SCIENTIFIC PROCEEDINGS OF RIGA TECHNICAL UNIVERSITY
Series - *Computer Science. Boundary Field Problems and Computer Simulation.* - 42nd thematic issue:
ISBN
2000

Solving the Travelling Salesman Problem by classical simulated annealing method

P. S. Grabusts
Riga Technical University, Riga, Latvia

Keywords: Optimization, Travelling Salesman Problem, Simulated Annealing

ABSTRACT: This study describes an optimization method called Simulated Annealing (SA) whose algorithm is employed in artificial neural networks, particularly, in the Boltzmann machine. The SA method is widely used in various combinatorial optimization tasks. In this paper the application of the SA method to a well-known task of combinatorial analysis - Travelling Salesman Problem (TSP) - is demonstrated and an experiment aimed to find the shortest tour distances between 25 Latvian towns is performed.

1 INTRODUCTION

Simulated Annealing (SA) is a stochastic optimization method that can be used to minimize the specified cost function (energy) given a combinatorial system with multiple degrees of freedom. This method enables one to find a global extremum for a function that has local minimums. The foundations of SA were introduced in (Kirkpatrick et al. 1983) and then developed in (Laarhoven et al. 1987). Nowadays the SA method is viewed of as an independent research area and is distinguished from artificial neural network techniques. It is possible to incorporate SA properties into the neural network architecture in a variety of ways particularly by including SA in the stage of learning. The prototype used was the Boltzmann machine to train which the SA method was employed.

2 PHYSICAL ANALOGY

2.1 *Annealing process*

This is the analogy with statistical mechanics and, specifically, with elements of solid-state physics that underlies the SA method. A practical example from metallurgy can be given here. What happens with the atomic structure of a solid when it is being annealed very quickly, i.e. when the temperature is decreasing? A sharp temperature falling can lead to the system's asymmetric structure or, in other words, to a non-optimal state containing errors. In the long run annealing leads to a state when a system becomes chilled or frozen and the thermal equilibrium is established.

The so-called Metropolises procedure determines iterative steps that control arriving at the best solution. This algorithm is employed to simulate reaching the atomic equilibrium at the given temperature. At each step of the algorithm atom is set with a slight probabilistic displacement (deviation): $x_i + \xi$, and a variation of system energy, ΔE, is calculated.

- If $\Delta E \leq 0$, then displacement is accepted and a configuration with the atom states changed is employed as the initial state at the next step;
- If $\Delta E > 0$, then a probability that the new state is accepted will be

$$P(\Delta E) = \exp(-\Delta E/kT) \quad (1)$$

where k = Boltzmann's constant; T= parameter: temperature.

Provided the system energy is employed as a cost function and system states are defined as $\{x_i\}$, one can see that the Metropolises procedure generates a variety of states for the given optimization task at the specific temperature.

2.2 *Combinatorial optimization*

Methods that are commonly employed to solve combinatorial optimization problems are distinguished into two deterministic and stochastic. Deterministic methods require not only huge computational resources but they also do not ensure finding the global minimum of a sufficient accuracy. Stochastic methods, in their turn, use statistical data with stochastic parameters, what affects the process of optimization, and solve the task from the probabilistic point of view.

Stochastic methods can be characterized by two phases:

- *general phase* where the function objective is evaluated as a set of model points, and
- *local phase* when this set is reconstructed in the direction of local searching.

It should be noted that in many methods these two stages are not strictly distinguished.

Another way to understand SA as a combinatorial optimization method is to imagine energetic surface as shown in Figure 1. When starting from a point optionally chosen, a pellet will always look for a path down. If a system of this kind is destroyed - say, as a result of some impact - then the pellet in most cases moves from A to B as the energy barrier from the side of A is less. If this impact is small, the pellet will more frequently move from A to B not from B to A. When the impact is strong, the pellet is expected to overcome the barrier quicker and more frequently, i.e. it may move both from A to B and from B to A. If one still wishes to impact the pellet movement, then a good compromise variant could be to start with the strongest impact and then to lessen it in sequence thus ensuring that the pellet could pass through the global minimum at some step.

