0} \pi(T) = \mathbf{e}_*\,, \tag{3.23}$$

where $\mathbf{e}_*$ is the following probability vector

$$(e_*)_i = \begin{cases} \dfrac{g(c_i)}{\sum\limits_{j\in\Omega_*} g(c_j)} & \text{if } i \in \Omega_*\,, \\ 0 & \text{if } i \in (\Omega \cap \Omega_*^c)\,. \end{cases} \tag{3.24}$$

The set Ω_* is the set of global minima and is defined in (2.1).

Proof. The proof of (3.23) is straightforward because of the properties of the function g. In fact for all $i \in \Omega_*$, (3.19) can be rewritten as

$$\pi_i(T) \;=\; \frac{g(c_i,T)}{\sum\limits_{j\in\Omega_*} g(c_j,T) + \sum\limits_{j\in(\Omega\cap\Omega_*^c)} g(c_j,T)} \;=$$

$$= \frac{1}{\sum_{j\in\Omega_*} \frac{g(c_j,T)}{g(c_i,T)} + \sum_{j\in(\Omega\cap\Omega_*^c)} \frac{g(c_j,T)}{g(c_i,T)}} . \tag{3.25}$$

If we let T approach zero and use (3.22), we have that for all $i \in \Omega_*$ and $j \in (\Omega \cap \Omega_*^c)$ the second sum at the denominator of (3.25) vanishes while the first term approaches

$$\sum_{j\in\Omega_*} \frac{g(c_j)}{g(c_i)} .$$

Similarly, if $i \in (\Omega \cap \Omega_*^c)$ the first sum at the denominator diverges. This completes the proof of the theorem. □

Next, we have to specify under which conditions on f and G, a $\pi_i(T)$ of the form described above is indeed the stationary probability function of the Markov chain associated to SA.

Theorem 3.2.2 *Let Ω'_j be defined by*

$$\Omega'_j = \{\omega_i : \omega_j \in \Omega_i\} ,$$

if $f \in \mathcal{F}$ and for all j

$$\sum_{i\in\Omega'_j} g(c_i,T)G_{ij}(T)f_T(\Delta c_{ij}) = \tag{3.26}$$
$$= g(c_j,T) \sum_{i\in\Omega_j} G_{ji}(T)f_T(\Delta c_{ji})$$

then $\pi_i(T)$ defined by (3.19) (3.22) is the stationary probability distribution of the Markov chain whose one-step transition probability is given by (2.5)-(2.8).

Proof. The proof is obtained by verifying that g, G, and f satisfy the conditions of Theorem 3.1.4. In view of the assumptions made on function G and of Propositions 3.2.1 and 3.2.2, the Markov chain defined by (2.5)-(2.8) is irreducible, aperiodic and positive recurrent. It is now immediate to see that $\pi_i(T)$ defined by (3.19) satisfies (3.7) because of (3.20) and (3.9) because of (3.21). Finally it takes just a little algebra to see that (3.8) is satisfied automatically once f is chosen as specified by (3.26). □

Theorem 3.2.2 is important since it suggests a way to construct a SA algorithm with guaranteed convergence properties. In fact one first selects a function $\pi_i(T)$ that ensures the convergence to the global optima (See Theorem 3.2.1) and then selects an acceptance function f and a generation rule G such that Propositions 3.2.1, 3.2.2 and Theorem 3.2.2 are satisfied. Propositions 3.2.1, 3.2.2 and Theorem 3.2.2 are sufficient condition for the algorithm to converge to the global optimum provided that π is achieved at each value of T.

If additional assumptions are placed on the generation function G and the Markov chain is constrained to be reversible, a simplified version of (3.26) gives an explicit expression for f as shown in the following

Corollary 3.2.1 *If the function $G_{ij}(T)$ is such that for all $i \in \Omega_j$ and $j \in \Omega_i$*

$$G_{ij}(T)G_{ji}(T) \neq 0 \tag{3.27}$$

and the Markov chain is reversible, (3.26) can be replaced by

$$\begin{aligned} g_i(c_i,T)G_{ij}(T)f_T(\Delta c_{ij}) = \\ = g_j(c_j,T)G_{ji}(T)f_T(\Delta c_{ji}) \ . \end{aligned} \tag{3.28}$$

Equation (3.28) yields the following explicit relation for f

$$\frac{f_T(\Delta c_{ij})}{f_T(\Delta c_{ji})} = \frac{G_{ji}(T)}{G_{ij}(T)} \frac{g_j(c_j,T)}{g_i(c_i,T)} \ .$$

□

Equation (3.27) is automatically satisfied by the definition of *symmetric generation rule*

$$G_{ij}(T) = G_{ji}(T) = G_{ij} \tag{3.29}$$

given in [67,68].

Remark 3.2.1 Simulated annealing as proposed by Kirkpatrick [12] has g, G and f defined as follows

$$g(c_i,T) \ = \ e^{\frac{-c_i}{T}} \ ,$$

$$G_{ij}(T) = \begin{cases} 1/|\Omega_i| & \text{for all } j \in \Omega_i\,, \\ 0 & \text{for all } j \in (\Omega_i)^c\,, \end{cases}$$

$$f_T(\Delta c_{ij}) = \min\left[1, e^{\frac{-c_j - c_i}{T}}\right]. \tag{2.3}$$

These functions satisfy the conditions of Corollary 3.2.1 and then, *a fortiori*, of Theorem 3.2.2.

Another SA algorithm can be generated just by replacing the acceptance function of (2.3) with the following one

$$f_T(\Delta c_{ij}) = \frac{e^{-\frac{c_j - c_i}{T}}}{1 + e^{\frac{c_j - c_i}{T}}} \tag{3.30}$$

and leaving functions G and g unchanged. □

The result stated in Corollary 3.2.1 is of interest when a SA algorithm has to be designed. In fact first we select the form of the stationary probability distribution, then we choose the generation function which suits the problem the best, and finally we use (3.26) to determine f. Note that (3.26) requires to know the cost of all the neighbors of solution i to determine the probability of transition from i to j. This is a limitation in practical applications since the cardinality of Ω_i may be quite large and we may even ignore which of the solutions are members of Ω_i. Corollary 3.2.1 offers a solution to this problem. In fact if we restrict ourselves to reversible Markov chains, than we can use (3.28) instead of (3.26) to define f. Notice that with (3.28) the transition probability is determined only by the cost of solutions i and j. In Remark 3.2.1 it is shown how this degree of freedom can be exploited to generate two different acceptance strategies that lead to the same asymptotic behavior. However, the choice of f affects the rate of convergence of the algorithm and, in Chapter 6 we will discuss how to select a function f that allows the fastest convergence at fixed T.

Note that when the control parameter T, approaches zero, both the acceptance functions f given by (2.3) and (3.30) become a unitary step function that

assigns probability one only to those transitions which improve the cost function and probability zero to the others (See Figure 3.1 and Figure 3.2). Hence, when T is

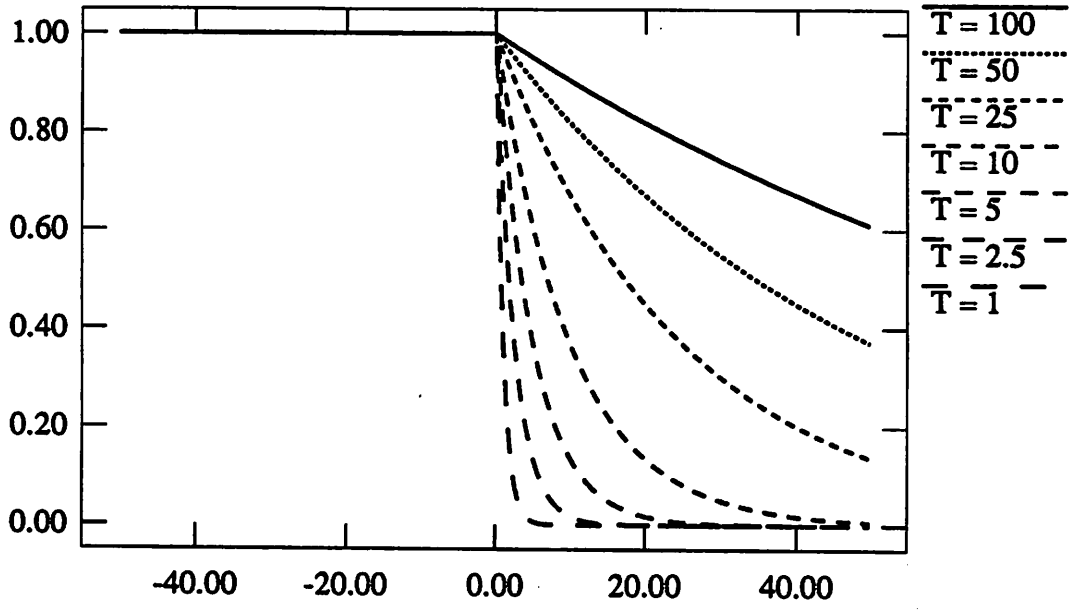

Figure 3.1: Acceptance function given by (2.3).

set to zero, the acceptance functions degenerate into the usual greedy strategy and select, among all the new configuration that are generated, those with cost lower than the present configuration only.

3.2.2 Rate of Convergence

Theorem 3.1.6 and Corollary 3.1.2 give us a tool to assess the convergence rate of SA. Corollary 3.1.2 in particular gives us a handy formula to measure the distance of $\mathbf{p}^{(k)}$ from π. In fact, from the definition of the constant matrix Π (3.6), it follows that

$$\begin{aligned} \|\mathbf{p}^{(k)} - \pi\| &\leq \|\mathbf{p}\|\|\mathbf{P}^k - \Pi\| \leq \\ &= \|\mathbf{P}^k - \Pi\| . \end{aligned} \tag{3.31}$$

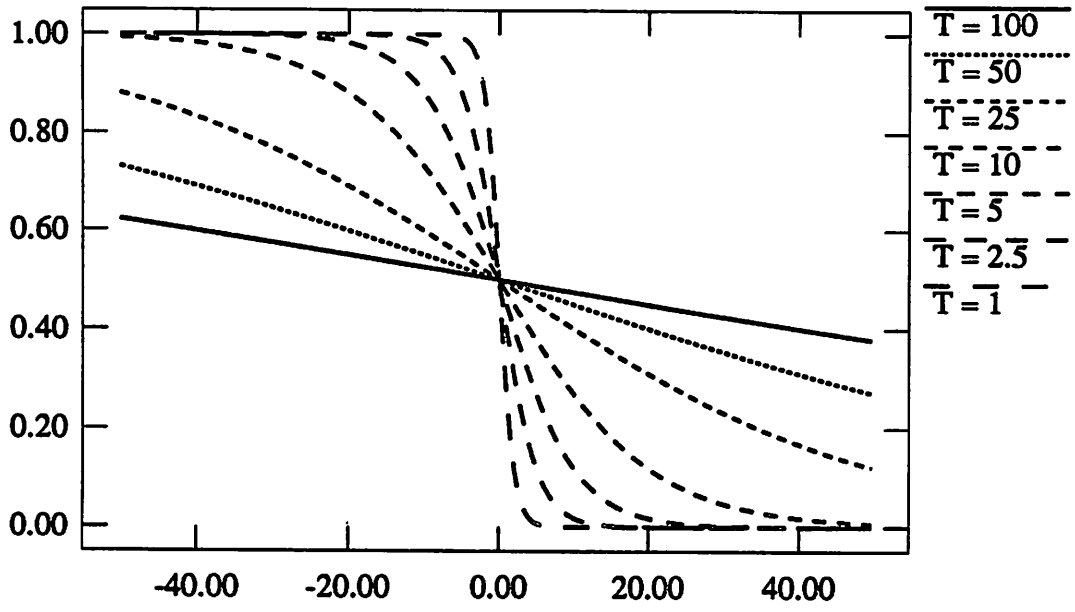

Figure 3.2: Acceptance function given by (3.30).

The substitution of (3.16) into (3.31) gives

$$\begin{aligned}
\|\mathbf{p}^{(k)} - \boldsymbol{\pi}\| &\leq \max_i \sum_{j=1}^{n} |P_{ij}^{(k)} - \pi_j| = \\
&= \max_i \sum_{j=1}^{n} |\sqrt{\frac{\pi_j}{\pi_i}} \sum_{h=1}^{n-1} \lambda_h^k \, r_i^{(h)} l_j^{(h)}| \leq \\
&\leq n \max_{i,j} \sqrt{\frac{\pi_j}{\pi_i}} \sum_{h=1}^{n-1} |\lambda_h^k| \, |r_i^{(h)}| \, |l_j^{(h)}| \leq \\
&\leq n \, |\lambda_1^k| \max_{i,j} \sqrt{\frac{\pi_j}{\pi_i}} \sum_{h=1}^{n-1} |r_i^{(h)}| \, |l_j^{(h)}| \leq \\
&\leq n \, |\lambda_1^k| \max_{i,j} \sqrt{\frac{\pi_j}{\pi_i}},
\end{aligned}$$

where the last inequality follows from orthonormality of $\mathbf{r}$ and $\mathbf{l}$. If we assume that

$$\pi_i(T) = \frac{e^{-\frac{c_i}{T}}}{Z(T)},$$

then we have

$$\|\mathbf{p}^{(k)} - \boldsymbol{\pi}\| \leq n \, |\lambda_1^k| \, e^{-\frac{\Delta c_{max}}{2T}}, \tag{3.32}$$

where

$$\Delta c_{max} = \max_{i,j \in \Omega} (c_j - c_i) .$$

The last inequality gives us some insight on the influence of the cost function on the rate of convergence. If we assume that there are two combinatorial optimization problems for which the matrix **P** has the first two eigenvalues equal, then SA will converge faster for the problem whose cost function has the larger Δc_{max}.

Equation (3.32) unfortunately has limited utility when SA is applied to solve practical problems. In fact, in these cases, to know of the second largest eigenvalue of matrix **P** is out of the question since the size of the matrix is extremely large and its entries are not even known. Furthermore what really interests us is to know λ_1 as a function of T and this is hopeless! However, intuitively the rate of convergence must be related to some extent to the topology of $\mathcal{G}$ and there are problems for which it is easier to make reasonable assumptions on topology of $\mathcal{G}$ than it is to estimate the second largest eigenvalue of **P**.

Intuition tells us that the Markov chain will converge faster if the probability of getting stuck in a region of the graph with relatively small stationarity probability is small compared to the probability of escaping that region. More precisely, if we consider an arbitrary subset Ω_s of the vertices of $\mathcal{G}$ such that $\emptyset \subset \Omega_s \subset \Omega$, the stationarity probability of being in Ω_s is given by

$$\sum_{i \in \Omega_s} \pi_i$$

while the probability to escape from Ω_s to the rest of the space $\{ \Omega_s \}^c$ is

$$\sum_{i \in \Omega_s, j \in \{\Omega_s\}^c} \pi_i P_{ij} .$$

We are interested in the ratio between the two probabilities defined above, namely

$$\Phi_s(T) = \frac{\sum_{\substack{i \in \Omega_s \\ j \in \{\Omega_s\}^c}} \pi_i(T) P_{ij}(T)}{\sum_{i \in \Omega_s} \pi_i(T)} .$$

$\Phi_s(T)$ is a called the conductance of set Ω^s [69]. The conductance of $\mathcal{G}$, $\Phi_{\mathcal{G}}(T)$, is defined by

$$\Phi_{\mathcal{G}}(T) = \min_{\emptyset \subset \Omega^s \subset \Omega} \max\{\Phi_s(T), \Phi_{\bar{s}}(T)\},$$

where $\Phi_{\bar{s}(T)}$ is the conductance of $\{\Omega^s\}^c$.

The relation between $\Phi_{\mathcal{G}}(T)$ and the second largest eigenvalue of $\mathbf{P}$, λ_1 in our notation, has been proven by Jerrum and Sinclair [69] and is stated in the following

Theorem 3.2.3 *Given a regular Markov chain with transition probability matrix* $\mathbf{P}$ *and underlying graph* $\mathcal{G}$*, the second largest eigenvalue* λ_1 *and the conductance of* $\mathcal{G}$ *satisfy the following inequality*

$$\lambda_1(T) \leq 1 - \frac{\Phi_{\mathcal{G}}(T)^2}{2}.$$

□

Theorem 3.2.3 by Jerrum and Sinclair is interesting because it establishes a relation between the topology of the graph and $\lambda_1(T)$. However its use to estimate the rate of convergence of SA is rather limited in practical applications. In fact, while it is possible to determine, for a number of combinatorial optimization problems, an expression for the conductance evaluated when T is infinite [70], the dependence of the conductance on T is not monotonic and hence it is impossible to find a bound for $\Phi_{\mathcal{G}}(T)$ based on $\Phi_{\mathcal{G}}(\infty)$ for finite values of T.

In Chapter 5 we will see that, using the results of the inhomogeneous theory it is possible to estimate the convergence rate of SA on the basis of quantities that relates the topology of the graph and the characteristics of the cost function.

Before we conclude the discussion on the rate of convergence a final comment is in order. The homogeneous ergodic theory gives us sharp bounds on the rate of convergence of the Markov chain to the asymptotic probability distribution. Unfortunately the use of these results requires the knowledge of either the eigenvalues of $\mathbf{P}$ or the conductance of the underlying graph and this poses serious restrictions to their application to estimate the rate convergence of SA to the

stationary probability in practical applications. However the results of the homogeneous theory are useful to select among the different acceptance functions, which guarantee the same asymptotic probability distribution, the one that will produce the faster convergence. The details will be presented in Section 6.1.1.

3.2.3 General Comments

Theorems 3.2.1 requires the algorithm to reach the stationarity probability distribution at each value of T. However reaching exactly stationarity requires, in general, an infinite number of moves as stated in the following theorem

Theorem 3.2.4 *Let* $\mathbf{P}$ *be a stochastic matrix. If there exist a finite* m *such that* $\mathbf{P}^{m+1} = \mathbf{P}^m$ *then there exist an integer* $p \leq m$ *such that* $\mathbf{P}^p = \boldsymbol{\Pi}$. *Furthermore, if* $\mathbf{P}$ *is reversible,* $p = 1$.

Proof. If $\mathbf{P}^{m+1} = \mathbf{P}^m$ then $\psi(\mathbf{P}) = 0$ with $\psi(z)$ being the polynomial

$$\psi(z) = z^m(z-1)\,.$$

From the Cayley-Hamilton theorem, it follows that the minimal polynomial $\varphi(z)$ of $\mathbf{P}$ divides $\psi(z)$ exactly and hence has the form $\varphi(z) = z^p(z-1)$ [64] with $p \leq m$. Since the solutions of equation $\varphi(z) = 0$ are the eigenvalues of $\mathbf{P}$, it follows that the only possible choice for them is either 0 or 1. Furthermore, from the Frobenius-Perron theorem, the largest eigenvalue is a simple solution of the characteristic equation for $\mathbf{P}$. As a consequence of the above considerations it follows that the Jordan form for $\mathbf{P}$ is given by

$$\mathbf{J} = \begin{bmatrix} 1 & \mathbf{o} & \mathbf{o} & \dots & \mathbf{o} \\ \mathbf{o} & \mathbf{J}_1 & \mathbf{o} & \dots & \mathbf{o} \\ \mathbf{o} & \mathbf{o} & \mathbf{J}_2 & \dots & \mathbf{o} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{o} & \mathbf{o} & \mathbf{o} & \dots & \mathbf{J}_q \end{bmatrix}, \tag{3.33}$$

where the i-th block on the diagonal has the following form

$$\mathbf{J}_i = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix}$$

and size n_i with $\sum_{i=1}^{q} n_i = n-1$. Using the Jordan form of $\mathbf{P}$, it is possible to extend the dyadic decomposition (3.15), to the general case in which $\mathbf{P}$ is not reversible, to obtain [64]

$$\mathbf{P}^k = \mathbf{1}\boldsymbol{\pi} + \mathbf{T}^{-1} \sum_{i=1}^{q} \tilde{\mathbf{J}}_i^{\,k} \mathbf{T}\,, \tag{3.34}$$

where $\tilde{\mathbf{J}}_i$ is obtained from $\mathbf{J}$ (3.33) by setting to zero everything but block $\mathbf{J}_i$. The term $\mathbf{T}^{-1} \sum_{i=1}^{q} \tilde{\mathbf{J}}_i^{\,k} \mathbf{T}$ is identically zero for all $k > \max_{1 \le i \le q}\{n_i\} - 1 = p - 1$. From equation (3.34) follows the theorem. □

Theorem 3.2.4 implies that only two cases are possible. Either $\boldsymbol{\pi}$ is achieved in p steps, or an infinite number of steps is necessary. This in turn means that if the Markov chain is reversible, $\boldsymbol{\pi}$ is obtained in one step or, equivalently, that we have enough information to recognize directly the best solution to the problem!

From the above discussion, any strategy which requires to achieve, at each value of T, stationarity exactly, is practically inapplicable. In fact, an SA algorithm performing an infinite number of iterations for each value of the parameter T is a conceptual, non implementable algorithm [7], in the sense that an internal loop is never exited.

However the results of the homogeneous theory can be used to guide the selection of an annealing schedule. If we assume that function g is continuous in its second argument, then also $\pi_i(T)$ is a continuous function. The continuity of $\pi_i(T)$ implies that the stationary probability distribution for a particular value of the controlling parameter, say $\hat{T}$, is a good approximation for the stationary probability distribution for all the values of T sufficiently close to $\hat{T}$. This result suggests a strategy for the control of T: Start with a value of T for which the

stationary probability distribution is easy to estimate, and update T so that only a few iterations are needed to obtain a good approximation to the new stationary probability distribution. We will elaborate more on this point in Chapter 7.1 where we discuss how to derive an "efficient" control strategy for the algorithm.

3.3 Related Work

The assumptions used in the previous section are, to the best of my knowledge, the least restrictive of those used to date. A number of other authors (See e.g. [55,56,67,68,71]) have derived similar results using, however, a set of more stringent assumptions. The conditions they used are implied by those presented in Section 3.2 but not vice versa.

The approach proposed in Section 3.2 to derive the asymptotic features of **SA** is a constructive one. It is similar to the approach by Metropolis et al. [14] in their original work and it is a well known technique in Monte Carlo simulation [16,17,18].

The other proofs of the convergence of **SA**, with the only exception of the the work by Rossier et al. [71], follow a different approach: A number of assumptions are placed on the acceptance function and on the generation rule and it is proved that the resulting stationary probability distribution satisfies (3.23).

In particular Aarts and van Laarhoven [68] and Otten and van Ginneken [67] assume that G is symmetric (See equation (3.29)) and that f satisfies the following:

i) For all i, j, k such that $c_i \leq c_j \leq c_k$

$$f_T(\Delta c_{ij}) f_T(\Delta c_{jk}) = f_T(\Delta c_{ik}) \, .$$

ii) If $\Delta c \leq 0$

$$f_T(\Delta c) = 1 \, .$$

iii) If $\Delta c > 0$

$$0 < f_T(\Delta c) < 1 ,$$
$$\lim_{T \to 0} f_T(\Delta c) = 0 .$$

With these assumptions and using the detailed balance equation (3.11), the solution to (3.26) becomes

$$\pi_i(T) = \frac{f_T(\Delta c_{i^* i})}{\sum_{j \in \Omega} f_T(\Delta c_{i^* j})} , \tag{3.35}$$

where $i^* \in \Omega_*$.

Lundy and Mees [56] relax the condition on the generation rule (See equation (3.29)) a little to obtain the following

$$G_{ij} = \begin{cases} 1/|\Omega_i| & \text{if } \ j \in \Omega_i , \\ 0 & \text{otherwise} . \end{cases} \tag{3.36}$$

The use of (3.36) instead of (3.29) modifies (3.35) to become

$$\pi_i(T) = \frac{|\Omega_i| f_T(\Delta c_{i^* i})}{\sum_{j \in \Omega} |\Omega_j| f_T(\Delta c_{i^* j})} . \tag{3.37}$$

Note that if (3.29) and (3.36) are used together, the size of the neighbor set becomes constant, i.e. $\mathcal{G}$ becomes a regular graph in which each node has degree $|\Omega_i|$, and (3.37) reduces to (3.35).

Anily and Federgruen [55] use a generation rule similar to (2.6) but the dependence on T is dropped and $\hat{G}_{ij}$ is assumed symmetric. With this selection of the generation rule we have the following expression for the stationary probability distribution

$$\pi_i(T) = \frac{G_i f_T(\Delta c_{i^* i})}{\sum_{j \in \Omega} G_j f_T(\Delta c_{i^* j})} .$$

Faigle and Schrader [72] follow a rather different approach which leads to a different convergence proof. The main result is summarized in the following proposition:

Proposition 3.3.1 *Let G_{ij} be such that (3.29) is satisfied and $\mathcal{G}$ is connected. Let $h_T(c_i, c_j)$ be a real function such that:*

i) For all i, j

$$h_T(c_i, c_j) > 0 \,,$$
$$h_T(c_i, c_l)\, h_T(c_l, c_j) = h_T(c_i, c_j) \,.$$

ii) For all i, j *such that* $c_i < c_j$

$$h_T(c_i, c_j) < 1 \,,$$
$$\lim_{T \to 0} h_T(c_i, c_j) = 0 \,,$$

then there exist a constant $g \geq 0$ *which depends on* G *only so that, for all* $j \in \Omega$

$$\pi_j(T) \leq g\, h_T(c_*, c_j) \,.$$

□

Proposition 3.3.1 extends the family of functions that can be used as acceptance functions. For example it is possible to use the following

$$h_T(c_i, c_j) = \frac{c_i}{c_j} T^{-(c_j - c_i)} \,,$$

which accounts for c_i and c_j explicitly and not for Δc_{ij} only. Of course the choice of h_T will determine the expression for π together with the rate at which the memory of initial condition is lost.

Chapter 4

Asymptotic Behavior: Inhomogeneous Theory

In the previous chapter we have considered the case in which, for each value of the temperature, the Markov chain is allowed to reach the stationary probability distribution $\pi(T)$ and then behavior of $\pi(T)$ as T approaches zero is studied. In other words a number of conditions on the Markov chain have been established for the following to be true

$$\lim_{T\to 0}\left(\lim_{m\to\infty}\Pr\{\, X_m = i \,\}\right) = \lim_{T\to 0}\pi_i(T) = \begin{cases} l(i,|\Omega_*|) & \text{if } i \in \Omega_*\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{4.1}$$

The function l depends on the assumptions made on the generation and acceptance functions. Note that the order in which the limits in (4.1) are taken is essential to the validity of the result. Furthermore, since the Markov chain is required to achieve stationarity at each value of the temperature, no restrictions are placed on the sequence of temperatures as long as they satisfy the conditions of Assumption 3.2.1.

In this chapter we study the convergence of the algorithm to the global optimum from a different perspective. We will drop the requirement that the Markov chain reaches stationarity at each value of T. In practice, we will require that only a finite number of iterations is performed by the algorithm at each value of T and we will determine what are the conditions under which the algorithm converges to the global optimum.

Of course, if the Markov chain does not reach stationarity at each value of T, the dependence of the probability transition matrix $\mathbf{P}$ on T cannot be disregarded and the homogeneous model has to be replaced with an inhomogeneous one.

In the remaining of the present chapter, we will restrict ourselves to instances of SA for which the following assumption holds:

Assumption 4.0.1 *The generate function G is independent of T and symmetric: i.e. $\hat{G}_{ij} = \hat{G}_{ji}$. The acceptance function is as in (2.3).*

□

Note that Assumption 4.0.1 is equivalent to assume that the Markov chain is reversible.

4.1 Inhomogeneous Markov Chains

In this section, akin to the analogous section for homogeneous Markov chains, definitions and fundamental results from the theory of inhomogeneous Markov chains are recalled. Theorems are reported without proof which can be found in standard texts on inhomogeneous Markov chains like [61,63,73].

Let us start with the definition of convergence for inhomogeneous Markov chains. Intuitively, the inhomogeneous Markov chain will converge if, for any initial probability distribution vector $\mathbf{p}(0)$,

$$\lim_{m \to \infty} \|\mathbf{p}(m) - \mathbf{e}_*\| = 0 , \tag{4.2}$$

where $\mathbf{p}(m)$ denotes the state probability vector after m transitions of the Markov chain are performed and $\mathbf{e}_*$ is defined by (3.24). Note that in this case T is not fixed. Hence the entries of the transition probability matrix are functions of the parameter T and the process is non stationary. For this reason condition (4.2) has to be replaced by the more precise requirement that the chain be *strongly ergodic* as stated in the following definition:

Definition 4.1.1 (Strong Ergodicity) *An inhomogeneous Markov chain is strongly ergodic if there exists a constant probability vector* $\mathbf{e}_*$ *such that for all* m

$$\lim_{k\to\infty} \sup_{\mathbf{p}(0)} \| \mathbf{p}(m,k) - \mathbf{e}_* \| = 0 ,$$

where

$$\mathbf{p}(m,k) = \mathbf{p}(0)\mathbf{P}(m,k) . \tag{4.3}$$

□

To use this definition to check ergodicity is impractical. A more viable strategy is to determine first whether the Markov chain satisfies a weaker form of ergodicity and then combine weak ergodicity with further conditions to achieve strong ergodicity.

Definition 4.1.2 (Weak Ergodicity) *An inhomogeneous Markov chain is weakly ergodic if, for all* m,

$$\lim_{k\to\infty} \sup_{\mathbf{p}(0),\mathbf{q}(0)} \| \mathbf{p}(m,k) - \mathbf{q}(m,k) \| = 0 , \tag{4.4}$$

where $\mathbf{p}(0)$ *and* $\mathbf{q}(0)$ *are two arbitrary initial state probability vectors with* $\mathbf{p}(m,k)$ *and* $\mathbf{q}(m,k)$ *defined as in (4.3).* □

Notice how the definition of weak ergodicity is significantly less restrictive than strong ergodicity. Weak ergodicity implies only "loss of memory" of the initial condition. Strong ergodicity instead implies in addition to "loss of memory", convergence to a constant vector.

Example 4.1.1 [Isaacson and Madsen] Consider a Markov chain with the transition probability matrix $\mathbf{P}(m)$ given by

$$\mathbf{P}(2m-1) = \begin{pmatrix} 1-\frac{1}{2m-1} & \frac{1}{2m-1} \\ 1-\frac{1}{2m-1} & \frac{1}{2m-1} \end{pmatrix}$$

and by

$$\mathbf{P}(2m) = \begin{pmatrix} \frac{1}{2m} & 1-\frac{1}{2m} \\ \frac{1}{2m} & 1-\frac{1}{2m} \end{pmatrix} .$$

Consider an arbitrary initial probability vector $\mathbf{p}(0)$. $\mathbf{p}(m,k)$ is given by

$$\mathbf{p}(m,k) = \begin{cases} (1-\frac{1}{k} \quad \frac{1}{k}) & \text{if } k \text{ is odd,} \\ (\frac{1}{k} \quad 1-\frac{1}{k}) & \text{if } k \text{ is even.} \end{cases}$$

The Markov chain is weakly ergodic but it is not strongly ergodic.

Consider now another Markov chain whose transition probability matrix $\mathbf{P}(m)$ is given by

$$\mathbf{P}(m) = \begin{pmatrix} \frac{1}{2} - \frac{1}{m+1} & \frac{1}{2} + \frac{1}{m+1} \\ \frac{1}{2} - \frac{1}{m+1} & \frac{1}{2} + \frac{1}{m+1} \end{pmatrix} .$$

As in the previous case take an arbitrary initial probability vector $\mathbf{p}(0)$ and compute $\mathbf{p}(m,k)$. In this case we obtain

$$\mathbf{p}(m,k) = (\frac{1}{2} - \frac{1}{k+1} \quad \frac{1}{2} + \frac{1}{k+1}) .$$

Then for all m

$$\lim_{k\to\infty} \mathbf{p}(m,k) = (\frac{1}{2} \quad \frac{1}{2}) .$$

Hence the Markov chain is strongly ergodic [1]. □

The definition of weak ergodicity is more tractable than the definition of strong ergodicity but it remains practically unusable to check whether a Markov chain is weakly ergodic.

