Environment. Technology. Resources. Rezekne, Latvia
Proceedings of the 14th International Scientific and Practical Conference. Volume 2, 40-45

Simulated Annealing Method in the Classic Boltzmann Machines

Peter Grabusts
Rezekne Academy of Technologies
Faculty of Engineering
Rezekne, Latvia
peteris.grabusts@rta.lv

Oleg Uzhga-Rebrov
Rezekne Academy of Technologies
Institute of Engineering
Rezekne, Latvia
rebrovs@tvnet.lv

Abstract. **The classical Boltzmann machine is understood as a neural network proposed by Hinton and his colleagues in 1985. They added noise interferences to the Hopfield model and called this network a Boltzmann machine drawing an analogy between its behaviour and physical systems with the presence of interferences. This study explains the definition of "simulated annealing" and "thermal equilibrium" using the example of a partial network. A technique for calculating the probabilities of transition states at different temperatures using Markov chains is described, an example of the application of the SA - travelling salesman problem is given. Boltzmann machine is used for pattern recognition and in classification problems. As a disadvantage, a slow learning algorithm is mentioned, but it makes it possible to get out of local minima. The main purpose of this article is to show the capabilities of the simulated annealing algorithm in solving practical tasks.**

Keywords: Boltzmann machine, simulated annealing, thermal equilibrium, travelling salesman problem.

I. Introduction and Theoretical Background

Two articles by Hopfield [1],[2] were essential to the notion of the connection between brain processes and physical systems.

The disadvantage of Hopfield networks is their tendency to stabilize at local but not global energy function minima. This shortage is largely overcome by a class of artificial neural networks (ANN) known as the Boltzmann machine (BM). The classic BM machine is a neural network proposed by G. Hinton and his colleagues in 1985 and described in the study [3]. In the BM changes in neuronal states are given by statistical, but not deterministic, relationships. There is a strong analogy between these methods and the process of cooling metal, so the methods themselves are often referred to as "simulated annealing" (SA).

One possible type of BM machine architecture is shown in Fig. 1.

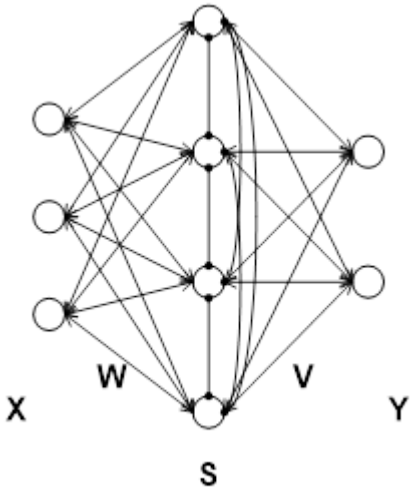


Fig. 1. An example of Boltzmann machine.

The elements of a BM are divided into two functional groups - a non-empty set of visible elements (neurons) and a (possibly empty) set of hidden elements. Visible elements (X and Y in Fig. 1) - the interface between the network and the external environment. During the training, the visible elements will be strengthened in specific positions, which are determined by the needs of the task to be solved, namely, any set of visible elements can be strengthened. The hidden elements (S in Fig. 1), if any, are never strengthened and are used to help describe the links in a set

Print ISSN 1691-5402
Online ISSN 2256-070X
https://doi.org/10.17770/etr2023vol2.7214

of input vectors, which cannot be assigned directly as the relationship between the visible elements.

The BM training procedure has 4 steps:

(1) The "artificial temperature" T is given a high initial value.

(2) The input vectors are passed through the network, then the output values and the target function are calculated.

(3) The weight values are changed, then network output values and the target function are recalculated.

(4) If the value of the objective function is decreased (improved), the change in weight is retained.

If the change in weight leads to an increase in the objective function, then the probability of maintaining this change is calculated using the Boltzmann distribution:

$$p(c) = e^{-c/kT} \quad (1)$$

where p - probability of target function change, k - constant (analogous to Boltzmann constant), T – "artificial temperature".

The concept of "artificial temperature" is related to the already mentioned SA. It is a stochastic optimization method used to optimize the objective function (energy).

The SA method allows to find a global extremum for a function that has local minima. The principle of SA was announced in classical study [4] and developed in studies [5]-[10]. Nowadays, the SA method is considered to be an independent field of research and is separated from the ANN methods. SA is based on an analogy with statistical mechanics, specifically with solid-state physics elements. A practical example from metallurgy is what happens to the atomic structure of the body by rapidly cooling it, lowering its temperature. A sharp drop in temperature can lead to an asymmetrical structure of the system or, in other words, to a suboptimal state (with errors). Cooling eventually causes the system to freeze and thermal equilibrium is reached.