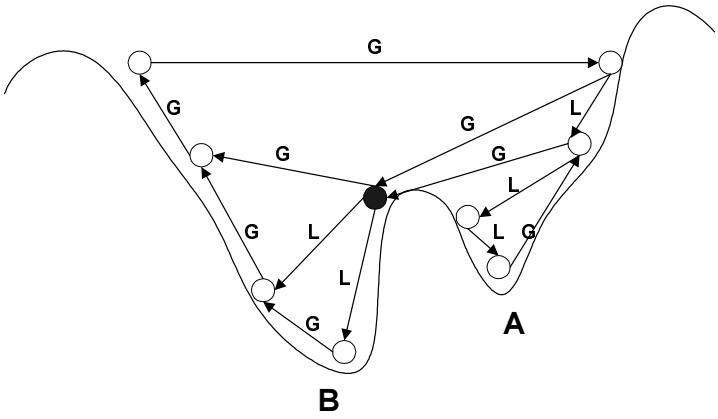

Figure 1. A simple energy landscape (G- global, L- local).

To enable practical application of the SA method, it is necessary to specify the following parameters:

1. Cost function W - analogous to the energy surface - the minimization of which is the purpose of the procedure.
2. A set of possible solutions conforming the energy surface or the state of physical system.
3. Generator of configuration state random changes.
4. Control parameter T that characterizes the artificial system temperature, and annealing schedule that characterizes in which way the temperature is decreased.

In a general form (Laarhoven et al. 1987), the combinatorial optimization problem can be formalized as (R,W) where R = space of states and W = cost function, W: R→R. The task now is to find a state at which W assumes its minimal value, that is state i_0 meets the following condition:

$$W_{opt.}=W(i_0)=\min W(i) \qquad (2)$$

where W_{opt} = optimal (minimal) value. Then the Metropolises algorithm can be used to generate a variety of states, W(i), and among those there can be $W(i_0)$.

3 THE SA ALGORITHM

The SA algorithm is based on the Boltzmann probabilistic distribution:

$$\Pr(E) \sim e^{(-E/kT)} \tag{3}$$

Expression (3) determines that if the system is in the state of thermal equilibrium at temperature T, its energy probabilistically distributes over all different energy states E. Even at a low temperature there exists a possibility for a system of being in high energy state. The system has the corresponding probability to pass from the local energy minimum state to a better, more global minimum.

The SA algorithm can be written as follows:

```
T=T0;
While(T>Tfreeze)
      Do until(Thermal Equilibrium is reached){
            choice (Config. i -> Config. j);
            if(ΔWij<0) then
              accept changes(Config. j);
            else
              r=random number[0,1);
              if (exp(-ΔWij/T)>r) then
                 accept changes(Config. j);
                 else
                    ignore changes(Config. i);}
          T=T*Tf;}
```

where

T_0 = initial temperature;
W_i = current configuration;
W_j = choice configuration;
T_f = temperature variation coefficient;
$\exp(-\Delta W_{ij}/T)$ = Boltzmann's factor.

To understand the essence of SA, the following analogy can be suggested. Assume a space of state is set as a basket of apples each state being represented as an apple. At each step one apple is drawn out and weighted. How to find the biggest apple? In the deterministic (full) searching an apple is drawn out of the basket, is weighted and is not put to the basket back. The best apple is kept aside. In the SA method, an apple is weighted and put back to the basket. Hence it can be drawn out and weighted several times.

The SA technique differs from other gradient descent optimization methods in that it does not “stick” in the local minimum found. Although SA is a rather slow procedure, it, however, ensures finding the global optimal solution.

The SA method is widely used in various combinatorial optimization tasks. It is employed in graph theory (to split and to paint graphs), in integrated circuit modeling (Laarhoven et al. 1987), neural net applications (Coughlin & Baran 1995) etc. In what follows a well-known combinatorial analysis task - the Travelling Salesman Problem (TSP) - is suggested as an application of the SA method.