Weak ergodicity is equivalent to loosing memory and from the results on homogeneous Markov chain it follows that memory is lost once the rows of the transition probability matrix $\mathbf{P}(m)$ become all equal, or, in other words, when $\mathbf{P}(m)$ becomes equal to $\Pi(m)$ for all m. In the inhomogeneous case, we allow only one transition for each value of m and hence $\mathbf{P}(m)$ never approaches the corresponding $\Pi(m)$. On the other hand, what is really of interest to us is that the sequence of matrices $\mathbf{P}(m,k)$ approaches, in the sense of (4.4), a constant matrix Π_* whose rows are all equal. To measure how far is $\mathbf{P}$ from Π_*, we could use the norm of the difference between the two matrices. However this is impractical since it requires

[1] Note that in both the Markov chains memory is actually lost in only one step. This is due to the structure of the probability transition matrix which has all the rows which are equal (See Theorem 3.2.4).

that we already know the expression for the entries of Π_* and this is out of the question in most of the cases. Alternatively we can assume that $\mathbf{P}(m,k)$ converges to an unknown constant matrix and use the following measure of the distance

Definition 4.1.3 (Ergodic Coefficient) *Given an inhomogeneous probability transition matrix* $\mathbf{P}$*, its ergodic coefficient is defined by*

$$\begin{aligned}\tau(\mathbf{P}) &= \frac{1}{2}\max_{ij}\sum_{k=1}^{|\Omega|}|P_{ik}-P_{jk}| = \\ &= 1 - \min_{ij}\sum_{k=1}^{|\Omega|}\min(P_{ik},P_{jk})\,. \end{aligned} \tag{4.5}$$

□

$\tau(\mathbf{P})$ is a positive function defined on the set of stochastic matrices and is called the *ergodic coefficient* for the stochastic matrix $\mathbf{P}$. Several slightly different versions of the ergodic coefficient are available in the literature of inhomogeneous Markov chains [61,73]. The definition given by (4.5) is the one given by Seneta in [63] [2].

From Definition 4.1.3 we have the following theorems that relate ergodicity coefficients and norms for stochastic matrices [61,63].

Theorem 4.1.1 *If* $\mathbf{P}$ *is any stochastic matrix and* $\mathbf{Q}$ *is any matrix such that for all* i

$$\sum_j Q_{ij} = 0\,,$$

then

$$\|\mathbf{QP}\| \le \|\mathbf{Q}\|\tau(\mathbf{P})\,.$$

Theorem 4.1.2 *Let* $\mathbf{P}$ *and* $\mathbf{Q}$ *be two stochastic matrices, then*

$$\tau(\mathbf{QP}) \le \tau(\mathbf{Q})\tau(\mathbf{P})\,.$$

□

[2] Dobrushin [74] has been the first to define and apply ergodic coefficients. In his paper, Dobrushin called $1-\tau(\mathbf{P})$ the ergodic coefficient.

The ergodic coefficient can be used to test weak ergodicity for a stochastic matrix as stated in the following theorem [61,63] :

Theorem 4.1.3 *A time inhomogeneous Markov chain is weakly ergodic if and only if there exists a sequence of integers* $\{ k_i \}$ *such that*

$$\sum_{i=0}^{\infty} [\, 1 \, - \, \tau(\mathbf{P}(k_i, k_{i+1}))] \, = \, \infty \; . \tag{4.6}$$

□

The last step is now to introduce a tool to check strong ergodicity. The strategy is to assume that the Markov chain is weakly ergodic and to add conditions on the limit vector to prevent oscillatory behavior like the one of the first Markov chain in Example 4.1.1. This is accomplished through the following theorem [61,75]

Theorem 4.1.4 *If for all m there exists a probability vector* $\pi(m)$ *such that*

$$\pi(m)\mathbf{P}(m) \, = \, \pi(m) \, ,$$

and

$$\sum_{m=0}^{\infty} \| \, \pi(m) \, - \, \pi(m+1) \, \| \, \leq \, \infty \, ,$$

and the Markov chain is weakly ergodic, then it is also strongly ergodic. Moreover if

$$\lim_{m \to \infty} \pi(m) \, = \, \mathbf{e}_* \, ,$$

then for all m

$$\lim_{k \to \infty} \sup_{\mathbf{p}(0)} \| \, \mathbf{p}(m,k) \, - \, \mathbf{e}_* \, \| \, = \, 0 \, .$$

□

Note that all the conditions given in this section are sufficient conditions for strong ergodicity and hence for convergence according to (4.2).

4.2 Asymptotic Properties of Simulated Annealing

In this section we will use the results of the previous section to prove convergence for **SA** once a monotonic sequence $\{T_k\}$ is used.

The idea is to determine the condition under which the Markov chain describing the behavior of **SA** is weakly ergodic. Then we will apply Theorem 4.1.4 to prove strong ergodicity.

4.2.1 Bounds

Before we start proving the convergence results we need to establish few bounds related to the connectivity of the $\mathcal{G}$, to the shape of the cost function c, and to the generate function G.

The first two bounds are related to the topology of $\mathcal{G}$. Define

$$r = \min_{i \in (\Omega \cap \Omega_M)^c} \max_{j \in \Omega} d(i,j) \tag{4.7}$$

to be the *radius* of the graph, where $d(i,j)$ is the *distance* of j from i measured as the length (number of edges) of the minimum length path from i to j in $\mathcal{G}$, and Ω_M, the set of local maxima, is defined by

$$\Omega_M = \{\, i : c_i \geq c_j \text{ for all } j \in \Omega_i \,\} \ .$$

Let $\hat{\imath}$ be the index of the vertex where the minimum in (4.7) is attained. The vertex $\hat{\imath}$ is called the *center* of $\mathcal{G}$. The radius r represents an upper bound on the number of transitions that have to be performed on the Markov chain so that there exists at least one column of $\mathbf{P}$, namely column $\hat{\imath}$, with all its entries strictly greater than zero. Note that the radius is well defined since it is assumed that $\mathcal{G}$ is connected. Furthermore, because of the symmetry of G_{ij}, $\mathcal{G}$ is also strongly connected. The second bound is a lower bound on the degree of $\mathcal{G}$ and it is defined as follows:

$$n_0 = \min_i |\Omega_i| \ .$$

The third bound is related to the cost function c and is an upper bound on the local slope of the cost function. Its expression is given by

$$L = \max_{i \in \Omega} \max_{j \in \Omega_i} |c_j - c_i|. \tag{4.8}$$

L plays the role of a Lipschitz-like constant.

Finally the fourth bound is a lower bound on the generation function

$$w = \min_{i \in \Omega} \min_{j \in \Omega_i} \frac{\hat{G}_{ij}}{G_i}. \tag{4.9}$$

4.2.2 Estimation of the Coefficient of Ergodicity

Consider a pair of states (i, j) that are neighbors in $\mathcal{G}$, i.e. $j \in \Omega_i$. From (2.3), (4.8), and (4.9), for all m

$$P_{ij}(m) \geq w \ \exp\Big(-\frac{L}{T_m} \Big) .$$

The entries on the diagonal of $\mathbf{P}(m)$, with the only exception of the entries relative to states that are local maxima, i.e. the states of set Ω_M, are monotonically increasing functions of m. To see this, recall the definition of $P_{ii}(m)$ and rewrite equation (2.8) as follows

$$P_{ii}(m) = 1 - \sum_{\substack{j: c_j \leq c_i \\ j \in \Omega_i}} G_{ij} - \sum_{\substack{j: c_j > c_i \\ j \in \Omega_i}} G_{ij} \ \exp\Big(-\frac{c_j - c_i}{T_m} \Big) . \tag{4.10}$$

The contribution of the first summation, i.e. the contribution of "down-hill" transitions, is independent of m. The contribution of the second summation, i.e. the contribution of "up-hill" transitions, is monotonically decreasing with m. Equation (4.10) can be rewritten as

$$\begin{aligned} P_{ii}(m) &= \sum_{\substack{j: c_j > c_i \\ j \in \Omega_i}} G_{ij} \Big\{ 1 - \exp\Big(-\frac{c_j - c_i}{T_m} \Big) \Big\} \geq \\ &\geq n_0 w \Big\{ 1 - \exp\Big(-\frac{L}{T_m} \Big) \Big\} . \end{aligned} \tag{4.11}$$

From (4.11) follows that there exist an index m_0 and a corresponding temperature T_{m_0}

$$T_{m_0} < -\frac{L}{\ln \frac{n_0}{n_0 + 1}} ,$$

such that for all $m \geq m_0$ and for all $i \in (\Omega \cap \Omega_m)^c$

$$P_{ii}(m) > w \ \exp\Big(-\frac{L}{T_m} \Big). \tag{4.12}$$

Equations (4.7) and (4.12) can be used to find a bound for $P_{ii}(m-r, m)$. In fact for every $m \geq m_0$ and for every $i \in \Omega$

$$\begin{aligned} P_{ii}(m-r,m) \ &\geq \ \prod_{l=m-r}^{m-1} w \ \exp\Big(-\frac{L}{T_l} \Big) \geq \\ &\geq \ w^r \ \exp\Big(-\frac{L}{T_{m-1}} \Big). \end{aligned} \tag{4.13}$$

To ease the notation, from now on we shall abbreviate $\tau(\mathbf{P}(m,k))$ to $\tau(m,k)$. If (4.13) is plugged into the expression for the ergodic coefficient given by (4.5) we have

$$\begin{aligned} \tau(kr-r, kr) \ &= \ 1 - \min_{ij}\{ \ \min(P_{ii}, P_{ji}) \ \} \ \leq \\ &\leq \ 1 - w^r \ \exp\Big(-\frac{rL}{T_{kr-1}} \Big), \end{aligned} \tag{4.14}$$

for all $k \geq k_0 = m_0/r$.

4.2.3 Weak Ergodicity

We are now ready to apply Theorem 4.1.3 to state a sufficient condition which guarantees that the Markov chain associated with SA is weakly ergodic. In fact with the estimation (4.14), condition (4.6) becomes

$$\sum_{k=k_0}^{\infty} w^r \exp\Big(-\frac{rL}{T_{kr-1}} \Big) = \infty. \tag{4.15}$$

Note that up to this point the sequence of parameters $\{ \ T_m \ \}$ had only to comply with the conditions of Assumption 3.2.1. In particular the dependency of T_m on m has not been specified. Equation (4.15) introduces a condition on the update function that combined with the assumption that at least one move is attempted at each value of T_m allow us to prove the following:

Theorem 4.2.1 (Sufficient Condition) *The inhomogeneous Markov chain associated with* **SA** *with the following update function*

$$T_m = \frac{\gamma}{\log(m + m_0 + 1)}, \tag{4.16}$$

where $m = 1, 2, \ldots$ *and* m_0 *is any parameter such that* $0 \leq m_0 < \infty$, *is weakly ergodic if*

$$\gamma \geq rL \ .$$

Proof. Substituting (4.16) in (4.6) we have, for all $k \geq k_0$

$$\tau(kr - 1, kr) \leq 1 - \frac{a}{(k + k_0)^{\frac{rL}{\gamma}}} \tag{4.17}$$

where a is given by

$$a = \frac{w^r}{r^{rL/\gamma}} \ . \tag{4.18}$$

It is obvious that, for any l

$$\sum_{k=l}^{\infty} (1 - \tau(kr - r, kr)) = \infty \ ,$$

if $rL/\gamma \leq 1$. The application of Theorem 4.1.3 completes the proof. □

It is worth noticing that the result we just proved, retains its validity even if more than one transition is performed at each value of T. The only difference will be in the values of the constants in (4.16) and not in the logarithmic law.

Moreover note that the logarithmic law has a major drawback for practical applications of the schedule (4.16). In fact the logarithmic law requires that an infinite sequence of temperatures has to be generated. Furthermore, since the logarithmic rule is deduced from the sufficient condition for weak ergodicity (4.15), it is immediate to see that the tail of the sequence of temperatures is essential to guarantee weak ergodicity. Hence there is no reason to believe that, if only a finite number of temperatures are allowed, the sequence should be extracted from (4.16). In Chapter 7 it will be shown that, at least in one case, the optimal finite sequence of temperatures is not patterned after (4.16).

4.2.4 Properties of the Stationary Probability Distribution

In this section we present two useful properties of the stationary probability distributions $\pi(T)$ that are necessary to the proof of strong ergodicity in the next section and to the analysis of finite time behavior in a later chapter. In particular we are interested in the behavior of $\pi(T)$ as a function of the parameter T.

In Chapter 3 we have already shown that as T approaches zero, $\pi(T)$ has a limit which is given by the optimum vector $\mathbf{e}_*$. In this section we will investigate the monotonicity of the convergence.

In view of Assumption 4.0.1, the expression for π becomes

$$\pi_i(T) = \frac{G_i e^{-\frac{c_i}{T}}}{Z(T)} \tag{4.19}$$

with $Z(T)$ an appropriate scale factor.

Let us define the average value and the variance of the cost with respect to the distribution $\pi(T)$ as follows:

$$C(T) = \sum_{j \in \Omega} c_j \, \pi_j(T) \,, \tag{4.20}$$

$$\sigma_C^2(T) = \sum_{j \in \Omega} (c_j - C(T))^2 \, \pi_j(T) \,.$$

From the definition of $\pi(T)$ it is immediate to compute its derivative with respect to T

$$\frac{d\pi_i(T)}{dT} = \frac{\pi_i(T)}{T^2} \{ \, c_i - C(T) \, \} \,. \tag{4.21}$$

Similarly, from (4.20), the expression for the derivative of $C(T)$ with respect to T is given by

$$\frac{dC(T)}{dT} = \frac{1}{T^2} \sigma_C^2(T) \,. \tag{4.22}$$

Define the set Ω_C as follows

$$\Omega_C = \{ \, i \, : \, c_i > \sup_T C(T) \, \} \,.$$

With the above definitions we can now prove the following proposition:

Proposition 4.2.1 *Consider π_i as a function of T. Three cases are possible:*

i) If $i \in \Omega_*$ *then for all* $m \geq 0$

$$\pi_i(T_{m+1}) - \pi_i(T_m) > 0 \ ;$$

ii) If $i \in \Omega_C$ *then for all* $m \geq 0$

$$\pi_i(T_{m+1}) - \pi_i(T_m) < 0 \ ;$$

iii) If $i \in (\Omega_* \cup \Omega_C)^c$ *, then there exists an unique integer* $\hat{m}_i$, $0 \leq \hat{m}_i < \infty$ *such that for all* $m \geq \hat{m}_i$

$$\pi_i(T_{m+1}) - \pi_i(T_m) < 0 \ ,$$

and for all $0 \leq m \leq \hat{m}_i - 1$

$$\pi_i(T_{m+1}) - \pi_i(T_m) > 0 \ .$$

Proof. From the expression for $d\pi_i(T)/dT$, the sign of the derivative evaluated at T is determined by the sign of $c_i - C(T)$. From (4.22), $C(T)$ is monotonically decreasing with T. Hence $C(T)$ attains its extrema at the extrema of the set on which it is defined, namely

$$\sup_T C(T) = \lim_{T \to \infty} C(T) = C(\infty),$$

and

$$\inf_T C(T) = \lim_{T \to 0} C(T) = c_* \ .$$

Point *i)* of the proposition is proven once we recognize that $i \in \Omega_*$ implies $c_i \leq C(T)$ for every T. Similarly point *ii)* follows from $c_i > C(T)$ for every T. Finally the monotonicity of $C(T)$ guarantees that there exist a unique value of T such that $c_i = C(T)$. Combining this with the monotonicity of the sequence $\{T_m\}$, there exists a unique value $\hat{m}_i$ which satisfies point *iii)*. □

It is immediate to see that a corollary to Proposition (4.2.1) guarantees the existence of a unique $\tilde{m} < \infty$ such that for all $i \in (\Omega \cap \Omega_*)^c$, for every $m \geq \tilde{m}$

$$\pi_i(T_{m+1}) - \pi_i(T_m) < 0 \ . \tag{4.23}$$

$\tilde{m}$ is defined obviously by

$$\tilde{m} = \max_{i \in (\Omega \cap \Omega_*)^c} \hat{m}_i \tag{4.24}$$

$\tilde{m}$ marks the onset of the monotonic decrease of $\pi_i(T)$ for all but the least cost configurations and is fundamental to establish finite time rate of convergence of π to $\mathbf{e}_*$. To determine its value we have to study, for all $i \in (\Omega_* \cup \Omega_C)^c$, the behavior of the value of T, which solves the equation $c_i - C(T) = 0$, as a function of c_i.

Proposition 4.2.2 *For all $i \in (\Omega_* \cup \Omega_C)^c$, let T_i be the value of T which solves $c_i - C(T) = 0$. T_i is monotonic, strictly increasing with increasing c_i.*

Proof. Consider two states $i, j \in (\Omega_* \cup \Omega_C)^c$ and such that $c_j < c_i$. Rewrite (4.21) as follows:

$$\frac{d\pi_i(T)}{dT} = \frac{\pi_i(T)}{T^2}\{\, c_i - c_j \,\} + \frac{\pi_i(T)}{\pi_j(T)} \frac{d\pi_j(T)}{dT}$$

and evaluate it for $T = T_j$. From the definition of T_j and from (4.21), the second term of the right hand side is zero. Hence

$$\left.\frac{d\pi_i(T)}{dT}\right|_{T=T_j} > 0$$

and again from (4.21) and (4.22), $T_i < T_j$. □

From the monotonicity of T_i, the solution to (4.24) is obtained by solving

$$c_{\tilde{i}} - C(T) = 0 \tag{4.25}$$

where

$$c_{\tilde{i}} = \min_{i \in (\Omega_* \cap \Omega)^c} c_i \,.$$

Equation (4.25) can be rewritten as follows

$$\delta \sum_{i \in \Omega_*} G_i - \sum_{i : c_i > c_{\tilde{i}}} G_i (c_i - c_{\tilde{i}}) \exp\left(-\frac{c_i - c_*}{T} \right) = 0 \tag{4.26}$$

where

$$\delta = c_{\tilde{i}} - c_* \,, \tag{4.27}$$

and c_* is the cost of any of the states in Ω_*. In conclusion $\tilde{m}$ is the smallest integer such that $T_{\tilde{m}} \leq T_i$ where T_i is the solution to (4.26).

Summarizing the quasi-stationary probability distribution $\pi_i(T)$ converges with decreasing temperature (i.e. increasing time) to the optimum vector. The quasi-stationary probabilities of the least-cost configurations monotonically increase with decreasing temperature. For configurations with costs not less than the weighted mean cost, the opposite is true. Each configuration i with cost between least-cost c_* and $C(\infty)$ has an associated "critical temperature" T_i ; while the temperature is greater than T_i, the quasi-stationary probability of the configuration increases with decreasing temperature, and for temperatures less than T_i the opposite is true. Furthermore, the critical temperature is an increasing function of cost. All of the above properties hold for any sequence $\{T_m\}$ which satisfies Assumption 3.2.1.

4.2.5 Strong Ergodicity

To prove strong ergodicity it suffices to show that that **SA** satisfies the conditions of Theorem 4.1.4. This is done in the following proposition

Proposition 4.2.3 *For any sequence $\{T_k\}$ which satisfies Assumption 3.2.1 the corresponding quasi-stationary probabilities are such that*

$$\sum_{m=0}^{\infty} \|\pi(m+1) - \pi(m)\| \leq 2(\tilde{m}-1) < \infty ,$$

where $\tilde{m}$ is given in (4.24).

Proof. From statement *i)* of Proposition 4.2.1, and (4.23), for $m \geq \tilde{m}$

$$\begin{aligned} \|\pi(m+1) - \pi(m)\| &= \sum_{i\in\Omega_*} \{\, \pi_i(m+1) - \pi_i(m) \,\} - \sum_{i\in(\Omega_*\cap\Omega)^c} \{\, \pi_i(m+1) - \pi_i(m) \,\} \\ &= \sum_{i\in\Omega_*} \{\{\, \pi_i(m+1) - \pi_i(m) \,\} - \{\, 1 - \pi_i(m+1) - 1 + \pi_i(m) \,\}\} \\ &= 2 \sum_{i\in\Omega_*} \{\, \pi_i(m+1) - \pi_i(m) \,\} \ . \end{aligned} \tag{4.28}$$

By (4.28), follows

$$\sum_{m=\tilde{m}}^{\infty} \|\pi(m+1) - \pi(m)\| \leq 2 \ .$$

which combined with the observation that for all $0 \leq m \leq \tilde{m}$

$$\|\pi(m+1) - \pi(m)\| \leq 1$$

completes the proof. □

Using Theorem 4.2.1 and Theorem 4.1.4 we can now prove the central result of this chapter:

Theorem 4.2.2 (Strong Ergodicity) *The inhomogeneous Markov chain associated with* **SA** *is strongly ergodic if it is weakly ergodic and the annealing schedule satisfies (4.16). In this case for all m*

$$\lim_{k\to\infty} \sup_{\mathbf{p}(0)} \| \mathbf{p}(m,k) - \mathbf{e}_* \| = 0 \ . \tag{4.29}$$

In particular, the annealing schedule in (4.16) with $\gamma \geq rL$ gives a strongly ergodic Markov Chain for which (4.29) holds.

4.3 Related Work

Several different proofs of strong ergodicity for the Markov chain describing **SA** have become available. The approaches followed by the various authors are rather different and the objective of the present section is to compile a short survey of them.

4.3.1 Sufficient Conditions

S. Geman and D. Geman were the first to find sufficient conditions for strong ergodicity of **SA** through an elaborate analysis based on the theory of Markov fields instead of Markov chains [39]. The strategy followed to prove strong ergodicity is similar to the one presented in the previous section. The bound on the ergodic coefficient found in [39] leads to a slightly different expression for the update function given by

$$T_m \geq \frac{|\Omega|\Delta c_{max}}{\log(m+m_0)}, \tag{4.30}$$

where

$$\Delta c_{max} = \max_{\omega_i \in \Omega} c_i - \min_{\omega_j \in \Omega} c_j \ .$$

Anily and Federgruen [55,76] find a sufficient condition for strong ergodicity in a more general setting in which the acceptance function f_T is not constrained to be as in (2.3) but belongs to a more general class which includes acceptance functions of the type introduced by Hastings [77] and Bohachevsky et al. [78] [3]. The update function found by Anily and Federgruen is given by

$$T_m \geq \frac{nL}{\log(m + m_0)} \tag{4.32}$$

where L is given by (4.8) and n is the distance, measured in number of edges, from any state to one of the states in Ω_*. The form of (4.32) is the same as the one found by the other authors but the constant that appears at the numerator of the fraction is larger or equal to the one found in the previous section but smaller or equal to the one by Geman and Geman.

Gelfand and Mitter [58] have generalized the set up of the problem a little further. In fact instead of finding the condition for convergence to the optimal set Ω_* they derived the sufficient conditions for convergence to any arbitrary subset $\Omega_A \subset \Omega$. The update function derived by Gelfand and Mitter is given by

$$T_m \geq \frac{h_A}{\log(m + m_0)} \ . \tag{4.33}$$

To define h_A we need the following digression: Let $i = i_0, i_1, \ldots, i_p = j$ be a path of length p which connects state i to state j. Let $\Lambda_{i \to j}$ be the set of all paths of finite length which connects state i to state j and define the height h_{ij} of the connection between i and j as

$$h_{ij} = \min_{\Lambda_{i \to j}} \sum_{k=1}^{p} \max(0, c_{i_{k+1}} - c_{i_k}) \ . \tag{4.34}$$

[3] It is interesting to note how this extended class contains acceptance functions of the form

$$f_T(c_i, c_j) = \begin{cases} \exp\left(-\frac{(c_j - c_i)c_i^g}{T}\right) & \text{if } c_j > c_i \\ 1 & \text{otherwise} \end{cases} \tag{4.31}$$

where g is an arbitrary negative number. Notice that, unlike acceptance function such as (2.3), (4.31) dependes on both c_i and c_j and not only on their difference. In Section 6.1.2 the behavior of acceptance function (4.31) is compared with the behavior of (2.3).

We are now ready to define h_A as follows

$$h_A = \min_{\substack{i \in \Omega_A \\ j \in (\Omega_A)^c}} h_{ij} \ .$$

In words, h_A is the height of the lowest hill that the algorithm has to climb to go from any of the states in Ω_A to any of the states in Ω_A^c.

Note that regardless the similarity of the update function (4.33) with the analogous functions reported in (4.16), (4.30), and (4.32) the constant at the numerator of (4.33) has a different and perhaps more intuitive origin. The strategy followed by Gelfand and Mitter in their proof requires to find a bound on the probability that a particular set is exited by the algorithm and then determine the form for the update schedule that makes the escape probability zero as T goes to zero. This is the reason why a constant of the form h_A replaces constant of the type rL.

Finally, for the sake of completeness, we mention a third different approach followed to prove sufficient conditions for strong ergodicity due to Holley and Stroock [79]. A schedule of the type of (4.33) is presented in a more general context in which the state space does not have to be finite and the Markov Process is a continuous function of T.

4.3.2 Necessary and Sufficient Conditions

There are, to the best of my knowledge, three authors that proved necessary and sufficient conditions for strong ergodicity of SA. Here we will present, in some details, the conditions as derived by Hajek [54] and we will mention briefly the procedures due to the other authors. To get started, we need a few definitions.

Definition 4.3.1 *Given* (Ω, c) *with a generation function* G

i) A cup is defined as the set C *of states such that for some number* h *the following is true:*

$$C = \{\, j : j \textit{can be reached at height } h_{ij} \leq h \textit{ from } i \,\}$$

for all $i \in C$.

ii) The depth of the cup C is defined by

$$d(C) = \overline{c}(C) - \underline{c}(C)$$

where $\overline{c}(C)$ *is defined by*

$$\overline{c}(C) = \min\{\, c_j : j \in (\Omega_i \cup C^c),\ i \in C \,\}$$

and $\underline{c}(C)$ *by*

$$\underline{c}(C) = \min\{\, c_j : j \in C \,\} \ .$$

Definition 4.3.2 *A Markov chain is weakly reversible if for any pair of states i and j and for any real number h, i is reachable at height h from j if and only if j is reachable at height h from i.* □

The necessary and sufficient condition for strong ergodicity proven by Hajek is stated in the following:

Theorem 4.3.1 *Assume that the Markov chain is irreducible, weakly reversible, and that at least one iteration is performed at each temperature then:*

i) If $j \in (\Omega \cap \Omega_m)^c$, *i.e. j is not a local minima for c (See equation (2.2))*

$$\lim_{m\to\infty} \Pr\{\, X_m = j \,\} = 0 \ .$$

ii) If $j \in \Omega_m$, *i.e. j is a local minima for c*

$$\lim_{m\to\infty} \Pr\{\, X_m = j \,\} = 0$$

if and only if

$$\sum_{m=1}^{\infty} \exp\Big(-\frac{d_j}{T_m}\Big) = \infty$$

where d_j *is the depth of the cup that contains j.*

iii) i) and ii) imply

$$\lim_{m\to\infty} \Pr\{\, X_m \in \Omega_* \,\} = 1$$

if and only if

$$\sum_{m=1}^{\infty} \exp\Big(-\frac{d_*}{T_m}\Big) = \infty \qquad (4.35)$$

where d_* *is the maximum of the depths of all the states that are local but not global minima.*

□

The divergence of the series (4.35) implies the following form for the annealing schedule:

$$T_m = \frac{d_*}{\log(m + m_0)} . \tag{4.36}$$

Once again, not surprisingly, the law of the schedule (4.36) is logarithmic. What makes condition (4.36) necessary and sufficient, as opposed to only sufficient, is the choice of the constant which appears at the numerator. The result by Hajek has been generalized a little further by Tsitsiklis [80] by eliminating the assumption on weak reversibility.

Gidas' [57] necessary and sufficient condition for strong ergodicity holds in a more general context in which the Markov chain is allowed to have more than one ergodic component. In particular, Gidas' main theorem says that a schedule which is necessary and sufficient has the following form

$$\frac{\frac{1}{C_0 + \delta}}{\log(m + m_0)} \leq T_m \leq \frac{\frac{1}{C_0}}{\log(m + m_0)} \tag{4.37}$$

for some $\delta > 0$. If there are at most two ergodic components, C_0 is defined as follows:

$$\frac{1}{C_0} = \min_{\gamma} h_\gamma ,$$

where $\Omega_\gamma, \gamma = 1, 2$ are the ergodic components of the Markov chain and h_γ is defined in (4.34).

If the number of ergodic components exceeds two, the schedule will have again the logarithmic form (4.37) but the value of the constant at the numerator will be unknown.

Gidas' extended his condition even further to include chains of the type

$$I_m = \frac{1}{m} \sum_{k=0}^{m} F(X_k) , \tag{4.38}$$

where F is any function defined on X_k. Extension (4.38) is important since it guarantees that, if the logarithmic rule to update T is used, I_m is a consistent estimator for $E_\pi[F]$, i.e. I_m converges in probability to $E_\pi[F]$ for any choice of

the initial probability distribution p(0) [81]. The properties of estimator I_m will be used in Section 6.1.1 to determine the best form for the acceptance function.

The third result is due to Connors and Kumar [82]. They follow a different approach based on the definition of the order of recurrence of a state ω_i

$$\beta_i = \sup_{c \geq 0} \sum_{k=0}^{\infty} e^{-c/T_k} \pi_i(T_k) = \infty$$

and the order of recurrence of a transition β_{ij}

$$\beta_{ij} = \sup_{c \geq 0} \sum_{k=0}^{\infty} e^{-c/T_k} P_{ij}(T_k) \pi_i(T_k) = \infty \ .$$

The order of recurrence of a state, β_i, serves the scope of a potential defined on Ω and has a number of interesting properties. In particular β_i is monotonically decreasing with the cost of the state. β_{ij}, the order of recurrence of a transition, satisfies the detailed balance equation on any edge of $\mathcal{G}$ and from this property necessary and sufficient conditions for strong ergodicity are derived. The actual result proven by Connors and Kumar is identical to the result proven by Hajek but the use of the orders of recurrence provides a proof which is more elegant and intuitive.

4.4 Concluding Remarks

In this chapter we have presented a sufficient condition for strong ergodicity of SA. The condition has been obtained with the assumption that the generating function is *symmetric* (See Assumption 4.0.1) and the acceptance function is as in (2.3). The procedure followed requires to find first conditions for weak ergodicity (i.e. loss of memory of the initial conditions) and then obtain strong ergodicity by adding to them conditions on the convergence of the quasi-asymptotic probability distribution.

The condition for weak ergodicity induced a constraint on the law implemented by the update function, namely

$$T_m = \frac{k}{\log(m + m_0)} \ , \tag{4.39}$$

with k being a constant dependent on the problem. The conditions for the convergence of the quasi-asymptotic probability distribution to the optimal vector simply require that the sequence $\{T_m\}$ is as in Assumption 3.2.1. Notice how the more restrictive condition on sequence $\{T_m\}$ is required by weak ergodicity while the effort to upgrade weak ergodicity to strong ergodicity is relatively minor.

This is not at all surprising. In fact if we recall the results of Chapter 3, it is immediate to see that we were already able to show the converge of π to $\mathbf{e}_*$, i.e. strong ergodicity, under the strong assumption that the Markov chain was allowed to reach stationarity at each value of T. To require that stationarity is reached at each value of T is equivalent to require that memory is lost at each value of T. The weak ergodicity result from the inhomogeneous theory goes a lot further. It ensures that, even if only one transition is performed at each value of T, memory is lost if the sequence $\{T_m\}$ is as in (4.39).

Finally we presented a short review of the most significant efforts that have been made to prove strong ergodicity of SA. The results obviously have the same logarithmic law (4.39) and differ only in the selection of the constant which appears at the numerator of (4.39). In particular the constant found by Gidas provides necessary and sufficient conditions for SA to be strongly ergodic with the less restrictive conditions on the Markov chain.

Chapter 5

Finite Time Behavior

In this chapter we will use the results derived in Chapter 4 to study the finite time behavior of the Markov chain describing SA. In particular it will be assumed that G and f are as in Assumption 4.0.1, while the update function will be the one defined by (4.16).

The main result of this chapter gives us an estimate of the departure of the state of the Markov chain from the optimum vector $\mathbf{e}_*$ at *finite* time m. The form of the results stated in Theorem 5.2.1 below gives comprehensive indications of the factors affecting the rate of convergence and their implications in the design of optimum annealing schedules.

