View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by IntechOpen

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

11

Recurrent Neural Approach for Solving Several Types of Optimization Problems

Ivan N. da Silva, Wagner C. Amaral, Lucia V. Arruda
and Rogerio A. Flauzino
University of São Paulo, USP/EESC/SEL, CP 359, São Carlos, SP
Brazil

1. Introduction

An artificial neural network, more commonly known as neural network, is a mathematical model for information processing based on the biological nervous system, which has a natural propensity for storing experiential knowledge and making it available for use (Haykin, 1999). The main advantage of a neural network is in its ability to approximate functional relationships, particularly nonlinear relationships.

Neural networks have been applied to several classes of optimization problems and have shown promise for solving such problems efficiently. Most of the neural architectures proposed in the literature solve specific types of optimization problems (Dillon & O'Malley, 2002; Kakeya & Okabe, 2000; Xia et al., 2002). In contrast to these neural models, the network proposed here is able to treat several kinds of optimization problems using a unique network architecture.

The approach described in this chapter uses a modified Hopfield network, which has equilibrium points representing the solution of the optimization problems. The Hopfield network is modified by presenting an optimization process carried out in two distinct stages, which are represented by two energy functions. The internal parameters of the network have been computed using the valid-subspace technique (Aiyer et al., 1990; Silva et al., 1997). This technique allows us to define a subspace, which contains only those solutions that represent feasible solutions to the problem analyzed. It has also been demonstrated that with appropriately set parameters, the network confines its output to this subspace, thus ensuring convergence to a valid solution. Also in contrast to other neural approaches that use an energy function for each constraint to be satisfied, the mapping of optimization problems using the modified Hopfield network always consists of determining just two energy functions, which are denoted by E^{conf} and E^{op}. The function E^{conf} is a confinement term that groups all structural constraints associated with the problems, and E^{op} is an optimization term that leads the network output to the equilibrium points corresponding to optimal solutions.

In this chapter, the proposed approach has been applied to solve combinatorial optimization problems, dynamic programming problems and nonlinear optimization problems. In addition to providing a new approach for solving several classes of optimization problems through a unique neural network architecture, the main advantages of using the modified

Open Access Database www.i-techonline.com

Source: Recurrent Neural Networks, Book edited by: Xiaolin Hu and P. Balasubramaniam, ISBN 978-953-7619-08-4, pp. 400, September 2008, I-Tech, Vienna, Austria

Hopfield network proposed in this chapter are the following: i) the internal parameters of the network are explicitly obtained by the valid-subspace technique of solutions, which avoids the need to use training algorithm for their adjustments; ii) the application of the valid-subspace technique allows feasible solutions to be found, which are derived from the confinement of all structural constraints by E^{conf}; iii) The optimization and confinement terms are not weighted by penalty parameters, which could affect both precision of the equilibrium points and their respective convergence processes; iv) for all classes of optimization problems, the same methodology is adopted to derive the internal parameters of the network, and v) for industrial application, the modified Hofpfield network offers simplicity of implementation both in analogue hardware, making use of operational amplifiers and in digital hardware by using digital signal processors.
The organization of the present chapter is as follows. In Section 2, the modified Hopfield network is presented, and the valid-subspace technique used to design the network parameters is described. In Section 3, the mapping of optimization problems using the modified Hopfield network is formulated. In Section 4, simulation results are given to demonstrate the performance of the developed approach. In Section 5, the key issues raised in the chapter are summarized and conclusions drawn.

2. The modified Hopfield network

Hopfield networks are single-layer networks with feedback connections between nodes. In the standard case, the nodes are fully connected, i.e., every node is connected to all others nodes, including itself (Hopfield, 1984). The node equation for the continuous-time network with N neurons is given by:

$$\dot{u}_i(t) = -\eta.u_i(t) + \sum_{j=1}^{N} T_{ij}.v_j(t) + i_i^b \tag{1}$$

$$v_i(t) = g(u_i(t)) \tag{2}$$

where $u_i(t)$ is the current state of the i-th neuron, $v_i(t)$ is the output of the i-th neuron, i_i^b is the offset bias of the i-th neuron, $\eta.u_i(t)$ is a passive decay term, T_{ij} is the weight connecting the j-th neuron to i-th neuron.
In Equation (2), $g(u_i(t))$ is a monotonically increasing threshold function that limits the output of each neuron to ensure that the network output always lies in or within a hypercube. It is shown in Hopfield (1984) that if $\boldsymbol{T}$ is symmetric and η=0, the equilibrium points of the network correspond to values $\boldsymbol{v}(t)$ for which the energy function (3) associated with the network is minimized:

$$E(t) = -\frac{1}{2}\boldsymbol{v}(t)^T.\boldsymbol{T}.\boldsymbol{v}(t) - \boldsymbol{v}(t)^T.\boldsymbol{i}^b \tag{3}$$

Therefore, the mapping of optimization problems using the Hopfield network consists of determining the weight matrix $\boldsymbol{T}$ and the bias vector $\boldsymbol{i}^b$ to compute equilibrium points to represent the problem to be solved.
One of the major difficulties in mapping optimization problems onto a conventional Hopfield network involves deciding how constraints can be included. Basically, most of

these neural networks proposed in the literature for solving optimization problems code the constraints as terms in the energy function that are weighted by penalty parameters. The stable equilibrium points of these networks, which represent a solution of the optimization problem, gave the correct solution only when those parameters are properly adjusted, and both the accuracy and the convergence process can be affected. This weakness of penalty and barrier function methods has, of course, been well known since 1968 when it was discussed by Fiacco and McCormick in Fiacco & McCormick (1968). They investigated the numerical problem associated with the change of parameters in these functions. In such approaches, the energy function given in (3) is represented by:

$$E(t) = E^{op}(t) + c_1 \cdot E_1^{const}(t) + c_2 \cdot E_2^{const}(t) + \cdots + c_m \cdot E_m^{const}(t) \tag{4}$$

where c_i are positive constants that are weighing each one of the constraints E_i^{const}. Thus, the network is involved with the minimization of a single energy function (E^{op}) correspondent to the objective function of the problem and subject to the several constraints E_i^{const}. If any of these constraints is violated then the solution is not feasible, i.e., the multiple constraints terms E_i^{const} tend to cancel each other out. Moreover, the convergence processes of these networks depend on the correct adjustment of the penalty constants associated with the energy terms.

In this chapter, we have developed a modified Hopfield network that does not depend on penalty or weighting parameters, which overcomes shortcomings associated with the other neural approaches. In contrast to most of the other neural models, the network proposed here is able to treat several kinds of optimization problems using a unique network architecture. A modified energy function $E^m(t)$, composed just by two energy terms is used here, which is defined as follows:

$$E^m(t) = E^{op}(t) + E^{conf}(t) \tag{5}$$

where $E^{conf}(t)$ is a confinement term that groups the structural constraints associated with the respective optimization problem, and $E^{op}(t)$ is an optimization term that conducts the network output to the equilibrium points corresponding to a cost constraint. Thus, the minimization of $E^m(t)$ of the modified Hopfield network is conducted in two stages:

i) minimization of the term $E^{conf}(t)$:

$$E^{conf}(t) = -\frac{1}{2}\boldsymbol{v}(t)^T . \boldsymbol{T}^{conf} . \boldsymbol{v}(t) - \boldsymbol{v}(t)^T . \boldsymbol{i}^{conf} \tag{6}$$

where $\boldsymbol{v}(t)$ is the network output, $\boldsymbol{T}^{conf}$ is a weight matrix and $\boldsymbol{i}^{conf}$ is a bias vector belonging to E^{conf}. This results in a solution $\boldsymbol{v}(t)$ in the subspace generated from the structural constraints imposed by the problem. This subspace has been derived from analysis of the Hopfield network dynamics, where it is shown in Hopfield (1984) that the energy functions E_i^{const}(t) given in (4), which are defined by (3), are Lyapunov functions provided matrices $\boldsymbol{T}$ are symmetric. An investigation associating the equilibrium points of those Lyapunov functions with respect to the eigenvalues and eigenvectors of the matrices $\boldsymbol{T}$ shows that all feasible solutions can be grouped in a unique subspace of solutions with equation $\boldsymbol{v}(t+1) = \boldsymbol{T}^{conf}.\boldsymbol{v}(t) + \boldsymbol{i}^{conf}$, where $\boldsymbol{T}^{conf}$ is a projection matrix and $\boldsymbol{i}^{conf}$ is a vector orthogonal to $\boldsymbol{T}^{conf}$. By analyzing the convergence process dynamics, it is revealed that $\boldsymbol{v}$ evolves first along those

eigenvectors of $\boldsymbol{T}^{conf}$ with the large eigenvalues, then along those with negative eigenvalues. As consequence of the application of this subspace approach, which is named the valid-subspace method, a unique energy term can be used to represent all constraints associated with the optimization problem since $\boldsymbol{T}^{conf}$ to be a projection matrix ($\boldsymbol{T}^{conf}.\boldsymbol{T}^{conf} = \boldsymbol{T}^{conf}$) and $\boldsymbol{i}^{conf}$ a vector orthogonal to $\boldsymbol{T}^{conf}$, i. e., $\boldsymbol{T}^{conf}.\boldsymbol{i}^{conf} = \boldsymbol{0}$. A more detailed analysis of the valid-subspace method can be found in Silva et al. (1997).

