PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 42, NO. 2, PP. 193-200 (1998)

SOLVING THE RADIO LINK FREQUENCY ASSIGNMENT PROBLEM WITH BOLTZMANN MACHINE

György STRAUSZ* and Paul BOURRET**

*Department of Measurement and Instrument Engineering
Technical University of Budapest
H-1521 Budapest, Hungary
E-mail: strausz@mmt.bme.hu
** Département D'Etudes et de Recherches en Informatique
2, avenue Edouard Belin B.P. 4095
31055 - Toulouse Cedex, France
E-mail: bourret@cert.fr

Received: Sept 17, 1998

Abstract

Neural network models that became well known and popular in the 80's have been successfully applied to solve tasks in several domains. These systems seem to offer fast and robust solutions for several difficult problems. The comparison of the results often shows similar achievements for neural networks and conventional methods. There is nothing surprising in it, if we consider that the different types of artificial neural systems accomplish the same or similar procedures as the different search algorithms and other methods. Most of the neural networks realize a modification of previously known algorithms on an intrinsic parallel system. Although the underlying methods are similar, the parallel structure and the nonlinear processing elements offer us a new, more efficient method.

In this paper we present how to map a constraint satisfaction problem to achieve fast, optimal or near optimal solution. The application task, which has been solved, is the Radio Link Frequency Assignment Problem (RLFAP). In this problem we have to assign frequencies from a finite domain to several radio connections in such a way that the result should meet numerous constraints.

The first section briefly describes the neural network model we have used to solve the problem. The second part introduces the RFLAP task in more detail. In the following two sections first we show a possible method to map the problem to the neural network and after this we present and evaluate the achieved results. In the fifth part we finish the paper with some conclusions.

Keywords: combinatorial optimization, neural networks, simulated annealing, Boltzmann machine.

1. Boltzmann Machines

The RLFAP application task that will be presented in the next section, is a constraint satisfaction problem. In order to take advantage of the features of connectionist machines we mapped the problem to a Boltzmann ma-

chine. This neural network is a stochastic extension of the Hopfield model (HOPFIELD, 1982).

The structure of the network is a very simple, one-layer recurrent model. Each unit in the layer is binary and fully connected to other units. It was shown that the system is asymptotically stable if the matrix of the connection strengths (weights) is symmetric. The dynamic behavior of the deterministic network can be described with the states of the units:

$$u_i := g\left(\sum_j w_{ij}u_j - \Theta_i\right) , \tag{1}$$

where u_i is the value of unit i and w_{ij} is the connection strength between unit i and j and g is the step function:

$$g(u) = \left\{ \begin{array}{ll} +1 & \text{if } u \geq 0 \\ 0 & \text{if } u < 0 \end{array} \right\} . \tag{2}$$

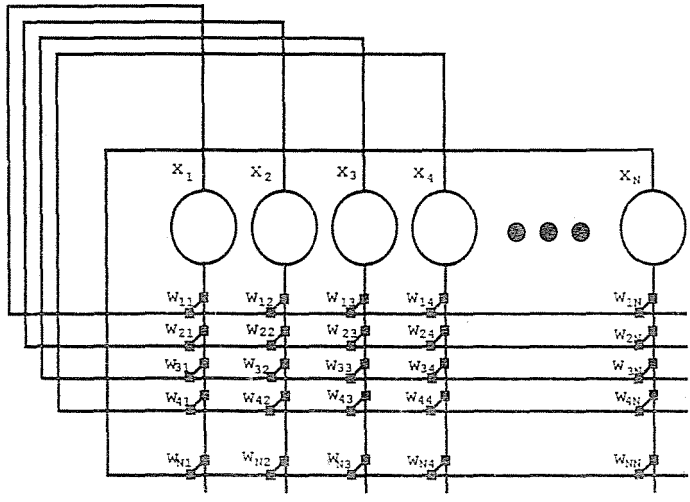

Fig. 1. Structure of the Hopfield type neural network

Solving the RLFAP task we use the advantageous feature of the network that we can assign an energy function to the system. The important property of an energy function or a Ljapunov function is that it always decreases (or remains constant) as the system evolves according to its dynamic rule. So as it was also shown in the paper of Hopfield that the network converges to a local minimum of the energy function. The function can be written as:

$$E := -\sum_i \sum_j w_{ij}u_i u_j . \tag{3}$$

Theoretically it is equivalent to minimize the energy function or to simulate the processing of the network, but from the point of view of our application using the energy function is more natural, so in the following we will focus on this approach.