The so-called Metropolis procedure [4] determines the iterative steps that control the achievement of the best solution. This algorithm is used to simulate equilibrium for a given temperature. In each step of this algorithm, the atom is given a small probable displacement - $x_i + \mathrm{y}$ and the system energy change E is calculated.

If $E < 0$, then the displacement is accepted and the configuration with the changed atomic states is used as the initial state in the next step.

If $E > 0$, then the probability, as a new state will be accepted, is:

$$P(E) = exp(-E / kT),$$

where k - is Boltzmann constant, T -parameter "temperature".

Using the system energy as a target function and defining the system states with $\{x_i\}$, the Metropolis procedure generates a series of states for a given optimization problem at a given temperature.

The BM together with the SA can be used to solve optimization problems. To understand SA as an optimization method, the energy surface can be imagined as shown in Fig. 2. The ball will always look for a way down when starting from an arbitrarily chosen point. If such a system is disrupted by acting on it in some way (e.g. shaking), the ball will most often move from A to B, as the energy barrier is lower on the A side. If this exposure is slight, then it is obvious that the ball will move more often from A to B not from B to A. If the exposure is strong, the ball will cross the barrier faster and more often and it can move from both A to B and from B to A.

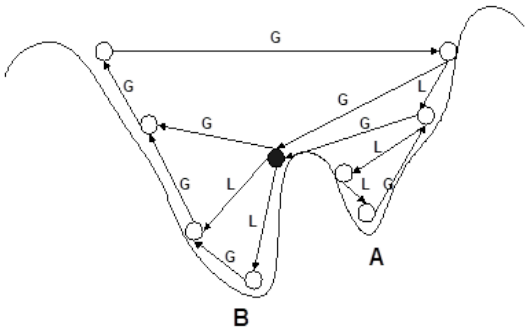


Fig. 2. Energy surface (G- global minimum, L- local minimum).

However, if you want to influence the movement of the ball, a good compromise would be to start with a stronger exposure and gradually reduce it. This would ensure that at some point the ball would pass through the global minimum.

In order to use the SA method in practice, it is necessary to determine:

1) the objective function W (analogous to the energy surface), the minimization of which is the aim of the procedure.

2) a set of possible solutions according to the energy surface or to the state of physical system.

3) configuration state random change generator.

4) control parameter T, which characterizes the artificial system temperature, and cooling mode, which characterizes how the temperature will be lowered.

The SA algorithm is based on Boltzmann's probabilistic distribution $Pr(E)$[11]. This expression states that if the system is in thermal equilibrium at temperature T, its energy is likely to be distributed between all the different energy states E. Even at low temperatures, there is a possibility that the system may be in a high energy state. The system has a corresponding probability of moving from a local energy minimum state to a better, more global minimum.

The steps of the SA algorithm in the pseudo-notation form are shown in Fig. 3.

```
(1)   T=T0;
(2)   While (T>Tfreeze)
(3)     Do until (Thermal equilibrium is reached) {
(4)           select (Config. i -> Config. j);
(5)           if(ΔWij<0) then
(6)             accept solution (Config. j);
(7)           else
(8)             r=random number [0,1);
(9)             if (exp(-ΔWij/T)>r) then
(10)              accept solution (Config. j);
(11)            else
(12)              reject solution (Config. i);}
(13)        T=T*Tf;}
```

Fig. 3. SA algorithm execution steps.

Here: T_0 – initial temperature; W_i – current configuration; W_j – selected configuration; T_f – decrease temperature; $exp(-W_{ij}/T)$ – Bolcmann factor.

SA differs from other gradient descent optimization procedures in that it does not "get stuck" in the local minimum found. Although SA is a relatively slow procedure, it guarantees finding the optimal global solution.

The main purpose of this article is to show the capabilities of the SA algorithm in solving practical tasks. One of the authors has already carried out research on a similar topic [12], [13]. In this article, the deficiencies found in the article [12] have been corrected and the description of the operation of the SA algorithm has been supplemented. An example of SA demonstration is taken from [13]. In the following, the essence of SA will be discussed in more detail.