ii) minimization of the term $E^{op}(t)$:

$$E^{op}(t) = -\frac{1}{2}\boldsymbol{v}(t)^{T}.\boldsymbol{T}^{op}.\boldsymbol{v}(t) - \boldsymbol{v}(t)^{T}.\boldsymbol{i}^{op} \tag{7}$$

where $\boldsymbol{T}^{op}$ is weight matrix and iop is bias vector belonging to E^{op}. This corresponds to move $\boldsymbol{v}(t)$ towards an optimal solution (the equilibrium points). Thus, the operation of the modified Hopfield network consists of three main steps, as shown in Fig. 1:

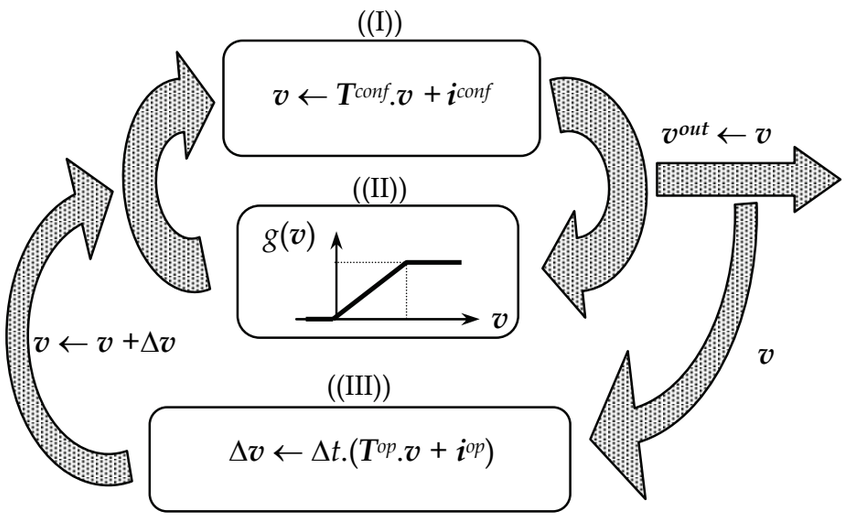


Fig. 1. The modified Hopfield network.

Step ((I)): Minimization of E^{conf}, corresponding to the projection of $\boldsymbol{v}(t)$ in the valid subspace defined by:

$$\boldsymbol{v}(t+1) = \boldsymbol{T}^{conf} \cdot \boldsymbol{v}(t) + \boldsymbol{i}^{conf} \Rightarrow \boldsymbol{v} \leftarrow \boldsymbol{T}^{conf} \cdot \boldsymbol{v} + \boldsymbol{i}^{conf} \tag{8}$$

where: $\boldsymbol{T}^{conf}$ is a projection matrix ($\boldsymbol{T}^{conf}.\boldsymbol{T}^{conf} = \boldsymbol{T}^{conf}$) and the vector $\boldsymbol{i}^{conf}$ is orthogonal to the subspace ($\boldsymbol{T}^{conf}.\boldsymbol{i}^{conf} = \boldsymbol{0}$). This operation corresponds to an indirect minimization of $E^{conf}(t)$. An analysis of the valid-subspace technique is presented in Aiyer et al. (1990) and Silva et al. (1997).

Step ((II)): Application of a nonlinear 'symmetric ramp' activation function constraining $\boldsymbol{v}(t)$ in a hypercube:

$$g_i(v_i) = \begin{cases} lim_i^{\inf} & \text{, if} \quad v_i < lim_i^{\inf} \\ v_i & \text{, if} \quad lim_i^{\inf} \le v_i \le lim_i^{\sup} \\ lim_i^{\sup} & \text{, if} \quad v_i > lim_i^{\sup} \end{cases} \tag{9}$$

where $v_i(t) \in [lim_i^{\inf}, lim_i^{\sup}]$. For combinatorial optimization and dynamic programming problems $g_i(v_i) \in [0, 1]$ and in this case $lim_i^{\inf} = 0$ and $lim_i^{\sup} = 1$. Although v is inside a set with particular structure, the modified
Hopfield network can represent a general problem. For example, if $\boldsymbol{v} \in \Re^n$ for nonlinear optimization problem, then $lim_i^{\inf} = -\infty$ and $lim_i^{\sup} = \infty$.

Step ((III)): Minimization of E^{op}, which involves updating of $\boldsymbol{v}(t)$ in direction to an optimal solution (defined by $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$) corresponding to network equilibrium points, which are the solutions for the optimization problem considered in a specific application. Using the 'symmetric ramp' activation function defined in (7) and given η=0, equation (2) subsequently becomes:

$$\boldsymbol{v}(t) = g(\boldsymbol{u}(t)) = \boldsymbol{u}(t) \tag{10}$$

By comparison with (1) and (6), we have:

$$\frac{d\boldsymbol{v}(t)}{dt} = \dot{\boldsymbol{v}} = -\frac{\partial E^{op}(t)}{\partial \boldsymbol{v}}$$

$$\Delta \boldsymbol{v} = -\Delta t.\nabla E^{op}(\boldsymbol{v}) = \Delta t.(\boldsymbol{T}^{op}.\boldsymbol{v} + \boldsymbol{i}^{op}) \tag{11}$$

Therefore, minimization of E^{op} consists of updating $\boldsymbol{v}(t)$ in the opposite direction to the gradient of E^{op}. These results are also valid when a 'hyperbolic tangent' activation function is used. In this step, the process used by the modified Hopfield network for solving the corresponding differential equations are identical to Euler's method and in optimization terms it represents a steepest descent algorithm with a fixed step size.
After each optimization step in ((III)), it is necessary to carry out several times the two steps involved with the confinement of constraints in order to ensure the feasibility of the problem is achieved, i.e., the steps ((I)) and ((II)) are continuously applied until the convergence of the output vector $\boldsymbol{v}$. In optimization terminology this method is therefore a gradient restoration algorithm with a fixed step size.
Therefore, according to Fig. 1 each iteration has two distinct stages. First, as described in Step ((III)), $\boldsymbol{v}$ is updated using the gradient of the term E^{op} alone. Second, after each updating, $\boldsymbol{v}$ is projected in the valid subspace. This is an iterative process, in which $\boldsymbol{v}$ is first orthogonally projected in the valid subspace (8), and then thresholded so that its elements lie in the range $[lim_i^{\inf}, lim_i^{\sup}]$. The convergence process is concluded when the values of vout during two successive loops remain practically constant, where the value of vout in this case is equal to $\boldsymbol{v}$.

3. Mapping optimization problems by the modified Hopfield network

In this section, the formulation of three types of optimization problems, namely combinatorial optimization problems, dynamic programming problems and nonlinear optimization problems, is presented.

3.1 Notation and definitions

The notation employed for vectors and matrices, which are used for mapping combinatorial optimization problems and dynamic programming problems, is as follows.

- The vector $\boldsymbol{p} \in \Re^n$ represents the solution set of an optimization problem consisted of n nodes (neurons). Thus, the elements belonging to $\boldsymbol{p}$ have integer elements defined by:

$$p_i \in \{1,...,n\} \text{ where } i \in \{1..n\} \tag{12}$$

The vector $\boldsymbol{p}$ can be represented by a vector $\boldsymbol{v}$, composed of ones and zeros, which represents the output of the network. In the notation using Kronecher products (Graham, 1981), we have:

- $\boldsymbol{\delta}$ is a matrix ($\boldsymbol{\delta} \in \Re^{nxn}$) defined by:

$$\delta_{ij} = \begin{cases} 1, \text{ if } i = j \\ 0, \text{ if } i \neq j \end{cases} \tag{13}$$

$\boldsymbol{\delta}(k) \in \Re^n$ is a column vector corresponding to k-th column of $\boldsymbol{\delta}$.

- $\boldsymbol{v}(\boldsymbol{p})$ is an $n.m$ dimensional vector representing the form of the final network output vector $\boldsymbol{v}$, which corresponds to the problem solution denoted by $\boldsymbol{p}$. The vector $\boldsymbol{v}(\boldsymbol{p})$ is defined by:

$$\boldsymbol{v}(\boldsymbol{p}) = \begin{bmatrix} \delta(p_1) \\ \delta(p_2) \\ \vdots \\ \delta(p_n) \end{bmatrix} \tag{14}$$

- vec($\boldsymbol{U}$) is a function which maps the mxn matrix $\boldsymbol{U}$ to the $n.m$-element vector $\boldsymbol{v}$. This function is defined by:

$$\boldsymbol{v} = \text{vec}(\boldsymbol{U}) = [U_{11}\, U_{21}...U_{m1} \quad U_{12}\, U_{22}...U_{m2} \quad U_{1n}\, U_{2n}\, ...U_{mn}]^T \tag{15}$$

- $\boldsymbol{V}(\boldsymbol{p})$ is an nxn dimensional matrix defined by:

$$\boldsymbol{V}(\boldsymbol{p}) = \begin{bmatrix} \delta(p_1)^T \\ \delta(p_2)^T \\ \vdots \\ \delta(p_n)^T \end{bmatrix} \tag{16}$$

where $[\boldsymbol{V}(\boldsymbol{p})]_{ij} = [\boldsymbol{\delta}(p_i)]_j$.