5.1 Components of Finite-Time Behavior

To determine the finite behavior of the Markov chain, we have to study the behavior of $\|\mathbf{p}(m) - \mathbf{e}_*\|$ as a function of the iteration counter m. To proceed it is useful to decompose $\mathbf{p}(m) - \mathbf{e}_*$ as follows:

$$\begin{aligned} \mathbf{p}(m) - \mathbf{e}_* &= \{\, \mathbf{p}(m) - \boldsymbol{\pi}(0)\mathbf{P}(0,m) \,\} + \\ &+ \{\, \boldsymbol{\pi}(0)\mathbf{P}(0,m) - \boldsymbol{\pi}(m) \,\} + \\ &+ \{\, \boldsymbol{\pi}(m) - \mathbf{e}_* \,\}. \end{aligned} \tag{5.1}$$

Taking the norm of (5.1) we obtain

$$\begin{aligned} \|\mathbf{p}(m) - \mathbf{e}_*\| \;\leq\; & \|\mathbf{p}(m) - \boldsymbol{\pi}(0)\mathbf{P}(0,m)\| \;+ \\ + \;& \|\boldsymbol{\pi}(0)\mathbf{P}(0,m) - \boldsymbol{\pi}(m)\| \;+ \\ + \;& \|\boldsymbol{\pi}(m) - \mathbf{e}_*\| \;. \end{aligned} \tag{5.2}$$

Decomposition (5.2) can be interpreted as follows: The first two terms measure the rate at which the memory of the initial condition is lost. In particular, the first term monitors the distance of $\mathbf{p}(0)$ from $\boldsymbol{\pi}(0)$ as the number of transitions increases. The second term of decomposition (5.2) measures the influence, on the behavior of $\boldsymbol{\pi}(k)$, of the rate of convergence of $\mathbf{P}(0,m)$ to the constant matrix. To understand better the role of the second term consider the following situation: Assume that each one of matrices $\mathbf{P}(k,k+1)$, $k = 0,1,\ldots,m-1$ is constant in the sense that all the rows of $\mathbf{P}(k,k+1)$ are identical. (See e.g. transition matrices of Example 4.1.1). Then convergence of $\boldsymbol{\pi}(k)$ to $\boldsymbol{\pi}(k+1)$ will occur in only one transition, i.e. $\boldsymbol{\pi}(k)\mathbf{P}(k,k+r) = \boldsymbol{\pi}(k+r)$, $k = 0,1,\ldots,m-1, r = 1,2,\ldots,m$. In this case the second term of (5.2) would be identically zero. Finally the third term monitors the rate at which the asymptotic probability distribution $\boldsymbol{\pi}(m)$ converges to the optimal vector $\mathbf{e}_*$.

In the next sections, each one of the three terms in the right hand side is bounded independently.

5.1.1 Bound for the First Term

To determine a bound for the first term in the right hand side of (5.2), we need to recall two results of the theory of stochastic matrices presented in Section 4.1. In view of Theorem 4.1.1, for the first term of the right hand side of (5.2),

$$\begin{aligned} \|\mathbf{p}(kr) - \boldsymbol{\pi}(0)\mathbf{P}(0,kr)\| \;&=\; \|(\,\mathbf{p}(0) - \boldsymbol{\pi}(0)\,)\mathbf{P}(0,kr)\| \;\leq \\ &\leq\; \|\mathbf{p}(0) - \boldsymbol{\pi}(0)\|\tau(0,kr) \;. \end{aligned} \tag{5.3}$$

To complete the bound of the first term of (5.2) it is necessary to bound $\tau(0,kr)$. The following proposition gives the required bound:

Proposition 5.1.1 *If $\gamma \geq rL$ and the annealing schedule (4.16) is applied so that τ satisfies (4.17) then*

i) For $l \leq k_0 \leq k$

$$\tau(lr - r, kr) \leq \left(\frac{k_0 + m_0/r}{k + m_0/r}\right)^a , \tag{5.4}$$

ii) For $k_0 \leq l \leq k$

$$\tau(lr - r, kr) \leq \left(\frac{l + m_0/r}{k + m_0/r}\right)^a , \tag{5.5}$$

where a is defined by (4.18), r by (4.7), and k_0 is such that (4.11) holds. m_0 is the parameter that controls the initial value of T.

Proof. Let us define

$$\nu = \frac{rL}{\gamma}$$

By means of Theorem 4.1.2, from (4.17) we have for $k_0 \leq l \leq k$

$$\begin{aligned}
\tau(lr - r, kr) &\leq \prod_{m=l}^{k} \tau(mr - r, mr) \leq \\
&\leq \prod_{m=l}^{k} \left(1 - \frac{a}{(m + m_0/r)^\nu}\right) \leq \\
&\leq \exp\left(-a \sum_{m=l}^{k} \frac{1}{(m + m_0/r)^\nu}\right) \leq \\
&\leq \exp\left(-a \sum_{m=l}^{k} \frac{1}{(m + m_0/r)}\right) \leq \\
&\leq \left(\frac{l + m_0/r}{k + m_0/r}\right)^a .
\end{aligned}$$

A similar bound can be derived for $l \leq k_0 \leq k$. □

Bounds (5.4) and (5.5) are fundamental to the analysis of the finite-time behavior of SA. Substituting (5.4) and (5.5) in (5.3) yields for all $k \geq k_0$

$$\|\mathbf{p}(kr) - \pi(0)\mathbf{P}(0, kr)\| \leq \|\mathbf{p}(0) - \pi(0)\| \left(\frac{k_0 + m_0/r}{k + m_0/r}\right)^a . \tag{5.6}$$

5.1.2 Bound for the Second Term

Let us define for all $m = 1, 2, \ldots$ the row vector $\mu(m)$

$$\mu(m) = \pi(0)\mathbf{P}(0,m) - \pi(m) .$$

The sequence of vectors $\{ \mu(i) \}$ satisfy the recursion

$$\mu(m+r) = \mu(m)\mathbf{P}(m,m+r) + \sum_{s=1}^{r} \{ \pi(m+s-1) - \pi(m+s) \}\mathbf{P}(m+s,m+r).$$

Noticing that $\mu(0) = \mathbf{o}$, the recursion is solved to give

$$\mu(kr) = \sum_{l=1}^{k} \epsilon(lr)\mathbf{P}(lr,kr) , \tag{5.7}$$

where

$$\epsilon(lr) = \sum_{s=1}^{r} \{ \pi(lr-s) - \pi(lr-s+1) \}\mathbf{P}(lr-s+1,lr) . \tag{5.8}$$

Applying Theorem 4.1.1 twice to obtain bounds for both $\|\epsilon(lr)\mathbf{P}(lr,kr)\|$ and $\|\epsilon(lr)\|$, from (5.7) and (5.8) respectively we obtain, for $k \geq 1$,

$$\|\mu(kr)\| \leq \sum_{l=1}^{k} \tau(lr,kr) \sum_{s=1}^{r} \|\pi(lr+1-s) - \pi(lr-s)\|. \tag{5.9}$$

Now making use of (5.4) and (5.5), (5.9) yields

$$\begin{aligned} \|\mu(kr)\| \;\leq\; & \left(\frac{k_0 + m_0/r}{k + m_0/r}\right)^a \sum_{l=1}^{l_0} \sum_{s=1}^{r} \|\pi(lr+1-s) - \pi(lr-s)\| \; + \\ & + \frac{2}{(k+m_0/r)^a} \sum_{l=l_0+1}^{k} (l+1+m_0/r)^a \{ \pi_*(lr) - \pi_*(lr-1) \} \end{aligned} \tag{5.10}$$

for $k \geq l_0 = \max\{ \hat{m}/r, k_0 - 2 \}$. π_* is given by

$$\pi_*(m) = \sum_{\omega_i \in \Omega_*} \pi_i(m) . \tag{5.11}$$

Now writing $\hat{\pi}_*(m)$ for $\{ 1 - \pi_*(m) \}$, we have

$$\begin{aligned} & \sum_{l=l_0+1}^{k} (l+1+m_0/r)^a \{ \pi_*(lr) - \pi_*(lr-r) \} \leq \\ & \leq \sum_{l=l_0+1}^{k} \{ (l+1+m_0/r)^a - (l+m_0/r)^a \} \hat{\pi}_*(lr-r) + (l_0+1+m_0/r)^a \hat{\pi}_*(l_0 r) \leq \\ & \leq a \sum_{l=l_0+1}^{k} \frac{\hat{\pi}_*(lr-r)}{(l+m_0/r)^{1-a}} + (l_0+1+m_0/r)^a \hat{\pi}_*(l_0 r) \;, \end{aligned} \tag{5.12}$$

where, in the last step we have used the relation $a < 1$.

The substitution of (5.12) in (5.10) gives for $k \leq l_0$

$$\|\mu(kr)\| \leq \frac{D_\mu}{(k+m_0/r)^a} + \frac{2a}{(k+m_0/r)^a} \sum_{l=l_0+1}^{k} \frac{\hat{\pi}_*(lr-r)}{(l+m_0/r)^{1-a}} \ , \tag{5.13}$$

where

$$\begin{aligned} D_\mu &= (k+\frac{m_0}{r})^a \sum_{l=1}^{l_0} \sum_{s=1}^{r} \|\pi(lr+1-s) - \pi(lr-s)\| \ + \\ &+ \ 2(l_0+1+m_0/r)^a \ \hat{\pi}_*(l_0 r) \ . \end{aligned}$$

To proceed further it is necessary to estimate $\hat{\pi}_*(m)$ and this is undertaken in the following proposition.

Proposition 5.1.2 *With the above definitions of $\pi_*(m)$ and of $\hat{\pi}_*(m)$ we have, for $m = 0, 1, \ldots,$*

$$\begin{aligned} \hat{\pi}_*(m) &= 1 - \pi_*(m) = \frac{1}{2}\|\pi(m) - \mathbf{e}_*\| \ \leq \\ &\leq \sum_{j \in (\Omega \cap \Omega_*)^c} \frac{G_i/G_*}{(m+m_0+1)^{b_j}} \ , \end{aligned} \tag{5.14}$$

where b_j is given by

$$b_j = \{ \ c_j - c_* \ \}/\gamma \tag{5.15}$$

with c_* the minimum of the cost function and G_*

$$G_* = \sum_{j \in \Omega_*} G_j \ .$$

Proof. By the definition of $\pi(m)$ given in (4.19) and that of $\pi_*(m)$ given in (5.11) we have

$$\begin{aligned} 1-\pi_*(m) &= 1-\sum_{i\in\Omega_*}\frac{G_i e^{-\frac{c_i}{T_m}}}{Z(m)} = \\ &= \frac{\sum_{j\in(\Omega\cap\Omega_*)^c}\frac{G_j/G_*}{(m+m_0+1)^{b_j}}}{1+\sum_{j\in(\Omega\cap\Omega_*)^c}\frac{G_j/G_*}{(m+m_0+1)^{b_j}}} \leq \\ &\leq \sum_{j\in(\Omega\cap\Omega_*)^c}\frac{G_j/G_*}{(m+m_0+1)^{b_j}} . \end{aligned}$$

□

Observe that the bound given in (5.14) is asymptotically (i.e. as $m \to \infty$) tight .

We can now say that, for $l = 1, 2, \ldots,$

$$\hat{\pi}_*(lr-r) \leq \sum_{j\in(\Omega\cap\Omega_*)^c}\frac{\eta_j}{(l-1+m_0/r)^{b_j}} , \tag{5.16}$$

where, for all $j \in \Omega_*$

$$\eta_j = \frac{G_j/G_*}{r^{b_j}} . \tag{5.17}$$

By substituting (5.16) in (5.13) and then bounding the resulting expression we obtain

$$\|\mu(kr)\| \leq \frac{D_\mu}{(k+m_0/r)^a} + \sum_{j\in(\Omega\cap\Omega_*)^c}\frac{2a\eta_j}{a-b_j}\left[\frac{1}{(k+m_0/r)^{b_j}} - \frac{E^{a-b_j}}{(k+m_0/r)^a}\right] , \tag{5.18}$$

where

$$E = (l_0 - 1 + m_0/r).$$

The bound in (5.18) has been obtained with the assumption that $a \neq b_j, j \in \Omega_*$. If this is not true, then for the terms corresponding to states j for which $a = b_j$, a related expression is obtained by a slightly different bounding procedure.

5.1.3 Bound for the Third Term

The bound for the third terms comes directly from Proposition 5.1.2.

5.2 Final Results

Combining the results given in Sections 5.1.1, 5.1.2, and 5.1.3 we obtain the following conclusive theorem.

Theorem 5.2.1 *For every $k \geq l_0$, the following relation holds*

$$\begin{aligned} \|\mathbf{p}(kr) - \mathbf{e}_*\| \;\leq\; & \frac{D}{(k + m_0/r)^a} + \\ & + \sum_{j \in (\Omega \cap \Omega_*)^c} \frac{2a\eta_j}{a - b_j} \Big[\frac{1}{(k + m_0/r)^{b_j}} - \frac{E^{a-b_j}}{(k + m_0/r)^a} \Big] + \\ & + \sum_{j \in (\Omega \cap \Omega_*)^c} \frac{2\eta_j}{(k + m_0/r)^{b_j}} \;, \end{aligned} \tag{5.19}$$

where

$$D = D_\mu + \|\mathbf{p}(0) - \pi(0)\|(k_0 + m_0/r)^a \;.$$

a, b_j, and η_j are given by (4.18), (5.15), and (5.17) respectively.

Equation (5.19) can be further simplified if we consider that the dominant term of

$$\frac{1}{(k + m_0/r)^{b_j}} \;,$$

for $j \in (\Omega \cap \Omega_*)^c$ is given by

$$\frac{1}{(k + m_0/r)^{b}} \;,$$

where

$$b = \min_{j \in (\Omega \cap \Omega_*)^c} b_j = \frac{\delta}{\gamma} \;,$$

and δ which has been defined in (4.27).

A simple corollary to Theorem 5.2.1 is :

Corollary 5.2.1 **SA** *with the annealing schedule given by (4.16) has the following rate of convergence*

$$\|\mathbf{p}(kr) - \mathbf{e}_*\| = O(1/k^{\min(a,b)}) \;. \tag{5.20}$$

5.3 Concluding Remarks

We can see from (5.20) that the bound on the asymptotic rate of convergence is limited by $\min(a, b)$. Both a and b depend on δ and L derived from the cost function, w and r from the connectivity properties of the graph underlying the Markov chain and on γ from the annealing schedule. Note that with all parameters and time held fixed, higher γ corresponds to higher temperature and thus, in this sense to a slower schedule. Now γ has to satisfy the condition that gives weak ergodicity, i.e. $\gamma \geq \gamma_{WE}$ wherein by our analysis $\gamma_{WE} = rL$, but otherwise it is a free parameter. It is therefore of some interest to investigate the value of γ which maximizes $\min(a, b)$.

Recall the definition of a in (4.18) and that $b = \delta/\gamma$. Hence $a(\gamma)$ and $b(\gamma)$ are respectively increasing and decreasing with increasing γ, and it is easy to see that there exists an unique $\tilde{\gamma}$ such that $a(\tilde{\gamma}) = b(\tilde{\gamma})$. Furthermore, the problem

$$\max_{\gamma \geq rL} \{ \min(a, b) \}$$

has the solution

$$\gamma = \max(\gamma_{WE}, \tilde{\gamma}).$$

The above procedure for optimizing the algorithm is often feasible since for many combinatorial optimization problems estimates of r, L and δ are available.

Example 5.3.1 Consider the placement problem discussed in Chapter 2 and consider the case in which n macro-cells have to be placed on a one-dimensional grid with n positions. If we assume that each one of the solutions in the neighbor of every state has uniform probability of being generated, we have the following estimate for r and w

$$r = n - 1$$

and

$$w = \frac{2}{n(n-1)}.$$

Substituting the estimates of w and r in the expression for a and b we have

$$a = \frac{w^r}{r^{rL/\gamma}} = \frac{\left[\frac{2}{n(n-1)}\right]^{n-1}}{(n-1)^{\frac{L(n-1)}{\gamma}}},$$

$$b = \frac{\delta}{\gamma}.$$

Clearly a is the smallest of the two. L and γ are constants of the problem and if we assume that they can be neglected we have that

$$a \approx \left[\frac{2}{n(n-1)^2}\right]^{n-1}$$

which in turn gives a very slow rate of convergence. □

The above discussion has been on the effect of γ (from the annealing schedule) on the bound of the rate of convergence at finite, but large time. For behavior after a smaller number of iterations, the more detailed relation (5.19) has to be considered. Observe that in the right hand side of this equation the only factors which depend on time kr are $1/(k+m_0/r)^a$ and $1/(k+m_0/r)^{b_j}$ for $\omega_j \in (\Omega \cap \Omega_*)^c$. We may glean qualitative information on the dependence of the rate of convergence of γ by investigating the dependence of a and b_j on γ. First recall that smaller γ gives larger b_j for each ω_j and, as already noted, smaller a. Hence reducing γ has the effect of reducing the third term and increasing the first term in the right hand side of (5.19). The dependence of the middle term is more involved since it has features of both terms reflected in it. Roughly, it is small only when both the first and the third terms are small, i.e. in the mid-range of γ.

With the benefit of analysis we can even go back to (5.2) and deduce qualitatively the effect of γ on each of the three terms there. The first term measures how effectively the difference between $\mathbf{p}(0)$ and $\pi(0)$ is forgotten at step m of the algorithm. The bound in (5.6) corroborates our intuitive understanding that this rate is aided by having higher γ, i.e. higher temperature and slower cooling. The third term, for which we have the most explicit information (See Proposition 5.1.2) depends on the rate at which the quasi-stationary probability distribution approaches

its asymptotic value, the optimum distribution. This term benefits from small γ. The middle term benefits from a matching of the two rates. The point in the analysis where this is most manifest is in (5.13). The two rates are matched and the term minimized in the mid-range of γ. In all, the above discussion illuminates the balancing of opposite mechanism that an optimal annealing schedule must reflect.

The analysis can be brought to bear on an important question [1] : To what extent does SA exploit the connectivity of the configuration in a particular case? The comparison is therefore between a given partially connected graph and a construct in which the connectivity is artificially increased. A first observation is that the artificial increase of connectivity leads to a deteriorating component in the performance, insofar as the departure of the quasi-stationary probability distribution at a particular T from the optimum distribution (See third term in (5.3)) is greater. This is easily seen by tracing the effect of increased connectivity on G_j/G_*, in Proposition 5.1.2. On the other hand, the effect on the coefficient of ergodicity and, in particular, on the parameter a in the bound for it given in Proposition 5.1.1, depends on the characteristics of the case being considered. To see this observe that the parameter a depends on w, r, and L and typically the first two decrease while the last increases with the increase of the connectivity in the construct.

Results similar to those presented in this section have been obtained also by Anily and Federgruen [76] and by Connors and Kumar [82].

Finally a general comment is in order. The convergence results obtained with the homogeneous theory were rather accurate but impossible to use in practice, since they required to know the eigenvalues of the transition matrix or the conductance of the graph $\mathcal{G}$ underlying the Markov chain.

In this section, for specific acceptance, generate, and update functions we were able to determine a bound on the departure of the finite time distribution from the asymptotic one that requires to know only bounds on the connectivity of the graph and on the local slope of the cost function. Not surprisingly however, the

[1] We are indebted to H.S. Witsenhausen for posing it

bound turns out to be quite loose and gives a rate of convergence that requires a number of iterations which is comparable with the number of iterations necessary to perform an exhaustive search of the state space.

So why bother to estimate the rate of convergence if it requires either quantities impossible to estimate or gives bound that are too loose to be used? Because the convergence results are useful not so much as a precise estimation on the amount of time necessary to achieve ergodicity but rather as a tool to establish the relative merits of the different choices of the variables of the annealing, namely the acceptance functions, the generate functions etc.

In this section we have discussed how the constant γ affects the rate of convergence of each single terms of (5.2) and what are the effects of an artificial increase in the connectivity of $\mathcal{G}$. In Chapter 6 we will see how convergence rate results can be used to select, among a sub-class of the admissible functions, the acceptance function that guarantees the fastest convergence to the stationary probability distribution at fixed T.

Chapter 6

The Acceptance and Generate Functions

SA, since its inception in 1983 [12], has been identified with a probabilistic algorithm which generates new solutions and accepts them according to the *Metropolis rule*

$$f_{ij} = \min\left[1, e^{-\frac{c_j - c_i}{T}}\right]. \tag{2.3}$$

However we have seen in Chapter 3 that SA is actually a class of algorithms for which convergence to the global optimum is guaranteed, provided that the acceptance and generate functions satisfy some mild conditions. The conditions are general enough that, given the asymptotic probability distribution of the solutions π, there is still freedom in the selection of both the acceptance function and the generate function.

Even the more restrictive conditions imposed by the inhomogeneous theory leave, in the most general interpretation proposed by Anily and Federgruen, some freedom in the selection of the acceptance function [76]. It is then legitimate to ask whether, among the different choices for the acceptance function, there is one which provides SA with the best rate of convergence to the optimal solution.

In this chapter we show that, if the stationary probability distribution is

as in

$$\pi_i(T) = \frac{G_i e^{-\frac{c_i}{T}}}{Z(T)} \tag{4.19}$$

and the generate function is symmetric, (2.3) is the best choice for the acceptance function in the sense that, fixed T, the underlying Markov chain has the fastest rate of convergence to the stationary probability distribution π.

From the theory we also know that the acceptance function is not the unique responsible for the rate of convergence of the Markov chain. In fact, the rate of convergence is determined by the eigenvalues of the transition matrix **P** and the entries of **P** depend not only on the form of the acceptance function but also on the generate function. In Section 6.2 we describe how clever implementations of the generate function can significantly improve the rate of convergence of **SA** to the optimal solution.

Section 6.3 is dedicated to the influence of an approximate evaluation of the cost function on the asymptotic behavior of **SA**. The issue of an approximate evaluation of the cost function arises in two different circumstances. First, there are problems for which to evaluate exactly the cost for a given solution is expensive in terms of computing time, while approximations to the cost can be computed easily [25]. Second, the implementation of **SA** on a computer with a parallel architecture and shared memory may lead to incorrect evaluations of the cost function. In fact, let us assume that the implementation is such that the variables of the problem are partitioned among the processors. Every processor generates a move by perturbing one or more of the variables it controls and, to decide whether to accept or reject the new solution, it has to evaluate the cost function. The processor knows exactly the value of the variables it owns but, unless all the other processors are idle, it has only an out-of-date information of the variables it does not own. Consequently, the evaluation of the cost for the perturbed solution may be incorrect and the processor may decide to accept a solution when it should have been rejected or vice versa. In both circumstances, it is important to determine whether, even in the presence of these errors, the asymptotic convergence properties of the algorithm are retained.

The origin of expression (4.19) for the stationary probability distribution

is implicitly justified by Kirkpatrick et al. [12] on the basis of an analogy between classical statistical mechanics and combinatorial optimization. However, there is no reason to believe that this analogy should hold for any combinatorial optimization problem with any choice of cost function. The aim of the last section is to provide a justification for the selection of (4.19) which does not require to use the analogy to statistical mechanics but is based only on fundamental results from Statistical Information Theory.

6.1 Acceptance Function

The criteria that should inspire the choice of the acceptance function for a practical implementation of SA are discussed first. We show that, if the Markov chain is reversible, given the form of the stationary probability distribution and a class of possible acceptance functions, there is an optimal strategy to select the acceptance function which makes the convergence to the asymptotic probability distribution the fastest. Then we briefly review other forms for the acceptance function that have been proposed in literature.

6.1.1 The Optimal Acceptance Function

In this section we assume that T is fixed and hence we will avoid to mention explicitly the dependence of the variables on T. Furthermore we will restrict our attention to reversible Markov chain according to Definition 3.1.7.

Among the possible ways to build a reversible Markov chain from a given stationary probability distribution π, let us consider the class of chains whose transition probabilities P_{ij} are of the form

$$P_{ij} = G_{ij} f_{ij} \,, \tag{6.1}$$

$$P_{ii} = 1 - \sum_{i \neq j} P_{ij} \,, \tag{6.2}$$

where G_{ij} represents the probability to generate state j from state i and f_{ij}, defined

by

$$f_{ij} = \frac{s_{ij}\pi_i G_{ij}}{\pi_j G_{ji} + \pi_i G_{ij}} \ , \tag{6.3}$$

is the probability that state j is actually accepted. s_{ij} is any symmetric function with the only limitation that corresponding acceptance probability f_{ij} obtained from (6.3) satisfies $0 \leq f_{ij} \leq 1$. The class of transition probabilities defined by (6.1), (6.2), and (6.3) has been introduced by Hastings in his work on Monte Carlo sampling methods [77].

It is immediate to see that the transition probabilities P_{ij} defined by (6.1), (6.2), and (6.3) satisfy the detailed balance equation (3.11). Furthermore, if

$$s_{ij} = \begin{cases} 1 + \frac{\pi_i G_{ij}}{\pi_j G_{ji}} & \text{if } \pi_j G_{ji} \geq \pi_i G_{ij} \\ 1 + \frac{\pi_j G_{ji}}{\pi_i G_{ij}} & \text{if } \pi_j G_{ji} \leq \pi_i G_{ij} \end{cases} \tag{6.4}$$

then

$$f_{ij} = \min\left[1, \frac{\pi_j G_{ji}}{\pi_i G_{ij}}\right] \ . \tag{6.5}$$

Similarly if $s_{ij} = 1$, then

$$f_{ij} = \frac{\pi_j G_{ji}}{\pi_i G_{ij} + \pi_j G_{ji}} \ . \tag{6.6}$$

Notice that if π is given by (4.19) and G_{ij} is symmetric, (6.5) becomes (2.3), while (6.6) becomes

$$f_{ij} = \frac{e^{-\frac{c_j - c_i}{T}}}{1 + e^{\frac{c_j - c_i}{T}}} \ .$$

To establish which of the different choices of s_{ij} leads to a faster rate of convergence we could, in principle, construct the corresponding transition probability matrices, compute their eigenvalues and determine which of the choices gives the smallest second largest eigenvalue. However we know by now that any strategy involving the computation of the eigenvalues is hardly feasible since, in general, we do not even know all the entries of the transition probability matrix $\mathbf{P}$. On the other hand we are not interested in the absolute value of the rate of convergence but only in establishing the relative merits of different choices of s_{ij}. To this end

we can proceed as follows: Let us consider the following random variable

$$I_N = \frac{1}{N}\sum_{n=1}^{N} c(X_n) ,$$

where X_n is the random variable that represents the state occupied by the Markov chain after n iterations. I_N is the estimator of

$$E_\pi[c] = \sum_{i\in\Omega} c_i\pi_i \ .$$

where $E_\pi[c]$ is the expected value of c with respect to the asymptotic probability distribution π. Since I_N is a random variable, we can use its variance $\sigma^2(I_N)$ defined by

$$\sigma^2(I_N) = E_\pi[(I_N - E_\pi[c])^2] \ ,$$

as a measure of the distance of the estimator I_N from the true value, $E_\pi[c]$. Now it is clear that, if the probability distribution after n iterations, $\mathbf{p}^{(n)}$, converges to π in the sense defined in Chapter 3 then

$$\lim_{N\to\infty} \sigma^2(I_N) = 0 \ . \tag{6.7}$$

Notice however that the converse is not true in general.

Kemeny and Snell have shown [83] that the variance of the estimator I_N is independent of the initial probability distribution on the solution space, $\mathbf{p}$, and has the following asymptotic expression

$$\begin{aligned} v(\mathbf{c},\pi,\mathbf{P}) &= \lim_{N\to\infty} N\sigma^2(I_N) = \\ &= \mathbf{c}\{\mathbf{DZ} + (\mathbf{DZ})^T - \mathbf{D} - \mathbf{D}\Pi\}\mathbf{c}^T = \\ &= \mathbf{c}\{2\mathbf{DZ} - \mathbf{D} - \mathbf{D}\Pi\}\mathbf{c}^T , \end{aligned} \tag{6.8}$$

where $\mathbf{c} = [c_1, c_2, \ldots, c_{|\Omega|}]$ is a row vector and c_i is the cost of the i-th state. $\mathbf{D}$ is a diagonal matrix whose non zero entries are $D_{ii} = \pi_i$ (See (3.12)), Π is a matrix with all the rows equal to π (See (3.6)), and $\mathbf{Z} = (\mathbf{I} - \mathbf{P} + \Pi)^{-1}$ is the fundamental matrix of the Markov chain [83]. From (6.8), if N is large enough, the variance

of the estimator I_N can be approximated with its asymptotic value $v(\mathbf{c}, \pi, \mathbf{P})$ as follows

$$\sigma^2(I_N) \approx \frac{v(\mathbf{c}, \pi, \mathbf{P})}{N} .$$

Equation (6.8) is crucial since it links v explicitly to the matrix $\mathbf{P}$ and gives us a tool to relate the entries of $\mathbf{P}$, and hence the s_{ij}'s, to the variance of I_N, and hence to the rate of convergence. To see this we can proceed as follows: Consider two stochastic matrices $\mathbf{P}_1$ and $\mathbf{P}_2$ and define $\mathbf{P}_1 \leq \mathbf{P}_2$ if each off-diagonal element of $\mathbf{P}_2$ is greater than or equal to the corresponding element of $\mathbf{P}_1$. With the above definition, we have the following sufficient condition on the variances of the estimators defined for the Markov chains corresponding to $\mathbf{P}_1$ and $\mathbf{P}_2$

Theorem 6.1.1 (Peskun) *Suppose that both the stochastic matrices* $\mathbf{P}_1$ *and* $\mathbf{P}_2$ *correspond to regular, reversible Markov chains with the same stationary probability distribution* π. *If* $\mathbf{P}_1 \leq \mathbf{P}_2$ *then*

$$v(\mathbf{c}, \pi, \mathbf{P}_2) \leq v(\mathbf{c}, \pi, \mathbf{P}_1)$$

Proof. For each one of the off-diagonal elements of $\mathbf{P}$, we have

$$\frac{\partial v(\mathbf{c}, \pi, \mathbf{P})}{\partial P_{ij}} = 2\mathbf{c}\Big(\mathbf{D}\frac{\partial \mathbf{Z}}{\partial P_{ij}}\Big)\mathbf{c}^T . \tag{6.9}$$

If we use the following matrix identity

$$\frac{\partial \mathbf{Z}}{\partial P_{ij}}\mathbf{Z}^{-1} + \mathbf{Z}\frac{\partial \mathbf{Z}^{-1}}{\partial P_{ij}} = 0$$

and we substitute it in (6.9) we obtain

$$\frac{\partial v(\mathbf{c}, \pi, \mathbf{P})}{\partial P_{ij}} = -2\mathbf{c}\Big(\mathbf{D}\mathbf{Z}\frac{\partial \mathbf{Z}^{-1}}{\partial P_{ij}}\mathbf{Z}\Big)\mathbf{c}^T . \tag{6.10}$$

From the definition of $\mathbf{D}$ and $\mathbf{Z}$, $\mathbf{D}\mathbf{Z}^{-1}$ and its inverse $\mathbf{Z}\mathbf{D}^{-1}$ are symmetric which, in turn, implies that $\mathbf{D}(\mathbf{Z}\mathbf{D}^{-1})\mathbf{D} = \mathbf{D}\mathbf{Z}$ is symmetric. The symmetry of $\mathbf{D}\mathbf{Z}$ allows us to rewrite (6.10) as follows

$$\frac{\partial v(\mathbf{c}, \pi, \mathbf{P})}{\partial P_{ij}} = -2(\mathbf{Z}\mathbf{c}^T)^T\Big(\mathbf{D}\frac{\partial \mathbf{Z}^{-1}}{\partial P_{ij}}\Big)(\mathbf{Z}\mathbf{c}^T) .$$

By the definition of $\mathbf{Z}$

$$\mathbf{D}\frac{\partial \mathbf{Z}^{-1}}{\partial P_{ij}} = \mathbf{D}\frac{\partial \mathbf{P}}{\partial P_{ij}} \ .$$

The matrix $\mathbf{D}(\partial \mathbf{P}/\partial P_{ij})$ is made of all zeros with the only exception of the entries $(i,i),(i,j),(j,i),(j,j)$ which are equal to $-\pi_i, \pi_i, \pi_j, -\pi_j$ respectively. As a consequence the matrix $\mathbf{D}(\partial \mathbf{P}/\partial P_{ij})$ is positive definite which, in turn, means that $v(\mathbf{c}, \boldsymbol{\pi}, \mathbf{P})$ is a decreasing function of the off-diagonal elements of $\mathbf{P}$. Therefore $\mathbf{P}_1 \leq \mathbf{P}_2$ implies $v(\mathbf{c}, \boldsymbol{\pi}, \mathbf{P}_2) \leq v(\mathbf{c}, \boldsymbol{\pi}, \mathbf{P}_1)$. □

Theorem 6.1.1 implies that the asymptotic variance of the estimator I_N can be reduced by making the off-diagonal elements of the transition matrix large. Intuitively this means that the more a transition to a different state is likely to be accepted, the faster the estimator I_N approaches its asymptotic value.