II. Boltzman's Machine Operation Principle

To understand the principles of BM operation, in future we will use the notation and a network with three nodes (neurons) from [5] (see Fig. 4):

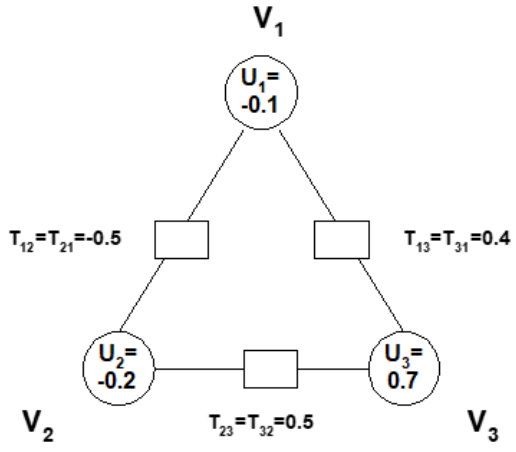


Fig. 4. The simple net nodes.

For the above net, the following holds:

$$V_i = 1, if \sum_{i \neq j} T_{ij} V_j > U_i \text{ and } V_i = 0, if \sum_{i \neq j} T_{ij} V_j < U_i \quad (2)$$

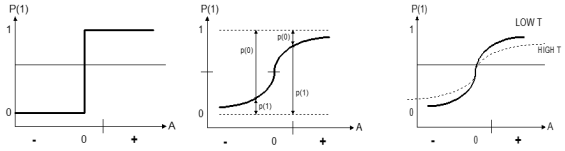
where V_i - activated state of the neuron,

U_i - threshold of the neuron; T_{ij} - weight coefficients between neurons i and j.

$$\sum_{i \neq j} T_{ij} V_j - U_i \quad (3)$$

is the activation of neuron.

The temperature effect can be shown as a change in probabilities (see Fig. 5). (BFPF - Boltzmann probability activation function).



(a) – threshold function (b) - BFPF (c) - BFPF

Fig. 5. The "temperature" effect.

Hinton [3] showed that Boltzmann function correctly characterizes this effect:

$$p(1) = \frac{1}{1 + e^{\frac{-A}{T}}} \quad (4)$$

We will further use this formula to determine BFPF. This function is shown in Fig. 6 for temperatures 0.5 and 0.25.

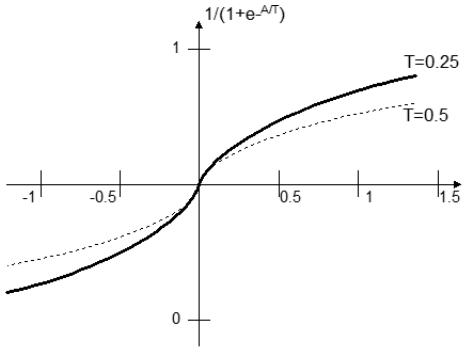


Fig. 6. "S"- shaped BFPF.

BM characteristics require:

1) For each state $V_1V_2V_3$ calculate the activation function according to the formula (3).

For example: $V_1V_2V_3 = 010$:

$$A_1 = T_{12}V_2 + T_{13}V_3 - U_1 = -0.5 + 0.1 = -0.4$$
$$A_2 = T_{12}V_1 + T_{23}V_3 - U_2 = 0 + 0 + 0.2 = 0.2$$
$$A_3 = T_{13}V_1 + T_{23}V_2 - U_3 = 0.5 - 0.7 = -0.2$$

2) For each neuron, the probability is calculated using formula (4) at different temperatures $[p(0) = 1 - p(1)]$ (see Table 1).

TABLE 1 PROBABILITIES AT DIFFERENT TEMPERATURES

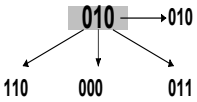
Neuron number	**T=0.25**		**T=1.0**	
	p(1)	***p(0)***	***p(1)***	***p(0)***
1	0.17	0.83	0.4	0.6
2	0.69	0.31	0.55	0.45
3	0.3	0.7	0.45	0.55

3) Using the values from point 2), it is necessary to calculate the transition probabilities to other states. In the general case - if the network consists of N elements - you can stay in the same state or go to N other states. For example, from state 010 the following transitions can be obtained:

The transition probabilities can be calculated by the formula:

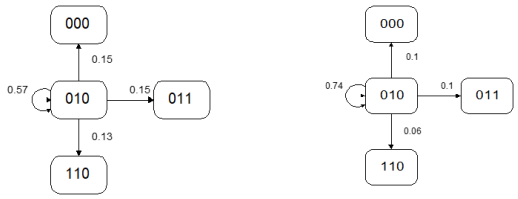
$$[V_j(p(1)_j + (1 - V_j)p(0)_j] / 3 \qquad (5)$$

We define for T=1:



010 -> 110 - $p(1)_1$=0.13
010 -> 000 - $p(0)_2$=0.15
010 -> 011 - $p(1)_3$=0.15
010 -> 010 - p=0.57

The result of applying this formula for state 010 for each neuron for various temperatures is shown in Fig.7.