- $\boldsymbol{P}\otimes\boldsymbol{Q}$ denotes the Kronecher product of two matrices. If $\boldsymbol{P}$ is an nxn matrix, and $\boldsymbol{Q}$ is an mxm matrix, then ($\boldsymbol{P}\otimes\boldsymbol{Q}$) is an ($n.m$)x($n.m$) matrix given by:

$$\boldsymbol{P} \otimes \boldsymbol{Q} = \begin{bmatrix} P_{11}\boldsymbol{Q} & P_{12}\boldsymbol{Q} & \cdots & P_{1n}\boldsymbol{Q} \\ P_{21}\boldsymbol{Q} & P_{22}\boldsymbol{Q} & \cdots & P_{2n}\boldsymbol{Q} \\ \vdots & \vdots & \ddots & \\ P_{n1}\boldsymbol{Q} & P_{n2}\boldsymbol{Q} & \cdots & P_{nn}\boldsymbol{Q} \end{bmatrix} \tag{17}$$

- $\boldsymbol{w}\otimes\boldsymbol{h}$ denotes the Kronecher product of two vectors. If $\boldsymbol{w}$ is an n-element vector and $\boldsymbol{h}$ an m-element vector, then ($\boldsymbol{w}\otimes\boldsymbol{h}$) is an $n.m$-element vector given by:

$$\boldsymbol{w} \otimes \boldsymbol{h} = \begin{bmatrix} w_1.\boldsymbol{h} \\ w_2.\boldsymbol{h} \\ \vdots \\ w_n.\boldsymbol{h} \end{bmatrix} \tag{18}$$

The properties of the Kronecher products (Graham, 1981) utilized are:

$$(\lambda \boldsymbol{w} \otimes \gamma \boldsymbol{h}) = \lambda\gamma(\boldsymbol{w} \otimes \boldsymbol{h}) \tag{19}$$

$$(\boldsymbol{w} \otimes \boldsymbol{h})^T(\boldsymbol{x} \otimes \boldsymbol{g}) = (\boldsymbol{w}^T\boldsymbol{x})(\boldsymbol{h}^T\boldsymbol{g}) \tag{20}$$

$$(\boldsymbol{P} \otimes \boldsymbol{Q})(\boldsymbol{w} \otimes \boldsymbol{h}) = (\boldsymbol{P}\boldsymbol{w} \otimes \boldsymbol{Q}\boldsymbol{h}) \tag{21}$$

$$(\boldsymbol{P} \otimes \boldsymbol{Q})(\boldsymbol{E} \otimes \boldsymbol{F}) = (\boldsymbol{P}\boldsymbol{E} \otimes \boldsymbol{Q}\boldsymbol{F}) \tag{22}$$

$$\text{vec}(\boldsymbol{Q}.\boldsymbol{V}.\boldsymbol{P}^{\text{T}}) = (\boldsymbol{P} \otimes \boldsymbol{Q}).\text{vec}(\boldsymbol{V}) \tag{23}$$

- $\boldsymbol{o}^n$ and $\boldsymbol{O}^n$ are respectively the n-element vector and the nxn matrix of ones, that is:

$$\left.\begin{matrix} [\boldsymbol{o}]_i = 1 \\ [\boldsymbol{O}]_{ij} = 1 \end{matrix}\right\} \text{for } i, j \in \{1..n\} \tag{24}$$

- $\boldsymbol{R}^n$ is an nxn projection matrix (i.e., $\boldsymbol{R}^n.\boldsymbol{R}^n = \boldsymbol{R}^n$) defined by:

$$\boldsymbol{R}^n = \boldsymbol{I}^n - \frac{1}{n}\boldsymbol{O}^n \tag{25}$$

The sum of the elements of each row of a matrix is transformed to zero by post-multiplication with $\boldsymbol{R}^n$, while pre-multiplication by $\boldsymbol{R}^n$ has the effect of setting the sum of the elements of each column to zero.

3.2 Formulation of combinatorial optimization problems

The combinatorial optimization problem considered in this chapter is the matching problem in bipartite graphs. However, several other types of combinatorial optimization problems, such as the salesman and N-queens problems, can be also solved by the proposed neural approach.

A graph G is a pair $G = (V,E)$, where V is a finite set of $2n$ nodes or vertices and E has as elements subsets of V of cardinality two called edges (Papadimitriou & Steiglitz, 1982). A matching M of a graph $G = (V,E)$ is a subset of the edges with the property that no two edges of M share the same node. The graph $G = (V,E)$ is called bipartite if the set of vertices V can be partitioned into two sets of n nodes, U and W, and each edge in E has one vertex in U and one vertex in W.

For each edge $[u_i, w_j] \in E$ is given a number $P_{ij} \geq 0$ called the connection weight of $[u_i, w_j]$. The goal of the matching problem in bipartite graphs is to find a matching of G with the minimum total sum of weights. Several problems, such as pattern recognition in computational vision, processes involving signal transmission, design of thin film circuits and schedule of operation processes, can be modeled as a matching problem in bipartite graphs.

As an example, for a bipartite graph with four nodes ($2n = 4$) represented in Fig. 2, the sets V, E, U, W and the matrix $\boldsymbol{P}$ are given by:

$$V = \{u_1, u_2, w_1, w_2\} \tag{26}$$

$$E = \{[u_1,w_1], [u_1,w_2], [u_2,w_1], [u_2,w_2]\} \tag{27}$$

$$U = \{u_1, u_2\} \tag{28}$$

$$W = \{w_1, w_2\} \tag{29}$$

$$\boldsymbol{P} = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} = \begin{bmatrix} 2.9 & 0.8 \\ 1.3 & 3.5 \end{bmatrix} \tag{30}$$

In this case, the minimum bipartite graph represented by the matching M will be chosen either by the subset $M_1 = \{[u_1,w_1], [u_2,w_2]\}$ or $M_2 = \{[u_1,w_2], [u_2,w_1]\}$. As the sum of the edges of M_2 is lower than that of M_1, then the subset M_2 corresponds to minimum bipartite graph, i. e., $M = M_2$.

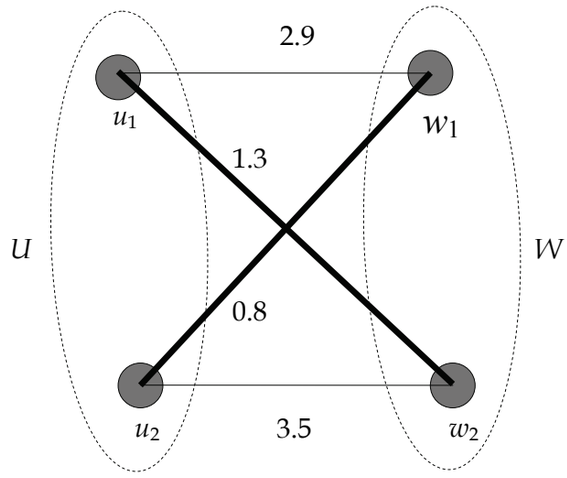


Fig. 2. Bipartite graph composed by four nodes.

In order to represent the association between nodes of U and W belonging to matching M, we have used the vector $\boldsymbol{p} \in \Re^n$, where the element $p_i \in \{1,\ldots,n\}$ represents the edge linking the i-th node of U to respective node of W, which is given by the own value of p_i. Using the definitions presented in subsection 3.1, for the matching problem illustrated in Fig. 2 the values of $\boldsymbol{p}$, $\boldsymbol{v}(\boldsymbol{p})$ and $\boldsymbol{V}(\boldsymbol{p})$ representing the solution given by M are defined by:

$$\boldsymbol{p} = [2 \quad 1]^T \tag{31}$$

$$\boldsymbol{v}(\boldsymbol{p}) = [\underbrace{0 \quad 1}_{\delta(p_1)^T} \quad \underbrace{1 \quad 0}_{\delta(p_2)^T}]^T \tag{32}$$

$$\boldsymbol{V}(\boldsymbol{p}) = \begin{bmatrix} \delta(p_1)^T \\ \delta(p_2)^T \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{33}$$

The equations of $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ are developed to force the validity of the structural constraints. These constraints mean that each edge in E has just one activated node in U and one activated node in W. Using the matrix $\boldsymbol{V}(\boldsymbol{p})$ to represent the structural constraints, we have:

$$[\boldsymbol{V}(\boldsymbol{p})]_{ij} \in \{1,0\}$$

$$\sum_{j=1}^{n}[\boldsymbol{V}(\boldsymbol{p})]_{ij} = 1 \tag{34}$$

In this case, a valid subspace for the matching problem in bipartite graphs can be represented by the following relationship:

$$\boldsymbol{I}^{conf} = \boldsymbol{V} = \frac{1}{n}\boldsymbol{o}^{n}.\boldsymbol{o}^{n^{T}} \tag{35}$$

It is now necessary to guarantee that the sum of the elements of each line of the matrix $\boldsymbol{V}$ takes value equal to 1. This procedure is represented in the modified Hopfield network by the projection matrix $\boldsymbol{T}^{conf}$, i.e., the multiplication of $\boldsymbol{T}^{conf}$ by $\boldsymbol{V}$ should also guarantee these constraints. Using the properties of the matrix $\boldsymbol{R}^{n}$, we have:

$$\boldsymbol{V}.\boldsymbol{R}^{n} = \boldsymbol{T}^{conf}.\boldsymbol{V} \tag{36}$$

$$\mathbf{I}^{n}.\mathbf{V}.\mathbf{R}^{n} = \mathbf{T}^{conf}.\mathbf{V} \tag{37}$$

