Group Updates and Multiscaling: An Efficient Neural Network Approach to Combinatorial Optimization

Aristidis Likas and Andreas Stafylopatis
Computer Science Division
Department of Electrical and Computer Engineering
National Technical University of Athens
157 73 Zographou, Athens, Greece

Abstract

A multiscale method is described in the context of binary Hopfield–type neural networks. The appropriateness of the proposed technique for solving several classes of optimization problems is established by means of the notion of group update which is introduced here and investigated in relation to the properties of multiscaling. The method has been tested in the solution of partitioning and covering problems, for which an original mapping to Hopfield-type neural networks has been developed. Experimental results indicate that the multiscale approach is very effective in exploring the state-space of the problem and providing feasible solutions of acceptable quality, while at the same it offers a significant acceleration.

1 Introduction

The Hopfield neural network model [7, 8] and closely related models such as the Boltzmann Machine [3, 1] have proved effective in dealing with hard optimization problems and yield near-optimal solutions with polynomial time complexity [6, 20]. The basic idea is to encode the objective function and the problem constraints in terms of an appropriate *energy function* which can be minimized by the neural network architecture.

The energy function corresponding to a Hopfield-type neural network with n units, connection weights w_{ij} (with $w_{ii} = 0$) and threshold values θ_i has the form:

$$E(\vec{y}) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j w_{ij} - \sum_{i=1}^{n}\theta_i y_i \tag{1}$$

where $\vec{y} = (y_1, \ldots, y_n)$ is the state of the network. We are concerned with networks of *binary* units which operate *asynchronously*. At each time instant one unit is selected randomly and the change in the network's energy that will result if the selected unit i changes state is computed:

$$\delta E_i(\vec{y}) = (2y_i - 1)(\sum_{j=1}^{n} w_{ji} y_j + \theta_i) \tag{2}$$

If $\delta E_i(\vec{y}) < 0$ then the change is accepted, otherwise it is rejected. If the weights are symmetrical ($w_{ij} = w_{ji}$) it is ensured that the network will settle into a state corresponding to a local minimum of the energy function [7, 18], where $\delta E_i(\vec{y}) > 0$ for all $i = 1, \ldots, n$. In solving optimization problems, our objective is to reach the global energy minimum.

The Boltzmann Machine Optimizer [1, 2] is based on the idea of applying an annealing schedule [11, 12] to the operation of the Hopfield network from an initial high temperature down to a temperature value near zero. At each temperature, the network operates as described previously, but a different update rule is used in case $\delta E_i(\vec{y}) > 0$: the state of unit i changes according to some probability function which in general depends on the quantity $exp(\delta E_i(\vec{y})/T)$ where T is the temperature parameter. As T tends to zero, the probability also tends to zero and the update rule becomes the same as in the pure Hopfield case.

In the following, we first describe the multiscale method for binary Hopfield-type networks. According to this method, a smaller network is constructed by grouping the units of the original network. The minimization of the objective energy function is carried out by performing iterations either in the original or in the coarse-scale network. The above technique has been originally introduced in [13], where a general description can be found within a slightly different notation framework. Moreover, the method in its general formulation was tested in [13], in the context of the Set Partitioning problem, using the neural network architecture described in [20] and a random allocation of units to groups. In order to provide insight into the operation of multiscaling we introduce in this paper the notion of *group update* in binary Hopfield-type networks; instead of selecting a single unit to update at each time step, we consider a group of units which are selected and updated simultaneously. We prove that there exists a direct relation between performing single updates in the coarse-scale network and performing group updates in the original fine-scale network. The idea of group update helps highlight several issues, which can be very useful in applying the multiscaling approach for solving optimization problems. Indeed, we consider the solution of the Set Partitioning and the Set Covering problems, for which an efficient grouping of units can be devised based on an original mapping of the problems to a network architecture. The performance of this approach is evaluated in terms of numerical experiments which yielded very good results.

2 Multiscaling in Binary Networks

The idea of *multiscaling* originates from the methods of numerical analysis [4, 14] and particularly the *multigrid methods* [9], which have been successfully applied to the solution of partial differential equations. The basic principle is that we can speed up convergence in a large optimization problem by introducing a smaller approximate version at a coarser scale and alternating the relaxation steps between the fine scale and the coarse scale instances of the problem.

The above idea is used in [15, 16] to optimize a general Hopfield objective function of *continuous* neural variables. A quite general methodology is suggested, that cannot be applied in the case where the fine and coarse scale vectors have binary elements. This happens because the mapping from the original variables to the coarse scale variables and vice versa cannot take a simple form as suggested in [16], since the linear combination of binary variables does not yield a binary variable. Indeed, the fact that both vectors contain binary valued elements makes necessary the dependence of the mappings on the current state of the fine-scale network. The multiscale method discussed next uses a transformation of special nature and is appropriate for the widely used class of neural network optimizers with binary units.

Consider a Hopfield-type neural network with n binary units (n-net), symmetric connection strengths w_{ij} (with $w_{ii} = 0$) and threshold values θ_i. Suppose now that we wish to switch to a smaller network with p units (p-net) that is also assumed to be a Hopfield-type neural network with binary units. The mapping from the n-net to the p-net is based on partitioning the units of the n-net into p disjoint groups, such that each group G_k, $k = 1, \ldots, p$, corresponds to a unit k of the p-net. We shall use the notation $\mathcal{P}$ to represent the partition.

If the state of the n-net is $\vec{y^\star} = (y_1^\star, \ldots, y_n^\star)$ the corresponding energy function $E(\vec{y^\star})$ can be written:

$$\begin{aligned} E(\vec{y^\star}) &= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i^\star y_j^\star w_{ij} - \sum_{i=1}^{n}\theta_i y_i^\star \\ &= -\frac{1}{2}\sum_{G_k\in\mathcal{P}}\sum_{G_l\in\mathcal{P}}\sum_{i\in G_k}\sum_{j\in G_l} y_i^\star y_j^\star w_{ij} \\ &\quad - \sum_{G_k\in\mathcal{P}}\sum_{i\in G_k}\theta_i y_i^\star \end{aligned} \tag{3}$$

Suppose that while being in state $\vec{y^\star}$ we decide to switch to the small network. Suppose also that we have a way of appropriately constructing the p-net. At some later time, after operation of the small network, we decide to switch back to the n-net. Then, if the final state of the p-net is $\vec{\xi} = (\xi_1, \ldots, \xi_p)$, the following transformation is used for obtaining the new state $\vec{y} = (y_1, \ldots, y_n)$ of the original network:

$$y_i = y_i^\star + u_i \xi_k, \quad i \in G_k \tag{4}$$

for all $G_k \in \mathcal{P}$. The quantities u_i are defined as

$$u_i = 1 - 2y_i^\star, \;\; i = 1, \ldots, n \tag{5}$$

The advantage of the above transformation (4) is that it can assure that the new state vector $\vec{y}$ of the n-net contains binary components. There is an interesting physical interpretation of the transformation: for each group $G_k \in \mathcal{P}$, if the final state of the corresponding unit k of the p-net is 1, all units belonging to group G_k change their state with respect to the state vector $\vec{y^\star}$. Inversely, if the resulting value ξ_k is 0, the units belonging to group G_k retain the state they had before switching to the p-net.