The result of Theorem 6.1.1 can be used to determine which, among the possible choices of s_{ij} as in (6.3), gives the fastest convergence in the sense of (6.7).

Theorem 6.1.2 (Peskun) *For any given generate function G_{ij}, if s_{ij} is given by*

$$s_{ij} = \begin{cases} 1 + \dfrac{\pi_i G_{ij}}{\pi_j G_{ji}} & \text{if } \pi_j G_{ji} \geq \pi_i G_{ij} \\ 1 + \dfrac{\pi_j G_{ji}}{\pi_i G_{ij}} & \text{if } \pi_j G_{ji} \leq \pi_i G_{ij} \end{cases} \tag{6.4}$$

the corresponding acceptance function will give, among the functions in the class defined by (6.3), the fastest rate of convergence in the sense of (6.7).

Proof. From the definition of P_{ij} it follows that $f_{ij} \leq 1$ which implies that

$$s_{ij} \leq (1 + (\pi_i G_{ij}/\pi_j G_{ji})) \ .$$

A similar inequality holds for s_{ji}. If the inequalities are combined we have

$$s_{ij} \leq 1 + \min\left[\frac{\pi_i G_{ij}}{\pi_j G_{ji}}, \frac{\pi_j G_{ji}}{\pi_i G_{ij}}\right] \ , \tag{6.11}$$

and (6.11) is satisfied with the equality sign, if s_{ij} is selected as in (6.4). Hence an s_{ij} as in (6.4) gives a transition probability matrix with the largest off-diagonal elements. Therefore from Theorem 6.1.1 s_{ij} as in (6.4) gives among the functions in Hasting's class, the fastest rate of convergence in the sense of (6.7). □

If the results of Theorems 6.1.1 and 6.1.2 are applied to SA we have the following

Proposition 6.1.1 *If the following stationary probability distribution*

$$\pi_i(T) = \frac{G_i e^{-\frac{c_i}{T}}}{Z(T)}$$

and a symmetric generate function are selected for an instance of SA*, the acceptance function which gives the fastest rate of convergence to* $\pi(T)$ *in the sense of (6.7) is given by*

$$f_{ij} = \min\left[1, e^{\frac{-c_j - c_i}{T}}\right].$$

6.1.2 Other Acceptance Functions

The results derived in the previous section assume that the asymptotic probability distribution π is selected first and the transition probability matrix P is built such that the detailed balance equation is satisfied.

However the selection of the transition probability as suggested in the previous section assumes that the number of iterations is large. In practical applications, the number of iterations has to be kept as small as possible. Hence there have been a number of authors that decided to select the transition probabilities which experimentally give the best trade-off between number of iterations and quality of the solution produced.

Bohachevsky, Johnson, and Stein [78] propose an acceptance function given by

$$f_{ij} = \begin{cases} e^{-\frac{(c_j - c_i)c_i^g}{T}} & \text{if } c_j > c_i\,, \\ 1 & \text{otherwise}\,, \end{cases}$$

where g is any arbitrary constant negative number. The acceptance function has been applied to find the minimum of a two-dimensional continuous function for which the optimum is known. The authors do not report explicit figures on the performance of their acceptance function compared with the standard one given in (2.3), but they claim that, for a class of two-dimensional continuous functions, the

acceptance function defined above performs constantly better than an acceptance function of the form (2.3). The acceptance function by Bohachevsky et al. does not depend on the difference of cost only but depends on the absolute value of the two costs. Another such acceptance function is the following

$$f_{ij} = \frac{c_i}{c_j} T^{-(c_j - c_i)}$$

proposed by Faigle and Schrader [72].

Note that, unlike Faigle's acceptance functions, Bohachevsky's belongs to the class of acceptance functions for which Anily and Federgruen showed that a logarithmic update rule for T like (4.39) is sufficient to guarantee strong ergodicity for the underlying Markov chain [76].

6.2 Generate Function

The generate function plays an important role. In fact, a poor selection of the move set may lead to a situation, such as the one shown in Example 2.1.2, in which the solution space is not fully accessible. A poor choice of the move set may still make the solution space reachable but with a very slow convergence to the global optimum.

Given the dependency of the move set on the problem, very little can be said in general. However there are two strategies that proved to work rather well when applied to solve combinatorial optimization problems like those arising in the layout of integrated circuits.

6.2.1 Range Limiting

The notion of range limiting was introduced first by Sechen in 1984 [19]. The idea behind it is quite simple and is based on the following observation: For any given value of T there is a subset of the moves that are most likely to be accepted by the algorithm. For example, towards the end of the execution of SA, when T assumes small values, only moves that improve the value of the cost function or

that degrade it by a limited amount are likely to be accepted. Therefore generating moves that produce a sizable degradation in the cost function, will result, most of the time, in a rejection and in a consequent waste of time. The range limiting strategy uses this observation to update the generation function trying to keep the ratio of accepted moves versus attempted moves high so that SA uses the CPU time efficiently.

The main drawback of range limiting is that it requires to recognize moves likely to produce large variations of the cost even before they are attempted and select them with a probability which is inversely proportional to the estimate of the amount of the perturbation generated. For range limiting to perform well, the move set has to be rich enough so that, even if some of the moves are discarded, SA has still a relatively large set of moves to choose from.

In optimization problems such as placement problems, the notion of the amount of variation produced by a move is directly related to geometrical considerations, e.g. the amount of displacement. Furthermore the set of moves is rich since any displacement of any of the cells generates a feasible moves. The combination of accurate prediction of the cost variation produced by a move with the richness of the move set is the reason why range limiting is particular effective in these applications.

For other problems where it is more difficult to predict of the effect produced on the cost by a move and/or the move set is not rich enough, we do not expect range limiting to perform well. In these cases, the range limiting strategy should be generalized. One such generalization is proposed by Greene and Supowit [84]. The idea is to compile a list of the variations in cost produced by each of the possible moves and select one of them with a probability proportional to acceptance probability. Once the move is generated, it is always accepted with probability one. It is immediate to see that this variation of SA is probabilistically equivalent to the original algorithm. However, it should be evident that the strategy proposed by Greene and Supowit has a basic limitation: It requires to know, for each solution, the cost of all the neighbors and this is, in most of the practical applications, out of the question. In fact unless the information is known *a priori,* to build the list of

all variations requires the exploration of the neighborhood which is time consuming. On top of this, it is necessary to store the cost of any of the solutions which comprise the neighborhood and this requires a large amount of computer memory.

6.2.2 Adaptive Generate Functions

In the previous section, we presented some of the limitation of the range limiting concept in its simplest implementation. In this section, we describe an extension of the idea which provides a method to determine, from the data collected during the execution of the algorithm, the selection probabilities for the moves. A complete account of the method is given in [27,85].

Hustin's method [27,85] is based on the assumption that a desirable move is a move which produces a sizable perturbation of the cost while maintaining a reasonable chance to be accepted. Then the moves have to be selected such that the more desirable the move, the more often it is attempted.

To rank the desirability of the move m, a quality factor Q_m is defined as follows

$$Q_m = \frac{\sum s(m)}{\frac{a_c(m)}{a_t(m)}},$$

where $a_c(m)$ and $a_t(m)$ are the number of moves of type m which are accepted and attempted respectively. The size $s(m)$ of the move m is the absolute value of the change in cost produced by the move. The selection probability for a move of type m is then determined by

$$P(m) = \frac{Q_m}{\sum_{m \in \mathcal{M}} Q_m}.$$

The quality factor is the product of two contributions: The average size of the moves of type m and the acceptance ratio of moves of the same type. A small quality factor means that either the move has a large size but is rarely accepted or it is accepted quite often but its size is small. Hence the best quality factor indicates the move which has the best trade-off between size and acceptance ratio. Of course, since both size and acceptance ratio are functions of T, it follows that

quality factors have to be recomputed every time a new temperature is generated.

The use of quality factor gives a criterion to select among the available moves that, unlike range limiting, does not rely on the particular problem at hand. In fact, the size of the move is determined from the data collected during the execution of the algorithm and no *a priori* estimation is necessary.

The effectiveness of a generate function based on quality factors is enhanced if it is combined with a rich move set. Hustin proved this by collecting the results obtained by his program Tim with different move sets. As expected, for a given quality of the solution, Hustin showed experimentally that the execution time decreases as the move set becomes richer [1]. In particular the generate function based on quality factors takes full advantage of the presence of complex moves. A complex move consists of a combination of simple moves which are attempted in sequence with the cost evaluated only after the last move of the sequence has been completed. Pictorially, complex moves allow the algorithm to tunnel through hills instead of climbing them.

The idea that a proper use of tunneling moves or, more in general, of large-size moves is useful to speed up the convergence of the algorithm is not new [86] and has also been observed in a different context by Szu [87,88]. Szu applied SA to find the global minimum of a continuous one dimensional function. In his implementation, a move is generated by selecting at random, from a specified distribution, a variable δx which is added to the present value of the independent coordinate to obtain the new one. He experimented with two different distributions: A Gaussian distribution and a Cauchy distribution. Both the Gaussian and the Cauchy distributions are symmetric with respect to the origin but while the moments of the Gaussian are all finite, those of the Cauchy are not. As a consequence, if δx_G is the random variable extracted from the Gaussian distribution and δx_C is the random variable extracted from the Cauchy distribution, for all $\alpha > 0$

$$\Pr\{\ |\delta x_C| \geq \alpha\ \} \geq \Pr\{\ |\delta x_G| \geq \alpha\ \}\ .$$

[1] Hustin claims for Tim, his placement program, up to 24 times faster execution time compared to Timberwolf3.2 [22] for solution with the same quality.

The results found by Szu show that SA with the larger size moves, i.e. those with δx_C extracted from the Cauchy distribution, produce results of the same quality with shorter execution times.

6.3 Errors in the Cost Evaluation

In the theory developed so far, it is always assumed that the evaluation of the cost is carried out exactly. In this section, we focus on the case in which the cost evaluation is approximated. Aim of this section is to find under what conditions on the error, the transition probabilities of the "noisy" process converge, as T approaches zero, to the transition probabilities of the original process.

At time n, given the value of the temperature T_n, assume that the uncorrupted process is in state i and that the state j, $j \neq i$, is selected as a candidate for the new state. Assume that the difference in cost $c_j - c_i$ is measured with an additive noise λ. Assume that λ is a real valued random variable independent of the particular choice of the state and with probability distribution given by

$$\Pr\{ W_n \leq \lambda \} = \Upsilon_n(\lambda) \ . \tag{6.12}$$

The probability distribution $\Upsilon_n(\lambda)$ is actually a sequence of distributions indexed in n. Now we can define on Ω a process Z_n which is conditionally dependent only on Z_{n-1} and on the error W_{n-1} and assume that its transition probabilities are given, for all $i \neq j$, by

$$\Pr\{ Z_{n+1} = j \mid Z_n = i, W_n = \lambda\} = \begin{cases} G_{ij}(T_n) \exp\left(-\frac{c_j - c_i + \lambda}{T_n}\right) & \text{if } c_i - c_j \leq \lambda \ , \\ G_{ij}(T_n) & \text{otherwise} \ . \end{cases} \tag{6.13}$$

With the above assumptions, Z_n is a Markov process. Furthermore, since W_n is independent of Z_n, the transition probability of Z_n becomes

$$\Pr\{ Z_{n+1} = j \mid Z_n = i\} = E_{\Upsilon_n}\{\Pr\{ Z_{n+1} = j \mid Z_n = i, W_n\}\} \ .$$

A sufficient condition for the transition probabilities of Z_n to approach those of X_n, as T approaches zero, is to require that the probability distribution of the error

concentrates, as T approaches zero, its mass around the origin. The details of the sufficient conditions are presented in the following theorem:

Theorem 6.3.1 *Let Z_n be the Markov process obtained from the Markov process X_n when the transition probability is affected by an additive noise λ. If the sequence of distributions $\{\Upsilon_n(\lambda)\}$ is such that for every $x > 0$*

$$\lim_{T_n \to 0} \int_{-x}^{x} \exp\left(-\frac{\lambda}{T_n}\right) d\Upsilon_n(\lambda) = 1 \tag{6.14}$$

and

$$\lim_{T_n \to 0} \frac{1}{T_n}\left[\int_{-\infty}^{-x} \exp\left(-\frac{\lambda}{T_n}\right) d\Upsilon_n(\lambda) + \int_{x}^{\infty} \exp\left(-\frac{\lambda}{T_n}\right) d\Upsilon_n(\lambda)\right] = 0\,, \tag{6.15}$$

then for all $i \neq j$

$$\lim_{T_n \to 0} \Pr\{\, Z_{n+1} = j \mid Z_n = i \,\} = \lim_{T_n \to 0} \Pr\{\, X_{n+1} = j \mid X_n = i \,\}\,. \tag{6.16}$$

Proof. The proof simply requires to compute the transition probabilities of the "noisy" process Z_n. From (6.12), (6.13), and the independence of W_n and Z_n, the transition probability for Z_n is given, for all $i \neq j$, by

$$\begin{aligned} \Pr\{\, Z_{n+1} = j \mid Z_n = i\} &= E_{\Upsilon_n}\{\Pr\{\, Z_{n+1} = j \mid Z_n = i, W_n\}\} = \\ &= \int_{\lambda \geq c_i - c_j} G_{ij}(T_n) \exp\left(-\frac{c_j - c_i + \lambda}{T_n}\right) d\Upsilon_n(\lambda) + \\ &+ \int_{\lambda < c_i - c_j} G_{ij}(T_n) d\Upsilon_n(\lambda) = \\ &= a_n + b_n \end{aligned}$$

To proceed we split the analysis into two separate cases:

i) Down-hill transitions, i.e. $c_i \geq c_j$.

ii) Up-hill transitions, i.e. $c_i < c_j$.

For down-hill transitions, we have to show that (6.15) is sufficient to guarantee that $a_n \approx o(T_n)$ and (6.14) implies

$$\lim_{T_n \to 0} b_n = \lim_{T_n \to 0} G_{ij}(T_n)\ . \tag{6.17}$$

From the expression for a_n it follows

$$\begin{aligned} a_n &= \int_{\lambda \geq c_i - c_j} G_{ij}(T_n) \exp\Big(-\frac{c_j - c_i + \lambda}{T_n}\Big) d\Upsilon_n(\lambda) = \\ &= G_{ij}(T_n) \exp\Big(-\frac{c_j - c_i}{T_n}\Big) \int_{\lambda \geq c_i - c_j} \exp\Big(-\frac{\lambda}{T_n}\Big) d\Upsilon_n(\lambda) . \end{aligned}$$

Since the term

$$G_{ij}(T_n) \exp\Big(-\frac{c_j - c_i}{T_n}\Big)$$

is bounded as T approaches zero, from (6.15) and from the condition $c_i \geq c_j$,

$$\lim_{T_n \to 0} \frac{1}{T_n} \int_{\lambda \geq c_i - c_j} \exp\Big(-\frac{\lambda}{T_n}\Big) d\Upsilon_n(\lambda) = 0 ,$$

which implies

$$\lim_{T_n \to 0} \frac{a_n}{T_n} = 0 . \tag{6.18}$$

From the expression for b_n

$$\begin{aligned} b_n &= \int_{\lambda < c_i - c_j} G_{ij}(T_n) d\Upsilon_n(\lambda) = \\ &= \int_{-\infty < \lambda < -(c_i - c_j)} G_{ij}(T_n) d\Upsilon_n(\lambda) + \int_{-(c_i - c_j) < \lambda < c_i - c_j} G_{ij}(T_n) d\Upsilon_n(\lambda) . \end{aligned}$$

Hence taking the limits and using (6.14) on the second term of the second equality and (6.15) on the first term of the second equality we obtain (6.17).

If the above computations are repeated for up-hill transitions we obtain

$$\lim_{T_n \to 0} a_n = \lim_{T_n \to 0} G_{ij}(T_n) \exp\Big(-\frac{c_j - c_i}{T_n}\Big) \tag{6.19}$$

and

$$\lim_{T_n \to 0} b_n = o(T_n) . \tag{6.20}$$

The four conditions (6.17), (6.18), (6.19), and (6.20) are equivalent to (6.16). □

A restricted version of Theorem 6.3.1 was proved earlier by Gelfand and Mitter [89,90]. In their work, Gelfand and Mitter assume that the error has a Gaussian distribution with zero mean and variance σ_n^2. With this assumption the conditions of Theorem 6.3.1 reduce to the following condition on the variance

$$\lim_{T_n \to 0} \sigma_n^2 \approx o(T_n^4) .$$

Theorem 6.3.1 requires that the sequence of error distributions must concentrate their mass around the origin as T_n approaches zero. In other words this means that the error in the measurements has to go to zero in distribution faster than T_n.

The experiments with parallel implementation of SA presented by Casotto, Romeo, and Sangiovanni Vincentelli [91], show indeed that the error, introduced by their algorithm in the measurements, goes to zero faster than T_n. The explanation of this behavior is simple. The algorithm implementations features a mechanism to control the moves based on quality factors. As a consequence, towards the end of the execution, the algorithm tends to attempt only moves whose size is small. The processor which is performing the move on a particular cell, owns also the cells that are placed in the neighborhood and this, combined with the fact that the size of the move is small, reduces the amount of error in the cost evaluation. The results were confirmed by the parallel implementations of SA by Darema-Rogers, Kirkpatrick, and Norton [92], by Kravitz and Rutenbar [93], and by Banerjee and Jones [28].

In the case of serial implementations where the error is introduced due to approximations in the evaluation of the cost function, there are no experimental results on the actual behavior of the error as a function of T. However, if the quality factors are used to control the generation mechanism, the size of the moves becomes a decreasing function of T_n hence, unless the error has some deterministic components, its amount should vanish as T approaches zero.

6.4 Stationary Probability Distribution

The theory developed in the previous chapters hinges upon the assumption that the stationary probability distribution π is known. Kirkpatrick et al., in their original paper [12], implicitly assumed, using the Metropolis sampling rule as acceptance function, that π was given by the exponential distribution

$$\pi_i(T) = \frac{G_i e^{-\frac{c_i}{T}}}{Z(T)}.$$

The justification for the Metropolis rule and hence for π, provided in [12] was based on the analogy between combinatorial optimization problems and classical statistical mechanics. However, there is no reason to believe that the exponential density should represent, adequately, the state distribution for any combinatorial optimization problem with any choice of the cost function. The aim of this section is to provide a justification, based on Statistical Information Theory, for the selection of the exponential as stationary probability distribution.

The idea behind the procedure is as follows. Consider an arbitrary combinatorial optimization problem (Ω, c) and suppose that the only information available about the problem is the average value of the cost function on the set of solutions, $\bar{c}$. Knowing only the average value of the cost does not provide us with enough information to compute the probability distribution of the solution exactly. The best procedure we can follow is to assign some *a priori* probability distribution that agrees with the information available but that expresses the "maximum uncertainty" about every other unknown. The rationale behind this reasoning is to leave the maximum possible freedom in picking a more accurate distribution as soon as new information becomes available.

The measure of the degree of uncertainty is given by the *statistical entropy* or the *informational uncertainty* [94,95]. Therefore the probability distribution will be found by maximizing the statistical entropy subject to the constraints imposed by the available knowledge.

Here are the details. Assume that

$$N = |\Omega| \tag{6.21}$$

where $|\Omega|$ is the cardinality of set Ω. Assume that $c_1, c_2, \ldots, c_i, \ldots$. are the admissible values the function $c(\omega)$ achieves on Ω. Suppose that in the particular instance of (Ω, c), $n_i, i = 1, 2, \ldots$, configurations have cost c_i and let $\bar{c}$ be the average value of the cost. The quantity n_i/N is the unbiased estimator for p_i, the probability that a solution of (Ω, c) has cost c_i. Hence, to determine the expression for n_i for each i is equivalent to find the probability distribution on Ω .

The numbers $n_i, i = 1, 2, \ldots$, can, in principle, assume any arbitrary posi-

tive value with the only constraints that

$$\sum_i n_i = N \tag{6.22}$$

and

$$\sum_i c_i n_i = N\bar{c}\,, \tag{6.23}$$

where N is given by (6.21) and $\bar{c}$ is a datum of the problem.

If $\mathcal{N}$ is the set of all collections of numbers $\{n_i\}$ which satisfies (6.22) and (6.23), the total number of feasible solutions is given by

$$\Theta = \sum_{\{n_i\}\in\mathcal{N}} \frac{N!}{\prod_i n_i!}\,. \tag{6.24}$$

The statistical entropy for problem (Ω, c) is then defined by

$$S = k \ln \Theta \tag{6.25}$$

with k a positive constant which embodies the choice of the units in which information is measured. The probability distribution which maximizes the uncertainty is obtained by solving the following constrained optimization problem

$$S_* = \max_{\{n_i\}\in\mathcal{N}} S\,. \tag{6.26}$$

To find the solution to (6.26), we can use the Implicit Lagrange Multipliers method [96]. However, before we can apply the method, we have to manipulate (6.26) to put it into a more manageable form. The explicit evaluation of Θ from (6.24) in closed algebraic form is rather difficult, owing to the restriction imposed by $\mathcal{N}$. In spite of that, if the number of solutions is large, we can resort to the following lemma [97]

Lemma 6.4.1 *Given a combinatorial optimization problem with N solutions, if N is sufficiently large the statistical entropy can be approximated with*

$$S \approx k \ln t(n)\,, \tag{6.27}$$

where $t(n)$ is given by

$$t(n) = \max_{\{n_i\}\in\mathcal{N}} \frac{N!}{\prod_i n_i!}\,. \tag{6.28}$$

Proof. The proof of the lemma in its generality is rather cumbersome and is therefore omitted here. In [97, page 28] a simplified version of the proof in the case in which the solutions can only have two different values of the cost is presented. □

If we assume that N is large enough for approximation (6.27) to make sense [2], we can replace (6.25) with

$$S = k \ln t(n) . \tag{6.29}$$

Substituting (6.28) in (6.29) and applying Stirling's formula we have

$$\begin{aligned} S &= k \ln t(n) = \\ &= k\{N \ln N - N - \sum_i \{p_i N \ln p_i N - p_i N\}\} = \\ &= -kN\{p_i \ln p_i\} \end{aligned} \tag{6.30}$$

where n_i have been replaced by $p_i N$. The *Lagrangian function* for (6.26) is then given by

$$\mathcal{L}(\alpha', \beta, p) = -kN\{p_i \ln p_i\} + \alpha'(\sum_i p_i - 1) + \beta(\sum_i c_i p_i - \bar{c}) . \tag{6.31}$$

If we assume that $\mathcal{L}$ is a differentiable function in α', β, and p, the stationary points (either a maximum or a minimum) of (6.26) are determined by the following set of equations

$$\frac{\partial \mathcal{L}(\alpha', \beta, p)}{\partial \alpha'} = 0 , \tag{6.32}$$

$$\frac{\partial \mathcal{L}(\alpha', \beta, p)}{\partial \beta} = 0 , \tag{6.33}$$

$$\frac{\partial \mathcal{L}(\alpha', \beta, p)}{\partial p_i} = 0 \quad i = 1, 2, \ldots \tag{6.34}$$

Substituting (6.31) in (6.32) and (6.33), we obtain again equations (6.22) and (6.23) respectively but written in terms of probabilities p_i instead of relative frequencies n_i. The substitution of (6.31) in (6.34) gives:

$$-\ln p_i + \alpha + \beta c_i = 0 \quad i = 1, 2, \ldots \tag{6.35}$$

[2] The assumption is well taken in practical applications. For example the placement problem of Example 2.1.1 with only 15 macro-cell gives $N \simeq 10^{12}$.

where $\alpha = \alpha' - 1$. Equations (6.35) are decoupled and can be solved separately for p_i to obtain the optimal solution π_i given by

$$\pi_i = e^{\alpha+\beta c_i} \quad . \tag{6.36}$$

Notice that the expression for π_i is a function of the two, yet unknown, Lagrange multipliers α and β. Equations (6.22) and (6.23) provide the values for α and β:

$$e^{-\alpha} = \sum_i e^{\beta c_i} \tag{6.37}$$

and

$$\frac{\sum_i c_i e^{\beta c_i}}{\sum_i e^{\beta c_i}} = \overline{c} \quad . \tag{6.38}$$

The next step is to prove that the choice of π_i indicated by (6.36), (6.37), and (6.38) actually maximizes the statistic entropy S. Let us rewrite (6.30) as follows:

$$\frac{S(p)}{kN} = -\sum_i p_i \ln p_i \quad .$$

Consider any other collection of numbers $\{p_i\}$ which satisfy (6.22) and (6.23) and study the difference

$$\begin{aligned} \frac{S(\pi) - S(p)}{kN} &= -\sum_i \pi_i \ln \pi_i + \sum_i p_i \ln p_i \; = \\ &= -\sum_i \pi_i \ln \pi_i + \sum_i p_i \ln p_i \; + \\ &\quad + \sum_i p_i \ln \pi_i - \sum_i p_i \ln \pi_i \; = \\ &= -\sum_i (\pi_i - p_i) \ln \pi_i - \sum_i p_i \ln \frac{\pi_i}{p_i} \, . \end{aligned} \tag{6.39}$$

Since both π_i and p_i satisfy (6.22) and (6.23), it follows that

$$\sum_i (\pi_i - p_i) \ln \pi_i = \sum_i (\pi_i - p_i)(\alpha + \beta c_i) = 0 \tag{6.40}$$

Substitution of (6.40) into (6.39) gives:

$$\begin{aligned} \frac{S(\pi) - S(p)}{kN} &= -\sum_i p_i \ln \frac{\pi_i}{p_i} = \\ &= -\sum_i p_i \ln(1 + \frac{\pi_i - p_i}{p_i}) \geq \\ &\geq -\sum_i p_i \frac{\pi_i - p_i}{p_i} = 0 \, . \end{aligned}$$

Summarizing, the probability distribution that maximizes the information uncertainty given the constraints (6.22) and (6.23) is

$$\pi_i = \frac{e^{\beta c_i}}{\sum_i e^{\beta c_i}} . \tag{6.41}$$

To see how T comes into the picture, it is necessary to go back to statistical mechanics and recall that

$$\frac{\partial S}{\partial \overline{c}} = -\frac{1}{T} . \tag{6.42}$$

If (6.30) and (6.41) are substituted in (6.42), the computation of the derivative gives

$$\frac{\partial S}{\partial \overline{c}} = \beta . \tag{6.43}$$

From (6.42) and (6.43) follows

$$\beta = -\frac{1}{T} .$$

The method used to find π_i can be generalized to include further information available *a priori* [32,98,99]. For example, if instead of knowing only the first moment of the cost function, there are other r observable functions $f_j, j = 1, 2, \ldots, r$, for which the value of the first moment

$$\sum_i f_j(\omega_i) p_i = \overline{f_j}$$

is known, the equation for π_i becomes

$$\pi_i = \exp\{\alpha + \sum_{j=1}^{r} \beta_j f_j(\omega_i)\} .$$

To conclude, a few comments are in order about the normalizing function

$$Z(\beta_1, \ldots, \beta_r) = \sum_i e^{\sum_{j=1}^{r} \beta_j f_j(\omega_i)} = e^{-\alpha} . \tag{6.44}$$

Function Z is well known in Statistical Information Theory and it is usually referred to as *partition function* or *Gibbs function.* To know Z is equivalent to know exactly the probability distribution of π_i as it is evident from its expression (6.44). Furthermore Z serves also the scope of the moment generating function for the distribution

π_i in the sense that all the moments of f_j can be computed from Z by [98]

$$\overline{f_j^n} = \sum_i f_j^n(\omega_i)\pi_i = \\ = \frac{(-1)^n}{Z(\beta_1,\beta_2,\ldots,\beta_r)} \frac{\partial^n Z(\beta_1,\beta_2,\ldots,\beta_r)}{\partial \beta_j^n} .$$

Finally note that Z is useful to compute not only the moments but also the variance of f_j

$$\Delta f_j = \sum_i (f_j(\omega_i) - \overline{f_j})^2 \pi_i = -\frac{\partial^2 \ln Z(\beta_1,\beta_2,\ldots,\beta_r)}{\partial \beta_j^2} \qquad j = 1,2,\ldots,r .$$

Chapter 7

The Annealing Schedule

In Chapter 3 it has been shown that for any monotonically decreasing sequence of temperatures $\{ T_m \}$ (See Assumption 3.2.1), **SA** converges asymptotically with probability one to the global optimum. However the algorithm is required to execute, at each value of T, an infinite number of transitions to reach the stationary probability distribution (See Theorem 3.2.4).

In Chapter 4 it has been proved that if only one iteration is performed at each value of T, there exists an annealing schedule which is necessary and sufficient to guarantee that **SA** achieves the global optimum with probability one. However, the sequence of temperatures $\{ T_m \}$ of the optimal annealing schedule

$$T_m = \frac{k}{\log(m + m_0)} \ , \tag{4.39}$$

indicates that an infinite number of steps has to be taken for T to reach the zero value.

It is clear that neither of the convergence results outlined above offers a viable strategy to control **SA** when it is applied to solve a practical problem. After all, any combinatorial optimization problem can always be solved in a finite amount of time by using an exhaustive search!

Clearly, in any practical implementation the asymptotic convergence of the algorithm can only be approximated and, consequently, the algorithm is no longer guaranteed to produce the global optimum. Along these lines, it is interesting to

establish how fast the algorithm approaches the stationary probability distribution.

In Chapter 3 it has been proved that the rate of convergence, at fixed T, can be estimated once it is known the second largest eigenvalue of the transition matrix **P** or the value of the conductance $\Phi(T)$ of the underlying graph $\mathcal{G}$. However, as already pointed out, the knowledge of either the second largest eigenvalue of **P** or of $\Phi(T)$ is out of the question in the practical cases.

The bound derived in Chapter 5 is a more useful result. In fact it depends on a number of parameters, namely w, L, r, δ, that are definitely easier to estimate than the eigenvalues or the conductance and are independent of T. However, estimating these parameters in general cases may still be a very difficult task, and even if there are cases in which the estimation of some of them is possible, the bound turns out to be so loose that the number of iterations required is comparable to the number of iterations necessary to perform an exhaustive search of the state space (See Example 5.3.1).

The preceding discussion shows how results from the asymptotic theory cannot be used verbatim to derive an annealing schedule for practical applications of SA. If SA has to be used to solve combinatorial optimization problems efficiently, it is important to answer this other question: Given a finite amount of time, what is the annealing schedule which produces the best solution? The answer to this question is, in general, not known. On the other hand, there is no apparent reason to believe that the optimal finite sequence of temperatures should obey the logarithmic updating rule (4.39). Here is an intuitive explanation of why it should not do so. The sequence produced with the logarithmic rule (4.39) is derived from the condition that the probability to escape any solution which is not the global minimum tends to one as T tends to zero. In Chapter 4 it has been shown that the "escape" condition is tantamount to require

$$\sum_{m=m_0}^{\infty} \exp\left(-\frac{k}{T_m}\right) = \infty \ .$$

The behavior of the series is not changed if any finite collection of its terms is dropped. If we interpret the above comment in terms of the sequence of temperatures, it means that, no matter how small is the value of the initial temperature,

the tail of the series is always sufficient to guarantee that convergence to the global optimum is preserved. This in turn means that the choice of any finite initial subsequence does not have any influence whatsoever on the convergence.

Intuition is substantiated by the work of Strensky [100]. In fact he proved that, at least in a very particular and simple example, the optimal finite schedule consists of a sequence of temperatures that does not follow the logarithmic rule. Quite surprisingly, Strensky's sequence of temperatures starts and ends with two subsequences, of different lengths, in which the temperature is set to zero. The subsequence of intermediate temperatures is monotonically decreasing but it consist of too few entries to establish reliably whether or not it resembles the logarithmic rule. There is no evidence that Strensky's annealing schedule is extendable to more general cases and the peculiarity of the sequence of temperatures found by Strensky makes it hard to believe that such an extension is possible. Apart from any skepticisms, the real problem with this method, as Strensky himself points out in his paper, is that the procedure to find the optimal schedule requires the knowledge of the complete transition probability matrix and a computing effort which is larger than the effort necessary to explore the state space exhaustively.

In conclusion, the inhomogeneous theory developed in Chapter 4, even if more sound from a theoretical point of view than the homogeneous one, offers little or no guidance to develop a finite time schedule.