Using (35) and (37) in equation of the valid subspace ($\boldsymbol{V} = \boldsymbol{T}^{conf}.\boldsymbol{V} + \boldsymbol{I}^{conf}$),

$$\boldsymbol{V} = \boldsymbol{I}^{n}.\boldsymbol{V}.\boldsymbol{R}^{n} + \frac{1}{n}\boldsymbol{o}^{n}.\boldsymbol{o}^{n^{T}} \tag{38}$$

Applying operator vec(.) given by (23) in (38),

$$\text{vec}(\boldsymbol{V}) = \text{vec}(\boldsymbol{I}^{n}.\boldsymbol{V}.\boldsymbol{R}^{n}) + \frac{1}{n}\text{vec}(\boldsymbol{o}^{n}.1.\boldsymbol{o}^{n^{T}})$$

$$\text{vec}(\boldsymbol{V}) = (\boldsymbol{I}^{n} \otimes \boldsymbol{R}^{n}).\text{vec}(\boldsymbol{V}) + \frac{1}{n}(\boldsymbol{o}^{n} \otimes \boldsymbol{o}^{n}) \tag{39}$$

Changing vec($\boldsymbol{V}$) by $\boldsymbol{v}$ in equation (39), we have:

$$\boldsymbol{v}(t+1) = (\boldsymbol{I}^{n} \otimes \boldsymbol{R}^{n}).\boldsymbol{v}(t) + \frac{1}{n}(\boldsymbol{o}^{n} \otimes \boldsymbol{o}^{n}) \tag{40}$$

Thus, comparing (40) and (8) the parameters $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ are given by:

$$\boldsymbol{T}^{conf} = (\boldsymbol{I}^{n} \otimes \boldsymbol{R}^{n}) \tag{41}$$

$$\boldsymbol{i}^{conf} = \frac{1}{n}(\boldsymbol{o}^{n} \otimes \boldsymbol{o}^{n}) \tag{42}$$

Equations (41) and (42) satisfy the properties of the valid subspace, i.e., $\boldsymbol{T}^{conf}.\boldsymbol{T}^{conf} = \boldsymbol{T}^{conf}$ and $\boldsymbol{T}^{conf}.\boldsymbol{i}^{conf} = \boldsymbol{0}$. In relation to example illustrated in Fig. 2 the matrix $\boldsymbol{T}^{conf}$ and the vector $\boldsymbol{i}^{conf}$ are respectively given by:

$$\boldsymbol{T}^{conf} = \begin{bmatrix} 0.5 & -0.5 & 0 & 0 \\ -0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & -0.5 \\ 0 & 0 & -0.5 & 0.5 \end{bmatrix} \tag{43}$$

$$\boldsymbol{i}^{conf} = [0.5 \quad 0.5 \quad 0.5 \quad 0.5]^T \tag{44}$$

The energy function E^{op} of the modified Hopfield network for the matching problem in bipartite graphs is projected in order to find a solution corresponding to the minimum total sum $\xi(\boldsymbol{p})$ referent to the values P_{ij} associated with each edges of M, which is defined by:

$$E^{op} = \xi(\boldsymbol{p}) = \text{trace}(\boldsymbol{V}(\boldsymbol{p})^T.\boldsymbol{P}) \tag{45}$$

In this case, when E^{op} is minimized, the optimal solution corresponds to the minimum energy state of the network. The parameters $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$ are then obtained from the corresponding cost constraint given by above equation. Using the properties of Kronecher product in (45), we have:

$$E^{op} = \text{vec}(\boldsymbol{V}(\boldsymbol{p})^T).\text{vec}(\boldsymbol{P}) = \boldsymbol{v}(\boldsymbol{p})^T.\text{vec}(\boldsymbol{P}) \tag{46}$$

Comparing (46) and (7), the parameters $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$ are given by:

$$\boldsymbol{T}^{op} = \boldsymbol{0} \tag{47}$$

$$\boldsymbol{i}^{op} = -\text{vec}(\boldsymbol{P}) \tag{48}$$

Using the definition of vec(.) provided in (15), the vector $\boldsymbol{i}^{op}$ in relation to example illustrated in Fig. 2 is given by:

$$\boldsymbol{i}^{conf} = [-2.9 \quad -1.3 \quad -0.8 \quad -3.5]^T \tag{49}$$

To illustrate the performance of the proposed neural network, some simulation results involving the matching problem in bipartite graphs are presented in Section 4.

3.3 Formulation of dynamic programming problems

A typical dynamic programming problem can be modeled as a set of source and destination nodes with n intermediate stages, m states in each stage, and metric data $d_{xi,(x+1)j}$, where x is the index of the stages, and i and j are the indices of the states in each stage (Hillier & Lieberman, 1980). The goal of the dynamic programming problem considered in this chapter is to find a valid path which starts at the source node, visits one and only one state node in each stage, reaches the destination node, and has a minimum total length (cost) among all possible paths.

The equations of $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ are developed to force the validity of the structural constraints. These constraints, for dynamic programming problems, mean that one and only one state in each stage can be actived. Thus, the matrix $\boldsymbol{V}(\boldsymbol{p})$ is defined by:

$$[\boldsymbol{V}(\boldsymbol{p})]_{ij} \in \{1,0\}$$

$$\sum_{j=1}^{m}[\boldsymbol{V}(\boldsymbol{p})]_{ij} = 1 \tag{50}$$

A valid subspace ($\boldsymbol{V}=\boldsymbol{T}^{val}.\boldsymbol{V} + \boldsymbol{I}^{conf}$) for the dynamic programming problem can be represented by:

$$\boldsymbol{I}^{conf} = \boldsymbol{V} = \frac{1}{m}\boldsymbol{o}^{n}.\boldsymbol{o}^{m^{T}} \tag{51}$$

Equation (51) guarantees that the sum of the elements of each line of the matrix $\boldsymbol{V}$ takes values equal to 1. Therefore, the term $\boldsymbol{T}^{conf}.\boldsymbol{V}$ must also guarantee that the sum of the elements of each line of the matrix $\boldsymbol{V}$ takes value equal to zero. Using the properties of the matrix $\boldsymbol{R}^{n}$, we have:

$$\boldsymbol{V}.\boldsymbol{R}^{m} = \boldsymbol{T}^{conf}.\boldsymbol{V}$$

$$\boldsymbol{I}^{n}.\boldsymbol{V}.\boldsymbol{R}^{m} = \boldsymbol{T}^{conf}.\boldsymbol{V} \tag{52}$$

Using (51) and (52) in equation of the valid subspace ($\boldsymbol{V} = \boldsymbol{T}^{conf}.\boldsymbol{V} + \boldsymbol{I}^{conf}$),

$$\boldsymbol{V} = \boldsymbol{I}^{n}.\boldsymbol{V}.\boldsymbol{R}^{m} + \frac{1}{m}\boldsymbol{o}^{n}.\boldsymbol{o}^{m^{T}} \tag{53}$$

Applying operator vec(.) given by (23) in (53),

$$\text{vec}(\boldsymbol{V}) = \text{vec}(\boldsymbol{I}^{n}.\boldsymbol{V}.\boldsymbol{R}^{m}) + \frac{1}{m}\text{vec}(\boldsymbol{o}^{n}.1.\boldsymbol{o}^{m^{T}})$$

$$\text{vec}(\boldsymbol{V}) = (\boldsymbol{I}^{n} \otimes \boldsymbol{R}^{m}).\text{vec}(\boldsymbol{V}) + \frac{1}{m}(\boldsymbol{o}^{n} \otimes \boldsymbol{o}^{m}) \tag{54}$$

Changing vec($\boldsymbol{V}$) by $\boldsymbol{v}$ in equation (54), we have:

$$\boldsymbol{v}(t+1) = (\boldsymbol{I}^{n} \otimes \boldsymbol{R}^{m}).\boldsymbol{v}(t) + \frac{1}{m}(\boldsymbol{o}^{n} \otimes \boldsymbol{o}^{m}) \tag{55}$$

Thus, comparing (55) and (8) the parameters $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ are given by:

$$\boldsymbol{T}^{conf} = (\boldsymbol{I}^{n} \otimes \boldsymbol{R}^{m}) \tag{56}$$

$$\boldsymbol{I}^{conf} = \frac{1}{m}(\boldsymbol{o}^{n} \otimes \boldsymbol{o}^{m}) \tag{57}$$

Equations (56) and (57) satisfy the properties of the valid subspace, i.e., $\boldsymbol{T}^{conf}.\boldsymbol{T}^{conf} = \boldsymbol{T}^{conf}$ and $\boldsymbol{T}^{conf}.\boldsymbol{i}^{conf} = \boldsymbol{0}$.