4 THE TRAVELLING SALESMAN PROBLEM

The task of a travelling salesman is to find the smallest tour length between N cities provided he visits each only once and returns to the starting point at the end of the trip. This task can be solved by using different techniques of combinatorial analysis and graph theory. Solving the TSP by the SA algorithm is described by Laarhoven et al. (1987) and Coughlin & Baran (1995).

In the n-city TSP, a distance matrix $D=(d_{ij})$, $i,j = 1,2\ldots\ldots n$ is given; d_{ij} denotes the distance between cities i and j. A tour through the n cities is defined as a closed walk that visits each city exactly once. Each tour can be represented by an element π of the set of all cyclic permutations of the n cities {1,...,n}, if π is defined such that $\pi(i)$, $i=1,\ldots,n$ is the successor of city i in the tour. Thus, the set of configurations consists of

all cyclic permutations of {1,.....,n} (there are ½ (n-1)! such permutations for a TSP with a symmetric distance matrix) and the cost of a permutation is defined as the length of the corresponding tour:

$$C(\pi)=\sum_{i=1}^{n} d_{i\pi(i)} \tag{4}$$

The TSP now is to minimize the cost over all possible permutations. Consider the class of problem instances, where the n cities are points in two-dimensional Euclidean space, whose locations are drawn independently from the uniform distribution over the unit square, and let d_{ij} be defined as the Euclidean distance between the locations of i and j. Then $C_{opt}^{(D)}$ is the smallest tour length in an instance of this class with distance matrix D.

Laarhoven et al. (1987) suggest a number of numerical expressions to evaluate the tours:

$$\lim_{n \to \infty} = \frac{C_{opt}^{(D)}}{\sqrt{n}} = \theta \tag{5}$$

where $\theta \approx 0.7949$.

In order to be able to apply the SA algorithm, a neighborhood structure must be defined, i.e. for each tour one has to define a neighborhood as the set of tours that can be reached from the present tour in one transition. A well-known neighborhood structure for TSP (which is, almost without exception, used in applications of SA to TSP) is defined by the generation mechanism for transitions and the corresponding transitions are known as k-opt transitions. The simplest case, a 2-opt transition, is carried out by selecting two cities in the present tour and reversing the order in which the cities in between this pair of cities are visited (see Figure 2).

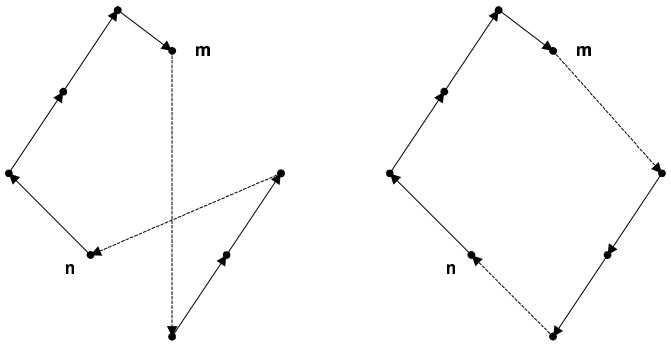

Figure 2. 2-opt example (left – current tour, right - tour after reversing the order between cities m and n).

A neighborhood of a tour is now defined as the set of all tours that can be reached from the current tour via a 2-opt transition (there are ½(n-1)n such neighbors).

The temperature variation coefficient for cooling schedule, T_f, is 0.925 (Laarhoven et al. 1987). In this way the running time for each transition is $O(n^{p1})$ where $p1<1$ and thus the total running time for the algorithm is $O(n^p)$ where $1<p<2$.

5 EXPERIMENTS

To implement the SA method and to evaluate its performance, the following case study has been chosen. Assume a map of Latvia is given with the regional centers marked. It is required to solve the TSP-25 task with the help of the SA method, i.e. to determine the shortest tour length between 25 Latvian towns.

The Latvian map shown in Figure 3 is taken from Grolier Multimedia Encyclopedia of 1995 available in the Internet at (http://www.rec.hu/REC/Maps/lat_map.html).