The homogeneous approach instead gives us some hints. The idea is as follows. We know that the closer we are to the stationary probability distribution π, the faster the convergence to it will be. Suppose that SA is started with an initial temperature, T_0, at which it is relatively easy to achieve a good approximation of the stationary probability distribution. Now assume that m iterations have been completed and that the current probability distribution of the states is a good approximation to $\pi(T_m)$. If the value of T_m is reduced by a small but finite amount to give T_{m+1}, the new stationary probability distribution will not be too far from the old one and hence the approximation to $\pi(T_m)$ should be a good stating point to achieve the approximation to $\pi(T_{m+1})$.

The initial value of T_0, the quality of the approximation to the stationarity

probability distribution, the amount by which T is reduced, and the stop criterion qualify the annealing schedules.

The annealing schedules can be subdivided into two classes: Static schedules and dynamic schedules. A static schedule is a schedule whose parameters are fixed before the algorithm is started. A typical example of such a static schedule is given by the geometric schedule introduced in Section 2.1. In a geometric schedule the temperature is updated according to the geometric rule (2.4) and the equilibrium is considered achieved once a minimum number of moves, proportional to the size of the problem, is attempted at each fixed value of T.

Dynamic schedules have parameters that are modified iteratively with the information that is collected during the operation of the algorithm.

Static schedules are easier to implement than dynamic ones but have the disadvantage that their parameters have to be tuned on the particular application and as such, it is difficult to port the resulting schedule to different applications. On the contrary, the parameters of dynamic schedules are problem independent. Intuitively, we can think of SA running on a problem, as a dynamical system whose control system is provided by the annealing schedule. A static schedule is equivalent to feed-forward control action. Conversely, a dynamic schedule implements a feedback control action. It should be no surprise to discover that static schedules are very sensitive to the the particular value of their parameters unlike dynamic schedules and that the performances of dynamic schedules are superior, in every respect, to those of static schedule.

The remaining of this chapter is dedicated to present the details of a dynamic annealing schedule and to review briefly the other significant approaches to the design of annealing schedules that have been proposed in the literature.

7.1 Adaptive Schedules

An annealing schedule based on the homogeneous approach is completely specified once we assign four parameters: The initial value of the temperature; the temperature update rule; the rule by which we determine if a satisfactory ap-

proximation of the stationarity distribution has been achieved; the rule to stop the algorithm.

To be widely usable, the annealing schedule should be problem independent. This means that the parameters of the four rules should be determined automatically by the algorithm with as little outside intervention as possible. The only way to do this is to make the parameters depend on the data that are collected during execution and let the algorithm modify the tuning of the parameters as soon as new information becomes available.

Before we start describing the details of the adaptive schedule we have to be a little more specific about the model of the algorithm we will be using to derive the schedule.

We want to be able to decide if, given the present value of T, the probability distribution of the states of the Markov chain is close enough to the stationary probability distribution or, in other words, if the Markov chain has reached equilibrium. To keep track of the probability distribution of the states of the Markov chain is very expensive. In fact, to keep track of the relative frequencies of the states, any time a solution is accepted, we must check if the solution has been found already and update the corresponding relative frequency counter or create a new one. The check is rather time consuming since any solution consists of a vector of $2n$ coordinates, with n the number of independent variables considered by SA. Furthermore, to memorize the solutions already found is costly also in terms of memory requirements since the complete vector has to be stored.

On the other hand, we decide whether or not to visit a new state only considering the costs of the states involved in the transition. Hence collecting the statistics of the cost should provide the necessary information to determine the parameters of the annealing schedule.

The theory developed in Chapter 3 and 4 is built on the fact that the algorithm can be modeled by a Markov chain. However the stochastic process

$$Y_n = c(X_n)\,, \tag{7.1}$$

where X_n is the Markov process defined on the solution space and c is the cost

function is, in general, non Markov. The following example presents a case in which X_n is Markov but Y_n is not

Example 7.1.1 Consider a Markov chain with four states [1]. Assume that the generation function is uniform and independent of T and assume that the acceptance function is given by (2.3). The cost function c is such that

$$c_i = \begin{cases} 1 & \text{if } i = 1 \\ 0 & \text{otherwise} \end{cases} . \tag{7.2}$$

and the process Y_n is given by (7.1). The details of the two process are presented in Figure 7.1.

To check if the process Y_n is Markov, we have to prove, following the definition [101], that for any given collection $(y_1, y_2, \ldots, y_n)$

$$\begin{aligned} \Pr\{ Y_n = y_n \quad | \quad & Y_{n-1} = y_{n-1}, Y_{n-2} = y_{n-2}, \ldots, Y_1 = y_1 \} = \\ = \quad & \Pr\{ Y_n = y_n \mid Y_{n-1} = y_{n-1} \} . \end{aligned} \tag{7.3}$$

We will show that there exist at least one case in which (7.3) is not satisfied for the process Y_n, in fact

$$\Pr\{ Y_n = 1 \mid Y_{n-1} = 0, Y_{n-2} = 0, Y_{n-3} = 1 \} \neq \Pr\{ Y_n = 1 \mid Y_{n-1} = 0 \} , \tag{7.4}$$

which in turn implies that Y_n is not Markov.

To check (7.4) simply requires to compute the two terms of the inequality. For the left hand side term we have

$$\begin{aligned} & \Pr\{ Y_n = 1 \mid Y_{n-1} = 0, Y_{n-2} = 0, Y_{n-3} = 1 \} = \\ & = \Pr\{ X_n = \omega_1 \mid Y_{n-1} = 0, X_{n-2} = \omega_2, Y_{n-3} = 1 \} \Pr\{ X_{n-2} = \omega_2 \mid X_{n-3} = \omega_1 \} + \\ & + \Pr\{ X_n = \omega_1 \mid Y_{n-1} = 0, X_{n-2} = \omega_4, Y_{n-3} = 1 \} \Pr\{ X_{n-2} = \omega_4 \mid X_{n-3} = \omega_1 \} . \end{aligned} \tag{7.5}$$

By symmetry considerations and by

$$\Pr\{ X_{n-2} = \omega_2 \mid X_{n-3} = \omega_1 \} = \Pr\{ X_{n-2} = \omega_4 \mid X_{n-3} = \omega_1 \} = \frac{1}{2} ,$$

[1] We are indebted to G. Sorkin for proposing this example

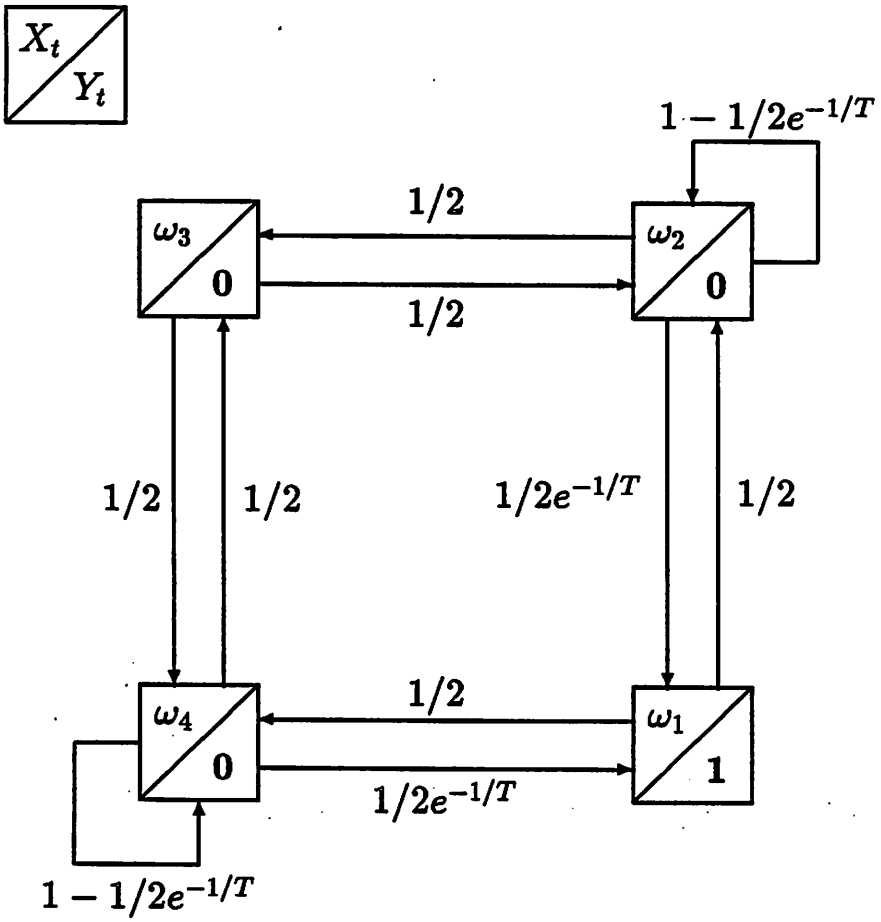

Figure 7.1: Example of a non Markov process $Y_n = c(X_n)$ with X_n Markov. The transition probabilities in the figure refer to process X_n.

equation (7.5) becomes

$$\begin{aligned}
&\Pr\{ Y_n = 1 \,|\, Y_{n-1} = 0, Y_{n-2} = 0, Y_{n-3} = 1\} = \\
&= \{ \Pr\{ X_n = \omega_1 \,|\, X_{n-1} = \omega_2, X_{n-2} = \omega_2, X_{n-3} = \omega_1 \} \\
&\quad \Pr\{ X_{n-1} = \omega_2 \,|\, Y_{n-1} = 0, X_{n-2} = \omega_2, X_{n-3} = \omega_1 \} \} + \\
&+ \{ \Pr\{ X_n = \omega_1 \,|\, X_{n-1} = \omega_3, X_{n-2} = \omega_2, X_{n-3} = \omega_1 \} \\
&\quad \Pr\{ X_{n-1} = \omega_3 \,|\, Y_{n-1} = 0, X_{n-2} = \omega_2, X_{n-3} = \omega_1 \} \} = \\
&= \Pr\{ X_n = \omega_1 \,|\, X_{n-1} = \omega_2 \} \Pr\{ X_{n-1} = \omega_2 \,|\, Y_{n-1} = 0, X_{n-2} = \omega_2 \} \quad (7.6)
\end{aligned}$$

where the last inequality follows from the assumption that X_n is Markov and from

$$\Pr\{ X_n = \omega_1 \,|\, X_{n-1} = \omega_3, X_{n-2} = \omega_2, X_{n-3} = \omega_1 \} = 0 \,.$$

From (7.1) it follows that the event $\{ X_{n-1} = \omega_2 \}$ implies the event $\{ Y_{n-1} = 0 \}$ and hence we can apply the Bayes rule and obtain

$$\begin{aligned} &\Pr\{ X_{n-1} = \omega_2 \,|\, Y_{n-1} = 0, X_{n-2} = \omega_2 \} = \\ &= \frac{\Pr\{ X_{n-1} = \omega_2 \,|\, X_{n-2} = \omega_2 \}}{\Pr\{ Y_{n-1} = 0 \,|\, X_{n-2} = \omega_2 \}} = \\ &= \frac{\frac{1}{2}\left(1 - e^{-1/T}\right)}{\frac{1}{2}\left(1 - e^{-1/T}\right) + \frac{1}{2}} \,. \end{aligned} \tag{7.7}$$

Substituting (7.7) in (7.6) gives

$$\begin{aligned} &\Pr\{ Y_n = 1 \,|\, Y_{n-1} = 0, Y_{n-2} = 0, Y_{n-3} = 1 \} = \\ &= \frac{1}{2} e^{-1/T} \frac{\frac{1}{2}\left(1 - e^{-1/T}\right)}{\frac{1}{2}\left(1 - e^{-1/T}\right) + \frac{1}{2}} \,. \end{aligned} \tag{7.8}$$

The right hand side term of(7.4) gives

$$\begin{aligned} &\Pr\{ Y_n = 1 \,|\, Y_{n-1} = 0 \} = \\ &= \Pr\{ X_n = \omega_1 \,|\, X_{n-1} = \omega_2 \} \Pr\{ X_{n-1} = \omega_2 \,|\, Y_{n-1} = 0 \} \\ &+ \Pr\{ X_n = \omega_1 \,|\, X_{n-1} = \omega_4 \} \Pr\{ X_{n-1} = \omega_4 \,|\, Y_{n-1} = 0 \} \\ &= \frac{1}{3} e^{-1/T} \,. \end{aligned} \tag{7.9}$$

Comparing (7.8) with (7.9) we obtain, for all $T > 0$,

$$-1 = e^{-1/T}$$

which is never satisfied. □

From the previous example it follows that, unless we gather enough evidence that the process Y_n is indeed Markov, we cannot apply the strategy outlined above to determine a viable annealing schedule.

There is a second point that needs clarification. To check equilibrium we have to compare the measured data with their predicted stationary distribution. From the theory developed in previous chapters, we know that if the acceptance function is in

$$f_T(\Delta c_{ij}) = \min\Big[1, \exp\Big(-\frac{\Delta c_{ij}}{T}\Big)\Big],$$

and the generation function is symmetric, the stationary probability distribution of the states is given by

$$\pi_i(T) = \frac{G_i e^{-\frac{c_i}{T}}}{Z(T)}. \tag{4.19}$$

Of course the stationary probability distribution of the cost process Y_n will not have the form (4.19), but given (4.19) and the acceptance function, we can predict the cost distribution at any given T once the cost distribution as T approaches infinity is known. Hence we have to introduce a model for the cost distribution and justify it in view of experimental observations.

The strategy used to check that process Y_n is Markov together with the model for the cost distribution are presented in the next section.

7.1.1 Preliminary Assumptions

Is The Process Y_n Markov?

To use the Definition (7.3) to prove that Y_n is Markov is impossible in practice since it requires the knowledge of the conditional distribution of Y_n which in turns implies the knowledge of $c(X_n)$ for all the possible outcomes of the random variable X_n. In every annealing schedules presented in the literature of which I am aware, process Y_n is assumed to be Markov. As seen in the previous section, this assumption may not hold. It needs to be carefully checked. Ideally, one should prove rigorously that, given the problem we are interested in, Y_n is indeed Markov. However a formal proof is too difficult. Here we assume that Y_n can be considered Markov if the value of the random variable Y_n is shown to depend on the value of Y_{n-1} only. Therefore we will pick some "illustrative" examples and verify, numerically, that Y_n satisfies the above stated criterion.

The procedure used to check that Y_n can be considered Markov, is standard in the theory of time series analysis [102,103]. Here are the details. The Markov property says that the behavior of the process at time n is completely determined by the conditional distribution of Y_n given Y_{n-1}. In other words, all the information needed to predict the value of Y_n is contained in the distribution of Y_{n-1}. Then the idea is to use the data generated by the process Y_n, namely y_n, and verify that the best prediction for Y_n requires only the knowledge of the statistics of Y_{n-1}. If the result of the test is positive, we will conclude that the process Y_n can be considered Markov and hence we can apply the strategy derived from the theory of Chapter 3.

Rewrite the process Y_n in its prediction form as follows

$$Y_n = E[\, Y_n \mid Y_{n-1} \,] + \varepsilon_n \,, \tag{7.10}$$

where $E[\, Y \mid \mathcal{Y} \,]$ is the expectation of random variable Y given the event $\mathcal{Y}$ and ε_n is the residual. The conditional expectation is, by definition, the best step predictor and if the residual ε_n is uncorrelated with the time series Y_n, (7.10) is equivalent to state that the outcome of random variable Y_n is influenced only by the value taken on by random variable Y_{n-1}.

To check (7.10) in practice is impossible since it requires to know the conditional probability distribution of the cost which is unknown. However if we can show that there is a predictor, not necessarily the best one, for which the residual is uncorrelated, then we have reached the conclusion that the process Y_n can be considered Markov.

Assume that y_n is the time series of the costs collected during an execution of annealing at fixed temperature. Let $\tilde{y}_n$

$$\tilde{y}_n = (y_{n-1} - E[y_n]) r_{yy}(1) + E[y_n] \tag{7.11}$$

be the optimal least-square one-step predictor, where $r_{yy}(1)$ is the one step correlation coefficient defined by

$$r_{yy}(1) = \frac{\sum_{n=1}^{N} y_n y_{n-1}}{\sqrt{\sum_{n=0}^{N-1} y_n^2 \sum_{n=1}^{N} y_n^2}}$$

and $E[y_n]$ is the expected value of y_n. If we assume that the process that generated the time series y_n is stationary, then $E[y_n] = E[y]$ for all n and the predictor (7.11) can be replaced by

$$\tilde{y}_n = (y_{n-1} - E[y])r_{yy}(1) + E[y] \; . \tag{7.12}$$

To complete the procedure, we have to compute the prediction error

$$e_n = \tilde{y}_n - y_n \; ,$$

and check that the cross correlation coefficients $r_{ye}(k)$, $k > 1$, defined by

$$r_{ye}(1) = \frac{\sum_{n=1}^{N} y_n e_{n-1}}{\sqrt{\sum_{n=0}^{N-1} y_n^2 \sum_{n=1}^{N} e_n^2}}$$

are zero. The intuitive explanation for this procedure is as follows. The time series e_n represents all the information that can still be extracted from the time series y_n after linear prediction. If e_n has a nonzero correlation with y_n, then the prediction error still carries some useful information and hence either the process is not Markov or the linear predictor is not sufficiently good.

The actual implementation of the test is as follows. We execute SA to generate the time series y_n at fixed value of T. The set of data is partitioned into two sub-sets: The *training set* and the *test set.*

The data from the training set are used to compute the parameters of (7.12), namely $E[y]$ and $r_{yy}(1)$. The obtained linear predictor is then used to check the Markov assumption for the data of the test set .

The data used are generated by running SA on three placement problems [2] . For each of the problems two temperatures representative of different regions of operation of SA are selected. The first temperature is assumed to be infinite, i.e. all the solutions that are generated are accepted. In this case SA is performing a pure random walk on the state space. The second temperature is

[2]The details of the placement problems are presented in Section 8.1.1.

chosen so that the ratio between accepted and attempted moves of is about 50%. In the examples used, this is obtained by setting $T = 10$.

The parameter of the predictor for the three examples are reported in Table 7.1.1. In particular, $r_{yy}(1)$ is the parameter which appears in equation (7.12) while $r_{yy'}(1)$ is the one step cross correlation between the data in the Training set and the data in the Test set. The results of the test are reported in Table 7.2 [3].

$r_{yy}(1)$	sofr255		sohr255		sore255	
	10	∞	10	∞	10	∞
$r_{yy}(1)$	0.993	0.986	0.987	0.985	0.992	0.972
$r_{yy'}(1)$	0.975	0.975	0.989	0.988	0.985	0.962
Notes: Number of data points 3000						

Table 7.1: Parameters of the linear predictor.

From the results contained in Table 7.2, it is evident that the correlation of the prediction error, e_n, with the data y_n is very small compared to $r_{yy}(1)$ reported in Table 7.1.1. Furthermore the sign of the correlation coefficients is not constant and this can be interpreted as an indication that the prediction error is mostly due to the noise which corrupts the time series y_n.

As a final remark, notice that the results presented show that, for the placement problems we used as test case, the best prediction of the outcome of process Y at time n depends only on the outcome of Y at time $n-1$. This does not mean that either the process Y_n is Markov according to the proper definition, or that the findings for the placement problems are to be expected if SA is run on any combinatorial optimization problem. In practice, the analysis presented above should be repeated any time SA with the annealing schedule is applied to a new problem.

[3]The size of both test set and training set is 3000 data points. The data points are the last 6000 data extracted from a sequence of 15000 data points generated by SA. The first 9000 points are dropped to make sure that no transient effects are present in the data.

$r_{ey}(k)$	sofr255		sohr255		sore255	
	10	∞	10	∞	10	∞
1	-0.012	0.023	0.009	0.005	-0.004	-0.008
2	-0.011	0.038	0.011	-0.002	-0.004	-0.006
3	-0.012	0.029	0.019	0.001	-0.004	0.002
4	-0.003	0.028	0.023	-0.003	0.002	0.000
5	-0.013	0.016	0.010	-0.002	0.007	0.000
6	-0.020	0.010	0.014	-0.009	0.011	0.001
7	-0.026	0.008	0.021	-0.012	0.004	-0.009
8	-0.029	0.011	0.022	-0.014	0.011	-0.007
9	-0.026	0.013	0.022	-0.016	0.009	-0.009
10	-0.029	0.017	0.015	-0.015	0.005	-0.012
20	-0.042	-0.001	0.010	-0.015	0.016	-0.013
30	-0.035	-0.011	0.007	-0.013	0.015	-0.019
40	-0.045	0.030	0.003	-0.020	0.022	-0.030
50	-0.045	0.041	0.004	-0.030	0.005	0.043
60	-0.040	0.020	-0.009	-0.021	0.007	0.037
70	-0.026	-0.014	-0.008	-0.001	-0.004	0.026
80	-0.027	0.012	-0.008	0.001	-0.032	0.020
90	-0.046	0.003	0.002	-0.020	-0.034	0.018
100	-0.018	0.003	0.028	-0.025	-0.049	0.010

Table 7.2: Correlation of the error with the data.

Cost Distribution Model

The second assumption is related to the model for the density of the random variable Y_n, namely $\Psi(c)$. The form of $\Psi(c)$ is unknown in general, but whatever its form is, we know that it must satisfy some constraints: First of all, $\Psi(c)$ has to have a finite left tail. In fact the probability to find a cost below the optimal cost must be zero. Second, experimental results show that states with cost close to optimum are rare. We can express this constraint by requiring that only a small portion of the probability measure $\Psi(c)$ is located in the neighbor of c_* and that $\Psi(c)$ decreases rapidly with the size of the neighbor. There are many ways to express the above constraint and one such way is:

$$\sum_i \log(c_i - c_*)\Psi(c_i) = b\,. \tag{7.13}$$

Other constraints are:

i) Finite expected value

$$\sum_i c_i \Psi(c_i) = \overline{c} \ .$$

ii) Normalization constraint

$$\sum_i \Psi(c_i) = 1 \ .$$

To determine the form of $\Psi(c)$, we can follow the same procedure used to derive the exponential form for the π (See Section 6.4) and maximize the statistical entropy subject to the constraints defined above using the method of Implicit Lagrange Multipliers. The Lagrangian function in this case becomes

$$\begin{aligned} \mathcal{L}(\lambda', \beta, \mu, \Psi) &= S + \lambda'(\sum_i \Psi(c_i) - 1) \\ &\quad + \eta(\sum_i c_i \Psi(c_i) - \overline{c}) + \mu(\sum_i \log(c_i - c_*)\Psi(c_i) - b) \ . \end{aligned} \tag{7.14}$$

Taking the derivatives of (7.14) with respect to $\Psi(c_i)$ and to the multipliers λ', η, and μ, we have the following set of equations:

$$\Psi(c_i) = (c_i - c_*)^\mu \, e^{\lambda + \eta(c_i - c_*)} \ , \tag{7.15}$$

$$e^{-\lambda} = \sum_i (c_i - c_*)^\mu \, e^{\eta(c_i - c_*)} \ ,$$

$$\frac{\sum_i (c_i - c_*)^{\mu+1} \, e^{\eta(c_i - c_*)}}{\sum_i (c_i - c_*)^\mu \, e^{\eta(c_i - c_*)}} = \overline{c} \ , \tag{7.16}$$

$$\frac{\sum_i \log(c_i - c_*) \, (c_i - c_*)^\mu \, e^{\eta(c_i - c_*)}}{\sum_i (c_i - c_*)^\mu \, e^{\eta(c_i - c_*)}} = b \ , \tag{7.17}$$

where $\lambda = \lambda' - 1$. Note that $\Psi(c_i)$ defined by (7.15 - 7.17) is the discrete version of the Gamma density [104]

$$\gamma(c; c_*, \alpha, p) = \frac{\alpha^p}{\Gamma(p)} e^{\alpha(c - c_*)} (c - c_*)^{p-1} \tag{7.18}$$

with

$$\Gamma(p) = \int_0^\infty x^{p-1} e^{-t} dt \ .$$

If we assume that the discrete cost distribution $\Psi(c_i)$ can be replaced with the corresponding continuous form, we can replace (7.15 - 7.17) with the more manageable (7.18). The parameters α and p are obtained easily by solving the continuous version of (7.16) and (7.17) for the Lagrange multipliers η and μ.

The continuous gamma distribution is completely specified by the two parameters α, the scale factor, and p the shape factor. Furthermore α and p, define also the first two moments of $\gamma(c; c_*, \alpha, p)$ as shown by

$$E[c] - c_* = \frac{p}{\alpha} \tag{7.19}$$

and

$$\sigma^2(c) = E[c - E[c]]^2 = \frac{p}{\alpha^2} \ . \tag{7.20}$$

The assumption that (7.13) adequately represents the experimental observation that states with cost close to the optimum are rare, is crucial to obtain the gamma distribution for the cost. However, (7.13) is not the only way to represent the constraint. On the other hand, the gamma density has a number of other characteristics that justify its adoption and hence support the use of (7.13). First of all, for very low values of T, the shape of the gamma resembles an exponential density, while at high values of T it approaches a Gaussian density. This fact is important since it is in good agreement with the observed densities of costs produced by runs of SA.

Second gamma is non symmetric. The asymmetry becomes more evident as T is reduced, and again this is in good agreement with experimental observations.

Finally, if we apply SA to an initial cost density which is a gamma, the density is modified by the exponential sampling rule to become

$$\gamma_T(c; c_*, \alpha, p) = \frac{e^{-c/T}\gamma(c; \alpha, p, c_*)}{Z(T)} \ , \tag{7.21}$$

where $Z(T)$ is the usual normalizing term. If (7.18) is substituted into (7.21) we obtain the following

$$\gamma_T(c; c_*, \alpha, p) = \frac{(\alpha + 1/T)^p}{\Gamma(p)} e^{(\alpha + 1/T)(c - c_*)} (c - c_*)^{p-1} \ . \tag{7.22}$$

The function γ_T is again a gamma density with the scale factor $(\alpha + 1/T)$, i.e. gamma is closed under the exponential sampling [4]. Figure 7.2, shows the influence of the parameter T on the shape of the cost density.

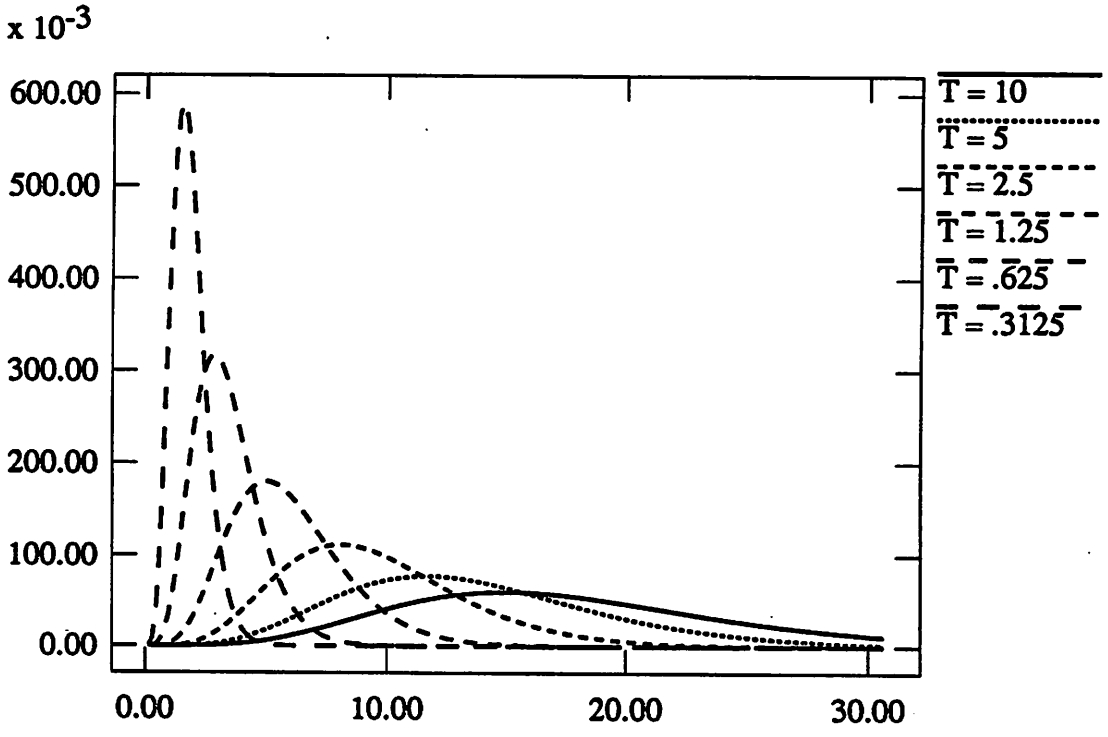

Figure 7.2: Gamma density as a function of T. c_* is assumed 0

From expression (7.22) we can compute the expected value and the variance of the density as a function of T

$$E_T[c] = \frac{p}{\alpha + 1/T} + c_* \tag{7.23}$$

and

$$\sigma_T^2(c) = \frac{p}{(\alpha + 1/T)^2} \ . \tag{7.24}$$

Equations (7.23) and (7.24) can be used to estimate the expected value of the cost and its variance at a given value of T once we know the corresponding values

[4]The closure under the exponential sampling is a property which is common to a number of densities other than gamma. Among them, the Gaussian distribution has the same property as shown by White [105] and by Otten and van Ginneken [67].

for the cost distribution before the sampling procedure is started or, equivalently, for T approaching infinity. In fact, if (7.19) is substituted into (7.23) and (7.20) into (7.24) we obtain the following expressions

$$E_T[c] = E_\infty[c] - \frac{\sigma_\infty^2(c)}{T}\left(\frac{\alpha}{\alpha + 1/T}\right) \tag{7.25}$$

and

$$\sigma_T^2 = \sigma_\infty^2(c)\left(\frac{\alpha}{\alpha + 1/T}\right)^2 \tag{7.26}$$

with α given by

$$\alpha = \frac{E_\infty[c] - c_*}{\sigma_\infty^2(c)} \ ,$$

and $E_\infty[c]$ and $\sigma_\infty^2(c)$ given by (7.19) and (7.20) respectively.

We conclude this section introducing a useful property of functions that are closed under the exponential sampling:

Proposition 7.1.1 *Let $\Psi(c,T)$ be a density function defined on the set C with the property that for all $c \in C$*

$$\Psi(c,T) = \frac{\Psi(c)e^{-\frac{c}{T}}}{Z(T)} \ , \tag{7.27}$$

$Z(T)$ being the partition function which normalizes $\Psi(c,T)$. Then $\Psi(c,T)$ satisfies the following differential equation

$$\frac{d\Psi(c,T)}{dT} = \frac{1}{T^2}\Psi(c,T)(c - E_\Psi[c]) \ , \tag{7.28}$$

where $E_\Psi[c]$ is the expected value of c given the density $\Psi(c,T)$.

Proof. The proof follows directly by computing the derivative of (7.27). □

Consider now the following definition of distance between densities [106]

$$\Xi(p,q) = \sum_{c \in C} \frac{(p(c) - q(c))^2}{q(c)} \ , \tag{7.29}$$

where p and q are two probability densities defined on the same set C [5]. Proposition 7.1.1 provides a simple way to estimate the distance between densities $\Psi(c,T)$

[5] To be formally correct, we should require that p and q are defined on the same measurable space C obtained combining the sample space C with a σ - algebra S of events defined on C [94].

and $\Psi(c, T+\delta T)$. In fact if it is assumed that δT is small enough so that $\Psi(c, T+\delta T)$ can be replaced with its Taylor expansion truncated after the linear term, substitution of (7.28) into (7.29), gives

$$\sum_{c \in C} \frac{(\delta \Psi(c,T))^2}{\Psi(c,T)} = \sum_{c \in C} \frac{(c - E_\Psi[c])^2}{T^4} \Psi(c,T) \, (\delta T)^2 = \\ = \frac{\sigma_\Psi^2(c)}{T^4} \, (\delta T)^2 \, . \tag{7.30}$$

Equation (7.30) gives an estimate, for small variation of T, the distance between two densities as a function of the variance of the cost at T and T^4.

7.1.2 Initial Temperature

The choice of the initial temperature T_0 is the result of a trade off. The higher the temperature, the easier it is to achieve the stationary probability distribution since the space is explored more freely and all the states are almost equally probable. However, the higher the initial temperature, the longer the sequence of temperatures that have to be examined before the algorithm terminates.