The energy function E^{op} of the modified Hopfield network for the dynamic programming problem, which is defined in (58), is projected to find a minimum path among all possible paths. In this equation, the first term defines the weight (metric cost) of the connection

linking the i-th neuron of stage x to the j-th neuron of the following stage (x+1). The second term defines the weight of the connection linking the i-th neuron of stage x to the j-th neuron of previous stage (x–1). The third term provides the weight of the connection linking the source node to all others nodes of the first stage, while the fourth term provides the weight of the connection linking the destination to all other nodes of the last stage. When E^{op} is minimized, the optimal solution corresponds to the minimum energy state of the network.

$$E^{op} = \frac{1}{4}[\sum_{x=1}^{n-1}\sum_{i=1}^{m}\sum_{j=1}^{m}\underbrace{d_{xi,(x+1)j}.v_{xi}.v_{(x+1)j}}_{1st.\,\text{Term}} + \sum_{x=2}^{n}\sum_{i=1}^{m}\sum_{j=1}^{m}\underbrace{d_{(x-1)j,xi}.v_{xi}.v_{(x-1)j}}_{2nd.\,\text{Term}}] + \\ + [\sum_{x=1}^{1}\sum_{i=1}^{m}\underbrace{d_{source,xi}.v_{xi}}_{3rd.\,\text{Term}} + \sum_{x=n}^{n}\sum_{i=1}^{m}\underbrace{d_{xi,destination}.v_{xi}}_{4th.\,\text{Term}}] \tag{58}$$

Therefore, optimization of E^{op} corresponds to minimizing each term given by (58) in relation to v_{xi}. From (58), the matrix $\boldsymbol{T}^{op}$ and vector $\boldsymbol{i}^{op}$ can be given by:

$$[\boldsymbol{T}^{op}]_{pq} = -[\boldsymbol{P}]_{xi,yj}.[\boldsymbol{Q}]_{xy} \quad \begin{cases}[\boldsymbol{P}]_{xi,yj} = \frac{1}{2}d_{xi,yj} \\ [\boldsymbol{Q}]_{xy} = \delta_{(x+1)y} + \delta_{(x-1)y}\end{cases} \tag{59}$$

$$\boldsymbol{i}^{op} = -[\underbrace{d_{source,11}\ \ \mathrm{d}_{\text{source,12}}\ \cdots \mathrm{d}_{\text{source,1m}}}_{m}\ \underbrace{0\ \ 0\ldots 0\ \ 0}_{m.(n\text{-}2)} \\ \underbrace{d_{n1,\text{destination}}\ \ \mathrm{d}_{n2,\text{destination}}\ \cdots d_{nm,destination}}_{m}] \tag{60}$$

where: $\boldsymbol{T}^{op} \in \Re^{nmxnm}$; $\boldsymbol{i}^{op} \in \Re^{n.m}$; $p = m.(x - 1) + i$; $q = m.(y - 1) + j$; $x, y \in \{2..n - 1\}$; $i, j \in \{1..m\}$.
In the next subsection, the formulation of nonlinear optimization problems by the modified Hopfield network is presented.

3.4 Formulation of nonlinear optimization problems

Consider the following general nonlinear optimization problem, with m-constraints and n-variables, given by the following equations:

$$\text{Minimize: } E^{op}(\boldsymbol{v}) = f(\boldsymbol{v}) \tag{61}$$

$$\text{subject to: } E^{conf}(\boldsymbol{v}): h_i(\boldsymbol{v}) \le \boldsymbol{0}\ , \quad i \in \{1..m\} \tag{62}$$

$$\boldsymbol{z}^{min} \le \boldsymbol{v} \le \boldsymbol{z}^{max} \tag{63}$$

where $\boldsymbol{v}$, $\boldsymbol{z}^{min}$, $\boldsymbol{z}^{max} \in \Re^n$; $f(\boldsymbol{v})$ and $h_i(\boldsymbol{v})$ are continuous, and all first and second order partial derivatives of $f(\boldsymbol{v})$ and $h_i(\boldsymbol{v})$ exist and are continuous. The vectors $\boldsymbol{z}^{min}$ and $\boldsymbol{z}^{max}$ define the bounds on the variables belonging to the vector $\boldsymbol{v}$. The conditions in (62) and (63) define a bounded polyhedron. The vector $\boldsymbol{v}$ must remain within this polyhedron if it is to represent a valid solution for the optimization problem (61). A solution can be obtained by a modified Hopfield network, whose valid subspace guarantees the satisfaction of condition (62). Moreover, the initial hypercube represented by the inequality constraints in (63) is directly

defined by the 'symmetric ramp' function given in (9), which is used as neural activation function, i.e. $\boldsymbol{v} \in [\mathbf{z}^{min}, \mathbf{z}^{max}]$.
The parameters $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ are calculated by transforming the inequality constraints in (62) into equality constraints by introducing a slack variable $\boldsymbol{w} \in \Re^n$ for each inequality constraint:

$$h_i(\boldsymbol{v}) + \sum_{j=1}^{m} \delta_{ij} . w_j = 0 \tag{64}$$

where w_j are slack variables, treated as the variables v_i , and δ_{ij} is defined by the Kronecker impulse function:

$$\delta_{ij} = \begin{cases} 1 \text{ , if } i = j \\ 0 \text{ , if } i \neq j \end{cases} \tag{65}$$

After this transformation, the problem defined by equations (61), (62) and (63) can be rewritten as:

$$\text{Minimize: } E^{op}(\boldsymbol{v}^+) = f(\boldsymbol{v}^+) \tag{66}$$

$$\text{subject to: } E^{conf}(\boldsymbol{v}): h_i(\boldsymbol{v}^+) \leq 0 \text{ , } \quad i \in \{1..m\} \tag{67}$$

$$z_i^{min} \leq v_i^+ \leq z_i^{max} \text{ , } \; i \in \{1..n\} \tag{68}$$

$$0 \leq v_i^+ \leq z_i^{max} \text{ , } \; i \in \{n+1..N\} \tag{69}$$

where $N = n + m$, and $\boldsymbol{v}^+ = [\boldsymbol{v}^T \;\; \boldsymbol{w}^T]^T \in \Re^N$ is a vector of extended variables. Note that E^{op} does not depend on the slack variables $\boldsymbol{w}$. Also an equality constraint of the form $h_i(.) = 0$ is incorporated in the above optimization problem by transforming into two inequalities, i.e., $h_i(.) \leq 0$ and $h_i(.) \geq 0$.
The projection matrix $\boldsymbol{T}^{conf}$ belonging to the valid-subspace equation given in (8) is obtained from the projection of $\boldsymbol{v}^+$, which is obtained after a minimization step of $E^{op}(\boldsymbol{v}^+)$, onto the tangent subspace of the surface bounded by constraints given by (67). In Luenberger (1984), it has been shown that a projection matrix to the system defined in (67) is given by:

$$\boldsymbol{T}^{conf} = \boldsymbol{I} - \nabla \boldsymbol{h}(\boldsymbol{v}^+)^T . (\nabla \boldsymbol{h}(\boldsymbol{v}^+) . \nabla \boldsymbol{h}(\boldsymbol{v}^+)^T)^{-1} . \nabla \boldsymbol{h}(\boldsymbol{v}^+) \tag{70}$$

where:

$$\nabla \boldsymbol{h}(\boldsymbol{v}^+) = \begin{bmatrix} \frac{\partial h_1(\boldsymbol{v}^+)}{\partial v_1^+} & \frac{\partial h_1(\boldsymbol{v}^+)}{\partial v_2^+} & \cdots & \frac{\partial h_1(\boldsymbol{v}^+)}{\partial v_N^+} \\ \frac{\partial h_2(\boldsymbol{v}^+)}{\partial v_1^+} & \frac{\partial h_2(\boldsymbol{v}^+)}{\partial v_2^+} & \cdots & \frac{\partial h_2(\boldsymbol{v}^+)}{\partial v_N^+} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_m(\boldsymbol{v}^+)}{\partial v_1^+} & \frac{\partial h_m(\boldsymbol{v}^+)}{\partial v_2^+} & \cdots & \frac{\partial h_m(\boldsymbol{v}^+)}{\partial v_N^+} \end{bmatrix} = \begin{bmatrix} \nabla \boldsymbol{h}_1(\boldsymbol{v}^+)^T \\ \nabla \boldsymbol{h}_2(\boldsymbol{v}^+)^T \\ \vdots \\ \nabla \boldsymbol{h}_m(\boldsymbol{v}^+)^T \end{bmatrix} \tag{71}$$

Inserting the value of (70) in the expression of the valid subspace in (8), we have:

$$\boldsymbol{v}^{+} \leftarrow [\boldsymbol{I} - \nabla \boldsymbol{h}(\boldsymbol{v}^{+})^{T}.(\nabla \boldsymbol{h}(\boldsymbol{v}^{+}).\nabla \boldsymbol{h}(\boldsymbol{v}^{+})^{T})^{-1}.\nabla \boldsymbol{h}(\boldsymbol{v}^{+})]. \boldsymbol{v}^{+} + \boldsymbol{i}^{conf} \quad (72)$$