The performance of the deterministic network is often not sufficient as it stuck in a local minimum that sometimes belongs to a high-energy state. In order to find the global minimum or at least a 'low energy state' local minimum we use the stochastic extension of the network, the Boltzmann machine (HINTON, 1983). The idea is to get out of spurious local minimum by using simulated annealing. We add 'thermal' noise to the process that makes units to change their value following a probability rule instead of the deterministic one. A chosen unit changes its state with the following probability:

$$P(u_i \to 1 - u_i) = \frac{1}{1 + e^{\frac{-\Delta E}{T}}}, \tag{4}$$

where (in case of binary units):

$$\Delta E = E(1 - u_i) - E(u_i) \tag{5}$$

and T is the 'temperature' parameter that controls the cooling process. It was proven (AARTS and KORST, 1989) that using a sufficiently high starting temperature and a sufficiently slow cooling schedule the system can reach global minimum. Although the known theoretical results do not help in practice to choose optimal parameters for the annealing process, using heuristically determined parameters the network usually stabilizes in the global or 'near global' local minimum.

Final success of the simulation depends strongly on the parameters of the annealing process. Although there are theoretical results about appropriate cooling schedules, the choice of parameters is mainly based on heuristic decisions. The most important parameters can be divided into three groups:

- starting temperature
- speed of the annealing
- stopping criterion

The starting temperature should be chosen such that the network behaves almost random at the beginning. The parameter itself is determined by the magnitude of the coefficients (weight values) of the simulated system.

The speed of the annealing process should be set as slow as possible. Theoretically an infinite slow annealing results to settle the system in the optimal solution. This way the speed of the annealing is constrained by the time limit of the simulation.

The speed of the cooling is controlled by two parameters: the number of iterations on a given temperature and the temperature decay rate. We

have used exponential cooling schedule that is the most simple and common method (AARTS and KORST, 1989). At each temperature $10N$ iteration was made, where N is the number of units in the Hopfield type neural network, and the parameter was decreased by a factor of 0.9.

Stopping criterion is determined in a way that the criterion is met when the network does not change its state any more. In asynchronous simulation this parameter is usually set to be $3N - 10N$ iteration.

2. Presentation of the Radio Link Frequencies Assignment Problem

As a matter of fact there are several kinds of RFLAP according to the kinds of constraints, which have to be taken into account and to the optimization criteria (if there is one).

The basic problem is only a constraint satisfaction problem. Let L be the radio links $(L_1, L_2, \ldots, L_3)$ and F_i be the set of frequencies $(f_i^1, \ldots, f_i^k)$ which can be assigned to L_i given the kind of devices which provide the radio link L_i.

The frequencies assigned to the links must meet mainly three kinds of constraints (let v_i be the value of the frequency assigned to L_i):

I.) Equality constraints.

$$|v_i - v_j| = d\,.$$

In fact this kind of constraints has to be met for two links providing a two ways radio contact. Thus variable v_i must meet at most one such a constraint (and in most cases every variable must meet such a constraint).

II.) Inequalities constraints.

$$|v_i - v_j| > d_{ij}\,.$$

The inequality constraints come from the computation of the electromagnetic waves propagation according to the landscape of the area where the related links have to be established. According to other criteria these constraints are split into several subsets which have different priorities. These constraints are in fact soft constraints which may be violated if there is no solution for an assignment meeting all constraints.

III.) Unary constraints.

$$v_i = v_i^0\,.$$

This means that some devices providing the radio links are already tuned to a given frequency and should not be modified when we solve the problem.