The connection weights x_{kl} of the p-net can be computed as follows. Using the transformation equation (4) the energy of the new state $\vec{y}$ of the n-net is:

$$\begin{aligned} E(\vec{y}) &= -\frac{1}{2} \sum_{G_k \in \mathcal{P}} \sum_{G_l \in \mathcal{P}} \sum_{i \in G_k} \sum_{j \in G_l} (y_i^\star + u_i \xi_k)(y_j^\star + u_j \xi_l) w_{ij} \\ &\quad - \sum_{G_k \in \mathcal{P}} \sum_{i \in G_k} \theta_i (y_i^\star + u_i \xi_k) \end{aligned} \tag{6}$$

Taking into account the energy definition, after some algebra equation (6) yields:

$$E(\vec{y}) = E(\vec{y^\star}) + \hat{E}(\vec{\xi}) \tag{7}$$

where the quantity $\hat{E}(\vec{\xi})$ is given from the following:

$$\begin{aligned} \hat{E}(\vec{\xi}) &= -\frac{1}{2} \sum_{G_k \in \mathcal{P}} \sum_{G_l \in \mathcal{P}} \xi_k \xi_l \Big(\sum_{i \in G_k} \sum_{j \in G_l} u_i u_j w_{ij} \Big) \\ &\quad - \sum_{G_k \in \mathcal{P}} \xi_k \Big(\sum_{i \in G_k} \theta_i u_i + \sum_{G_l \in \mathcal{P}} \sum_{i \in G_k} \sum_{j \in G_l} u_i y_j^\star w_{ij} \Big) \end{aligned} \tag{8}$$

We can easily observe that the quantity $\hat{E}(\vec{\xi})$ represents the energy of a network of size p whose state is given by the vector $\vec{\xi} = (\xi_1, \ldots, \xi_p)$ and whose connection weights x_{kl} $k = 1, \ldots, p$, $l = 1, \ldots, p$ take the form:

$$x_{kl} = \sum_{i \in G_k} \sum_{j \in G_l} u_i u_j w_{ij} \tag{9}$$

To be consistent with the energy definition (1) the quantities x_{kk} should be equal to zero. We can achieve this by making a slight change to equation (8). We add the terms $x_{kk} = \sum_{i \in G_k} \sum_{i' \in G_k} u_i u_{i'} w_{ii'}$ to the threshold values ω_k of the p-net and set $x_{kk} = 0$, $k = 1, \ldots, p$. Since the ξ_k values are binary and hence $\xi_k^2 = \xi_k$, we find:

$$\hat{E}(\vec{\xi}) = -\frac{1}{2} \sum_{k=1}^{p} \sum_{l=1}^{p} x_{kl} \xi_k \xi_l - \sum_{k=1}^{p} \omega_k \xi_k \tag{10}$$

where

$$x_{kl} = \begin{cases} \sum_{i \in G_k} \sum_{j \in G_l} u_i u_j w_{ij} & \text{if } k \neq l \\ 0 & \text{if } k = l \end{cases} \tag{11}$$

and the threshold ω_k of each unit k of the small network is:

$$\begin{aligned} \omega_k \;\; = & \sum_{i \in G_k} \theta_i u_i + \sum_{G_l \in \mathcal{P}} \sum_{i \in G_k} \sum_{j \in G_l} u_i y_j^\star w_{ij} \\ & + \frac{1}{2} \sum_{i \in G_k} \sum_{i' \in G_k} u_i u_{i'} w_{ii'} \end{aligned} \tag{12}$$

Alternatively, using equation (2) we obtain the following expression:

$$\omega_k = -\sum_{i \in G_k} \delta E_i(\vec{y^\star}) - \frac{1}{2} \sum_{i \in G_k} \sum_{i' \in G_k} u_i u_{i'} w_{ii'} \tag{13}$$

It is important to note that, since the original network has symmetrical weights the p-net has also symmetrical weights (i.e., $x_{kl} = x_{lk}$ for all $k = 1, \ldots, p$, $l = 1, \ldots, p$), as we can easily observe from equation (11). This fact ensures convergence of the p-net when used in Hopfield-type operations. Another observation is that in order to compute the weight x_{kl} only the weights w_{ij} connecting the units belonging to groups k and l are involved.

The procedure of multiscaling can now be described as follows: while being in a state $\vec{y^\star}$ of the original network, we create a small p-net as defined above. Then we can perform iterations on this network and, using equation (4), we can return to the original network whose energy is now given by equation (7). A characteristic of the method is that the computation of the weights x_{kl} and the values ω_k depends strongly on the state $\vec{y^\star}$ of the n-net at the time of transformation. Thus, the computation must be repeated each time we wish to switch to a p-net. This imposes an overhead that may restrict the number of transitions between the networks. It should be stressed, however, that the cost of switching back to the n-net is negligible.

As a conclusion we can state that it is possible to achieve multiscale operation in a binary Hopfield-type network by apropriately partitioning its units into groups and constructing a binary Hopfield-type coarse-scale network. The operation of the latter emulates the interaction among the groups of units of the original network. According to the result of this interaction all units participating in a group either change or retain their state. A transition of a unit of the coarse-scale network to the 'on' state corresponds to a *group update* in the underlying fine-scale network. Therefore, the operation of multiscaling suggests a more general way of performing state transitions in a Hopfield network: a group of units can be selected for update and, based on the resulting energy difference (which has to be defined properly), all units of the group either accept or reject the state transition. In the following section a formal definition of the notion of group update is presented and the exact correspondence between performing group updates in the fine-scale network and performing single updates in the coarse-scale network is established.

3 Group Updates

The operation of the Hopfield model is based on the notion of single updates. At each step we randomly select a unit and calculate the difference in the energy of the network that would result if the state of this unit was altered. Based on the energy difference, we decide whether the state of the unit must change or not. We shall use the term *trial* to denote the above operation. Changing the state of a unit will be referred to as *update*. Hence, a trial is eventually followed by an update.

In this section, we introduce the notion of *group update* which constitutes a generalization of the notion of single update. The difference lies in the fact that instead of selecting a single unit to consider for update, we can select a group containing a number of units. In a manner analogous to the single unit case, we first perform a trial by calculating the difference in the energy that would result if the state of *all* units in the group was altered. Then a decision must be made whether a group update must take place, i.e., whether the states of *all* these units must change or not. It is clear that group operation differs from *synchronous* operation [1], which consists of simultaneously performing individual trials and updates and also from *block-sequential* operation [10]; according to the latter operation mode, the state vector is divided into subvectors which are sequentially updated, where within each subvector individual updates are performed simultaneously (synchronous operation within each subvector).

Consider a discrete Hopfield-type network with n binary units as defined in the previous section. Suppose that while being in state $\vec{y^\star} = (y_1^\star, \ldots, y_n^\star)$ we consider a group G consisting of m units and perform a trial on this group. If the trial is successful, a group update takes place and the new state $\vec{y} = (y_1, \ldots, y_n)$ of the network will be such that:

$$y_i = \begin{cases} y_i^\star + u_i & \text{if } i \in G \\ y_i^\star & \text{if } i \notin G \end{cases} \tag{14}$$

where $u_i = 1 - 2y_i^\star$, $i = 1 \in G$. Equation (14) simply expresses the fact that units in G change state.