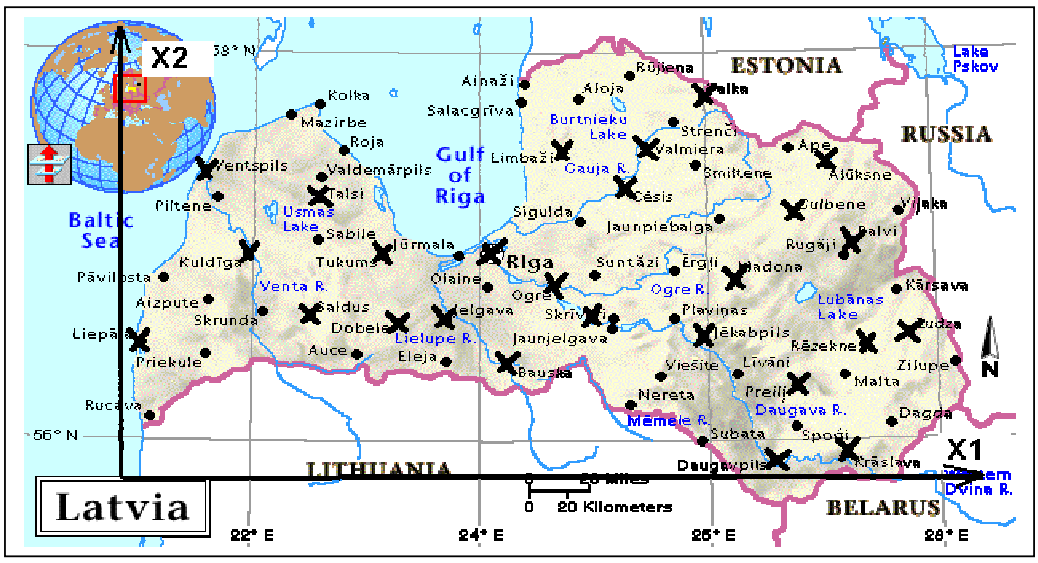

Figure 3. Latvian map with the regional centers marked with a cross.

Denotations of the towns and their co-ordinates are specified in Table 1.

Table 1. Denotations and co-ordinates of towns.

No.	Region	Town	X1	X2
0	A	Rīga	0,41	0,26
1	B	Limbaži	0,49	0,39
2	C	Valmiera	0,58	0,40
3	D	Cēsis	0,56	0,34
4	E	Valka	0,65	0,46
5	F	Alūksne	0,79	0,37
6	G	Gulbene	0,76	0,32
7	H	Balvi	0,82	0,28
8	I	Ludza	0,88	0,17
9	J	Rēzekne	0,83	0,16
10	K	Krāslava	0,82	0,03
11	L	Daugavpils	0,73	0,02
12	M	Preiļi	0,76	0,11
13	N	Jēkabpils	0,65	0,16
14	O	Madona	0,69	0,23
15	P	Aizkraukle	0,51	0,19
16	Q	Ogre	0,48	0,23
17	R	Bauska	0,43	0,14
18	S	Jelgava	0,36	0,19
19	T	Dobele	0,31	0,18
20	U	Tukums	0,29	0,26
21	V	Talsi	0,21	0,33
22	W	Ventspils	0,10	0,36
23	X	Kuldīga	0,14	0,26
24	Y	Saldus	0,21	0,19
25	Z	Liepāja	0,02	0,16

For the purpose of TSP simulation, a software program was written. The results of its simulation can be seen in Figure 4.