Therefore we want to select a finite value of T_0 such that the corresponding probability density is relatively close to the density for T that approaches infinity. We cannot use the estimation for the distance between two gamma densities since we are well outside the range in which the approximation given by (7.30) can be used reliably. In fact, for (7.30) to be true, T has to be finite and δT has to be small enough for the linear approximation to hold.

However we can use (7.25) to determine the initial value for T. From (7.25), since $\alpha \gg 1/T$, $E_\infty[c] - E_T[c]$ is small if $\sigma_\infty^2(c) \ll T$. Hence the criterion we propose is as follows:

$$T_0 = K\sigma(c)$$

with K a small positive number.

It should be noted that, since at high values of T, only a limited number of moves is necessary to achieve a good approximation of the equilibrium distribution, the overhead of selecting a value for K which is too high has relative effect on

the efficiency of the schedule. This assumption is validated by the analysis of the sensitivity of the solution, measured by the value of its cost and by the amount of time required to obtain it, with respect to parameter K.

7.1.3 The Decrement of T

T_m is updated to give T_{m+1} when it is recognized that the present cost density is a good approximation of the stationary cost density at T_m. From the homogeneous theory, it follows that the time required to obtain a good estimate of the stationary cost density at the new value of T_{m+1} is proportional to its distance from the initial density. The idea then, is to control T so that, at two subsequent values of T, the stationary densities are close enough for the current cost distribution to be a good starting point to obtain quickly the desired approximation to the stationary distribution at T_{m+1}.

From equation (7.30), it follows that the distance between two stationary cost densities, γ_T and $\gamma_{T+\delta T}$ is less than a specified number λ^2 if

$$\frac{\sigma_T(c)}{T^2}\delta T \leq \lambda\,. \tag{7.31}$$

The sequence of temperatures that satisfies the bound on the distance of the two stationary distribution imposed by (7.31), can be obtained integrating in time the following differential equation

$$\frac{dT}{dt} = \frac{\lambda T^2}{\sigma_T(c)}\,, \tag{7.32}$$

where $\sigma_T(c)$ is a function of time given by (7.26). However, if (7.26) is rewritten in the following more general form

$$\frac{\sigma_T(c)}{T} = \frac{\sigma_{T^*}(c)}{T^*}\frac{\alpha T^* + 1}{\alpha T + 1}\,, \tag{7.33}$$

we see that for small variations of the difference $(T-T^*)$, the second term to the right hand side is approximately one and, as such, it can be neglected. Hence $\sigma_T(c)/T$ can be considered constant with respect to T. If equation (7.32) is rewritten as

$$\frac{dT}{T} = \frac{\lambda T}{\sigma_T(c)}dt\,, \tag{7.34}$$

we can take advantage of the observation that $\sigma_T(c)/T$ is constant and integrate (7.34) by separation of variables to obtain

$$\log T_{k+1} - \log T_k = \frac{\lambda T}{\sigma_T(c)}$$

which ultimately gives

$$T_{k+1} = T_k \exp\Big(-\frac{\lambda T_k}{\sigma_{T_k}(c)}\Big) . \tag{7.35}$$

Note that at high values of T the ratio of the variance of the cost with respect to T is rather small. Consequently, the exponent of (7.35) becomes large and this causes large variations of T which will invalidate the procedure followed for its derivation. For this reason in the actual implementation of (7.35), we have to make sure that the ratio T_{k+1}/T_k is lower bounded so that the assumptions for (7.35) to be true, are still valid.

7.1.4 The Equilibrium Condition

To reach an equilibrium means that the steady-state probability distribution of the accessible states is established. However, dynamic monitoring of the steady-state condition for all of the accessed states is in practice hardly feasible. The condition we propose is completely based on statistical information collected during the operation of the algorithm.

Once equilibrium is established, the cost distribution approaches a stationary distribution [6] . From the assumption that the cost is distributed as a gamma, we can predict the stationary values for both the expected value and the variance once we know the corresponding values as T approaches infinity.

To achieve the estimates as T approaches infinity, the following procedure is used. The acceptance probability is fixed to one and a fixed number of moves, proportional to the number of the degrees of freedom of the problem, is executed by SA. For every data that becomes available, a random number r is extracted from a uniform distribution on the interval $[0, 1]$ and confronted with a fixed number $s \leq 1/2$. If $r \leq s$ the corresponding datum is assigned to time series #1 otherwise

[6] This is guaranteed in view of the assumption that the process $Y_n = c(X_n)$ is Markov.

it is assigned to time series #2. Stationarity is achieved if the statistics of the two series show, at a specified level of significance, that the data are extracted from the same population.

To check that the data of the two series are extracted from the same population, we proceed along the lines of the standard techniques for testing statistical hypotheses [107]. The data from series #1 are used to estimate the expected value $\overline{E}_\infty$ and the variance $\overline{\sigma^2_\infty}$. With the estimated $\overline{E}_\infty$ and $\overline{\sigma^2}_\infty$, a gamma density, γ_∞, is then built and the following two hypotheses are formulated:

H_0 : The data from Series #2 belong to a population which has density γ_∞.

H_1 : The data of the Series #2 belong to a different populations.

The test used is based on the Kolmogorov-Smirnov statistics [107, page 345].

If the result of the test is in favor of hypothesis H_0, $\overline{E}_\infty$ and $\overline{\sigma^2}_\infty$ are accepted as estimates of the corresponding quantities and the initialization procedure is concluded. In the opposite case, the generation at infinite temperature is continued to obtain another batch of data and the testing procedure is repeated.

Given the estimates $\overline{E}_\infty$ and $\overline{\sigma^2}_\infty$, equations (7.25) and (7.26) give the estimates for $E_T[c]$ and $\sigma^2_T(c)$. A γ_T density with parameters $E_T[c]$ and $\sigma^2_T(c)$ is then generated and the testing procedure described above is repeated, at each value of T, to verify that the costs generated by SA have density γ_T.

The test described above gives correct results if the data are representative of the corresponding population. As a consequence, some precautions are necessary in its implementation. The initial set of data, most likely reflects the transient due to the change in temperature and therefore its influence on the outcome of the test should be relatively small. This is accomplished by requiring that the test is not applied if a minimum number of data has not been generated yet.

To determine minimum number of data that have to be neglected is impossible. In fact to determine the duration of the transient exactly requires the knowledge of the second eigenvalue of $\mathrm{P}(T)$. However, we can use the number of data that was necessary to reach stationarity when T was considered infinite as an estimate of the minimum amount of data necessary for the test to make sense.

On the other hand, we have to account for the case in which our estimates are incorrect and the actual values of $E_T[c]$ and $\sigma_T^2(c)$ are significantly different from the predicted ones. In this case the outcome of the test will be always negative and SA will enter an infinite loop. This is avoided simply by putting a fixed upper bound on the number of moves that are attempted.

However if the limit on the number of moves is hit, we have an indication that the estimates we are using are no longer adequate to predict the future values of both $E_T[c]$ and $\sigma_T^2(c)$. To limit the entity of future errors, the algorithm retunes the reference values that will be used for future predictions. To this purpose, equations (7.25) and (7.26) are replaced with

$$E_{T_2}[c] = E_{T_1}[c] - \frac{\sigma_{T_1}^2(c)}{T_2}\left(1 - \frac{T_2}{T_1}\right)\left(\frac{\alpha + 1/T_1}{\alpha + 1/T_2}\right) \tag{7.36}$$

and

$$\sigma_{T_2}^2(c) = \sigma_{T_1}^2(c)\left(\frac{\alpha + 1/T_1}{\alpha + 1/T_2}\right)^2, \tag{7.37}$$

where T_2 is the temperature at which we need the estimates and T_1 is the reference temperature. Note that as T_1 approaches infinity, (7.36) and (7.37) approach (7.25) and (7.26) respectively. The advantage of (7.36) and (7.37) over (7.25) and (7.26) is that the most recently computed values of $E_{T_1}[c]$ and $\sigma_{T_1}(c)$, can be used to produce more reliable estimations.

The limit on the number of attempted moves is hit more frequently at low values of temperature. The reason for this is twofold: First, at low temperatures, the ratio between attempted moves and accepted moves is rather small and this jeopardizes the reliability of the statistics since the number of useful samples is greatly reduced. Second, towards the end, the algorithm tends to be trapped in a local minimum. Because of this, measured data have both expected value and variance which are consistently lower that the predicted ones. Without re-adjustment, a large number of moves has to be attempted before the measured statistics approach the predicted ones. Intuitively, we have to wait until when SA exits the present local minimum and starts visiting a different portion of the state space and we know that the probability that such event occurs, is exponentially decreasing with both

T and the height of the lowest path that takes out of the domain of attraction of the local minimum.

A comment is in order about the Kolmogorov-Smirnov test. The test requires to compare the measured density with the computed one. In practice this is accomplished by grouping the data in n bins, constructing the cumulated histogram of data S_n, computing the predicted distribution Γ and comparing them by studying the statistics of the random variable D_n

$$D_n = \max_c |\Gamma(c) - S_n(c)| .$$

D_n has a χ^2 distribution with n degrees of freedom and hence its statistics depend on n and, ultimately, on the number of bins that are used to build the histogram. Since the range of variability of data is known only approximately before the data are actually generated, to distribute the bins uniformly over the range of variability may lead to a histogram which is too inaccurate. On the other hand, to build an accurate histogram, may require to alter the distribution of the bins over the variability range iteratively and this procedure becomes rather cumbersome if the size of the sample is very large.

To avoid this difficulty and speed up the execution time, the procedure described above to test the hypotheses, is modified as follows: Compute the weighted moving average of the data

$$\overline{y}_n(r) = \frac{\sum_{k=n-r}^{n} y_k f_{n-k}}{\sum_{k=0}^{r} f_k}$$

with r is the order of the moving average. $f_k \leq 1$, $k = 0, 1, \ldots, r$ are the *oblivion factors* used to assign higher weights to more recent observations [102]. If

$$|E_T[c] - \overline{y}_n(r)| \leq \delta \sigma_T^2(c) \tag{7.38}$$

then compute the histogram and apply the Kolmogorov-Smirnov test. If the test is satisfied with the desired level of significance, then conclude that equilibrium is achieved, otherwise continue.

The parameter δ is of course related to the level of significance selected. We found, in the majority of our experiments, that the outcome of the Kolmogorov-Smirnov test with level of significance 5% is positive if equation (7.38) is satisfied with $\delta = .75$. Furthermore, in the case of a negative outcome of the Kolmogorov-Smirnov test, an additional data sample of size proportional to the size of the problem was always sufficient for the test to give a positive result.

A Simplified Equilibrium Criterion

To conclude this section we present a simplified version of the equilibrium condition that has been developed earlier than the one presented above and that can be considered its prototype.

The idea behind the simplified criterion is the same, the implementation however is different. In the simplified criterion, the equilibrium is considered achieved if the ratio of the number of new generated states with their costs within the range $(E_T[c] - \delta)$, $(E_T[c] + \delta)$ to the total number of the newly accepted states also reaches a stable value, say ρ. If $\Psi(c)$ is the normalized distribution of cost ρ is given by

$$\rho = \Psi(\delta/\sigma) - \Psi(-\delta/\sigma) \ .$$

Given ρ and the size of the problem, a batch of N_a accepted moves is generated by **SA**. During the execution, the number n_a of accepted solutions whose cost is in the range $(E_T[c] - \delta)$, $(E_T[c] + \delta)$, is monitored and compared against ρN_a. If n_a exceeds ρN_a before N_a moves have been accepted, the equilibrium is considered achieved. Conversely, n_a is reset to zero and the algorithm is required to generate a new batch of solutions.

Both ρ and N_a should be updated dynamically to reflect the variations of $\Psi(c)$ as T is decreased. However, in this simplified criterion they are both fixed at the beginning of the algorithm.

Similarly to the condition described above, an upper and a lower bounds on the number of generated moves are introduced.

The annealing schedule with the simplified equilibrium criterion described

above is hereinafter referred to as *the simplified schedule.*

7.1.5 The Stopping Criterion

The scope of the stopping criterion is to recognize that the algorithm has reached a local minimum and T is so small that the probability to escape from it is negligible.

To determine the occurrence of such a situation, after equilibrium is achieved, we compare the difference between the maximum and minimum costs among the accepted states at that value of T with the maximum change in cost in any accepted move at the same value of T. If they are the same, apparently all the states accessed are of comparable costs and there is no need to continue to use SA. T is then set to zero and the algorithm becomes a standard "greedy" random selection algorithm. This mechanism to terminate the annealing has been found to be quite successful without any negative side effects.

7.2 Related Work

Many authors have tackled the problem of finding an "optimal" finite time annealing schedule. In this section we will review briefly some different interpretations of the criteria presented above. Among all the schedules that we will refer to, only the schedule proposed by Delosme and Lam [108] is truly dynamic. In all the others, the criterion to determine if equilibrium has been achieved requires that a minimum number of moves, proportional to the size of the problem, is attempted. Moreover the annealing schedule by Delosme and Lam is different from the others also because it requires to attempt only one move at each value of T.

7.2.1 Initial Temperature

The idea to relate the value of the initial temperature to the value of the standard deviation of the costs at infinite value of T dates back to White's work [105]. White proposed a criterion for the selection of the initial temperature

based on the assumption that the cost has approximately a Gaussian distribution with parameters $E_\infty[c]$ and $\sigma^2_\infty(c)$. If the exponential sampling rule is applied, the following expression for $E_T[c]$ is obtained

$$E_T[c] \approx E_\infty[c] - \frac{\sigma^2_\infty(c)}{T} .$$

White proposes to take the initial value of T such that

$$E_\infty[c] - E_T[c] \leq \sigma(c) .$$

which implies $T \geq \sigma(c)$ similarly to what proposed above. A similar criterion is used by Delosme and Lam.

Aarts and van Laarhoven [68] suggest to select T_0 such that a specified acceptance ratio ρ_0 is achieved. In particular if n moves are attempted at temperature T and the generation probability is uniform, the corresponding acceptance ratio ρ_T is given by

$$\rho_T = \frac{1}{n}\sum_{i=1}^{n} \min\left[1, e^{-\frac{\Delta_i}{T}}\right] , \tag{7.39}$$

where Δ_i is the variation in cost of the i-th move. If $n^+(n^-)$ is the portion of the moves with positive (negative) Δ_i, and Δ^+ is the average of the positive variation in cost, (7.39) can be solved for T to give

$$T_0 = \frac{\log(n^+/(n^+ - (1-\rho_0)n^-))}{\Delta^+} .$$

Criteria similar to Aarts' in spirit but slightly different in the implementation have been proposed by other authors [26,30,56].

7.2.2 The Decrement of T

The condition on the decrement of T which appears most frequently in literature requires to select the new value of T such that for all $i \in \Omega$

$$\frac{1}{1+\lambda_A} \leq \frac{\Psi(c, T_k)}{\Psi(c, T_{k+1})} . \tag{7.40}$$

Aarts and van Laarhoven [68] assume that cost is normally distributed and derive, from (7.40), the following updating rule

$$T_{k+1} = \frac{T_k}{1 + \frac{\log(1+\lambda_A)T_k}{3\sigma_{T_k}(c)}} . \tag{7.41}$$

Notice that if the exponential factor in (7.35) is expanded in power series, the series is truncated after the first term, and λ_A is such that

$$\lambda = \frac{\log(1+\lambda_A)}{3}$$

with λ as in (7.35), we obtain the update rule (7.41). In conclusion, if $\lambda T_k/\sigma_{T_k}(c)$ is small, the rule proposed by Aarts and van Laarhoven, (7.41), gives the same temperature decrement provided by rule (7.35).

Otten and van Ginneken combine the requirement of equation (7.40) with experimental observations to give the following criterion:

$$T_{k+1} = T_k - \xi \frac{T_k^3}{\sigma_{T_k}^2(c)} .$$

For high values of T_k, the factor ξ is related to λ_A by the following relation

$$\xi = \frac{\sigma_{T_k}^2(c)\log(1+\lambda_A)}{T_k(E[c] - T_k\log(1+\lambda_A))} .$$

An approach similar to the one presented in Section 7.1.3 has been proposed by Nulton and Salamon [109]. They assume that each of the state has a different cost and require that the distance between the cost densities at two subsequent temperatures is smaller than a factor λ_N^2. They denote by $dE_T[c]/dt$ the time rate of variation of $E_T[c]$ during the annealing process. From statistical mechanics it follows that

$$\frac{dE_T[c]}{dt} = -\frac{E_{T+\delta T}[c] - E_T[c]}{\epsilon(T)} , \tag{7.42}$$

where $\epsilon(T)$ is a function of the second eigenvalue of the transition probability matrix $\mathbf{P}(T)$. Combining (7.42) with the condition

$$\Xi(\Psi(c, T+dT), \Psi(c,T)) = \lambda_N^2 ,$$

they obtain

$$\frac{dT}{dt} = -\frac{\lambda_N T}{\epsilon(T)\sqrt{(\sigma_T(c)/T)}} . \tag{7.43}$$

Integrating equation (7.43) with respect to time, gives the temperature variation. Notice that even if equation (7.43) has form similar to (7.32), the coefficients of the two differential equations are rather different. In particular notice that to integrate (7.43) the knowledge of $\epsilon(T)$ is required which in turn implies the knowledge of the second eigenvalue of the transition probability matrix $\mathbf{P}(T)$!

A complete different approach is followed by Delosme and Lam [108]. According to their definition, the system is considered to be in equilibrium if it satisfies the following condition

$$\mid \overline{E}_{T_{k+1}}[c] - E_{T_{k+1}}[c] \mid \leq \lambda_D \sigma_{T_{k+1}}(c) , \tag{7.44}$$

where $\overline{E}$ is the expected value of the cost not necessarily at equilibrium. Delosme and Lam assume that the dynamics of $\overline{E}_{T_k}[c]$ are adequately approximated by the first order autoregressive process given by

$$\overline{E}_{T_{k+1}}[c] = r_1(c, T_{k+1})(\overline{E}_{T_k}[c] - E_{T_{k+1}}[c]) + E_{T_{k+1}}[c] , \tag{7.45}$$

where $r_1(c, T_k)$ is the first term of the autocorrelation function for the time series of the cost.

Combining equation (7.44) with (7.45), Delosme and Lam obtain the following expression for the update rule

$$T_{k+1} = \frac{T_k}{1 + \lambda_D \frac{T_k}{\sigma_{T_k}(c)}(1 - r_1(c, T_k))} . \tag{7.46}$$

The final form of (7.46) is again similar to (7.41) with the difference that the coefficient which multiplies the ratio $T_k/\sigma_{T_k}(c)$ depends on the autocorrelation of the time series. In particular, higher values of $r_1(c, T_k)$ correspond to slower update of T while lower values of $r_1(c, T_k)$ force faster updates.

An intuitive explanation for the form of the interaction between T and $r_1(c, T_k)$ is as follows: A high value of $r_1(c, T_k)$, i.e. $r_1(c, T_k)$ close to one, means

that $\overline{E}_{T_k}[c]$ is a good prediction for the corresponding quantity at temperature T_{k+1} which in turn indicates that the process has its statistics still "close" to the stationary statistics at T_k. To put it in another way, the process is slow to respond to variations of T. Consequently, the pace at which T is updated must be slowed down. Similarly, if $r_1(c, T_k)$ is close to zero, the influence of the present value of $\overline{E}$ in determining the corresponding new value is limited compared to the influence of E (See equation (7.45)). Therefore the process is quick to follow variations of T and hence T can be updated at a faster rate.

One final comment is in order about the update rule proposed by Delosme and Lam. From (7.46) it follows that negative values of $r_1(c, T_k)$ generate annealing schedules which are faster than annealing schedules generated from time series with the same autocorrelation coefficient but with the sign changed. This behavior seems counter intuitive since only $|1 - r_1(c, T_k)|$ should matter in determining the amount of variation in T that the system can tolerate.

In a later version of their work [110], Delosme and Lam specify the model for both the cost distribution and for the generation strategy. This allows them to modify (7.46) to give

$$T_{k+1} = \frac{T_k}{1 + \lambda_A \left(\frac{T_k}{\sigma_{T_k}(c)} \right)^3 \alpha(\rho_T)},$$

where

$$\alpha(\rho_T) = \frac{4\rho_T(1 - \rho_T)^2}{(2 - \rho_T)^2}$$

and ρ_T is the acceptance ratio, namely the ratio of accepted moves versus attempted ones.

This result is very interesting since, for the first time, the temperature update is linked with move generation. In particular, the smaller is the acceptance ratio, the slower is the temperature updating. One explanation for this is the following. A small acceptance ratio implies that the number of new configurations that are actually accepted is small and this in turn means that the exploration of the solution space proceeds slowly. Therefore, more time is required to achieve quasi-equilibrium and this explains the slow down in the temperature update.

Notice that as the temperature approaches zero, the acceptance ratio ρ_T goes to zero which in turn implies that the temperature variation approaches zero. If the temperature becomes constant, unless other mechanisms intervene, the algorithm enters an infinite loop. However this situation does not occur. In fact $\sigma_{T_k}(c)$ approaches zero at the same rate as T and this means that the variations of the cost are progressively reduced and, eventually, the stopping criterion will detect that the cost has become stationary and it will terminate the algorithm.

7.2.3 The Equilibrium Condition

To reach an equilibrium means to establish the steady-state probability distribution of the accessible states. However, dynamic monitoring of the steady-state condition for all the accessed states is in practice hardly feasible. Partly because of this the condition that has to be satisfied for the system to reach the equilibrium is the least addressed issue in the annealing processes in the literature. Typically, either a fixed number of generated configurations [68] or certain minimum number of new accepted configurations [12] is used, in the hope that the system will reach equilibrium by then. Both methods are strongly dependent upon the problem that has to be solved and hence hardly portable.

Lam and Delosme [108] propose to take only one step at each temperature and let the update rule take care of the fact that the equilibrium has not been approximated well. Notice however, that the sequence of temperatures generated by Lam's annealing schedule does not resemble the sequence generated by the logarithmic rule (4.39).

7.2.4 The Stopping Criterion

A typical criterion for termination of the annealing process is as follows: The algorithm is terminated when the average cost does not change significantly for few consecutive values of T [12,108].

A different approach is followed by White [105]. He proposes to stop the

process when the final value of T is such that for every local minimum $i \in \Omega_m$

$$\min_{i \in \Omega_m} \min_{j \in \Omega_i} \exp(-\frac{c_j - c_i}{T}) < \frac{1}{\varepsilon} .$$

The above condition gives

$$T_f \leq \max_{i \in \Omega_m} \max_{j \in \Omega_i} \frac{c_j - c_i}{\log \varepsilon} .$$

Notice that $\max_{i \in \Omega_m} \max_{j \in \Omega_i}(c_j - c_i)$ is not known and has to be estimated.

Otten and van Ginneken [67] suggest to terminate **SA** as soon as any further reduction in T will produce an improvement of the expected value of the cost which is ε, $(\varepsilon < 1)$, times the average improvement registered during the execution of the algorithm. The explicit form for the condition is given by

$$\frac{\sigma_\infty^2(c)}{T(E_\infty[c] - E_T[c])} \leq \varepsilon .$$

A similar bound was introduced also by Aarts and van Laarhoven [68].

Chapter 8

Applications of Adaptive Annealing Strategies

In this chapter we describe the results obtained testing the annealing schedule introduced in Chapter 7. The chapter is divided into two sections.

In the first section we concentrate on a particular type of layout problem: Placement of cells on a two-dimensional grid. For this problem we compare the behavior of the new adaptive schedule to the behavior of its simplified version and to the behavior of the geometric schedule in which T is updated by

$$T_{k+1} = \alpha T_k$$

and equilibrium is considered achieved when a minimum number of new moves has been attempted .

In the second one, we discuss some earlier results obtained using the simplified version of the annealing schedule presented in Section 7.1.4. The test problems used are: The traveling-salesman and the standard cell placement problem. These particular problems have been chosen because, for both of them, algorithms based on the simulated annealing but controlled with a static schedule, are available. Furthermore the nature of the problems is different enough to prove the generality of the strategy proposed to control the annealing process.

8.1 The Adaptive Schedule

In this section we concentrate on a particular instance of placement problem and we address the following issues. First we study the improvements achieved by replacing the simplified mechanism used to detect equilibrium with the one presented in Section 7.1.4. Then we show how variations in the values of the fixed parameters that appear in the adaptive schedule have limited influence on the quality of the solution.

8.1.1 Test Case Characteristics

In all the circuits used as test case, the cells have to be placed on a prespecified rectangular grid. The circuits can be classified into three groups according to the characteristics of their connectivity.

The first group has a hierarchical structure. Each example is obtained by combining 4 groups of 4^k cells to obtain a circuit consisting of 4^{k+1} cells. An example of the result of the constructing procedure for $k = 1$ is given by the circuit shown in Figure 8.1. The first group is again subdivided into two subgroups. The only difference between the subgroups is the different set of weights assigned to the connections. Examples with the letter "h" at the end of the name, represent circuit with higher weights. Given the structure of the circuit, we know the value of the cost associated with the global optimum and this is a useful milestone to compare the final solution found by SA.

The second group of circuits consists of examples whose connectivity has been generated at random. In particular, for circuit sohr225 the netlist has been generated according to the following procedure. The n cells which comprise the circuit are partitioned into two subsets S_1 and S_2 of equal size. Then $k(n/2)^p$ pair of cells (c_i, c_j) with $c_i \in S_1$ and $c_j \in S_2$ are selected at random and a connection between the two is created. The same procedure is repeated recursively on S_1 and S_2 until when the subsets consist of two cells only. The coefficients k and p are selected according to the *rent's rule* [111]. Rent's rule gives an estimate of the

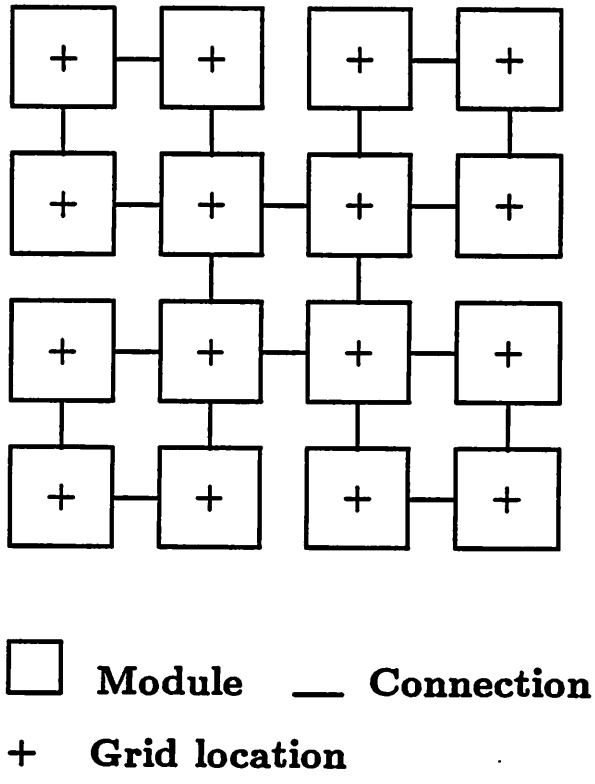

Figure 8.1: Hierarchical example with 16 cells.

number of connections given the number of cells if the example represents a digital circuit. Circuit sohr225 has been generated in particular using $n = 225$, $k = 0.6$, and $p = 2/3$.

Circuit sofr225, instead, has been generated by selecting uniformly at random, a number m of connections among the $n(n-1)/2$ possible connections for the n cells which comprise the circuit. To generate sofr225, m has been taken equal to the number of connections in sohr225 and $n = 225$.

Finally the last example, sore115 represents a real circuit.

In all the cases the move set consists of pairwise interchanges of cells and the cost function is the total wire length measured as the sum of the Manhattan lengths of all the interconnections.

Example Name	# Cells	# Nets	Optimal cost
test64	64	84	292
test256	256	340	2340
test1024	1024	1364	18274
test64h	64	84	1092
test256h	256	340	17476
test1024h	1024	1364	279620
sofr225	225	361	(*) 1000
sohr225	225	361	(*) 780
sore105	105	191	(*) 360
Notes: (*) Estimated value			

Table 8.1: Summary of the characteristics of the test cases.

8.1.2 Improvements Over the Simplified Schedule

To test the quality of the adaptive schedule, we compare the results obtained by using it against the results obtained using the simplified schedule and the geometric schedule. To keep the comparison as meaningful as possible, we fixed the parameters of the adaptive schedule and of the simplified schedule, while we optimized the parameters for the geometric schedule. In particular we run the geometric schedule with different choices of the coefficient α in (2.4) and of the number of attempts per cell and we picked the best combination of the parameters for each example. The set up of the parameters used for the geometric schedule is

Example Name	α	# Moves per Cell
test64	0.9	32
test256	0.9	64
test1024	0.95	32
test64h	0.85	32
test256h	0.85	64
test1024h	0.9	32
sofr225	0.9	64
sohr225	0.9	64
sore105	0.9	64

Table 8.2: Parameters used for the geometric schedule.

summarized in Table 8.2 while the parameters used for both the adaptive and the simplified schedule are given in Table 8.3.

K	3
δ	0.75
λ	0.5
maximum generation	32
minimum generation	4

Table 8.3: Parameters used for both the adaptive and the simplified schedule.

The results obtained are collected in Tables 8.4, 8.5, and 8.6. The columns

Example Name	Final Cost		Attempted Moves		Number
	$\overline{x}$	σ^2	$\overline{x}$	σ^2	Samples
test64	359.6	1864.11	53610	5.14 e+07	50
test256	2700.6	33699.3	855230	7.6 e+09	30
test1024	20034	524722	3.28 e+06	2.16 e+11	20
test64h	1321.8	22166.8	48540	8.44 e+07	20
test256h	19960.8	2.55 e+06	273793	1.71 e+09	20
test1024h	301882	9.65 e+07	1.85 e+06	4.00 e+10	10
sofr225	1117.6	1924.34	234607	8.97 e+08	20
sohr225	906.45	2064.05	264216	1.94 e+09	20
sore105	413	382.2	63961.2	1.33 e+08	20

Table 8.4: Adaptive Schedule.

Example Name	Final Cost		Attempted Moves		Number
	$\overline{x}$	σ^2	$\overline{x}$	σ^2	Samples
test64	358.6	1858.4	59507.2	7.13 e+07	50
test256	2695.3	29922.0	898534	6.09 e+09	30
test1024	19941.0	676940.0	3.79 e+06	2.88 e+11	20
test64h	1319.2	21974.5	48455.6	6.17 e+07	20
test256h	19977.6	2.37 e+06	305267	1.96 e+09	20
test1024h	301943	1.02 e+08	1.89 e+06	4.20 e+10	10
sofr225	1117.0	2001.4	264405	1.39 e+09	20
sohr225	904.8	2067.3	288779	2.01 e+09	20
sore105	411	462.16	79620	1.35 e+08	20

Table 8.5: Simplified Schedule.

Example Name	Final Cost		Attempted Moves		Number
	$\overline{x}$	σ^2	$\overline{x}$	σ^2	Samples
test64	355.8	1397.15	116695	3.06 e+07	50
test256	2729	80297.4	1.23 e+06	9.15 e+09	30
test1024	21411	3.38 e+07	7.71 e+06	4.45 e+11	20
test64h	1217	9125.9	49664	5.82 e+06	20
test256h	19553.1	2.95 e+06	554598	1.56 e+09	20
test1024h	293690	8.54 e+07	2.59 e+06	3.70 e+10	10
sofr225	1054	895.7	486720	1.04 e+09	20
sohr225	833.5	1238.55	473320	6.34 e+08	20
sore105	386	57.94	193200	5.36 e+07	20

Table 8.6: Geometric Schedule.

denominated $\overline{x}$ and σ^2 contain the average value and the variance of either the final cost or the number of attempted moves. The number of samples used to compute the statistics is reported in the last column of the table.

The number of attempted moves is used as an estimator of the time required by the algorithm to produce the final solution. The time itself is not reproduced since it varies significantly with the load average of the computer at the time of the execution.

The analysis of the results shows that, while the values of the cost of the final solutions obtained with the three annealing schedules fluctuate only within few percentage points among each other, the number of attempted moves and hence the amount of time required varies significantly. Table 8.7 summarizes the results of Tables 8.4, 8.5, and 8.6. In particular, the geometric and simplified schedule are compared against the adaptive one. The variations Δ_c and Δ_m are defined by

$$\Delta_c = c_i/c_a \qquad i = G, S\ ,$$
$$\Delta_m = m_i/m_a \quad i = G, S\ ,$$

where c_i (m_i) is the cost (the number of attempted moves) required by the geometric (G) or simplified (S) schedules. c_A (m_A) is the cost (the number of attempted moves) required by the adaptive schedule.