Using equation (14) the energy of the new state $\vec{y}$ takes the form:

$$\begin{aligned} E(\vec{y}) &= -\frac{1}{2}\sum_{i\notin G}\sum_{j\notin G} y_i^\star y_j^\star w_{ij} - \frac{1}{2}\sum_{i\in G}\sum_{j\in G}(y_i^\star + u_i)(y_j^\star + u_j)w_{ij} \\ & -\frac{1}{2}\sum_{i\in G}\sum_{j\notin G}(y_i^\star + u_i)y_j^\star w_{ij} - \frac{1}{2}\sum_{i\notin G}\sum_{j\in G} y_i^\star(y_j^\star + u_j)w_{ij} \\ & -\sum_{i\notin G}\theta_i y_i^\star - \sum_{i\in G}\theta_i(y_i^\star + u_i) \end{aligned} \tag{15}$$

The above equation can be written:

$$\begin{aligned} E(\vec{y}) &= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i^\star y_j^\star w_{ij} - \sum_{i=1}^{n}\theta_i y_i^\star \\ & -\frac{1}{2}\sum_{i\in G}\sum_{j\in G} u_i u_j w_{ij} - \frac{1}{2}\sum_{i\in G}\sum_{j\in G} u_i y_j^\star w_{ij} - \frac{1}{2}\sum_{i\in G}\sum_{j\in G} y_i^\star u_j w_{ij} \\ & -\frac{1}{2}\sum_{i\in G}\sum_{j\notin G} y_j^\star u_i w_{ij} - \frac{1}{2}\sum_{i\notin G}\sum_{j\in G} y_i^\star u_j w_{ij} - \sum_{i\in G}\theta_i u_i \end{aligned} \tag{16}$$

Based on the symmetry of the weights we finally obtain:

$$\begin{aligned} E(\vec{y}) &= E(\vec{y^\star}) - \sum_{i\in G} u_i(\sum_{j=1}^{n} y_j^\star w_{ij} + \theta_i) \\ & -\frac{1}{2}\sum_{i\in G}\sum_{j\in G} u_i u_j w_{ij} \end{aligned} \tag{17}$$

Equation (17) provides the change $\Delta E_G(\vec{y^\star}) = E(\vec{y}) - E(\vec{y^\star})$ in the network's energy caused by the group update. The energy change can also be written in the form:

$$\Delta E_G(\vec{y^\star}) = \sum_{i\in G}\delta E_i(\vec{y^\star}) - \frac{1}{2}\sum_{i\in G}\sum_{j\in G} u_i u_j w_{ij} \tag{18}$$

where, according to (2), $\delta E_i(\vec{y^\star})$ is the change in the network's energy in case the network is in state $\vec{y^\star}$ and unit i is selected for a *single* update.

A more intuitive interpretation of equation (18) can be obtained in terms of the following proposition.

Proposition 1. *The change in the network's energy due to a group update from state $\vec{y^\star}$ is equal to the sum of the individual energy changes that would result if the units of the group sequentially changed state starting from $\vec{y^\star}$ and following an arbitrary order.*

Proof: Suppose that by sequentially performing those changes, the network's state will successively pass through states $\vec{y^1}$, $\vec{y^2}, \ldots, \vec{y^m}$, i.e., we consider an appropriate numbering of units such that the change of the state of unit $i \in G$ causes the transition from state $\vec{y^{i-1}}$ to state $\vec{y^i}$ (we have $\vec{y^\star} = \vec{y^0}$ and $\vec{y^m} = \vec{y}$). Since finally the states of all units in the group have changed, the total energy difference is:

$$\Delta E(\vec{y^\star}) = \sum_{i \in G} \delta E_i(\vec{y^{i-1}}) \tag{19}$$

In state $\vec{y^{i-1}}$ the states of units $1, \ldots, i-1$ have changed. Using the fact that $\delta E_i(\vec{y}) = (2y_i - 1)(\sum_{j=1}^{n} w_{ji} y_j + \theta_i)$ and $y_j^i = y_j^\star + u_j$, $j = 1, \ldots, i$, it is easy to verify that

$$\delta E_i(\vec{y^{i-1}}) = \delta E_i(\vec{y^\star}) - u_i(\sum_{k=1}^{i-1} u_k w_{ik}) \tag{20}$$

Substitution from (20) into (19) yields the expression given by (18) for the energy change resulting from the group update. ∎

It is obvious that in order for an operation style based on group updates to be successful, the identification and selection of appropriate groups of units must take place and this task is highly problem dependent. In many problem mappings there are constraints implying that certain groups of neural variables should have the same state or some pairs of other neural variables should have opposite states. If we start the operation of the Hopfield-type network from an arbitrary initial state satisfying those requirements and proceed by performing the appropriate group updates, we always move between valid states and thus the search is more constrained and effective than in the general case where the whole state space is searched. In such a case, despite the fact that group updates are computationally more expensive, there is a significant performance benefit, since the number of iterations required to reach an acceptable network state is much smaller.

As far as grouping schemes are concerned they can be distinguished into *state dependent* and *state independent*. In the former case, the composition of the group that is considered for update at each time step is determined based on the current state of the network, while in the latter case the groups are specified from the beginning and remain fixed during the operation of the network. We show in the next section that, if a state independent grouping scheme is considered, the use of multiscaling offers a significant performance advantage.

As a matter of fact, the multiscale method is closely related to the notion of group update not only in an intuitive, but also in a formal way, as illustrated by the following theorem.

Theorem 1. *Suppose that while being in state $\vec{y^\star}$ of the n-net, a p-net is created. Then, the change in the n-net's energy in state $\vec{y}$ caused by a group update concerning the units of the k-th group, is equal to the change in the p-net's energy caused by the single update of unit k in state $\vec{\xi}$, provided that the two networks are in consistent states, i.e., $y_i = y_i^\star + u_i\xi_k$ if $i \in G_k$.*

Proof: If the p-net is in state $\vec{\xi} = (\xi_1, \ldots, \xi_p)$, the change in the network's energy caused by a *single update* concerning unit k, is

$$\delta \hat{E}_k(\vec{\xi}) = (2\xi_k - 1)(\sum_{l=1}^{p} \xi_l x_{kl} + \omega_k) \tag{21}$$

Consider now the n-net being in state $\vec{y} = (y_1, \ldots, y_n)$, where $y_i = y_i^\star + u_i\xi_k$ if $i \in G_k$. For each $i = 1, \ldots, n$ we define

$$u_i' = 1 - 2y_i \tag{22}$$

From equations (4) and (5) we find that

$$u_i' = u_i(1 - 2\xi_k), \quad i \in G_k \tag{23}$$

According to equation (18), the energy change caused by a group update concerning group G_k of the n-net is:

$$\Delta E_{G_k}(\vec{y}) = \sum_{i \in G_k} \delta E_i(\vec{y}) - \frac{1}{2} \sum_{i \in G_k} \sum_{i' \in G_k} u_i' u_{i'}' w_{ii'} \tag{24}$$

Since

$$\delta E_i(\vec{y}) = -u_i'(\sum_{G_l \in \mathcal{P}} \sum_{j \in G_l} y_j w_{ij} + \theta_i) \tag{25}$$

we have

$$\begin{aligned} \Delta E_{G_k}(\vec{y}) \quad = \quad & - \sum_{i \in G_k} u_i' \sum_{G_l \in \mathcal{P}} \sum_{j \in G_l} y_j w_{ij} + \theta_i) \\ & - \frac{1}{2} \sum_{i \in G_k} \sum_{i' \in G_k} u_i' u_{i'}' w_{ii'} \end{aligned} \tag{26}$$