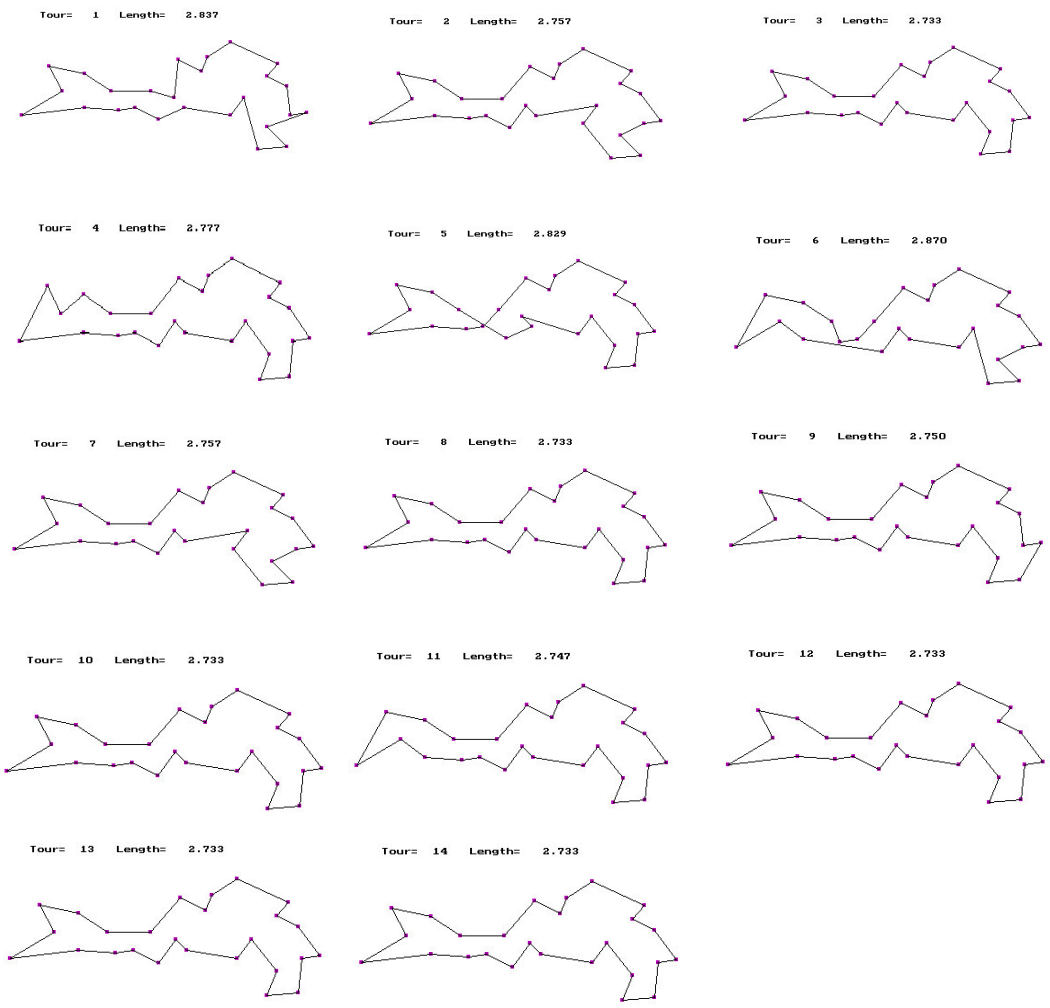

Figure 4. Program simulation.

The results of the program execution are presented below.