Results of the Lyapunov stability theory (Vidyasagar, 1993) should be used in (72) to guarantee the stability of the nonlinear system, and consequently, to force the network convergence to equilibrium points that represent a feasible solution to the nonlinear system. By the definition of the Jacobean, when $\boldsymbol{v}$ leads to equilibrium point implicates in $\boldsymbol{v}^{e} = \boldsymbol{0}$. In this case, the value of $\boldsymbol{i}^{conf}$ should also be null to satisfy the equilibrium condition, i. e., $\boldsymbol{v}^{e} = \boldsymbol{v}(t) = \boldsymbol{v}(t + 1) = \boldsymbol{0}$. Thus, $\boldsymbol{h}(\boldsymbol{v}^{+})$ given in equation (72) can be approximated as follows:

$$\boldsymbol{h}(\boldsymbol{v}^{+}) \approx \boldsymbol{h}(\boldsymbol{v}^{e}) + \boldsymbol{J}.(\boldsymbol{v}^{+} - \boldsymbol{v}^{e}) \quad (73)$$

where $\boldsymbol{J} = \nabla \boldsymbol{h}(\boldsymbol{v}^{+})$ and $\boldsymbol{h}(\boldsymbol{v}^{+}) = [h_1(\boldsymbol{v}^{+}) \;\; h_2(\boldsymbol{v}^{+}) \; ... \; h_m(\boldsymbol{v}^{+})]^{T}$.
In the proximity of the equilibrium point $\boldsymbol{v}^{e} = \boldsymbol{0}$, we obtain the following equation related to the parameters $\boldsymbol{v}^{+}$ and $\boldsymbol{h}(\boldsymbol{v}^{+})$:

$$\lim_{\boldsymbol{v}^{+} \to \boldsymbol{v}^{e}} \left\| \frac{\boldsymbol{h}(\boldsymbol{v}^{+})}{\boldsymbol{v}^{+}} \right\| = \boldsymbol{0} \quad (74)$$

Finally, introducing the results derived from (73) and (74) in equation given by (72), we obtain:

$$\boldsymbol{v}^{+} \leftarrow \boldsymbol{v}^{+} - \nabla \boldsymbol{h}(\boldsymbol{v}^{+})^{T}.(\nabla \boldsymbol{h}(\boldsymbol{v}^{+}).\nabla \boldsymbol{h}(\boldsymbol{v}^{+})^{T})^{-1}.\boldsymbol{h}(\boldsymbol{v}^{+}) \quad (75)$$

Therefore, equation (75) synthesizes the valid-subspace expression for treating systems of nonlinear equations. In this case, for nonlinear optimization problems the original valid-subspace equation given in (8), which is represented by step ((I)) in Fig. 1, should be substituted by equation (75). Thus, according to Fig. 1, successive applications of the step ((I)) followed by the step ((II)) make $\boldsymbol{v}^{+}$ convergent to a point that satisfies all constraints imposed to the nonlinear optimization problem.
The parameters $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$ associated to the energy function E^{op}, which is given by (7) and represented in (66), should be defined so that the optimal solution corresponds to the minimization of E^{op}. This procedure can be implemented by updating the vector $\boldsymbol{v}^{+}$ in the opposite gradient direction that of the energy function E^{op}. Since conditions (66)-(69) define a bounded polyhedron, the objective function (66) has always a minimum. Thus, the equilibrium points of the network can be calculated by assuming the following values to $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$:

$$\boldsymbol{i}^{op} = -\left[\frac{\partial f(\boldsymbol{v}^{+})}{\partial v_1^{+}} \;\; \frac{\partial f(\boldsymbol{v}^{+})}{\partial v_2^{+}} \cdots \frac{\partial f(\boldsymbol{v}^{+})}{\partial v_N^{+}} \right]^{T} \quad (76)$$

$$\boldsymbol{T}^{op} = \boldsymbol{0} \quad (77)$$

According to mentioned previously, the vector $\boldsymbol{v}^{+}$ is composed by both vectors $\boldsymbol{v}$ and $\boldsymbol{w}$, i. e., $\boldsymbol{v}^{+} = [\boldsymbol{v}^{T} \;\; \boldsymbol{w}^{T}]^{T}$, then the vector $\boldsymbol{i}^{op}$ given in (76) can be also represented by:

$$\boldsymbol{i}^{op} = -\left[\frac{\partial f(\boldsymbol{v}^+)}{\partial v_1}\ \frac{\partial f(\boldsymbol{v}^+)}{\partial v_2}\cdots\frac{\partial f(\boldsymbol{v}^+)}{\partial v_n}\ \frac{\partial f(\boldsymbol{v}^+)}{\partial w_1}\ \frac{\partial f(\boldsymbol{v}^+)}{\partial w_2}\cdots\frac{\partial f(\boldsymbol{v}^+)}{\partial w_m}\right]^T \tag{78}$$

As the optimization process of the cost function does not depend on the slack variables $\boldsymbol{w}$, equation (76) can then be replaced by the following one:

$$\boldsymbol{i}^{op} = -\left[\underbrace{\frac{\partial f(\boldsymbol{v}^+)}{\partial v_1}\ \frac{\partial f(\boldsymbol{v}^+)}{\partial v_2}\cdots\frac{\partial f(\boldsymbol{v}^+)}{\partial v_n}}_{n\text{-components}}\ \underbrace{0\ 0\ 0\ \ldots\ 0}_{m\text{-components}}\right]^T \tag{79}$$

To illustrate the performance of the proposed neural network, some simulation results involving nonlinear optimizations problems are presented in the next section.

4. Simulation results

In this section, some simulation results are presented to illustrate the application of the neural network approach developed in the previous sections for solving combinatorial optimization problems, dynamic programming problems and nonlinear optimization problems.

4.1 Combinatorial optimization problems

The modified Hopfield network has been used in the solution of the matching problem proposed in Papadimitriou & Steiglitz (1982), with matrix $\boldsymbol{P}$ given by:

$$\boldsymbol{P} = \begin{bmatrix} 7 & 2 & 1 & 9 & 4 \\ 9 & 6 & 9 & 5 & 5 \\ 3 & 8 & 3 & 1 & 8 \\ 7 & 9 & 4 & 2 & 2 \\ 8 & 4 & 7 & 4 & 8 \end{bmatrix}$$

A graphical representation of this problem is illustrated in Fig. 3(a). The parameters $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ to be used in the modified Hopfield network illustrated in Fig. 1 are obtained using equations given in (41) and (42), while the parameters $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$ are defined using (47) and (48). The elements of the vector $\boldsymbol{v}$ of the modified Hopfield network were randomly generated between 0 and 1.

The modified Hopfield network converged after 50 iterations, which is considered extremely fast when compared with other neural approaches used in combinatorial optimization. In comparative terms, the simulation of this problem by the conventional Hopfield network proposed in Hopfield & Tank (1985), using the same initial values for the output vector $\boldsymbol{v}$, reaches the final solution in 317 iterations. The edges set, representing the optimal solution, is given by {[1,3];[2,5];[3,1];[4,4];[5,2]}. The vectors $\boldsymbol{p}$ and $\boldsymbol{v}(\boldsymbol{p})$, and the matrix $\boldsymbol{V}(\boldsymbol{p})$ representing the obtained solution is provided by:

$$\boldsymbol{p} = [3 \quad 5 \quad 1 \quad 4 \quad 2]^T$$

$$\boldsymbol{v}(\boldsymbol{p}) = [\underbrace{00100}_{\delta(p_1)^T} \ \underbrace{00001}_{\delta(p_2)^T} \ \underbrace{10000}_{\delta(p_3)^T} \ \underbrace{00010}_{\delta(p_4)^T} \ \underbrace{01000}_{\delta(p_5)^T}]^T$$

$$\boldsymbol{V}(\boldsymbol{p}) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3(b) illustrates the minimum bipartite graph representing the final solution obtained by the modified Hopfield network. Figure 4 shows the evolution of the matrix $\boldsymbol{V}$ during the convergence process of the network. The minimization of the energy term E^{op} guarantees the minimum total sum among all edges (E^{op} = 15), where the value of Δt used in (11) were assumed as 0.01.

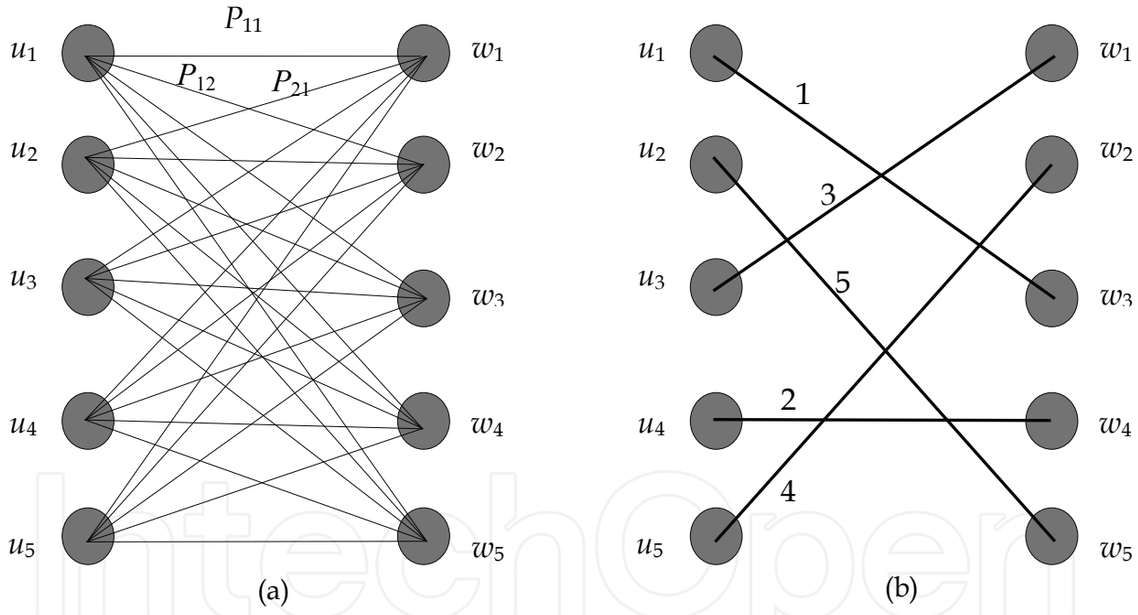


Fig. 3. Bipartite graph composed by ten nodes (a) and minimum bipartite graph (b).