Using equations (4) and (23) we obtain

$$\begin{aligned}\Delta E_{G_k}(\vec{y}) &= -(1-2\xi_k)(\sum_{G_l\in\mathcal{P}}\sum_{i\in G_k}\sum_{j\in G_l} u_i u_j w_{ij}\xi_l)\\ &\quad -(1-2\xi_k)(\sum_{G_l\in\mathcal{P}}\sum_{i\in G_k}\sum_{j\in G_l} u_i y_j^\star w_{ij})\\ &\quad -(1-2\xi_k)(\sum_{i\in G_k} u_i\theta_i)\\ &\quad -\frac{1}{2}(1-2\xi_k)^2(\sum_{i\in G_k}\sum_{i'\in G_k} u_i u_{i'} w_{ii'}) \end{aligned} \tag{27}$$

or

$$\begin{aligned}\Delta E_{G_k}(\vec{y}) &= -(1-2\xi_k)(\sum_{\substack{G_l\in\mathcal{P}\\ l\neq k}}\sum_{i\in G_k}\sum_{j\in G_l} u_i u_j w_{ij}\xi_l)\\ &\quad -(1-2\xi_k)(\sum_{i\in G_k}\sum_{i'\in G_k} u_i u_{i'} w_{ii'})\xi_k\\ &\quad -(1-2\xi_k)[\sum_{i\in G_k} u_i(\sum_{G_l\in\mathcal{P}}\sum_{j\in G_l} y_j^\star w_{ij}+\theta_i)]\\ &\quad -(1-2\xi_k)(\frac{1}{2}\sum_{i\in G_k}\sum_{i'\in G_k} u_i u_{i'} w_{ii'})\\ &\quad +(1-2\xi_k)(\sum_{i\in G_k}\sum_{i'\in G_k} u_i u_{i'} w_{ii'})\xi_k \end{aligned} \tag{28}$$

Substitution from (11) and (12) into (28) finally yields:

$$\Delta E_{G_k}(\vec{y}) = \delta\hat{E}_k(\vec{\xi})$$

∎

Therefore, the operation of the coarse-scale network is given a meaningful physical interpretation, since it constitutes an exact simulation of group updates in the fine-scale network. An important consequence of the above theorem is that, if for a neural network architecture there is an effective state independent grouping scheme, then instead of performing group updates it is computationally more efficient to create the corresponding coarse-scale network and proceed by performing single updates in this network. In the next section we consider optimization problems for which there is a naturally arising fixed (state independent) grouping scheme and thus the employment of multiscaling is very beneficial.

4 Application to Optimization

We shall now examine the appropriateness of the multiscale method in solving optimization problems by considering the *set partitioning* and *set covering* problems. For these two problems we have developed original mappings to the corresponding fine-scale Hopfield network, which are of interest by themselves and allow an effective exploitation of the advantages offered by multiscaling. A similar methodology can be applied to most of the optimization problems belonging to the broad category of *partitioning, covering, hitting* and *splitting* problems [5].

4.1 Application to Set Partitioning

The formulation of the Set Partitioning Problem (SPP) has as follows [5]: Given a finite set S containing M elements and a collection C of subsets S_i $(i = 1, \ldots, N)$ of S, find the minimum subset F of C such that all the subsets belonging to F are disjoint and they constitute a partition of S, i.e., their union is equal to S.

In order to deal with this problem we construct the following neural network architecture. Let k_i $(i = 1, \ldots, M)$ denote the number of occurences of the i-th element of S in the subsets belonging to C. For example, if $k_1 = 2$ there are two subsets in C that contain the first element of S. The units of the neural network are arranged in M rows, such that row i corresponds to the i-th element of S and contains k_i units, each corresponding to an occurence of that element in the subsets belonging to C. Thus, the unit (i, j) corresponds to the j-th occurence of the i-th element of S in the subsets belonging to C. If in the final state of the network a unit (i, j) is in the 'on' state, the subset S_l corresponding to the j-th occurence of the i-th element of S must be incorporated in the solution set.

The problem constraints require that every element of S should be covered and the subsets belonging to the solution set should be disjoint. This implies that in the final equilibrium state there must be exactly one 'on' unit in each row, i.e., each element must be covered by one and only one subset. We shall construct the network so that every equilibrium state fulfils the above requirement. It is obvious, however, that this property of equilibrium states solves the problem of disjointness only partially, as it is still possible to have 'on' units in different rows corresponding to non-disjoint subsets. This problem can be solved if we enforce the network to reach equilibrium points where either all units corresponding to the same subset are in the 'on' state or all are in the 'off' state. We shall call these equilibrium points *feasible* since they correspond to feasible (valid) solutions of the original problem [1]. The network is constructed in such a way that every feasible equilibrium point is characterized by lower energy than any non-feasible equilibrium point. Thus, the lower the energy of the equilibrium point attained by the network the better the chance of finding valid solutions of the Set Partitioning problem instance. It must be noted that the problem of finding even one valid solution (not nessecarily the optimal) of the SPP is NP-hard. Therefore it is not possible to construct a neural network with all equilibrium points corresponding to valid problem solutions. Another important characteristic of the proposed architecture is that the resulting energy function is *order preserving*, in the sense that for feasible equilibrium points, the lower the corresponding energy the lower the size of the resulting solution set F [1].

It can be ensured that there is exactly one 'on' unit in each row in an equilibrium state, by assigning a positive threshold value to each unit and imposing a strong negative connection between any pair of units belonging to the same row (no negative connections are assumed between units belonging to different rows). On the other hand, to favour feasible equilibrium states, a positive connection is imposed between any pair of units that correspond to the same subset.

Let us first consider positive connections and threshold values. We assign the value $\delta_1/|S_r|$ to the strength of the positive connection imposed between any pair of units (i,j) and (k,l) (with $i \neq k$) such that the j-th occurence of the i-th element of S and the l-th occurence of the k-th element of S correspond to the same subset S_r in C. The threshold value of unit (i,j) is determined as follows:

$$\theta_{ij} = \alpha - \frac{\delta_1(|S_r| - 1 + \delta_2)}{2|S_r|} \tag{29}$$

where S_r is the member of C corresponding to unit (i,j) and $|S_r|$ denotes the cardinality of the set S_r. The value of the parameter $\alpha > 0$ can be arbitrarily specified, while the values of δ_1 and δ_2 will be determined from the requirements of the problem as will be described next. Note that, actually, both positive connection weights and threshold values are common for all units corresponding to a given subset S_r.

Since in an equilibrium state each row contains exactly one 'on' unit, the network energy has the form:

$$E = -M\alpha - \sum_{r=1}^{N} \delta_1 \frac{z_r(z_r - |S_r| - \delta_2)}{2|S_r|} \tag{30}$$

where z_r ($r = 1, \ldots, N$) denotes the number of 'on' units that correspond to subset S_r.

The value of δ_2 can be determined from the requirement that every feasible equilibrium state should correspond to lower energy than any non-feasible equilibrium state. Let us define the function

$$f_s(z) = \frac{z}{s}(s - z + \delta_2) \tag{31}$$

with $f_s(0) = 0$ and $f_s(s) = \delta_2$, where s is a positive integer and z is an integer such that $0 \leq z \leq s$. Using the above function, equation (30) takes the form:

$$E = -M\alpha + \frac{\delta_1}{2} \sum_{r=1}^{N} f_{|S_r|}(z_r) \tag{32}$$

The energy corresponding to a feasible equilibrium state (i.e., either $z_r = |S_r|$ or $z_r = 0$ for all r) is given by

$$E = -M\alpha + |F| \frac{\delta_1 \delta_2}{2} \tag{33}$$

where $|F|$ is the size of the solution set F. The maximum value that can be assigned to $|F|$ is N, which means that the maximum energy of a feasible equilibrium state is $-M\alpha + N\delta_1\delta_2/2$.