In particular for example test1024 the simplified schedule requires 15% more time then the adaptive schedule, while the time required by the geometric

Example Name	Δ	Variation		
		A	G	S
test64	Δ_c	1.00	0.987	0.997
	Δ_m	1.00	2.17	1.11
test256	Δ_c	1.00	1.01	0.998
	Δ_m	1.00	1.44	1.05
test1024	Δ_c	1.00	1.06	0.995
	Δ_m	1.00	2.34	1.15
test64h	Δ_c	1.00	0.921	0.998
	Δ_m	1.00	1.02	0.998
test256h	Δ_c	1.00	0.980	1.00
	Δ_m	1.00	2.03	1.15
test1024h	Δ_c	1.00	0.973	1.00
	Δ_m	1.00	1.24	1.02
sofr225	Δ_c	1.00	0.944	0.999
	Δ_m	1.00	2.07	1.12
sohr225	Δ_c	1.00	0.919	0.998
	Δ_m	1.00	1.79	1.02
sore105	Δ_c	1.00	0.936	0.995
	Δ_m	1.00	3.02	1.24
Notes: A = Adaptive Schedule G = Geometric Schedule S = Simplified Schedule				

Table 8.7: Variation with respect to the adaptive schedule.

schedule exceed the time required by the adaptive schedule by 132%. However the amount of speed-up is not constant. There are cases in fact in which the improvement in speed obtained by the adaptive schedule with respect to the geometric one is more limited (See for example test1024h).

The adaptive schedule selects a more effective sequence of temperatures which in turns requires a smaller number of trials. In Figure 8.2 the sequence of temperatures produced by the adaptive schedule is compared with the sequence produced by the geometric schedule. The sequence of temperatures selected by the simplified schedule is not reported since it is hardly distinguishable by the one produced by the adaptive schedule. The data refer to sohr225.

Notice that the simplified schedule is slower than the adaptive schedule.

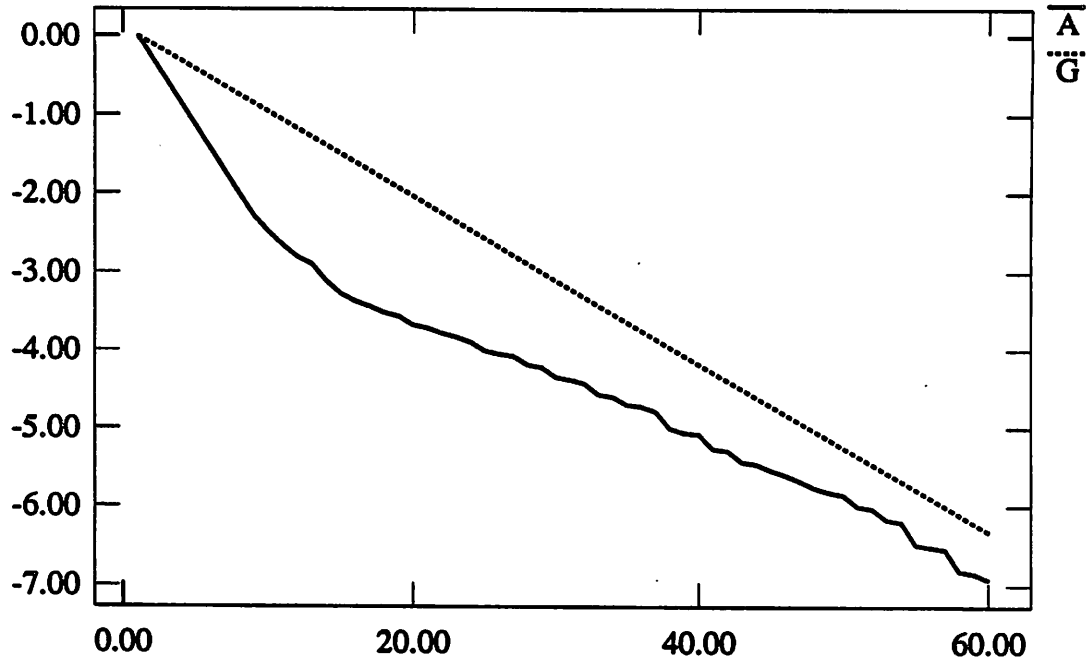

Figure 8.2: Logarithm of T/T_0 versus number of iterations. The straight line represents the updating rule of the geometric schedule with $\alpha = .9$. The piecewise line represents the updating rule of the adaptive schedule.

This is due to the fact that the simplified schedule assumes that equilibrium is achieved once the ratio of moves with cost within a certain range reaches a fixed value. Hidden in this criterion is the assumption that the shape of the distribution does not change as T is reduced. From experience however, it follows that, as T becomes smaller, the cost distribution becomes more and more skewed. Consequently, the simplified schedule ends up wasting time trying to achieve an equilibrium distribution which is incorrect.

To assess the quality of the mechanism we used to predict the expected value and the variance of the cost we collected the data from a run executed on sohr225. Three curves are reported in Figure 8.3. The first one is the measured average cost, the second one is the predicted average cost, and the third one is the minimum value of the cost found.

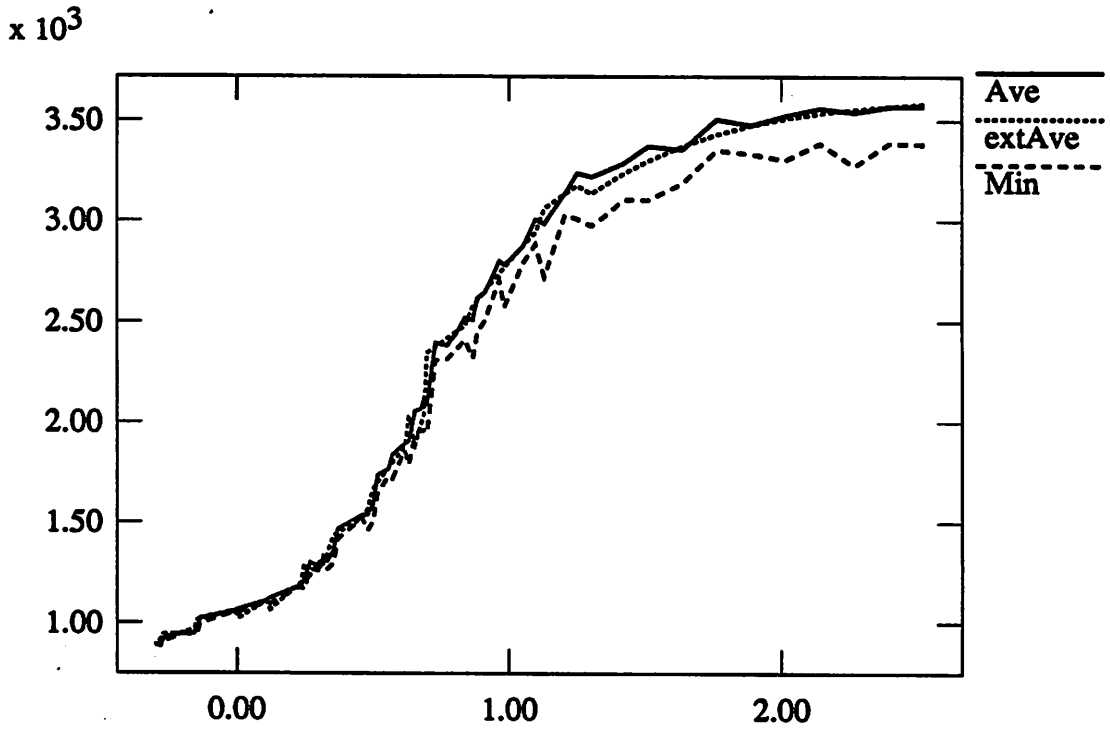

Figure 8.3: Cost versus logarithm of T.

The agreement between prediction and measurements is extremely good for temperature in the medium-high range. Only minor corrections of the predictions are necessary every time that T is updated. At low values of T, predictions have to be readjusted more significantly after every temperature update.

This behavior has two explanations. First of all, in the present implementation of the placement program, the mechanism to generate new moves is rather naive. In fact no control on the type of moves that are generated is implemented and, as a consequence, the acceptance ratio becomes very small towards the end of the execution of the algorithm. A small acceptance ratio reduces the speed at which the state space is explored and hence reduces the significance of the statistics that are collected which in turn affects the reliability of the predictions.

Second, as T approaches zero, the algorithm tends to settle into a local minima. This means that the statistics that are collected are biased. To show this problem we run again SA on the same example but this time we did not do any

adjustment to the estimates. In practice we measured the expected value and the standard deviation of the cost as T approaches infinity and we used (7.25) and (7.26) to compute the predictions. The results are presented in Figure 8.4. Note

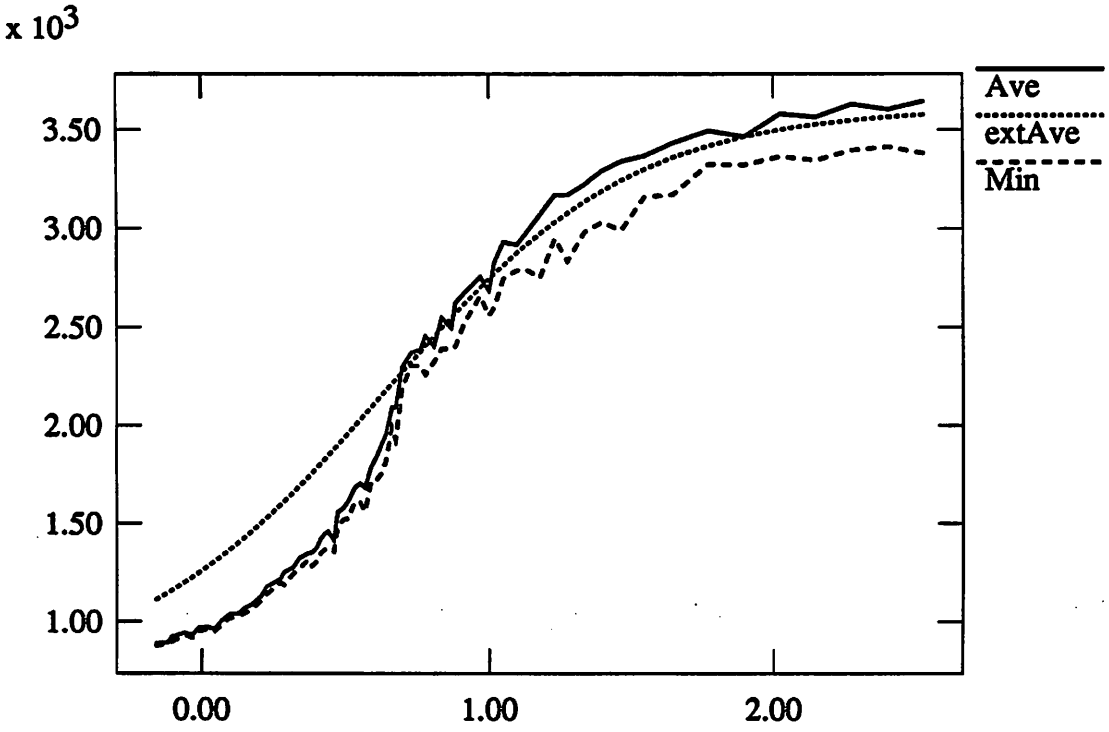

Figure 8.4: Cost versus logarithm of T. No prediction adjustments.

how the predicted value lays a little below of the measured one for high values of T. As T is reduced the gap narrows and, eventually, the measured value becomes smaller that the predicted one to indicate that the algorithm has been trapped in a local minima and that the measured statistics are becoming biased.

Finally, in Figure 8.5 and Figure 8.6 the predicted and measured Gamma densities are compared for three values of T. Figure 8.5 refers to a T at which the acceptance ratio is about .75. Figure 8.6 refers to a very low value of T for which the acceptance ratio is approximately .05. The histogram represents the density of measured data, and the curve represents the gamma density build from the predicted expected value and variance. Notice that while the agreement is fairly good at high values of T, at low values the measured data (those represented by the

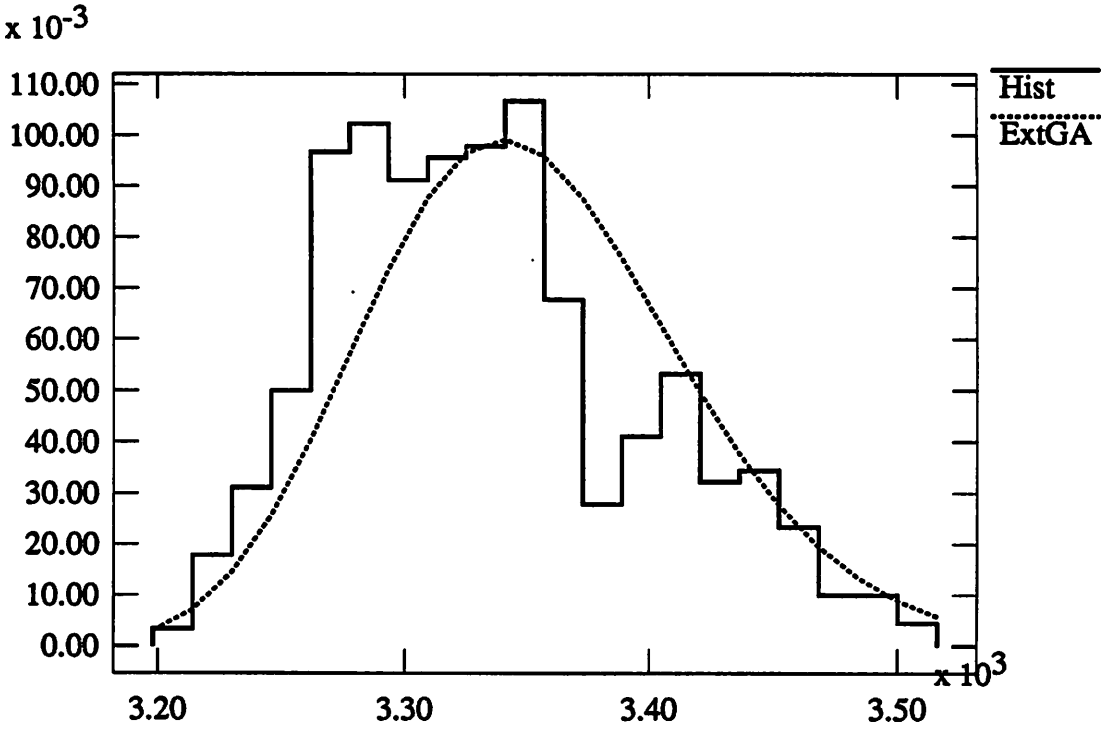

Figure 8.5: Cost density for high value of T.

discrete histogram in the figures) disagree significantly from the predicted curve. In particular in the case represented in Figure 8.6, the Kolmogorov-Smirnov test was never satisfied and the inner loop was exited because of the limit on the maximum number of attempted moves. The data refer to sohr225.

8.1.3 Sensitivity Analysis

The adaptive schedule gives good results with the same set of parameters for all the test cases we tried. However to assess the dependency of the solution on the particular choice of the parameters, we conducted a sensitivity analysis. We selected the three circuits test64, test256, and test1024, and we perturbed each of the parameters, one at a time, to analyze the effects of the perturbation on the result. We repeated the experiment ten times for each choice of the parameters and we recorded the average variation of the cost and of the number of attempted moves in Tables 8.8, 8.9, and 8.10. Δ_c refers to variation in cost while Δ_m refers to

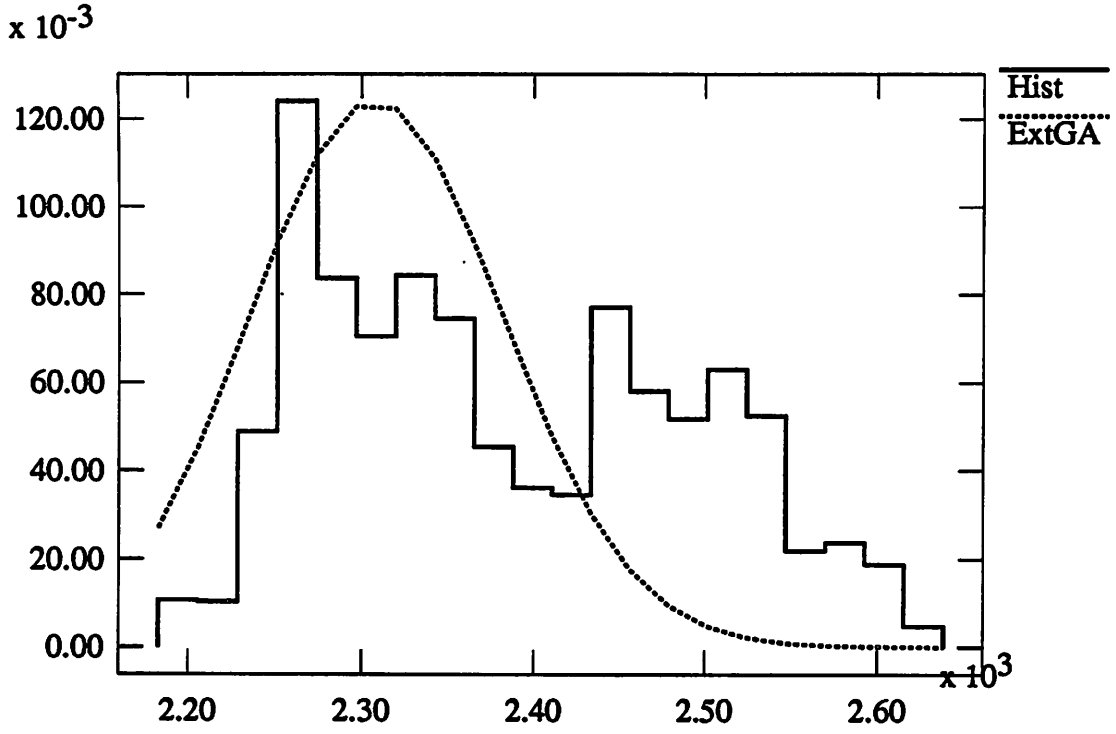

Figure 8.6: Cost density for low value of T.

variation in the number of attempted moves. Both values of Δ are defined by

$$\Delta = \left(\frac{r-p}{r}\right)100.$$

where r represents the unperturbed quantity and p the perturbed one.

Example Name		Cost Variation in %						
		-50%	-25%	-10%	0	10%	25%	50%
test64	Δ_c	-4.73%	-3.69%	7.90%	0.00%	-3.78%	2.83%	-5.94%
	Δ_m	11.47%	12.03%	14.83%	0.00%	6.88%	7.63%	11.36%
test256	Δ_c	2.57%	0.01%	2.60%	0.00%	-2.26%	4.05%	-0.39%
	Δ_m	1.04%	-1.15%	-7.60%	0.00%	-4.51%	-3.25%	-11.03%
test1024	Δ_c	2.72%	-0.05%	0.06%	0.00%	-3.22%	2.20%	0.89%
	Δ_m	6.86%	4.85%	0.50%	0.00%	3.71%	3.61%	-2.18%
Number of Samples = 10								

Table 8.8: Sensitivity with respect to parameter K

Example Name		Cost Variation in %						
		-50%	-25%	-10%	0	10%	25%	50%
test64	Δ_c	3.91%	2.55%	2.77%	0.00%	-3.58%	-4.94%	-1.09%
	Δ_m	-42.27%	-22.11%	-11.81%	0.00%	7.75%	7.58%	20.21%
test256	Δ_c	-3.12%	0.18%	-3.97%	0.00%	0.85%	3.22%	0.88%
	Δ_m	-67.50%	-34.91%	-16.84%	0.00%	-4.27%	1.65%	19.57%
test1024	Δ_c	0.92%	-0.51%	0.38%	0.00%	1.91%	0.17%	-0.48%
	Δ_m	-50.73%	-31.71%	-5.86%	0.00%	4.22%	9.74%	18.15%
Number of Samples = 10								

Table 8.9: Sensitivity with respect to parameter δ

Example Name		Cost Variation in %						
		-50%	-25%	-10%	0	10%	25%	50%
test64	Δ_c	0.45%	-1.45%	-0.50%	0.00%	-1.45%	-2.12%	-1.28%
	Δ_m	-38.24%	-10.94%	-0.74%	0.00%	6.81%	9.59%	14.45%
test256	Δ_c	5.23%	-0.11%	1.20%	0.00%	1.61%	0.68%	-4.15%
	Δ_m	-45.56%	-19.75%	-7.31%	0.00%	-10.02%	5.25%	8.61%
test1024	Δ_c	3.36%	3.16%	0.85%	0.00%	-1.09%	-1.16%	-0.99%
	Δ_m	-44.51%	-30.53%	-8.12%	0.00%	6.16%	14.43%	19.05%
Number of Samples = 10								

Table 8.10: Sensitivity with respect to parameter λ

Analyzing the results, we see that for all the parameters, the differences of cost are always contained within the -5% +5% range. The range of variations combined with the observation that the relative sign does not seem to follow any pattern, are hints that the fluctuations are due more to noise effects than to effective sensitivity of the solution with respect to the parameters.

As expected, the number of attempted moves is more susceptible to the variation of the parameters, particularly δ and λ. In fact a lower values of δ imply higher accuracy in checking the achievement of the equilibrium and hence a larger number of attempts. Similarly, the smaller λ, the smaller the decrement of T which is allowed, hence the larger the number of temperatures that are generated.

Notice however, how the variations in the number of attempted moves induced by positive perturbations, have an absolute value which is smaller that

the negative counterpart. This asymmetry means that the adaptive schedule does a better job in compensating positive variations than it does with negative ones. This can be explained as follows: If λ is increased, the algorithm is forced to take larger increments of T but this is partially compensated by the larger number of moves that is required to achieve equilibrium again. If δ is increased, the algorithm will be less precise in determining the convergence to equilibrium. This will be reflected by a large variability measured by σ which in turn will force a slower updating for T. On the other hand, if λ is reduced, the algorithm will take a smaller number of iterations at each value of T but it will never be allowed to take less iterations than the minimum number required for the accuracy of the estimation. Instead if δ is reduced, the extra time spent to achieve the equilibrium is not compensated by the reduction of σ which is bounded below by the noise of the measurements.

The sensitivity analysis presented above, far from being conclusive, tells at least that the improvements obtained by the adaptive schedule over the plain geometric one are not due to the particular choice of the parameters. Furthermore, if we combine this result with the observation that the parameters of the adaptive schedule were constant for all the test cases, we conclude that the adaptive schedule is fairly robust in the sense that it produces solutions with comparable quality in a smaller time.

8.2 The Simplified Schedule

The only difference between the simplified schedule and the adaptive schedule presented in the previous chapter, is in the mechanism used to check if equilibrium has been established. The simplified schedule has been the prototype for the adaptive schedule. It has been tested on the traveling salesman problem and on the placement of standard cells. The results obtained during the experimentations are reported in the following sections.

8.2.1 Traveling-salesman Problem

The Traveling-salesman problem is a well known $NP - hard$ problem [8, 112]. It consists in determining a tour on a graph such that every node is visited only once and the length of the tour is minimized. In the particular instance of the traveling-salesman problem we selected, the cities, nodes of the graph, are located at the vertices of a square array. The cost is the total Manhattan distance of a closed tour divided by the number of the cities. The intercity distance is one.

A move is carried out by randomly selecting two cities in the tour and reversing the order in which the cities in between are visited. The control parameters of the annealing schedule are reported in Table 8.11. Problems with the number of

parameter	value
K	20
λ	0.7
δ	0.5 σ
maximum generation	$N(N-1)/2$
minimum generation	N

Table 8.11: Parameter setting for the traveling salesman example. N number of cities.

cities N ranging from 49 to 400 are analyzed.

For the particular traveling-salesman problems considered, the global minima are known. Hence the quality of the solution produced by the algorithm can be carefully assessed. Using the control parameters reported in Table 8.11, the standard deviation from the global minimum is less than 2 % for each problem studied.

In Figure 8.7, the CPU time required for the annealings is compared with the result published in [68] in which a geometric annealing schedule is used.

Care has to be taken when comparing the CPU time from various studies. In fact two issues may affect the CPU time required to produce the result in addition to the actual efficiency of the algorithms compared. The first issue involves the targeted quality for the solution. SA may produce oscillations of the solution in the

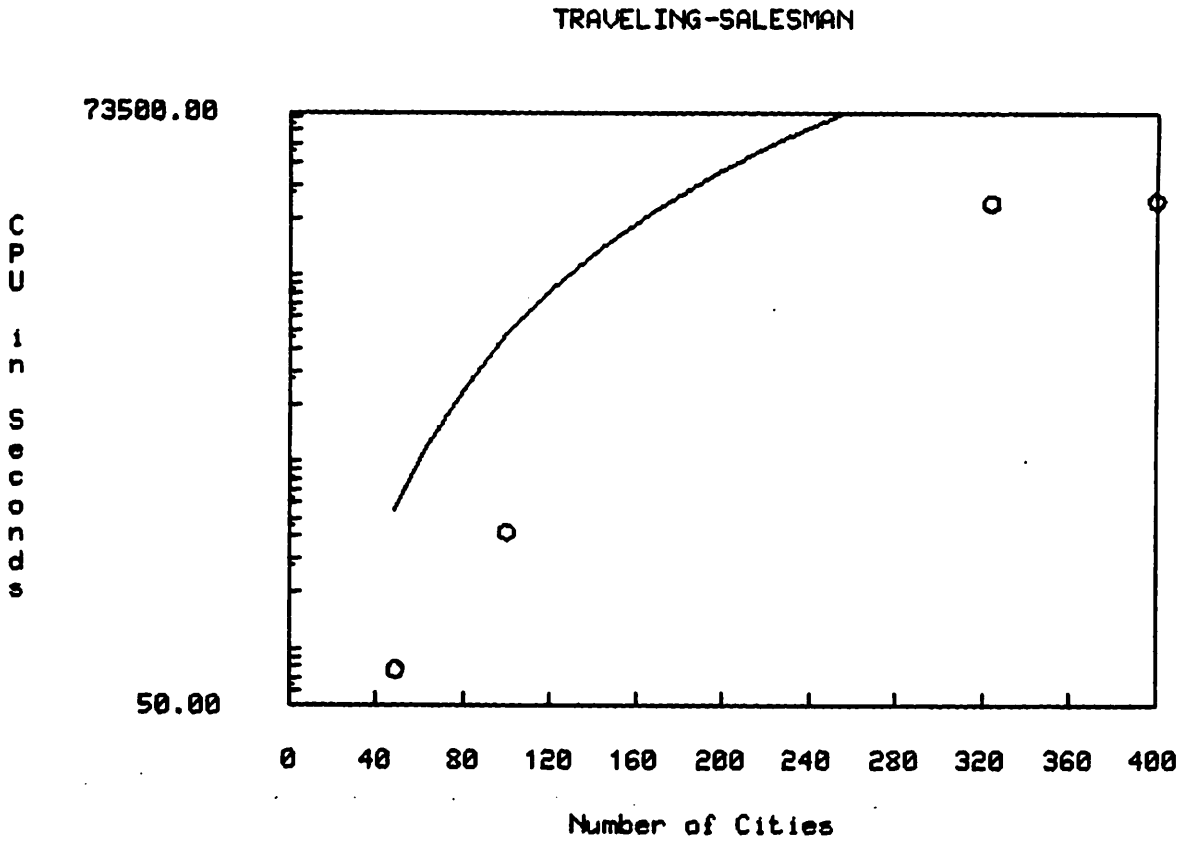

Figure 8.7: Comparison of performance of the new annealing process with the one reported in literature. Solid line is from literature; circles are from the new annealing process.

vicinity of the optimum and a stopping criterion that promptly detects such a situation can save few percents of the global running time. The second issue is related to the differences in the computing environment, both in computer hardware and software. Different efficiency of both operating systems and compilers may produce significant differences in CPU time. In the comparison above, the quality of the solution found by the algorithm is comparable in both studies. The computer hardware used, had the same architecture and the same speed performances measured in MIPS [1] . The operating system and the programming languages used in algo-

[1] Aarts et. al. used a Dec VAX 11-780, we used a DEC VAXstation II/GPX.

rithm implementation are, however, different. In our case we used C programming language and UNIX BSD 4.3 operating system. The program by Aarts et al. is written in Pascal with VMS operating system.

Besides the differences described above, a uniform speed-up of more than five times clearly demonstrates the efficiency of the proposed adaptive schedule. The CPU time saving is obtained mainly by reducing the number of moves attempted at high values of T and by using a more aggressive updating strategy for T.

8.2.2 Placement of Standard Cells

The standard cell layout style is used to assemble sub-blocks of integrated circuits. In this style the cells, each of which implements an elementary logic function, are placed in rows (or columns) and are connected to realize more complex logic functions. The connections are performed in the routing channels available between the rows. The objective of the placement is to select the position of the cells in the rows so that the area of the sub-block, namely the area of the cells plus the routing area, is minimized. Of course the minimization is obtained by reducing the routing area since the cell areas are data of the problem.

TimberWolf3.2 [22] is a very efficient simulated annealing based package for the placement and global routing of standard cells. TimberWolf3.2 features an annealing schedule in which the number of moves that are attempted at each value of T is proportional to the complexity of the circuit, i.e. number of cells, as shown in Table 8.12. T is updated according to a predetermined exponential law.

To test its efficiency, the simplified adaptive schedule has been installed in TimberWolf3.2. All the other features of the original program, namely the move-set, the penalty term to account for overlap, as well as the range-limiting feature in TimberWolf3.2 are left unchanged. The control parameters for the simplified annealing schedule are the same as those used in the traveling-salesman problem discussed in the previous section. In this case, N is the total number of cells.

Four circuits of size ranging from 183 to 800 cells are analyzed using both algorithms. In Table 8.13 the results obtained with the TimberWolf3.2 annealing

No. of cells	Att. per Cell
0 - 200	100
200 - 500	200
500 - 1000	300
1000 - 1500	400
1500 - 2000	500
2000 - 2500	600
2500 - 3000	700
3000 - 3500	800
3500 - 4000	900
4000 - 4500	1000
4500 -	1200

Table 8.12: TimberWolf optimal number of attempts per cell.

process and those obtained with the simplified annealing schedule are indicated with the subscript 1 and 2 respectively. The quantities Δ_t and Δ_w, given by

$$\Delta_t = t_1/t_2 \ ,$$
$$\Delta_w = w_1/w_2 \ ,$$

represent the ratio of CPU time and of estimated wire length obtained by SA with the two different annealing schedules. The results collected in Table 8.13 show that

Size	CPU time (sec.)			Wire Length		
	t_1	t_2	Δ_t	w_1	w_2	Δ_w
183	2301	984	2.33	295585	300448	0.98
286	7330	4536	1.61	317306	330263	0.96
469	11117	5140	2.16	487934	480981	1.01
800	38526	32400	1.19	1270561	1262354	1.007

Table 8.13: Comparisons with TimberWolf Annealing Process.

TimberWolf3.2 with the simplified schedule, is up 2.3 times faster than the same program with the original schedule while maintaining the quality of the solution. Similarly to what experienced with the adaptive schedule, the improvement in CPU time does not exhibit a constant trend with respect to the size of the problem to be solved. This is expected since the proposed schedule, even with the simplified equilibrium criterion, is still adaptive and the number of attempted moves is varied

according to the statistics of the problem to be solved, in order to obtain good quality solutions.

As stated in the previous section, CPU time required by SA varies drastically with the quality of the solution. The speed-up ratio may be reduced if a setting different from the one reported in Table 8.12 for the maximum number of generated moves is used for the original annealing schedule.

To test the above conjecture, TimberWolf3.2 with the original schedule was allowed to run for an amount of time comparable to the time required by the simplified schedule. The results we have obtained are collected in Table 8.14. The

Size	Wire length saving	CPU time
	simplified vs. original	simplified vs. original
183	+2 %	0.97
286	-3 %	1.15
469	+42 %	1.02
800	+1 %	0.97

Table 8.14: Comparisons between the two schedules with the same amount of CPU time.

results represented in Table 8.14 show that the two schedules produce solutions with comparable quality in three out of four cases. In the fourth case the quality of the solution obtained with the new schedule is considerably better. The conclusion we derive is that the adaptive schedule, even in its simplified version, succeeds in the task of determining the exact number of moves to be performed at each value of T necessary to produce good quality results.