4.2 Dynamic programming problems

The first dynamic programming problem to be solved by the modified Hopfield network is illustrated in Fig. 5, which is composed by three intermediate stages (n = 3) and two states in each stage (m = 2). The values of the weights $d_{xi,(x+1)j}$, which link the i^{th} neuron of stage x to the j^{th} neuron of the following stage (x+1), are also indicated in Fig. 5. The goal is to find the minimum path (from all possible paths), which starts at the source node and reaches the destination node, passing by only one state node in each stage.

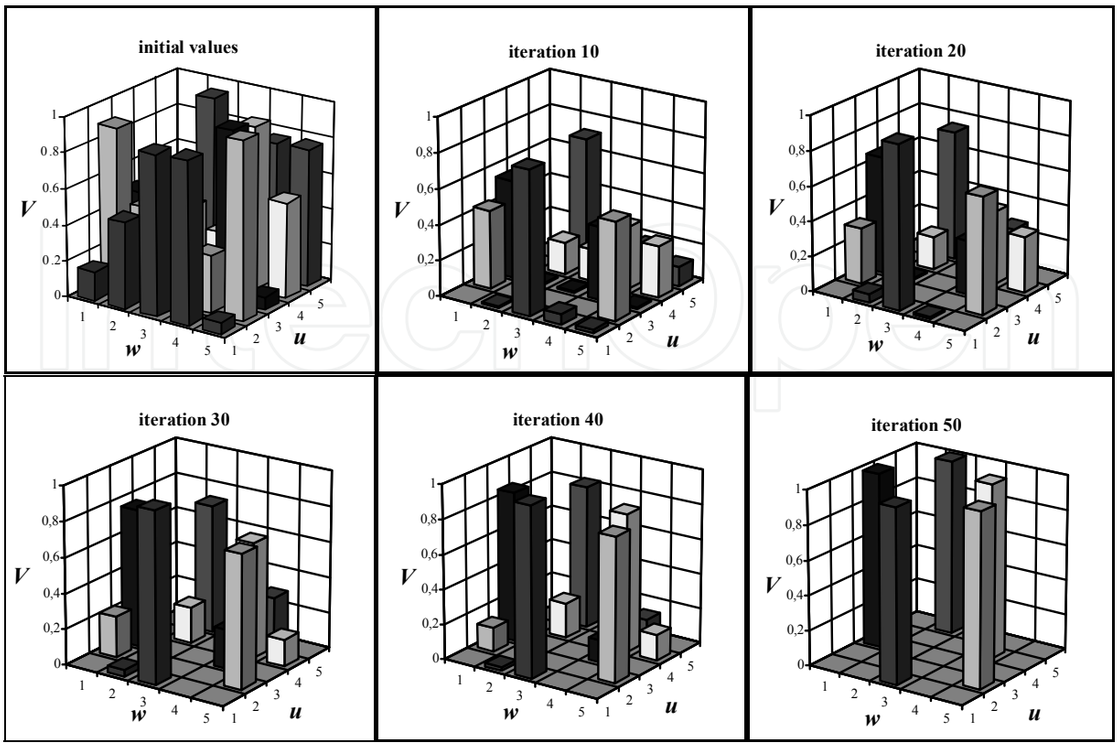


Fig. 4. Evolution of the matrix $\boldsymbol{V}$ for bipartite graph problem.

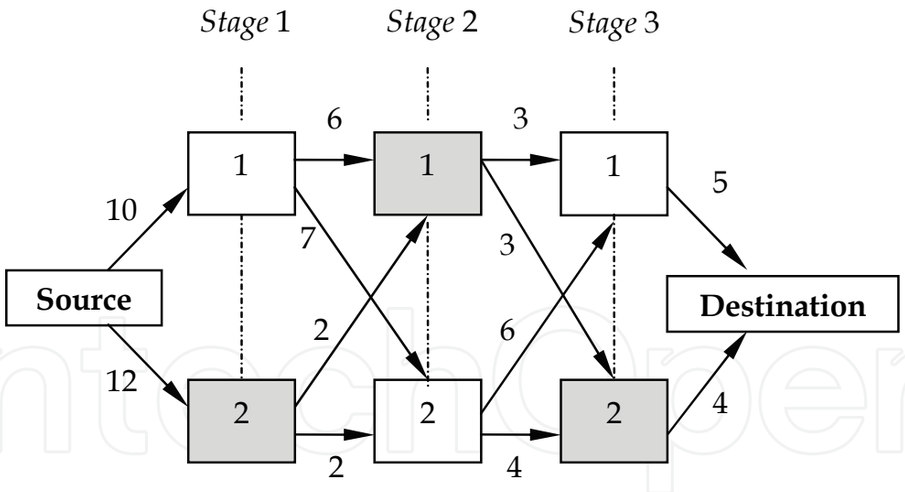


Fig. 5. The dynamic programming problem (m = 3 and n =2)

For this example the total number of possible paths is equal to 8, which is obtained by m^n. The optimal solution is given by the shaded states, i.e., state 2 in stage 1, state 1 in stage 2, and state 2 in stage 3. The modified Hopfield network applied in this problem always converges after three iterations. The vectors $\boldsymbol{p}$ and $\boldsymbol{v}(\boldsymbol{p})$, and the matrix $\boldsymbol{V}(\boldsymbol{p})$ representing the obtained solution are as follows:

$$\boldsymbol{p} = [2 \quad 1 \quad 2]^T$$

$$\boldsymbol{v}(\boldsymbol{p})^T = [0 \;\; 1 \quad 1 \;\; 0 \quad 0 \;\; 1]^T$$

$$\boldsymbol{V}(\boldsymbol{p}) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The minimization of the energy term E^{op} guarantees that the solution obtained represents the minimum path (E^{op} = 21) from all possible paths.

To illustrate that the proposed network can be used efficiently, various dynamic programming problems were simulated and the results compared with those obtained by the network proposed by Chiu et al. (1991). In this example, the number of stages and number of states has been increased step by step. The number of stages and number of states in each stage for the simulated examples were established by values belonging to the integer set defined by {2, 4, 8, 16, 32, 64}. The goal was to find a valid path, which starts at the source node, visits one and only node in each stage, and reaches the destination node, with the minimum possible total length. For such purposes, we have simulated both networks using the same initial values for the output vectors $\boldsymbol{v}$, which were randomly generated between 0 and 1 for all instances treated in this comparison.

The weights of the connection $d_{xi,(x+1)j}$ linking nodes (states) of the network were randomly selected from the integer set {1, 3, 5, 7, 9}. For those instances with n and m less than 32, each example was simulated twenty times using random initial conditions. Examples with n and m greater than or equal to 32 were simulated ten times.

The performance analysis for both networks was done using the average normalized path length (D), which is given by:

$$D = \frac{S_c}{n_s.(n+1)} \tag{80}$$

where S_c is the sum of the selected paths after network convergence; n_s is the number of simulations; n is the number of stages.

The simulation results are shown in Table 1. In this table, the D^{MHN} and D^{CN} columns provide, respectively, the results of the average normalized path length for the modified Hopfield network and the one proposed by Chiu et al. (1991). This table shows that the modified Hopfield network presented better results with a shorter normalized path length. For checking the results obtained by the modified Hopfield network, simulations using conventional dynamic programming were also carried out using the same instances described in Table 1. In all analyzed instances, the values reached to the objective functions were practically identical in both approaches. However, the conventional method obtains the final solutions more rapidly than the modified Hopfield network. On the other hand, the implementation of dynamic programming problem to specialist systems in a neural network environment can be more easily made by using the modified Hopfield network. For all problems treated in this subsection, the values of Δt used in (11) were assumed as 0.01.

The adverse facts that can influence on the performance of the network proposed by Chiu et al. (1991) and explain their less accurate results are the following: i) optimization and constraint terms involved in problem mapping are treated in a single stage, ii) interference between optimization and constraint terms affects the precision of the equilibrium points, and iii) the convergence process of the network depend on the correct adjustment of the weighting constants associated with the energy terms. However, the modified Hopfield

network presented here treats these terms in different stages. The terms $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ (belonging to E^{conf}) of the modified Hopfield network were developed to force the validity of the structural constraints associated with the dynamic programming problem, and the terms $\boldsymbol{T}^{op}$ and $\boldsymbol{i}^{op}$ were projected to find a minimum path among all possible paths.

Number of stages (n)	Number of states (m)	D^{MHN}	D^{CN}
2	2	3.13	3.25
4	4	2.03	3.12
8	8	1.34	2.00
16	16	1.06	1.85
32	32	1.03	1.61
64	64	1.02	1.39
16	2	3.14	3.21
16	4	1.79	2.98
16	8	1.26	1.85
16	32	1.13	1.79
2	16	1.17	1.53
4	16	1.02	1.60
8	16	1.09	1.76

Table 1. Simulation results (dynamic programming).