On the other hand, it can be readily verified that, if $0 < \delta_2 < 1$, all integer values of z in the interval $(0, s)$ (with $s > 1$) yield values of f_s greater than δ_2. Also the minimum of $f_s(z)$ in $0 < z < s$ $(s > 1)$ occurs for $z = 1$ and has the value $1 - (1 - \delta_2)/s$. Hence, a lower bound of $\sum_{r=1}^{N} f_{|S_r|}(z_r)$ for $0 < z_r < |S_r|$ is the quantity $1 - (1 - \delta_2)/s_{\min}$, where $s_{\min}$ denotes the minimum value of $|S_r|$ $(r = 1, \ldots, N)$. Therefore, the minimum energy of a non-feasible equilibrium state cannot be less than $-M\alpha + [1 - (1 - \delta_2)/s_{\min}]\delta_1/2$.

Consequently, the requirement that every feasible equilibrium state should correspond to lower energy than any non-feasible equilibrium state implies that

$$1 - \frac{1 - \delta_2}{s_{\min}} > N\delta_2 \tag{34}$$

which yields

$$\delta_2 < \frac{s_{\min} - 1}{N s_{\min} - 1} \tag{35}$$

Moreover, as can be observed from equation (33), the energy function is order preserving since between two arbitrary feasible equilibrium states the one that corresponds to lower energy results also in a solution set F of smaller size.

The value of the parameter δ_1 can be determined from the requirement that the threshold value θ_{ij} should be greater than zero for all units (i, j). Using (29) we readily obtain the following inequality:

$$\delta_1 < 2\alpha \frac{s_{\max}}{s_{\max} - 1 + \delta_2} \tag{36}$$

where $s_{\max}$ denotes the maximum value of $|S_r|$ $(r = 1, \ldots, N)$.

After having determined positive connection weights and threshold values, it remains to specify the negative weight between each pair of units belonging to the same row, which must ensure that it is not possible to have more than one 'on' unit in each row in an equilibrium state. We define the *positive potential* of a unit in a given state as the sum of the threshold of that unit (which is assumed positive) and of the positive excitation it accepts from other units. If unit (i, j) corresponds to subset S_r, its maximum positive potential over all states is $b_{ij} = \theta_{ij} + \delta_1(|S_r| - 1)/|S_r|$. (Obviously, the maximum positive potential is the same for all units corresponding to the same subset.) It can be easily verified that, if the strength of the negative connection weight is greater than the maximum positive potential of both units it connects, there can be no more than one 'on' unit in each row in an equilibrium state. We shall consider a common value b for the strength of all negative connections of the network, which is taken greater than the maximum b_{ij} over all units (i, j). It can be seen that $b_{ij} > \alpha$ for $|S_r| > 1$. For reasons that will become apparent next (Proposition 3), in any case the value of b is taken greater than 2α.

The proposed Hopfield-type network can operate as a Boltzmann Optimizer to minimize the energy function and, consequently, to find feasible solution sets of minimum size for the SPP. It is also apparent that the employment of multiscaling is very attractive for this problem, since there is a state independent grouping of the units of the network. According to this grouping, all units corresponding to the same subset in C should participate in the same group and one unit of the coarse-scale network is allocated to each of these groups. If creation of the small network takes place at a state of the original network, where all units belonging to the same group are in the same state, then the created network has an additional important feature, as illustrated by the following proposition.

Proposition 2. *The feasible equilibrium points of the fine-scale network correspond to equilibrium points of the coarse-scale network.*

Proof: Suppose that we create the coarse-scale network while being at a state $\vec{y^\star}$ having the property that all units belonging to the same group are in the same state. Suppose also that $\vec{y}$ is a feasible equilibrium state of the fine-scale network. Then, the corresponding state $\vec{\xi} = (\xi_1, \ldots, \xi_N)$ of the coarse-scale network satisfies that $y_{ij} = y^\star_{ij} + u_{ij}\xi_k$ for $(i, j) \in G_k$. In order to prove that $\vec{\xi}$ is an equilibrium state, it is sufficient to show that $\delta\hat{E}_k(\vec{\xi}) > 0$ for each unit $k = 1, \ldots, N$ of the coarse-scale network. From Theorem 1 this energy difference is equal to the energy difference caused by the corresponding group update $\Delta E_{G_k}(\vec{y})$ in the fine-scale network. Therefore it is sufficient to show that, in a feasible equilibrium state $\vec{y}$, holds that $\Delta E_{G_k}(\vec{y}) > 0$ for all groups G_k, $k = 1, \ldots, N$, (where each group G_k contains units corresponding to the same subset S_k).

We distinguish two cases:

i) All units belonging to group G_k are in the 'on' state. Since there is exactly one 'on' unit in each row, we have for each unit $(i,j) \in G_k$:

$$\delta E_{ij}(\vec{y}) = \theta_{ij} + (|S_k| - 1)\frac{\delta_1}{|S_k|} \tag{37}$$

Using equation (18) we find that

$$\Delta E_{G_k}(\vec{y}) = \sum_{(i,j)\in G_k} \theta_{ij} + |S_k|(|S_k| - 1)\frac{\delta_1}{|S_k|} - \frac{|S_k|(|S_k| - 1)}{2}\frac{\delta_1}{|S_k|} \tag{38}$$

which is greater than zero.

ii) All units belonging to group G_k are in the 'off' state. Since there is exactly one unit 'on' in each row, each unit $(i,j) \in G_k$ receives an inhibition of strength b from the unit that is 'on' in row i. Therefore:

$$\delta E_{ij}(\vec{y}) = -(\theta_{ij} - b) \tag{39}$$

and using (18) we obtain

$$\Delta E_{G_k}(\vec{y}) = |S_k|(b - \alpha) + |S_k|\frac{\delta_1(|S_k| - 1 + \delta_2)}{2|S_k|} - \frac{|S_k|(|S_k| - 1)}{2}\frac{\delta_1}{|S_k|} \tag{40}$$

which yields

$$\Delta E_{G_k}(\vec{y}) = |S_k|(b - \alpha) + \frac{\delta_1\delta_2}{2} > 0 \tag{41}$$

since $b > \alpha$. ∎

The equilibrium points of the coarse-scale network that correspond to feasible equilibrium points in the fine-scale network will be referred to as *feasible coarse equilibrium points.* All other equilibrium points of the coarse-scale network will be referred to as non-feasible coarse equilibrium points. It is possible to extend the result of Proposition 2 in terms of the following:

Proposition 3. *The feasible coarse equilibrium points correspond to lower energy than any non-feasible coarse equilibrium point.*

Proof: Suppose that we create the coarse-scale network while being at a state $\vec{y^\star}$ which, as already stated, satisfies the property that all units belonging to the same group are in the same state. Since $E(\vec{y^\star})$ is constant, we have from equation (7) that, if for two arbitrary states $\vec{\xi^1}$ and $\vec{\xi^2}$ the inequality $\hat{E}(\vec{\xi^1}) < \hat{E}(\vec{\xi^2})$ holds, then, equivalently, the corresponding states of the original network will satisfy $E(\vec{y^1}) < E(\vec{y^2})$. Consequently, in order to prove the above proposition, it is sufficient to show that for every feasible coarse equilibirum point the energy of the corresponding state of the original network is lower than the energy of states corresponding to non-feasible coarse equilibrium points.