(a) Boltzmann behavior (T=1.0) (b) T=0.25

Fig. 7. Transition probabilities for different temperatures.

4) Having obtained transition diagrams for all states, it is possible to draw a BM state diagram for each temperature. In practice, this turns out to be a laborious process, so another method is used. A chain of events $S_0, S_1, S_2, \dots S_{m-1}$ is given and a system is known for which these events that follow each other with known probabilities $p(i,j)$. The system can be represented by an $m \times m$ matrix. This method is known as Markov chain and makes it possible to find out the probability of system position in any state and at any time. The results are shown in Fig. 8. and Fig. 9. The shaded column is the state used for the example.

If the probability at time t is equal to $P_i(t)$, then the probability $P_j(t+1)$ of being in state S_j at time $t+1$ can be calculated using the formula:

$$P_i(t+1) = \sum P_i(t)p(i,j) \qquad (6)$$

Next State	CURRENT STATE							
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	0.55	0.31	0.1	0	0.13	0	0	0
S_1	0.02	0.08	0	0.02	0	0.04	0	0
S_2	0.23	0	0.74	0.23	0	0	0.27	0
S_3	0	0.31	0.1	0.58	0	0	0	0.17
S_4	0.2	0	0	0	0.71	0.26	0.26	0
S_5	0	0.3	0	0	0.08	0.47	0	0.1
S_6	0	0	0.06	0	0.08	0	0.24	0.1
S_7	0	0	0	0.17	0	0.23	0.23	0.63

Fig. 8. Markov chains for T=0.25.

Next State	CURRENT STATE							
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	0.53	0.22	0.15	0	0.16	0	0	0
S_1	0.11	0.35	0	0.11	0	0.13	0	0
S_2	0.18	0	0.57	0.18	0	0	0.2	0
S_3	0	0.22	0.15	0.54	0	0	0	0.17
S_4	0.18	0	0	0	0.56	0.19	0.2	0
S_5	0	0.21	0	0	0.14	0.5	0	0.15
S_6	0	0	0.13	0	0.14	0	0.42	0.15
S_7	0	0	0	0.17	0	0.18	0.18	0.53

Fig. 9. Markov chains for T=1.00.

For example, knowing that the probabilities of being in the time period $t = 0$ in any state are equal to 0.125 (i.e. 1/8). Then the probability of the system being in state S_3 can be calculated (see Fig. 9):

$$P_3(1) = P_0(0)p(0,3) + P_1(0)p(1,3) + P_2(0)p(2,3)$$
$$+P_3(0)p(3,3) + P_4(0)p(4,3) + P_5(0)p(5,3)$$
$$+ P_6(0)p(6,3) + P_7(0)p(7,3) =$$
$$= 0.125 * 0 + 0.125 * 0.22 + 0.125 * 0.15 +$$
$$+ 0.125 * 0.54 + +0.125 * 0 + 0.125 * 0 +$$
$$+ 0.125 * 0 + 0.125 * 0.17 =$$
$$= 0.125(0.22 + 0.15 + 0.54 + 0.17) = \mathbf{0.135}.$$

III. SIMULATED ANNEALING COUNTING

Using the possibility to calculate the probabilities of network nodes in any state and at any time, we can see what happens if the temperature of network changes. Let us turn to formula (6) and, using the Markov chain for $T = 1$ and $T = 0.25$, calculate the probabilities of finding network states at time t. The results are shown in Fig. 10.