Travelling Salesman Problem with Simulated Annealing
Initial length: 3.026

Tour= 1 Length= 2.837 2 - 4 - 5 - 6 - 7 - 9 - 8 - 12 - 10 - 11 - 14 - 13 - 15 - 17 - 18 - 19 - 24 - 25 - 23 - 22 - 21 - 20 - 0 - 16 - 1 - 3 - 2
Tour= 2 Length= 2.757 11 - 10 - 12 - 9 - 8 - 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 20 - 21 - 22 - 23 - 25 - 24 - 19 - 18 - 17 - 16 - 15 - 14 - 13 - 11
Tour= 3 Length= 2.733 11 - 12 - 14 - 13 - 15 - 16 - 17 - 18 - 19 - 24 - 25 - 23 - 22 - 21 - 20 - 0 - 1 - 3 - 2 - 4 - 5 - 6 - 7 - 8 - 9 - 10 - 11
Tour= 4 Length= 2.777 21 - 20 - 0 - 1 - 3 - 2 - 4 - 5 - 6 - 7 - 8 - 9 - 10 - 11 - 12 - 14 - 13 - 15 - 16 - 17 - 18 - 19 - 24 - 25 - 22 - 23 - 21
Tour= 5 Length= 2.829 22 - 21 - 20 - 17 - 15 - 16 - 13 - 14 - 12 - 11 - 10 - 9 - 8 - 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 18 - 19 - 24 - 25 - 23 - 22
Tour= 6 Length= 2.870 19 - 18 - 0 - 1 - 3 - 2 - 4 - 5 - 6 - 7 - 8 - 9 - 12 - 10 - 11 - 14 - 13 - 15 - 16 - 17 - 24 - 23 - 25 - 22 - 21 - 20 - 19
Tour= 7 Length= 2.757 22 - 21 - 20 - 0 - 1 - 3 - 2 - 4 - 5 - 6 - 7 - 8 - 9 - 12 - 10 - 11 - 13 - 14 - 15 - 16 - 17 - 18 - 19 - 24 - 25 - 23 - 22
Tour= 8 Length= 2.733 9 - 8 - 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 20 - 21 - 22 - 23 - 25 - 24 - 19 - 18 - 17 - 16 - 15 - 13 - 14 - 12 - 11 - 10 - 9
Tour= 9 Length= 2.750 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 20 - 21 - 22 - 23 - 25 - 24 - 19 - 18 - 17 - 16 - 15 - 13 - 14 - 12 - 11 - 10 - 8 - 9 - 7
Tour= 10 Length= 2.733 10 - 11 - 12 - 14 - 13 - 15 - 16 - 17 - 18 - 19 - 24 - 25 - 23 - 22 - 21 - 20 - 0 - 1 - 3 - 2 - 4 - 5 - 6 - 7 - 8 - 9 - 10

Tour= 11 Length= 2.747 10 - 11 - 12 - 14 - 13 - 15 - 16 - 17 - 18 - 19 - 24 - 23 - 25 - 22 - 21 - 20 - 0 - 1 - 3 - 2 - 4 - 5 - 6 - 7 - 8 - 9 - 10
Tour= 12 Length= 2.733 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 20 - 21 - 22 - 23 - 25 - 24 - 19 - 18 - 17 - 16 - 15 - 13 - 14 - 12 - 11 - 10 - 9 - 8 - 7
Tour= 13 Length= 2.733 24 - 19 - 18 - 17 - 16 - 15 - 13 - 14 - 12 - 11 - 10 - 9 - 8 - 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 20 - 21 - 22 - 23 - 25 - 24
Tour= 14 Length= 2.733 24 - 19 - 18 - 17 - 16 - 15 - 13 - 14 - 12 - 11 - 10 - 9 - 8 - 7 - 6 - 5 - 4 - 2 - 3 - 1 - 0 - 20 - 21 - 22 - 23 - 25 - 24

In accordance with the results of the experiment, the shortest tour between Latvian towns is the following (see Figure 5).

Figure 5. The shortest tour according to the experimental results.

6 CONCLUSIONS

This study describes the SA method and presents an example of its implementation as applied to solving the Travelling Salesman Problem. The SA algorithm is rather slow. This, however, provides finding an optimum. The SA method can further be used to model neural network models, Boltzmann machines.

Acknowledgement I thank Professor Arkady Borisov (Riga Technical University) for useful comments on the manuscript.

REFERENCES

Coughlin, J.P. & Baran, R.H. 1995. *Neural Computation in Hopfield Networks and Boltzmann Machines*. University of Delaware Press.
Kirkpatrick, S., Gelatt, C.D. & Vecchi, M.P. 1983. Optimization by simulated annealing. *Science* 220, 671-680.
Laarhoven, P.J.M. van & Aarts, E.H.L. 1987. *Simulated Annealing:Theory and Applications*. D. Reidel Publishing Company, Holland.

INFORMATION ABOUT AUTHOR

Peteris Grabusts, *Riga Technical University, Riga*, Latvia. E-mail:peter@ibm.cs.ru.lv.