If $\vec{\xi}$ is a feasible coarse equilibrium point, the energy of the corresponding state $\vec{y}$ is given by equation (33). Using (34) and (36) it can be shown that $N\delta_1\delta_2/2 < \alpha$, and, therefore, we have for every $|F| \leq N$ that $|F|\delta_1\delta_2/2 < \alpha$. Consequently, the energy E_f of a feasible equilibrium state satisfies:

$$E_f < -(M-1)\alpha \tag{42}$$

On the other hand, if a coarse equilibrium point $\vec{\xi}$ is not feasible, the resulting state $\vec{y}$ will have the following characteristics: (i) due to the group update operation, for every subset S_r, either $z_r = 0$ or $z_r = |S_r|$ $(r = 1, \ldots, N)$ and (ii) there will be at least one row having either none or more than one unit in the 'on' state.

Consider first the case where there is a row that does not contain any unit in the 'on' state, whereas all other rows contain exactly one 'on' unit each. The energy of state $\vec{y}$ takes the form:

$$E_1 = -(M-1)\alpha + |F|\frac{\delta_1\delta_2}{2} \tag{43}$$

Clearly, for every value of $|F|$ holds that

$$E_1 > E_f \tag{44}$$

Consider now the case where there is a row containing two units in the 'on' state, and the remaining rows contain exactly one 'on' unit each. The corresponding energy of the original network would be:

$$E_2 = -(M+1)\alpha + b + |F|\frac{\delta_1\delta_2}{2} \tag{45}$$

Given that $b > 2\alpha$, we find that $E_2 > -(M-1)\alpha$ and thus for every value of $|F|$:

$$E_2 > E_f \tag{46}$$

Inequality (46) becomes stronger in the case where there exist more than two units in the 'on' state in the same row. Moreover, it is obvious that inequalities (44) and (46) hold also in the case where there are more than one rows that violate the constraint of having exactly one 'on' unit. Therefore, in any case we see that the feasible coarse equilibrium points correspond to lower energy than the non-feasible ones. ∎

According to Propositions 2 and 3, the feasible solutions of a given instance of the Set Partitioning Problem correspond to the low energy states of the coarse-scale network. Consequently, the construction of the p-net allows us to approach (or even attain) the optimal solution despite the fact that the operation takes place at a coarse level. Therefore, one can use the multiscale method for the creation of the p-net and then perform iterations exclusively in that network in order to reach a feasible equilibrium state.

4.2 Application to Set Covering

The Set Covering Problem (SCP) has the following definition [5]: Given a finite set S containing M elements and a collection C of subsets S_i $(i = 1, \ldots, N)$ of S, find the minimum subset F of C such that the subsets belonging to F cover the set S, i.e., their union is equal to S.

The neural network architecture constructed to solve the problem features analogy to the one described in the last subsection. As previously, we denote by k_i $(i = 1, \ldots, M)$ the number of occurences of the i-th element of S in the subsets belonging to C and consider a network with its units arranged in M rows. Each row $i = 1, \ldots, M$ is composed of two collections of units. The first collection contains k_i units, with unit (i, j) corresponding to the j-th occurence of the i-th element of S in the subsets belonging to C. If in the final state of the network a unit (i, j) of the above type is in the 'on' state, the subset S_l corresponding to the j-th occurence of the i-th element of S must be incorporated in the solution set F.

As every element of S must be covered, there must be *at least one* unit in the 'on' state in each row in the final state of the network, i.e., there must be at least one subset covering each element. Satisfaction of this requirement can be achieved by letting each row i $(i = 1, \ldots, M)$ contain a collection of $k_i - 1$ additional units and by imposing the constraint that in every equilibrium state there must be exactly k_i 'on' units in row i. A simple way to do this is to set a negative connection of strength α between each pair of units belonging to the same row and specify the threshold values and the strength of the positive connections of the network so that the maximum positive potential of each unit (i, j) will be less than αk_i and the minimum positive potential will be greater than $\alpha(k_i - 1)$. The additional units shall be called *slack units* [19] and will be distinguished from the originally considered first collection of units, which correspond to subsets and shall be called *actual units*. The above construction scheme ensures that all equilibrium states of the network are feasible since they satisfy the covering constraint.

A positive connection of strength $\delta_1/|S_r|$ is imposed between any pair of actual units (i, j) and (k, l) (with $i \neq k$), such that the j-th occurence of the i-th element of S and the l-th occurence of the k-th element of S correspond to the same subset S_r in C. This is necessary, since it is desirable in equilibrium states to have either all units corresponding to the same subset in the 'on' state or all of them in the 'off' state. The above option leads to reduced redundancy in the final solution set, by avoiding the existence of subsets that do not contribute to the covering of S as all their elements are covered by other subsets of the solution. We call the equilibrium points having the above property *complete*. The weights and thresholds of the network are specified so that complete equilibrium states correspond to lower energy than any other equilibrium state.

The threshold value of unit (i, j) is specified as follows. If unit (i, j) is an actual unit corresponding to subset S_r then

$$\theta_{ij} = \alpha(k_i - \delta_3) - \frac{\delta_1(|S_r| - 1 + \delta_2)}{2|S_r|} \tag{47}$$

where $\delta_1 > 0$ and $0 < \delta_2, \delta_3 < 1$. If unit (i, j) is a slack one, then $\theta_{ij} = \alpha(k_i - \delta_3)$. (Note that, in the case of Set Covering, threshold values depend also upon the row index i.) We next discuss the specification of the parameters δ_1 and δ_2.

Since in an equilibrium state there are exactly k_i 'on' units in each row i, the corresponding energy has the form:

$$E = \sum_{i=1}^{M} \frac{k_i(k_i - 1)}{2}\alpha - \sum_{i=1}^{M} k_i(k_i - \delta_3)\alpha - \sum_{r=1}^{N} \delta_1 \frac{z_r(z_r - |S_r| - \delta_2)}{2|S_r|} \tag{48}$$

where z_r, $(r = 1, \ldots, N)$ denotes the number of 'on' units that correspond to subset S_r. The first two terms on the right-hand side of the above equation consitute a constant quantity which will be denoted by K. Thus,

$$E = K + \sum_{r=1}^{N} \delta_1 \frac{z_r(|S_r| - z_r + \delta_2)}{2|S_r|} \tag{49}$$

The value of δ_2 can be determined from the requirement that every complete equilibrium state should correspond to lower energy than any other equilibrium state. Using the function $f_s(z)$ defined in the previous subsection, equation (49) takes the form:

$$E = K + \frac{\delta_1}{2} \sum_{r=1}^{N} f_{|S_r|}(z_r) \tag{50}$$

The energy corresponding to a complete equilibrium state (i.e., either $z_r = |S_r|$ or $z_r = 0$ for all r) is given by

$$E = K + |F|\frac{\delta_1 \delta_2}{2} \tag{51}$$

where $|F|$ is the size of the subset collection F that corresponds to the solution of the problem. Using similar arguments as in the case of the SPP we find that, in order for every complete equilibrium state to correspond to lower energy than any non-complete equilibrium state, the following inequality must hold:

$$\delta_2 < \frac{s_{\min} - 1}{N s_{\min} - 1} \tag{52}$$

Moreover, the resulting energy function is order preserving for complete equilibrium states, as can be observed from equation (51).

The value of δ_1 can be derived based on the requirement that the maximum positive potential of each actual unit (i, j) must be less than αk_i and the minimum positive potential must be greater than $\alpha(k_i - 1)$. The maximum positive potential of actual unit (i, j) corresponding to subset S_r is $\theta_{ij} + \delta_1(|S_r| - 1)/|S_r|$, while the mimimum positive potential is equal to θ_{ij}. This leads to the following two inequalities which must be simultaneously satisfied by the value of δ_1:

$$\delta_1 < 2\alpha\delta_3 \frac{s_{\max}}{s_{\max} - 1 - \delta_2} \tag{53}$$

and

$$\delta_1 < 2\alpha(1 - \delta_3) \frac{s_{\max}}{s_{\max} - 1 + \delta_2} \tag{54}$$

Therefore, in order to obtain the desired network characteristics, first the values of $\alpha > 0$ and $0 < \delta_3 < 1$ must be arbitrarily specified, then the value of δ_2 must be determined so as to satisfy (52) and, finally, the value of δ_1 must be determined according to the inequalities (53) and (54).