T.	Time	P(0)	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)	P(7)
1.00	1	0.13250	0.08750	0.14125	0.13500	0.14125	0.12500	0.10500	0.13250
1.00	2	0.13326	0.07630	0.14966	0.13586	0.14770	0.12052	0.10211	0.13457
1.00	3	0.13350	0.07198	0.15417	0.13548	0.15002	0.11715	0.10321	0.13450
1.00	4	0.13372	0.07001	0.15694	0.13498	0.15094	0.11487	0.10457	0.13398
1.00	5	0.13396	0.06899	0.15873	0.13461	0.15133	0.11336	0.10555	0.13345
1.00	6	0.13420	0.06843	0.15993	0.13436	0.15151	0.11238	0.10617	0.13302
1.00	7	0.13441	0.06810	0.16074	0.13421	0.15159	0.11172	0.10655	0.13268
0.25	8	0.13054	0.01528	0.20866	0.13741	0.19140	0.09850	0.06050	0.15671
0.25	9	0.12228	0.01052	0.23237	0.13194	0.20334	0.08186	0.05802	0.15866
0.25	10	0.12019	0.00920	0.24609	0.13000	0.20520	0.07376	0.06000	0.15456
0.25	11	0.12024	0.00869	0.25585	0.12913	0.20451	0.06930	0.06104	0.15024
0.25	12	0.12100	0.00845	0.26317	0.12872	0.20314	0.06656	0.06138	0.14658
0.25	13	0.12189	0.00833	0.26875	0.12851	0.20169	0.06473	0.06143	0.14366
0.25	14	0.12272	0.00826	0.27306	0.12842	0.20038	0.06342	0.06137	0.14137
0.25	15	0.12341	0.00822	0.27639	0.12838	0.19926	0.06246	0.06128	0.13959

Fig. 10. Network states at time t.

Starting with equal probabilities (1/8) the system reaches time=7 at temperature $T = 1$. Having experimentally verified the following steps, it is clear that

at this temperature the probabilities practically do not change further. According to Hinton, this phenomenon is called "Thermal equilibrium". The probabilities begin to change when the temperature changes. Fig. 10. shows that the temperature drops to 0.25. Thermal equilibrium is reached again at time=15. If at this point we go to $T = 0$, we can get the final state of the network:

T.	Time	P(0)	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)	P(7)
0	28	0	0	0.494	0	0.313	0	0	0.193

At the end of the annealing process, the system reaches a stable position. If the initial conditions of the network from point 2) are met, then it is clear that the state S_2 with the lowest energy (-0.2) has the highest final probability - 0.494; S_4 - with energy -0.1 at the beginning has a final probability - 0.313; and finally - S_7 has the smallest final probability. For all other states, the final probabilities are zero.

The purpose of Boltzmann approximation was to prove that the final probabilities in a partial state strongly depend on the state energy.

IV. Using the SA Method for Optimization Purposes

The SA method is widely used in many combinatorial optimization tasks. It is used in graph theory, neural networks and other applications. As an application of the SA algorithm, the well-known combinatorics problem - the traveling salesman problem (TSP) - is proposed, the goal of which is to find the shortest path between N cities - visiting each city only once and returning to the original city at the end. It is a fairly well-known problem in combinatorics, which can be solved by various combinatorics or graph theory techniques. Methods of solving TSP with the SA algorithm have also been reviewed in the literature [14], [15].

The task of TSP is to minimize the objective function in all possible permutations. If n cities are located in 2-dimensional Euclidean space, then d_{ij} is the Euclidean distance between cities i and j, then C_{ij} is the shortest path for the given distance matrix D.

In order to use the SA algorithm for this type of task, some concepts need be introduced. For each path, we can define a neighbour as the set of paths that can be reached from the current path during one transition. Such a mechanism of neighbour structures for TSP is called k-opt transitions.

For the purpose of the experiment, a study was carried oud: to find the shortest path between 8 milk processing companies. It is necessary to solve the task of TSP-8 with the help of the SA method, i.e., find the shortest path between 8 objects. Locations of objects are given with GPS coordinates.

In this case, the SA algorithm executes in 20 steps, i.e., as a result of the experiment, it was determined that the state of thermal equilibrium was reached in the $20-th$ step (see Fig. 11). The shortest path calculated by the SA algorithm was 648 km (see Fig. 12.).

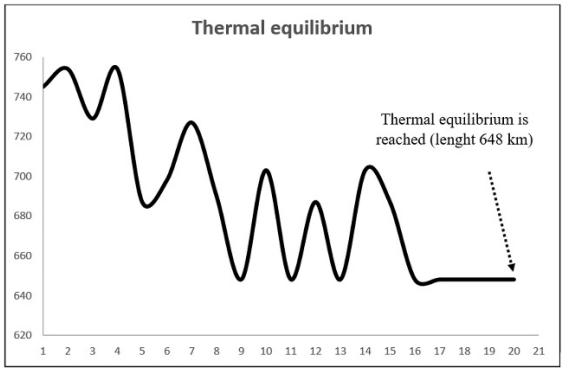


Fig. 11. Thermal equilibrium is reached.