Chapter 9

Conclusions

A class of algorithms for the solution of combinatorial optimization problems inspired by the technique known as Simulated Annealing has been presented. The algorithms in the class have the characteristics of being able to climb "hills", i.e. to accept intermediate solutions which increase the cost. The acceptance of up-hill moves is controlled by a probability distribution whose shape is adjusted by means of a parameter, the temperature T.

Any algorithm in the class is completely specified by assigning five parameters: The generate function, the acceptance function, the update function, the inner loop criterion, and the outer loop criterion. The generate function perturbs the current solution to produce the new one. The acceptance function determines whether or not a new generated solution has to be accepted. The update function controls the rate at which the temperature is reduced. The inner loop criterion determines the number of new solutions that have to be generated at each value of the temperature. The outer loop criterion establishes when the algorithm has reached a stationarity point for the cost function and should be terminated.

The structure of the algorithm prompts a number of questions on its behavior: Is the algorithm guaranteed to find the global optimum of the combinatorial optimization problem? Does the initial solution have any influence on the final one? Do the parameters of the algorithm have an influence on the rate of convergence to the optimal solution? To answer these questions a mathematical model of SA has

been proposed. The model represents the algorithm as a stochastic process whose transition probabilities depend only on the states involved in the transition. Due to the characteristics of SA, the process is a Markov chain.

Two theories have been developed to prove the convergence of the algorithm to the global optimum. The first one uses a simplified approach. It assumes that T is kept fixed for a number of iterations which is long enough for the Markov chain to achieve its stationary probability distribution π. With this assumption, the behavior of the Markov chain is completely specified by the stationary probability distribution.

The procedure followed to prove convergence is constructive and provides a method to build an instance of the SA algorithm. In fact, the first step of the procedure requires to select a stationary probability distribution $\pi(T)$ such that, as T goes to zero, only the solutions which are global optima are assigned non zero probabilities. Given $\pi(T)$, the results of the theory provide a rule to select the generate and acceptance functions such that the Markov chain converges to π. The rule is general enough to leave some degrees of freedom in the selection of both the generate and acceptance function even with $\pi(T)$ fixed.

Since stationarity has to be achieved at each value of T, the only limitation imposed on the update function is that the sequence of temperatures has to converge to zero.

The requirement that stationarity must be achieved at each value of T is the most severe limitation of the homogeneous theory. In fact, a part from trivial cases, to require stationarity is tantamount to require that an infinite number of iterations be performed.

The second theory proposed, based on inhomogeneous Markov chains, eliminates the requirement about stationarity at each value of T. The main results of the inhomogeneous theory says that convergence to the global optimum can still be achieved if only one iteration is executed at each value of T. However, in this case the selection of the acceptance function is more limited and the update

rule must obey the following

$$T_m = \frac{k}{\log(m + m_0)} .$$

We showed how the constant at the numerator of the above equation is dependent on the connectivity of the graph underlying the Markov chain and on the Lipschitz constant that bounds the local variation of the cost function. Different expressions for k were derived independently by other authors. k plays a crucial role in proving convergence. In fact while the logarithmic form is a necessary condition for the convergence, there are only a few values of k that make the logarithmic rule sufficient to guarantee convergence. Furthermore there exists a unique choice of k which makes the logarithmic rule both necessary and sufficient for the convergence of SA to the optimum.

The inhomogeneous model is also used to study the finite time behavior of the algorithm. A bound on the distance of the actual probability distribution from the optimum one after a finite number of iterations has been derived.

The bound indicates how the annealing schedule must be balanced between opposing requirements for optimum performance. A simple corollary to this result states that, for large number of iterations m, the L_1-norm of the difference of the present probability distribution from the optimum one is $O(r/m^{\min(a,b)})$, where r, a, and b are quantities characteristic of the problem to be solved. a and b respectively increase and decrease when the parameter k increases.

These results, though important because they show that the algorithm converges asymptotically to the global optimum with probability one, cannot be applied verbatim in practical implementations of SA. In fact the homogeneous theory requires an infinite number of iterations at each value of T while the inhomogeneous one needs only one iteration per temperature but an infinite sequence of temperatures.

The bound on the finite time behavior is relevant since it relates the rate of convergence with the connectivity of the graph underlying the Markov chain and with the local variation of the cost function. However, the number of iterations it requires is comparable with the number of iterations necessary to perform an

exhaustive exploration of the state space.

Among the two theories, the homogeneous one offers a better guidance to design SA algorithms for practical applications. In fact the strategy required by the theory for the asymptotic convergence can be successfully approximated as follows: Start with a value of T for which a good approximation of $\pi(T)$ can be obtained quickly. Reduce T by a finite but limited amount ΔT such that $\pi(T)$ is a good starting point to approximate $\pi(T - \Delta T)$. Keep the temperature fixed until a good approximation to $\pi(T - \Delta T)$ is achieved. Repeat the process until when T is sufficiently close to zero that the algorithm is, for practical purposes, trapped in a local minimum and the execution can be terminated.

The degrees of freedom left by the homogeneous theory can then be exploited to select the acceptance and generate functions that provide the fastest rate at which the approximation to $\pi(T)$ is achieved.

We presented a strategy to select the acceptance function that guarantees that the convergence to $\pi(T)$ at fixed T is the fastest possible. We also discussed a method to generate moves which keeps the ratio between accepted and attempted moves as high as possible.

In order for the approximate strategy to work efficiently, the largest amount ΔT which is tolerable by the algorithm and the criterion to judge the quality of the approximation must be determined. For this purpose, we presented an annealing schedule which efficiently uses the amount of time assigned to the algorithm. The schedule consists of four criteria whose parameters are determined iteratively from the data collected during the execution of the algorithm. The schedule is adaptive and the procedure to determine the value of the parameters makes the schedule independent of the problem to be solved.

The results obtained using SA with the adaptive schedule show that, for a given quality of the results, savings of up to 50% of the CPU time are possible with respect to SA implementations which feature non adaptive schedules. This shows that adaptive schedules are definitely an effective strategy. However an efficient annealing schedule alone is not sufficient. It is necessary to link the annealing schedule to the mechanism used to generate new moves. In fact the reliability of

the parameters determined by the adaptive schedule is related to the amount of information that can be extracted from data. Towards the end of the algorithm, unless the moves are selected carefully, the acceptance ratio approaches zero very quickly and, consequently, the amount of information carried by data is greatly reduced.

The work by Delosme and Lam presents the first results of such a joint control strategy. We believe that this is a very promising direction in which the study of the techniques to control the evolution of SA should be developed. In particular generation mechanisms such as the one based on quality factors appear to be the best suited to be combined with an adaptive annealing schedule.

Both theories developed, make very little assumptions on the data of the problem. Even for the results on the finite time behavior, which require the tightest set of assumptions, it is only assumed that the cost function has a bounded Lipschitz constant, and that the graph underlying the Markov chain is strongly connected. It should be no surprise to discover that the time required to achieve a reasonable approximation of the equilibrium is unrealistically large.

Given that, in practical applications of SA, the asymptotic convergence can only be approximated, it is interesting to investigate if there are characteristics of the cost function and of the move set for which the approximate strategy performs better. Intuitively the cost function on the solution spaces looks like a landscape with peaks and valleys and SA, during its execution, is actually performing a random walk on it. At any given value of T, SA has a sizable probability to climb hills within a certain range of cost, and the size of the range is a function of T. When T is high, SA can hardly distinguish valleys from peaks and hence the random walk is free to wander everywhere. As T is reduced, valleys become more and more evident to SA and it becomes more and more difficult for the random walk to leave one valley and enter another one. Eventually the random walk will remain confined to a valley and will never exit it [1].

[1]Notice that from the point of view of the asymptotic theory, the random walk is never trapped in any of the local minima. In fact the asymptotic results are obtained by requiring that, for any non zero temperature, the probability to escape any solution which is not a global minima never vanishes.

From the theory we know that the probability that the random walk is trapped in a given valley is proportional, for large number of iterations, to the sum of the asymptotic probabilities of the states that populate the valley. Consequently, it seem that the worst performance of the algorithm should be obtained if the cost function has a landscape which looks like a large plateau spotted with very narrow, scarcely populated, valleys, some of which with high lying minima. In this case, in fact, if the temperature is sufficiently high, the algorithm is likely to wander on the plateau entering and exiting different valleys. As the temperature is lowered below a critical value, the random walk will be trapped in the first valley it encounters without being able to escape from it for the rest of the execution. Conversely SA should perform better, if the deeper valley have also higher probability to capture the random walk.

To translate this simple intuition into desirable characteristics for the cost function is anything but easy owing to the structure of the solution space. Sorkin [113] is the only one, to the best of my knowledge, that investigates the characteristics of the cost function for a class of combinatorial optimization problems. Sorkin conjectures that the cost function has a fractal nature and to verify his conjecture he proposes to study the characteristics of the trajectories produced experimentally by SA. If the cost function were indeed fractal, the trajectories produced by SA should have the characteristics of a fractional brownian motion. Sorkin verifies his conjecture showing that, on a number of real-world combinatorial optimization problems, including placement problems for both real and randomly generated circuits, that the data collected executing SA exhibit the properties of fractional brownian motion.

Even if the evidence is not conclusive, the result is interesting. If Sorkin's conjecture holds true and the cost function does exhibit the fractal nature, it should be possible to take advantage of the extra information and use it to select more efficient generate functions and more effective cooling schedules.

The results discussed in this thesis relate to the application of SA to solve optimization problems with a finite state space. There is an equally vast literature that deals with the application of SA annealing to solve optimization problems

defined on a continuous state space.

Szu [88] and Vanderbilt and Louie [114] use SA to solve continuous optimization problems simply by exploring the state space with moves that randomly perturb one of the coordinates by a discrete amount. More interesting is the true continuous time continuous space generalization of SA offered by the Langevin algorithm [90,115,116,117,118,119]. The Langevin algorithm finds the minimum of the vector field $V(x)$ by integrating the following differential equation

$$dX_t = \nabla V(X_t)dt + \sqrt{2T_t}\, dw_t \qquad X_0 = x_0\,,$$

where w_t is the standard Wiener process and ∇ is the gradient operator. The term dw_t represents a noise term whose amplitude is controlled by T that is added to allow the algorithm to escape local minima in the vector field $V(x)$. In [115,116] it is shown that the necessary and sufficient condition for X_t to converge to the global optimum of $V(x)$ requires that T is updated with a rule which is the continuous analogue of the logarithmic rule found for the discrete case.

Gelfand [90] proposed a hybrid SA-Langevin algorithm whose small time behavior resembles that of the annealing algorithm and whose large time behavior is similar to the Langevin algorithm. It is not known if this new hybrid algorithm converges and if its performances are an improvement with respect to the SA algorithm and to the Langevin algorithm.

The convergence result proved for the Langevin algorithm suffers from the same limitations of the analogous asymptotic results proved for the SA algorithm. It will be interesting to investigate if adaptive annealing schedules similar to the one introduced for SA can be applied successfully also to the Langevin algorithm.

Finally the application of SA to neural networks has recently open another very promising and interesting area of research. A neural network is a distributed, massively parallel architecture in which each processing unit, a state, is patterned after a simplified representation of the functional behavior of a neuron. One of the architectures to implement neural networks, perhaps the most famous, has been proposed by Hopfield and Tank [120]. It consists of a network of states, each represented by a binary valued variable. The states are the vertices of a weighted

graph whose edges represent the constraint of the optimization problem. Neural networks provide a fast method to compute the solution to optimization problems. However, their major drawback is that the solution produced is in general only a local optimum for the problem. On the other hand **SA** is very slow to converge but it provides asymptotically the global optimum of the problem. The new idea is to combine the powerful concept of Hopfield's networks with a **SA** to obtain a fast algorithm which locates quickly the global optimum.

The introduction of **SA** in the neural network domain has resulted in two different algorithmic interpretations: The Boltzmann machine and the stochastic neural network. In Boltzmann machines [121,122,123] the implementation of the network is similar to Hopfield's the only difference being that the binary values of the states are determined by **SA** instead of by the Hopfield's greedy strategy. In stochastic neural networks [124], the states are represented by continuous variables and their value is computed with the Langevin algorithm.

The asymptotic theory developed for the **SA** algorithm and for the Langevin algorithm can be applied to Boltzmann machines (See Aarts and Korst [123]) to stochastic neural networks (See Levy and Adams [125]) respectively. However, once again, the asymptotic theory suggests an annealing schedule which is too slow to be of use in practical applications and hence leaves the problem of determining viable approximate strategies still open.

Bibliography

[1] L. Conway and C. Mead, *Introduction to VLSI Systems.* Addison Wesley, October 1980.

[2] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design A System Perspective.* Addison Wesley, 1985.

[3] R. A. Tapia, "Diagonalized Multiplier Methods and Quasi-Newton Methods for Constrained Optimization," *J. Opt. Theory Appl.*, vol. 22, pp. 135–194, 1977.

[4] M. Powell, "Algorithms for Nonlinear Constraints that Use Lagrangian Functions," *Mathematical Programming*, vol. 14, pp. 224–248, 1978.

[5] E. R. Panier, A. L. Tits, and J. N. Herskovits, "A QP-Free, Globally Convergent, Locally Superlinearly Convergent Algorithm for Inequality Constrained Optimization," *SIAM Journal on Control and Optimization*, vol. 26, no. 4, pp. 788–811, 1988.

[6] G. Dantzig, *Linear Programming and Extensions.* Princeton: Princeton University Press, 1963.

[7] E. Polak, *Computational Methods in Optimization a Unified Approach.* Academic Press, 1971.

[8] M. Garey and D. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness.* Freeman, 1979.

[9] S. Cook, "The Complexity of Theorem-Proving Procedure," *Proc. 3rd Ann. ACM Symp. On Theory of Computing*, pp. 151–158, 1971.

[10] R. Karp, "Reducibility Among Combinatorial Problems," *Complexity of Computer Computations*, pp. 85–103, 1972.

[11] J. Schwartz, "Fast Probabilistic Algorithms for Verification of Polynomial Identities," *Journal of ACM*, vol. 27, no. 4, October 1980.

[12] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 13 May 1983.

[13] V. Černy, "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm," *Journ. of Optimization Theory and Applications*, vol. 45, pp. 41–51, 1985.

[14] N. Metropolis, A. Rosenbluth, M. Rosenbluth, and A. Teller, "Equation of State Calculations by Fast Computer Machines," *Journal of Chem. Physics*, vol. 21, p. 1087, 1953.

[15] K. Binder, *Monte Carlo Methods in Statistical Physics.* Springer-Verlag, 1978.

[16] J. Hammersley and D. Handscomb, *Monte Carlo Methods.* New York: J. Wiley and Sons, 1964.

[17] H. Tijms, *Stochastic Modeling and Analysis: a Computational Approach.* Chichester: J. Wiley and Sons, 1986.

[18] B. Ripley, *Stochastic Simulation.* New York: J. Wiley and Sons, 1987.

[19] C. Sechen and A. Sangiovanni-Vincentelli, "The TimberWolf Placement and Routing Package," *Proc. 1984 Custom Integrated Circuit Conference*, pp. 522–527, 1984.

[20] F. Romeo, A. Sangiovanni-Vincentelli, and C. Sechen, "Research on Simulated Annealing at Berkeley," *Proc. CICC*, pp. 652–657, 1984.

[21] C. Sechen and A. Sangiovanni-Vincentelli, "The TimberWolf Placement and Routing Package," *IEEE Journal of Solid-State Circuits*, vol. 20, no. 2, p. 510, April 1985.

[22] C. Sechen and A. Sangiovanni-Vincentelli, "TimberWolf3.2: A New Standard Cell Placement and Global Routing Package," *Proc. 23rd Design Automation Conf.*, pp. 432–439, 1986.

[23] C. Sechen and K. Lee, "An Improved Simulated Annealing Algorithm for Row-Based Placement," *Proc. ICCAD*, pp. 478–483, 1987.

[24] J.Burns, A.Casotto, M.Igusa, F.Marron, F.Romeo, A.Sangiovanni-Vincentelli, C.Sechen, H.Shin, G.Srinath, and H.Yaghutiel, "Mosaico: An Integrated Macro-Cell Layout System," *Proceedings of VLSI '87*, pp. 133–148, August 10-12, 1987.

[25] K. Grover, "A New Simulated Annealing Algorithm for Standard Cell Placement," *Proc. ICCAD*, pp. 378–379, 1986.

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

[27] S. Hustin, *Tim, a New Standard Cell Placement Program Based on the Simulated Annealing Algorithm*. Master's thesis, University of California, Berkeley, March 1988.

[28] P. Banerjee and M. Jones, "A Parallel Simulated Annealing Algorithm for Standard Cell Placement on a Hypercube Computer," *Proc. ICCAD*, pp. 34–37, 1986.

[29] D. Wong, H. Leong, and C. Liu, *Simulated Annealing for VLSI Design*. Boston: Kluwer Academic, 1988.

[30] C. Aragon, D. Johnson, L. McGeoch, and C. Schevon, "Simulated Annealing Performance Studies," *Workshop on Statistical Physics in Engineering and Biology*, April 1984.

[31] A. E. Gamal and I. Shperling, "Design of Good Codes via Simulated Annealing," *Workshop on Statistical Physics in Engineering and Biology*, April 1984.

[32] E. Bonomi and J. Lutton, "The N-City Traveling Salesman Problem: Statistical Mechanics and the Metropolis Algorithm," *SIAM Review*, vol. 26, no. 4, pp. 551–568, October 1984.

[33] H. Fleisher, J. Giraldi, D. Martin, R. Phoenix, and M. Tavel, "Simulated Annealing as a Tool for Logic Optimization in a CAD Environment," *Proc. ICCAD*, pp. 203–205, 1985.

[34] J. Lam and J. Delosme, "Logic Minimization Using Simulated Annealing," *Proc. ICCAD*, pp. 348–352, 1986.

[35] S. Nahar, S. Sahni, and E. Shragowitz, "Simulated Annealing and Combinatorial Optimization," *Proc. 23rd Design Automation Conference*, pp. 293–299, 1986.

[36] D. Bell, F. McErlean, P. Stewart, and W. Arbuckle, "Clustering tuples in Databases," *The Computer Journal*, vol. 31, no. 3, pp. 253–257, 1988.

[37] J. Pincus and A. Despain, "Delay Reduction Using Simulated Annealing," *Proc. 23rd Design Automation Conf.*, 1986.

[38] D. Wong, H. Leong, and C. liu, "Multiple PLA Folding by the Method of Simulated Annealing," *Proc. CICC*, pp. 351–355, 1986.

[39] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. on Pattern Analysis and Machinery Intelligence*, vol. 6, pp. 721–741, 1984.

[40] P. Carnevalli, L. Coletti, and S. Patarnello, "Image Processing by Simulated Annealing," *Journal of Research and Development*, vol. 29, pp. 569–579, 1985.

[41] T. Sejnowski and G. Hinton, "Separating Figures from Ground with a Boltzmann Machine," *Vision, Brain, and Cooperative Computation*, pp. 703–724, 1985.

[42] E. Sontag and H.J.Sussman, "Image Restoration and Segmentation Using Simulated Annealing," *Proc. 24th Conf. on Decision and Control*, pp. 768–773, December 1985.

[43] J. Landa, K. Scheff, and H. Szu, "Binocular Fusion Using Simulated Annealing," *Proc. of IEEE First Annual International Conference on Neural Networks*, pp. IV–327 IV–334, June 1987.

[44] K. Scheff and H. Szu, "1-D Optical Cauchy Machine Infinite Film Spectrum Search," *Proc. of IEEE First Annual International Conference on Neural Networks*, pp. III–673 III–679, June 1987.

[45] A. Sokal, "New Monte Carlo Algorithms for Quantum Field Theory and Critical Phenomena, or How to Beat Critical Slowing Down," *Proc. 8th International Congress of Mathematical Physiscs*, June 1986.

[46] B. Kernighan and S. Lin, "An Efficient Procedure for Partitioning Graphs," *Bell System Technical Journal*, pp. 291–307, February 1970.

[47] C. M. Fiduccia and R. M. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," *Proc. 19th Design Automation Conf.*, pp. 175–181, 1982.

[48] M. Vecchi and S. Kirkpatrick, "Global Wiring by Simulated Annealing," *IEEE Trans. on Computer-Aided Design*, vol. CAD-2, no. 4, pp. 215–222, October 1983.

[49] D. Freedman, *Markov Chains*. Holden-Day, 1971.

[50] F. Romeo and A. Sangiovanni-Vincentelli, "Probabilistic Hill Climbing Algorithms: Properties and Applications," *1985 Chapel Hill Conference on Very Large Scale Integration*, pp. 393–417, 1985.

[51] F. Romeo, *Probabilistic Hill Climbing Algorithms: Properties and Applications.* Master's thesis, University of California, Berkeley, December 1986.

[52] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and Finite-Time Behavior of Simulated Annealing," *Advances in Applied Probability*, vol. 18, pp. 747–771, 1986.

[53] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and Finite-Time Behavior of Simulated Annealing," *Proc. 24th Conf. on Decision and Control*, December 1985.

[54] B. Hajek, "Cooling Schedules for Optimal Annealing," *Mathematics of Operations Research*, 1985.

[55] S. Anily and A. Federgruen, *Probabilistic Analysis of Simulated Annealing Methods.* New York: Graduate School of Business, Columbia University, 1985.

[56] M. Lundy and A. Mees, "Convergence of the Annealing Algorithm," *Mathematical Programming*, vol. 34, pp. 111–124, 1986.

[57] B. Gidas, "Non-Stationary Markov Chains and Convergence of the Annealing Algorithm," *J. Stat. Phys.*, vol. 39, pp. 73–131, 1984.

[58] S. B. Gelfand and S. K. Mitter, "Analysis of Simulated Annealing for Optimization," *Proc. 24th Conf. on Decision and Control*, pp. 779–786, December 1985.

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

[60] W. Rudin, *Principles of Mathematical Analysis.* New York: McGraw-Hill, 1976.

[61] D. Isaacson and R. Madsen, *Markov Chains: Theory and Applications.* New York: J. Wiley and Sons, 1976.

[62] S. Karlin, *A First Course in Stochastic Processes.* Academic Press, 1973.

[63] E. Seneta, *Non-negative Matrices and Markov Chains.* New York: Springer-Verlag, 1981.

[64] F. Gantmacher, *The Theory of Matrices.* New York: Chelsea Publishing Co., 1977.

[65] S. Ross, *Stochastic Processes.* New York: J. Wiley and Sons, 1983.

[66] J. Kielson, *Markov Chain Models: Rarity and Exponentiality.* New York: Springer-Verlag, 1979.

[67] R. J. M. Otten and L. P. P. van Ginneken, "Floorplan Design using Simulated Annealing," *Proc. ICCAD*, pp. 96–98, 1984.

[68] E. L. Aarts and P. M. van Laarhoven, "Statistical Cooling: A General Approach To Combinatorial Optimization Problems," *Philips J. Res*, vol. 40, pp. 193–226, 1985.

[69] A. Sinclair and M. Jerrum, "Approximate Counting, Uniform Generation and Rapidly Mixing Markov Chains," Tech. Rep., Dept. of Computer Sciences, University of Edinburgh, Edinburgh, Scotland, October 1987.

[70] G. Sorkin, "Private Communication," 1988.

[71] Y. Rossier, M. Troyon, and T. Liebling, "Probabilistic Exchange Algorithms and Euclidean Traveling Salesman Problems," Tech. Rep. RO 851125, EPF, Lousanne, 1985.

[72] U. Faigle and R. Schrader, "On the Convergence of Stationary Distributions in Simulated Annealing Algorithms," *Information Processing Letters*, vol. 27, no. 4, pp. 189–194, April 8, 1988.

[73] M. Iosifescu, *Finite Markov Processes and Their Applications.* New York: J. Wiley and Sons, 1980.

[74] R. Dobrushin, "Central Limit Theorem for Nonstationary Markov Chains, I, II," *Theory Prob. Appl*, pp. 65–80, 329–83, (English Translation), 1956.

[75] R. Madsen and D. Isaacson, "Strongly Ergodic Behavior for Non-Stationary Markov Processes," *Ann. Prob*, vol. 1, no. 2, pp. 329–335, 1973.

[76] S. Anily and A. Federgruen, "Simulated Annealing Methods with General Acceptance Probabilities," *Journal of Applied Probability*, vol. 24, pp. 657–667, 1987.

[77] W. Hastings, "Monte Carlo Sampling Methods using Markov Chains and Their Applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[78] I. Bohachevsky, M. Johnson, and M. Stein, "Generalized Simulated Annealing for Function Optimization," *Technometrics*, vol. 28, no. 3, pp. 209–217, August 1986.

[79] R. Holley and D. Stroock, "Simulated Annealing via Sobolev Inequalities," *Communications in Mathematical Physics*, vol. 115, no. 4, pp. 553–569, April 1988.

[80] J. Tsitsiklis, *Markov Chains with Rare Transitions and Simulated Annealing.* MIT Laboratory for Information and Decision Systems, August 1985.

[81] W. Davenport, *Probability and Random Processes; an Introduction for Applied Scientists and Engineers.* New York: McGraw-Hill, 1970.

[82] D. Connors and P. Kumar, "Balance of recurrence Orders in Time-Inhomogeneous Markov Chains with Applications to Simulated Annealing," *Probability in the Engineering and Informational Sciences*, vol. 2, no. 2, pp. 157–184, 1988.

[83] J. Kemeny and J. Snell, *Finite Markov Chains.* New York: Springer-Verlag, 1976.

[84] J. Greene and K. Supowit, "Simulated Annealing Without Rejected Moves," *IEEE Trans. on Computer-Aided Design*, vol. CAD-5, no. 1, pp. 221–228, January 1986.

[85] S. Hustin and A. S. Vincentelli, "Tim, a New Standard Cell Placement Program Based on the Simulated Annealing Algorithm," *Proceedings of International Workshop on Placement and Routing*, May 1988.

[86] M. Jackson, "WILDCAT: A VLSI Placement Program for Macrocell Layout," Tech. Rep., Dept EECS, College of Engineering, University Of California, 1985.

[87] Szu, "Fast Simulated Annealing," *AIP Conf. Neural Net. for Computing*, 1986.

[88] H. Szu, "Fast Simulated Annealing," *Phisics Letters A*, vol. 122, no. 3,4, pp. 157–162, June 1987.

[89] S. Gelfand and S. Mitter, "Simulated Annealing with Noisy or Imprecise Energy Measurements," Tech. Rep., Massachusetts Institute of Technology, Cambridge, 1987.

[90] S. Gelfand, "Analysis of Simulated Annealing Type of Algorithms," Tech. Rep. LIDS-TH-1688, Massachusetts Institute of Technology, Cambridge, 1987.

[91] A. Casotto, F. Romeo, and A. Sangiovanni-Vincentelli, "A Parallel Simulated Annealing Algorithm for the Placement of Macro-Cells," *IEEE Trans. on Computer-Aided Design*, vol. CAD-6, pp. 838–847, September 1987.

[92] F. Darema-Rogers, S. Kirkpatrick, and V.A.Norton, "Simulated Annealing on Shared Memory Parallel Systems," *IBM Journal on Research and Development*, 1987.

[93] S. Kravitz and R. Rutenbar, "Multiprocessor-based Placement by Simulated Annealing," *Proc. 23rd Design Automation Conf.*, pp. 567–573, 1986.

[94] S. Kullback, *Information Theory and Statistics.* London: J. Wiley and Sons, 1959.

[95] R. Gallagher, *Information Theory and Reliable Communication.* New York: J. Wiley and Sons, 1968.

[96] D. Luenberger, *Introduction to Linear and Nonlinear Programming.* Addison-Wesley, 1984.

[97] G. Rushbrooke, *Introduction to Statistical Mechanics.* London: Oxford University Press, 1967.

[98] E. Jaynes, "Information Theory and Statistical Mechanics," *Physical Review*, vol. 104, no. 4, pp. 620–630, May 15, 1957.

[99] A. Katz, *Principles of Statistical Mechanics; the Information Theory Approach.* San Francisco: W. H. Freeman, 1967.

[100] P. Strensky, "Optimal Annealing Schedules: A Case Study," Tech. Rep. 12923, I.B.M. T.J. Watson Research Center, 1987.

[101] E. Wong and B. Hajek, *Stochastic Processes in Engineering Systems.* New York: Springer-Verlag, 1985.

[102] T. Anderson, *The statistical Analysis of Time Series.* New York: J. Wiley and Sons, 1971.

[103] P. Brockwell and R. Davis, *Time Series: Theory and Methods.* New York: Springer-Verlag, 1987.

[104] K. Bowman and L. Shenton, *Properties of Estimators for the Gamma Distribution.* New York: Marcel Dekker Inc., 1988.

[105] S. White, "Concept of Scale in Simulated Annealing," *Proc. ICCD*, pp. 646–651, 1984.

[106] B. V. Gnedenko and A. Khinchin, *An Elementary Introduction to the Theory of Probability.* San Francisco: W. H. Freeman, 1961.

[107] P. G. Hoel, *Introduction to Mathematical Statistics.* New York: J. Wiley and Sons, 1962.

[108] J. Lam and J. Delosme, *An Adaptive Annealing Schedule.* New Haven, Connecticut: University of Yale, 1987.

[109] J. Nulton and P. Salamon, "Statistical Mechanics of Combinatorial Optimization," *Physical Review A*, vol. 37, no. 4, pp. 1351–1356, February 15, 1988.

[110] J. Lam and J. Delosme, "Simulated Annealing: A Fast Heuristic for Some Generic Layout Problems," *Proc. ICCAD*, pp. 510–513, 1988.

[111] W. Heller, W. Mikhail, and W. Donath, "Prediction of Wiring Space Requirements for LSI," *J. Design Automation and Fault-Tolerant Computing*, pp. 117–177, 1978.

[112] E. Lawler, J. Lenstra, A. G. R. Kan, and D. Shmoys, *The Traveling Salesman Problem.* Chichester: J. Wiley and Sons, 1985.

[113] G. Sorkin, *Combinatorial Optimization, Simulated Annealing and Fractals.* Master's thesis, University of California, Berkeley, December 1987.

[114] D. Vanderbilt and S. G. Louie, "A Monte Carlo Simulated Annealing Approach to Optimization over Continuous Variables," *Journal of Computational Physics*, vol. 52, no. 2, pp. 259–271, November 1984.

[115] B. Hajek, "A Tutorial Survey of Theory and Applications of Simulated Annealing," *Proc. 24th Conf. on Decision and Control*, pp. 755–760, December 1985.

[116] B. Gidas, "Global Minimization via the Langevin Equation," *Proc. 24th Conf. on Decision and Control*, pp. 774–778, December 1985.

[117] H. Kushner, "Asymptotic Global Behavior for Stochastic Approximations and Diffusions with Slowly Decreasing Noise Effects," Tech. Rep. 85-7, Lefshetz Center for Dynamical Systems, Brown University, 1985.

[118] S. Geman and C. Hwang, "Diffusions for Global Optimization," *SIAM Journal on Control and Optimization*, vol. 24, no. 5, pp. 1031–1043, September 1986.

[119] T. Chiang, C. Hwang, and S. Shiu, "Diffusions for global Optimization in $\mathbb{R}^n$," *SIAM Journal on Control and Optimization*, vol. 25, no. 3, pp. 737–753, May 1987.

[120] J. Hopfield and D. Tank, "Neural Computations of Decisions in Optimization Problems," *Biological Cybenetics*, vol. 52, no. 3, pp. 141–152, 1985.

[121] G. Hinton and T. Sejnowski, "Optimal Perceptual Inference," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 448–453, 1983.

[122] G. Hinton and T. Sejnowski, "Learning and Relearning in Boltzmann Machines," *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, pp. 282–317, 1986.

[123] E. Aarts and J. Korst, "Simulated Annealing and Boltzmann Machines: a Stochastic approach to Combinatorial Optimization and Neural Computing," Tech. Rep., Philips Research Laboratories, 1988.

[124] J. Cervantes and R. Hildebrant, "Comparison of Three Neuron-Based Computation Schèmes," *Proc. of IEEE First Annual International Conference on Neural Networks*, pp. III–657, III–671, June 1987.

[125] B. Levy and M. Adams, "Global Optimization with Stochastic Neural Networks," *Proceedings of the First International Conference on Neural Networks*, pp. III–681,III–690, June 1987.