A Boltzmann Optimizer based on the above defined network is used to find near-optimal solutions for the Set Covering Problem. As in the SPP case, a considerable benefit is expected from the employment of multiscaling, since there is a state independent grouping of the actual units of the network. According to this grouping, all actual units corresponding to the same subset in C should participate in the same group and one unit of the coarse-scale network is allocated to each of these groups. Slack units are maintained in the p-net as individual units. In analogy with the SPP, we create the small network from a state of the original network, where all actual units belonging to the same group are in the same state.

In analogy with the SPP, it can be proved that the states of the resulting coarse-scale network that correspond to complete equilibrium points of the fine-scale network are also equilibrium points. In addition, it can be shown that the complete coarse equilibrium points are of lower energy than any non-complete coarse equilibrium point. As a result, it is possible to find complete equilibrium states of the original network (and therefore near-optimal problem solutions) just by operating at the coarse-scale level.

It is possible for the network to reach a final solution where there are still redundant subsets in the sense that all their elements are covered by other subsets of the solution set. These subsets can be removed from the solution set F without violating the covering constraint. To achieve this, we add a processing stage after termination of the multiscale algorithm. During this stage we successively specify groups in the fine-scale network and perform the corresponding group updates. At each step, the specification of the group of units to be updated is done as follows. At first, all rows having at least one slack neuron in the 'off' state are marked. Then we search for subsets (contained in the final solution) satisfying the property that all their corresponding units are 'on' and belong to a marked row. If such a subset is found, we consider the group containing the actual units corresponding to that subset and one 'slack' unit in the 'off' state from each of the respective marked rows. A group update concerning this group will further reduce the energy of the network since it removes one redundant set. The above operation is repeated until no suitable group can be specified.

4.3 Experiments

In order to evaluate the efficiency of our approach a series of experiments has been conducted regarding both the Set Partitioning and the Set Covering problems. These problems are of particular interest, since they constitute the basic formulation of many scheduling problems. The objective of the experiments was to examine the suitability of the proposed original fine-scale architectures for providing valid near optimal solutions and then to assess the capability of the resulting coarse-scale networks to speed up execution time and attain low energy states despite the fact that they operate at a different level of granularity.

To generate the instances of the SPP first the number M of elements of the set S and the number N of subsets in the collection set C were specified. Then, in order to ensure the existence of at least one valid solution, we first constructed a number of subsets of S by allocating each of the elements of S to one and only one of these subsets so as to create a disjoint solution. The remaining subsets were randomly constructed by deciding with probability q whether an element of S would belong to a particular subset. By varying the value of q the density of the resulting problem instance could be adjusted. The value of q was chosen in the range from 0.05 (sparse instances) to 0.4 (dense instances). For the generated problem instances, first the optimal solution was found using exhaustive search. Then, the fine-scale architecture was used to solve the problem and finally the multiscale approach was applied.

As far as the fine-scale architecture is concerned, we tested its effectiveness on small and medium (N smaller than 100 and M smaller than 50) problem instances since for large problem instances the size of the resulting neural network was very high (over 1000 units). Both sparse and dense instances were considered. In what concerns the annealing schedule, we have adopted the following logarithmic scheme [20]:

$$T_n = \frac{T_{n-1}}{1 + \log f(n)} \tag{55}$$

where $f(n) = f(n-1)(1+r)$ (with $f(0) = 1$) and r is the parameter that adjusts the speed of the schedule. All experiments with the fine-scale architecture were conducted starting from temperature $T_0 = 10.0$ and using a temperature reduction rate $r = 10^{-7}$. Also the value of the parameter α was fixed to 10.0 for all experiments. It should be noted that the above starting temperature is appropriate for this specific choice of the value of α. For greater (smaller) values of α, the starting temperature must be higher (lower). At each temperature, a number of trials equal to $2n$ (where n is the number of network units) was performed and the annealing stopped if for 10 consecutive temperature values none of the units changed its state. The average value of the final annealing temperature over all experiments with the fine-scale network was 1.74. The results obtained from the experiments are shown in Table 1 for small and medium size instances and in Table 2 for big size instances. As a metric of the quality of a feasible solution obtained for a given instance I of the problem we use the *approximation ratio* which is defined as $h(I)/opt(I)$, where $h(I)$ denotes the cardinality of the obtained feasible solution and $opt(I)$ denotes the cardinality of the optimal solution [17]. In each row of Tables 1 and 2, the percentage of feasible solutions obtained and the approximation ratio averaged over all feasible solutions are depicted. The approximation ratio is not meaningful in cases where the percentage of feasible solutions is less than 50%. As can be observed, solutions of acceptable quality are obtained only for small problem instances (up to 50 subsets and 10 elements).

The above situation is remedied through the multiscale approach and the construction of the coarse-scale network, whose size is equal to the number of subsets N and independent

		Feasible %		*Approximation ratio*		*Average size*	*Average*
N	M	*Fine scale*	*Coarse scale*	*Fine scale*	*Coarse scale*	*Fine scale*	speedup
30	10	86.0	100.0	1.32	1.08	80	2.05
30	20	62.5	100.0	1.12	1.25	114	3.38
50	10	67.5	100.0	1.37	1.23	133	2.17
50	30	52.0	100.0	1.58	1.11	212	7.94
50	50	28.0	100.0	—	1.14	377	13.52
100	10	57.0	100.0	1.42	1.27	254	3.42
100	30	12.0	87.5	—	1.29	798	12.59
100	50	5.0	75.0	—	1.16	922	18.67

Table 1: SPP: Small and medium size instances

		Feasible %	*Approximation ratio*
N	M	*Coarse scale*	*Coarse scale*
200	50	100.0	1.19
200	100	95.0	1.23
200	150	91.0	1.37
350	50	100.0	1.22
350	100	92.0	1.27
350	150	86.0	1.36
500	50	83.0	1.33
500	100	72.0	1.42
500	150	68.0	1.44

Table 2: SPP: Big size instances

of the number of elements and the value of q. In all tested instances, first the fine-scale network was constructed and then the creation of the p-net took place at the zero state of the n-net, i.e., $y_i^\star = 0$ for all i. The operation of the p-net started from temperature $T_0 = 20.0$ and followed a temperature reduction rate $r = 10^{-7}$, while the average final temperature value was 3.6. As the weights and thresholds of the coarse-scale network are greater than in the fine-scale case (in absolute values) a higher value of T_0 was chosen for the annealing schedule [1]. After the operation of the p-net was terminated, the corresponding state of the n-net was found and the fine-scale network operated as a pure Hopfield network (with zero temperature) to provide the final solution to the problem.

In what concerns the quality of the resulting solution, when the above annealing schedule was employed, feasible solutions were obtained for most of the experiments. In accordance with Propositions 2 and 3, all feasible solutions were attained by performing iterations only in the p-net, i.e., the resulting state $\vec{y}$ of the n-net was a feasible equilibrium point and further operation at the fine-scale level provided no improvement. In the case of non-feasible coarse equilibrium points, the fine-scale network simply settled to the nearest equilibrium state (which was not feasible). This result establishes the capability of the coarse-scale network to effectively search the constrained state space and provide solutions of very good quality. The average case approximation ratio over all tested instances was 1.25. It must be noted that solutions of better quality were found in the case of dense than in the case of sparse problem instances.