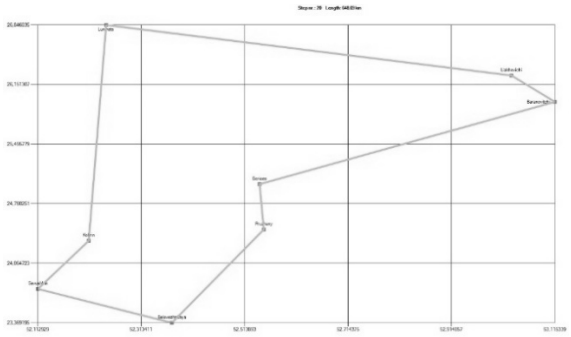


Fig. 12. The shortest path between objects.

V. Conclusion

This article describes an optimization method called simulated annealing. The SA method is widely used in various combinatorial optimization tasks. SA is a stochastic optimization method that can be used to minimize a given cost function when a combinatorial system with multiple degrees of freedom is used. This method allows you to find a global extremum for a function that has local minima. This article demonstrates the application of the SA method to a well-known combinatorial analysis problem, the Traveling Salesman Problem, and performs an experiment to find the shortest path between 8 companies.

The SA method is used to implement the BM operation. The purpose of this article was to show the capabilities of the SA algorithm in solving practical combinatorics problems.

References

[1] J.J. Hopfield, "Neural Networks and physical systems with emergent collective computational abilities." Proc.Natl. Acad.Sci. USA,79, P. 2554-2558, 1982. https://doi.org/10.1073/pnas.79.8.2554

[2] J.J. Hopfield, "Neurons with graded response have collective computational properties like those of two state neurons." Proc.Natl. Acad.Sci. USA,81, P. 3088-3092, 1984. https://doi.org/10.1073/pnas.81.10.3088

[3] D.H. Ackley, G.E. Hinton and T.J. Sejnowski,T.J. "A learning algorithm for Boltzmann machines," Cognitive Science,9, P. 147-169, 1985. https://www.cs.toronto.edu/~fritz/absps/cogscibm.pdf

[4] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, "Optimization by simulated annealing." Science, 220, P. 671-680, 1983. https://www.science.org/doi/10.1126/science.220.4598.671

[5] P.J. Laarhoven and E.H. Aarts, Simulated Annealing: Theory and Applications. Holland: D. Reidel Publishing Company, 1987. https://doi.org/10.1007/978-94-015-7744-1

[6] R.H. Otten and L.P. Ginneken, The Annealing Algorithm. Kluwer Academic Publishers, 1989.

[7] V. Granville, M. Krivanek and J-P. Rasson, "Simulated annealing: A proof of convergence." IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(6), 652–656, 1994. https://doi.org/10.1109/34.295910

[8] L. Ingber, "Simulated annealing: Practice versus theory." Math. Comput. Modelling, 18, 29–57, 1993. https://doi.org/10.1016/0895-7177(93)90204-C

[9] J.P. Coughlin and R.H. Baran, Neural Computation in Hopfield Networks and Boltzmann Machines. University of Delaware Press, 1985.

[10] B. Maxwell, Advanced Simulated Annealing. Clanrye International, 2015.

[11] E. Aarts and J. Korst, Simulated Annealing and Boltzmann Machines: A stochastic approach to combinatorial optimization and neural computing. John Wiley and Sons, 1989.

[12] P. Grabusts, Analysis of the simulated annealing method in classic Boltzmann machines. Environment. Tehnologies. Resources. Proceedings of the International Scientific and Practical Conference, Rezekne, Latvia, 1997. https://doi.org/10.17770/etr1997vol1.1857

[13] P. Grabusts, J. Musatovs and V. Golenkov, The Application of Simulated Annealing Method for Optimal Route Detection Between Objects, Procedia Computer Science, ICTE in Transportation and Logistics, ICTE 2018, Volume 149, 2019, Pages 95-101, https://doi.org/10.1016/j.procs.2019.01.112

[14] W.J. Cook, In Pursuit of the Traveling Salesman. Princeton: Princeton University Press, USA, 2011.

[15] D.L. Applegate, R. Bixby, V. Chvátal and W. Cook, The Traveling Salesman Problem: A Computational Study. Princeton University Press, 2006.