The RLFAP can be described in a very general way as follows:

Find a solution meeting all constraints and minimizing additional criterion $F(v_i)$ and if there is no solution minimize a criterion based on the violation of the constraints. Additional criterion is different for the problem instances, but in most cases we should minimize the number of used different frequencies or the maximal used frequency value.

The purpose of the project was to prove the viability of using Boltzmann Machine for solving the RLFAP. Therefore we have solved several instances of the problem, so called sceni ($i = 1, \ldots, 4$), by the neural network approach. To be able to compare both qualities of the obtained solutions and the needed processing time for solving them, these problems have already been solved by constraint programming approach.

3. Mapping the Problem to Boltzmann Machine

We presented the RLFAP constraint satisfaction task as an optimization problem, where the primary aim of the optimization is to meet all the constraints. Let us consider first the basic problem where there are no other optimization criteria than the above mentioned one. In this case we can use the number of violated constraints as the cost function of the problem. The solution to be found can be characterized by a finite set of discrete variables (radio links) $V = \{v_1, \ldots, v_n\}$, that should take a value from a finite domain (frequencies) $D = \{f_1, \ldots, f_p\}$. To use a Boltzmann machine for solving the problem generally we have to accomplish the following two steps:

a.) We should convert the optimization task into a 0–1 programming problem. In order to do it, instead of using real valued variable (v_i), we assign new binary variables for each possible link-frequency pair. So we create a variable x_{ij} for each (v_i, f_j) double. Let:

$$\begin{aligned} x_{ij} &= 1 \quad \text{if} \quad v_i = f_j \,, \\ x_{ij} &= 0 \quad \text{if} \quad v_i \neq f_j \,. \end{aligned}$$

Then we obtain our network, such that the state of each unit u_k determines the value of exactly one variable x_{ij}. (This mapping can be written as $u_k = m(x_{ij})$ or $U = M(X)$.) So each possible state vector of the network units contributes to a given (correct or not correct) solution of the optimization problem.

b.) In the second step we should define the connections of the network, the weight matrix (W) such that the energy function of the network and the cost function of the optimization problem should be feasible and order preserving. This means that the global minimum of the energy function should correspond to the optimal solution of the problem and the functions

should also satisfy the following condition:

$$F(m(X_p) > F(m(X_q) \Rightarrow E(U_m) > E(U_n)\,, \tag{6}$$

where E is the energy function, F is the cost function and $U_m = M(X_p)$, $U_n = M(X_q)$.

In our case to accomplish these conditions we have to consider both the constraints of the original problem and the constraint coming from the assignments of the variables x_{ij}, that creates the new condition:

$$\sum_j x_{ij} = 1\,. \tag{7}$$

In order to make the problem easier for the network we modify the structure and hardwire the second condition in it. We use the 1-of-K encoding for the units that was introduced by [Pet89]. It means that we force the network that among the units that contribute to the variables $x_{i1}, x_{i2}, \ldots, x_{ir}$ there should be always exactly one unit with the value of 1. This also causes that we should consider at the evaluation of the energy function that if a unit changes $0 \rightarrow 1$ then another unit changes $1 \rightarrow 0$ in the same time. In this case the asynchronous behavior of the network is modified in the sense that always two units change values (vice-versa) synchronously.

After this to find a solution that meets all the constraints of the RLFAP task we can choose the weights of the network in the following way:

$w_{pq} = 1$, if the assignment variables $v_i = f_j$ and $v_k = f_l$ does not violate a constraint, where $u_p = m(x_{ij})$ and $u_q = m(x_{kl})$.

$w_{pq} = 0$, otherwise.

It can be seen then easily that the energy function:

$$E = -\sum_p \sum_q (1 - w_{pq}) u_p u_q \tag{8}$$

is feasible and order preserving.

So far we presented how to map the basic RLFAP problem to a neural network. In our case we had to complete another task, too. Among the possible solutions we wanted to find an optimal one, where the optimization criterion was to minimize the number of used different values (frequencies).