In addition, a significant benefit in terms of execution time was achieved, in comparison to the case where the fine-scale network operates solely. This is due to the fact that the resulting p-net is much smaller and, thus, the number of trials performed at each temperature is considerably reduced. (The speedup can be estimated as $O(n^2/p^2)$, since, for a network of size l, $2l$ iterations are performed at each temperature with each iteration requiring l multiplications.) Since, due to the problem formulation, just one execution of the p-net is adequate to obtain near-optimal solutions, this benefit is significantly higher than the overhead imposed by the construction of the p-net and by the fact that the annealing starts at a higher temperature. As Table 1 indicates, the average speedup for small and medium size instances was in the range between 2 and 18, while for large instances the speedup could not be determined, as the big size of the fine-scale network did not allow us to find acceptable solutions in reasonable time (Table 2).

It should also be noted that the quality of the obtained solution depends heavily on the choice of the temperature reduction rate. We have found that the chosen rate value

N	M	*Approximation ratio*
30	10	1.12
30	20	1.18
50	10	1.27
50	30	1.03
50	50	1.13
100	10	1.18
100	30	1.31
100	50	1.24

Table 3: Set Covering Problem

provides solutions of good quality in reasonable execution time. In general, with smaller (larger) rate values the quality of the solution is better (worse), but the annealing procedure is slower (faster).

Similar experiments were performed for the Set Covering problem. Due to the existence of slack neurons, the size of the resulting networks was bigger in this case, thus the experiments were performed for problem instances with N between 30 and 100 and M between 10 and 50. Moreover, both sparse and dense problem instances were examined. The value of α was fixed to 10 and the value of δ_3 was fixed to 0.1 for all the experiments. The construction of a problem instance was similar to the SPP case, except that we were not concerned to create a disjoint solution. Care was taken so that every element participated in at least one subset. Due to the appropriate construction of the network all resulting solutions are feasible, i.e., all elements of S are covered.

Experiments with the fine-scale network have shown that the quality of the obtained solutions was poor, even for small problem instances. On the contrary, the multiscale method provided results of good quality. The multiscale scheme used in the experiments had the same characteristics as in the SPP case, except for the fact that the initial temperature of the p-net was set equal to 0.5. The annealing rate of the p-net was $r = 10^{-7}$ and the average final temperature was 0.11. In addition, as described in the previous section, a postprocessing stage was added to remove redundant subsets from the final solution set. Table 3 depicts the average case approximation ratio for instances of various sizes, while the overall average case approximation ratio is equal to 1.18. As can be observed, the obtained results are quite satisfactory.

5 Conclusions

We have introduced the notion of group update in the context of Hopfield-type neural networks having binary computing elements, and we have shown that the multiscale approach is an easy and fast way for performing group updates. It has been proved that a single update in the coarse-scale network is equivalent (in terms of the resulting energy difference) to the corresponding group update in the original fine-scale network. This fact makes the multiscale approach very attractive, especially in problems where state independent grouping schemes can be devised.

We have exploited the idea of multiscaling in solving optimization problems, such as the Set Partitioning and Set Covering problems. An original mapping of these problems to a binary Hopfield neural network architecture has been constructed, which is characterized by an order preserving energy function. Then, by appropriately grouping the units of this network, a coarse-scale network was created having the property that the low energy equilibrium points correspond to feasible equilibrium points of the original network. Thus, feasible solutions can be obtained by operating exclusively at the coarse level, allowing for both faster and more effective search of the problem state space. In such a case, multiscaling can be considered as a method for developing a coarse representation of the problem without losing any significant information. A similar construction methodology (concerning the creation of both the fine-scale and the coarse-scale networks) can also be applied to other relevant combinatorial optimization problems such as the Set Packing, Independent Set, Vertex Covering, Set Splitting, Hitting Set etc.

References

[1] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines, A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley & Sons Ltd., 1989.

[2] E. Aarts and J. Korst, *Boltzmann Machines for Travelling Salesman Problems*, European Journal of Oper. Res., Vol. 39, pp. 79-95, 1989.

[3] D. Ackley, G. Hinton and T. Sejnowski, *A Learning Algorithm for Boltzmann Machines*, Cognitive Science, Vol.9, pp. 147-169, 1985.

[4] F. Chatelin and W.L. Miranker, *Acceleration by Aggregation of Successive Approximation Methods*, Linear Algebra and its Applications, Vol. 43, pp. 17-47, 1982.

[5] M. Garey and D. Johnson, *Computers and Intractability: a Guide to the Theory of NP-completeness*, W.H. Freeman and Company, San Francisco, 1979.

[6] L. Herault and J. Niez, *Neural Networks and Combinatorial Optimization: a Study of NP-complete Graph Problems*, in *Neural Networks: Advances and Applications*, ed. E. Gelenbe, North-Holland, 1991.

[7] J. Hopfield, *Neural Networks and Physical Systems with Emergent Collective Computational Abilities*, Proc. of the Nat. Academy of Sciences USA, Vol. 79, pp. 2554-2558, 1982.

[8] J.J. Hopfield and D.W. Tank, *Neural Computation of Decisions in Optimization Problems*, Biological Cybernetics, Vol. 52, pp. 141-152, 1985.

[9] W. Huckbusch, *On the Multigrid Method Applied to Difference Equations*, Computing, Vol. 20, pp. 291-306, 1978.

[10] Y. Kamp and M. Hasler, *Recursive Neural Networks for Associative Memory*, John Wiley & Sons, 1990.

[11] S. Kirkpatrick, C.D. Gellat Jr. and M.P. Vecchi, *Optimization by Simulated Annealing*, Science, Vol. 220, pp. 671-689, 1983.

[12] S. Kirkpatrick, *Optimization by Simulated Annealing: Quantitative Studies*, Journ. Statist. Phys. 34, pp. 974, 1984.

[13] A. Likas and A. Stafylopatis, *Multiscaling in binary Hopfield-type neural networks: application to the Set Partitioning Problem*, International Journal of Neural Networks, Vol. 3, No. 2, June 1992.

[14] W.L. Miranker, *Numerical Methods for Stiff Equations*, Dordrecht: D. Reidel, 1981.

[15] E. Mjolsness, C. Garret and W.L. Miranker, *Multiscale Optimization in Neural Nets: Preliminary Report*, Proc. IJCNN, Vol. 3, pp. 781-786, San-Diego, CA, June 1990.

[16] E. Mjolsness, C. Garret and W.L. Miranker, *Multiscale Optimization in Neural Nets*, IEEE Trans. on Neural Networks, Vol. 2, No. 2, March 1991.

[17] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, 1981.

[18] S. Poljak and M. Sura, *On Periodical Behaviour of Societies with Symmetric Influences*, Combinatorica, 3(1), pp. 119-121, 1983.

[19] G. A. Tagliarini and E. W. Page, *Learning in Systematically Designed Networks*, Proc. IJCNN, Vol. 1, pp. 497-502, Washington D.C., June 1989.

[20] V. Zissimopoulos, V. Paschos and F. Pekergin, *On the Approximation of NP-complete Problems by Using Boltzmann Machine Method: The Cases of Some Covering and Packing Problems*, IEEE Trans. on Computers, Vol. 40, No. 12, pp. 1413-1419, Dec. 1991.