To accomplish this task we extended our model with new neuron-like elements and also completed with a second term the energy function. For each value in the domain we have an integer element (t_v) that outputs the number of variables currently taking the value. The best energy function term we found for this problem is:

$$E_2 = -c \sum_v^D \sum_z^D t_v t_z\,, \tag{9}$$

where D is the number of values in the domain and c is a sufficiently small factor to keep this term less than 1. (The reason to use c is that the primary criterion is still not to violate the constraints.) This term is feasible, and although not generally order preserving (if we consider the cost function to be the number of used values), but it keeps local order preserving feature for all possible two consecutive states that is sufficient when we use asynchronous updating. So the energy function was used to find optimal solution is the following:

4. Results

The described method was tested on 7 different CELAR databases (see *Table 1*). For all the investigated problems correct solutions were found.

To obtain a solution for the basic problem (where the only criterion is to meet all the constraints) the usage of the deterministic network ($T = 0$) was sufficient. In this way the time needed to find a solution was some seconds (if we do not count the data-loading phase). The size of the network varied from 2000 units up to 5000 units for the different problems.

Table 1. Data and results of the tested RLFAP problems

Problem	Number of variables	Number of constraints	Violated constraints in solution	Cost of the violated constraints	Number of used values in solution
scen01	916	5548	0	0	20
scen02	400	2760	0	0	14
scen03	200	1235	0	0	14
scen04	680	3967	0	0	46
scen06	200	1322	163	4870	46
scen07	400	2865	305	487547	46
scen08	916	5744	171	320	46

To get correct value assignments for the variable with using minimal number of different values, we should have to use simulated annealing. The necessary cooling schedule was different for the problems but for all of them a starting temperature $T_0 = 10$ and a cooling of $T := 0.9^*T$ after each three times the number of units cycle gave good results. The processor time used for solving the problems was in each case less than an hour on a Sun Sparcstation 10 machine.

For all the tested problems the quality of the solution was good. The achieved results are better or the same that have been received with other methods (see details in the table). The Boltzmann machine proved to be a robust and reliable method for solving the problem. There are several

possibilities for further investigations. We plan to test different cooling schedules, polynomial (AARTS and VAN LAARHOVEN, 1985) and logarithm cooling (HAJEK, 1988) strategies and the algorithm will be implemented on a 100 processor parallel machine that can increase the speed very much.

The presented work shows that Boltzmann machines offer an efficient method for solving constraint satisfaction tasks in case when suitable mapping of the problem can be accomplished.

References

[1] AARTS, E. H. L. – KORST, J.H. M. (1987): Boltzmann Machines and their Applications, *Lecture Notes in Computer Science*·258, Springer-Verlag, Berlin, pp. 34–50.

[2] AARTS, E. H. L. – KORST, J. H. M. (1989): Simulated Annealing and Boltzmann Machines, John Wiley and Sons, Chichester.

[3] AARTS, E. H. L. – VAN LAARHOVEN, P. J. M.: A New Polynomial Time Cooling Schedule, *Proc. IEEE Int. Conf. On Computer-Aided Design*, Santa Clara, pp. 206–208.

[4] HAJEK, B.: Cooling Schedules for Optimal Annealing, *Mathematics of Operation Research*, Vol. 13, No. 2, pp. 311–329.

[5] HINTON, G. E. – SEJNOWSKI, T. J. (1983): Optimal Perceptual Inference, *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Washington D.C., pp. 448–453.

[6] HINTON, G. E. – SEJNOWSKI, T. J. (1986): Learning and Relearning in Boltzmann Machines, in Rumelhart, D.E. and McClelland, J.L. (Eds.): Parallel Distributed Processing 1, Bradford Books, Cambridge (MA), pp. 282–317.

[7] HOPFIELD, J. J. (1982): Neural Networks and Physical Systems with Emergent Collective Computational Abilities, *Proc. National Academy of Sciences of the USA 81*, pp. 3088–3092.