Thus, the main advantages of using a modified Hopfield network to solve dynamic programming problems are i) consideration of optimization and constraint terms in distinct stages with no interference with each other, ii) use of the unique energy term (E^{conf}) to group all constraints imposed on the problem, and iii) lack of need for adjustment of weighting constants for initialization. In all examples, the network output vector $\boldsymbol{v}$ was initialized with small random values defined between 0 and 1. It should be noticed that the increase in the number of states and stages does not degrade the performance of the network, but rather shows its efficiency.

4.3 Nonlinear optimization problems

In this subsection, we provide three examples to illustrate the effectiveness of the proposed architecture to solve nonlinear optimizations problems.

Example 1. Consider the following constrained optimization problem proposed in Bazaraa & Shetty (1979) in page 491, which is composed by inequality constraints and bounded variables:

$$\begin{aligned}
&\text{Min } f(\boldsymbol{v}) = e^{v_1} + v_1^2 + 4v_1 + 2v_2^2 - 6v_2 + 2v_3 \\
&\text{subject to: } v_1^2 + e^{v_2} + 6v_3 \le 15 \\
&\qquad v_1^4 - v_2 + 5v_3 \le 25 \\
&\qquad v_1^3 + v_2^2 - v_3 \le 10 \\
&\qquad 0 \le v_1 \le 4 \\
&\qquad 0 \le v_2 \le 2 \\
&\qquad v_3 \ge 0
\end{aligned}$$

This problem has a unique optimal solution $v^* = [0.0 \quad 1.5 \quad 0.0]^T$, and the minimal value of $f(v^*)$ at this point is equal to –3.5. Using a value of $\Delta t = 0.01$ in (11), which is corresponding to step ((III)) in Fig. 2, the solution vector (equilibrium point) obtained by the modified Hopfield network is given by $v = [0.0002 \quad 1.5001 \quad 0.0000]^T$, with $E(v) = f(v) = -3.499$. However, if we assume the value $\Delta t = 0.0001$ the network reaches the optimal solution v^*. Figure 6 shows the trajectories of the modified Hopfield network starting from $v^0 = [2.33 \; 0.31 \; 0.16]^T$ and converging towards the equilibrium point.

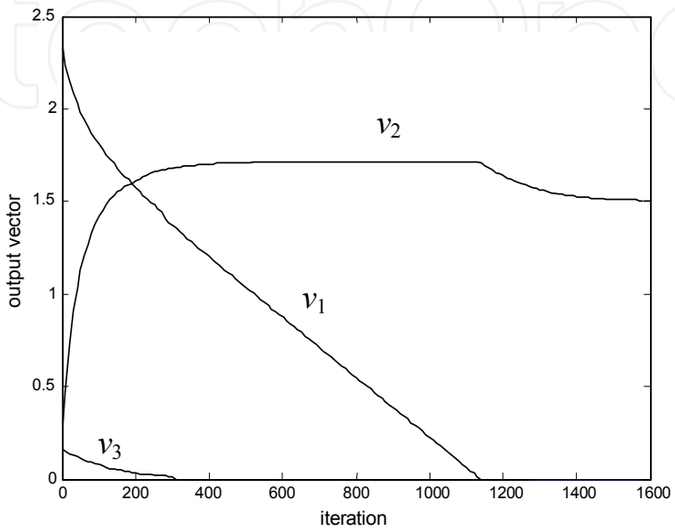


Fig. 6. Transient behavior of the modified Hopfield network in example 1.

To observe the global convergent behavior of the proposed network, we generated 15 initial points randomly distributed between 0 and 5. The bound constraints represented by the last three equations are directly mapped through the piecewise activation function defined in (9). All simulation results obtained by the modified Hopfield network show that the proposed architecture converges to v^*. The trajectories of the objective function starting from several initial points are illustrated in Fig. 7. All trajectories lead towards the same theoretical minimal value provided by $f(v^*) = -3.5$ when assumed $\Delta t = 0.0001$. These results show the efficiency of the modified Hopfield network for solving constrained nonlinear optimization problems.

A comparison using the SQP (Sequential Quadratic Programming) method and the modified Hopfield network was also done for this example. Both methods have found the same final solution. The SQP method reached the final solution in 35 iterations, whereas the modified Hopfield network needed 1587 iterations. However, convergence time to reach the final solution has not been directly proportional to number of iterations. For this example, using a microcomputer Pentium IV, the SQP method and the modified Hopfield network obtained the final solution in 3.65 and 5.86 seconds, respectively. This fact can be explained with respect to simplicity associated with the convergence process used by the modified Hopfield network, which consists of only three main steps as shown in Fig. 1. As well, as observed with the dynamic programming problems, the modified Hopfield network is an alternative method for solving constrained optimization problems and has the advantage of offering simplicity of implementation both in analogue hardware making use of operational amplifiers and in digital hardware by using digital signal processors.

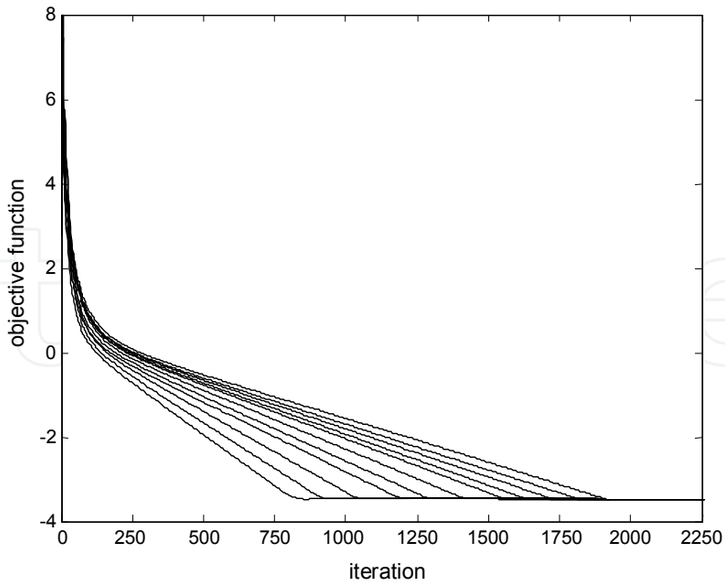


Fig. 7. Evolution of the objective function for 15 initial points in example 1.

To provide a more consistent analysis in relation to the efficiency of the proposed architecture, we make in the next example a comparison between the results produced by the modified Hopfield network with those provided by the network developed in Xia et al. (2002), and also by the topology presented in Kennedy & Chua (1988).

Example 2. Consider the following constrained optimization problem proposed in Xia et al. (2002), which is composed by inequality constraints:

$$\text{Min } f(\boldsymbol{v}) = v_1^3 + (v_1 - 2v_2)^3 + e^{v_1+v_2}$$

$$\text{subject to : } \boldsymbol{v} \in V$$

where $V = \{\boldsymbol{v} \in \Re^2 \mid v_1^2 + v_2^2 \leq 1\}$. This problem has a unique optimal solution given by $\boldsymbol{v}^*$ = [–0.5159 0.8566]^T with $f(\boldsymbol{v}^*)$ = –9.8075. All simulation results provided by the modified Hopfield network show that it is convergent to $\boldsymbol{v}^*$.

In Table 2, the results obtained by the modified Hopfield network using Δt = 0.0001 are compared with those provided by the projection neural network proposed in Xia et al. (2002), and also those given by the nonlinear circuit network developed in Kennedy & Chua (1988). Six different initial points were chosen, where two points {(1, 0); (0, -1)} are located in V and four {(-2, -2); (2, -2); (2, 2); (-2, 2)} are not in V. The results obtained by the modified Hopfield network are very close to the exact solution. The mean error between the solutions obtained by the network and the exact solution is less than 0.02%. We can verify that all solutions produced by the modified Hopfield network are quite stable.

According to Table 2 the nonlinear circuit network proposed in Kennedy & Chua (1988) can apparently approach $\boldsymbol{v}^*$ in only two cases. This was also observed in simulations performed in Xia et al. (2002). The projection neural network developed in Xia et al. (2002) produces solutions for all cases presented in Table 1, and we can observe that the final solutions depend on their initial values. It is also shown in table 1 that the modified Hopfield

network, independently of the initial values of v, has converged to the same final values for all simulations. To illustrate the global convergent behavior of the modified Hopfield network, Fig. 8 shows the trajectories of v starting from several initial points.

Initial Vector	Modified Hopfield Network	Projection Neural Network	Nonlinear Circuit Network
$v^{(0)} = [\,2 \;\; 2]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5195 \;\; 0.8641]^T$
$v^{(0)} = [-2 \;\; 2]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5196 \;\; 0.8641]^T$
$v^{(0)} = [-2 \;\; -2]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5161 \;\; 0.8564]^T$	$\propto$
$v^{(0)} = [\,2 \;\; -2]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5162 \;\; 0.8563]^T$	$\propto$
$v^{(0)} = [\,1 \;\; 0]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5162 \;\; 0.8564]^T$	$\propto$
$v^{(0)} = [\,0 \;\; -1]^T$	$v = [-0.5160 \;\; 0.8566]^T$	$v = [-0.5161 \;\; 0.8564]^T$	$\propto$

Table 2. Comparison of the simulation results in example 2.

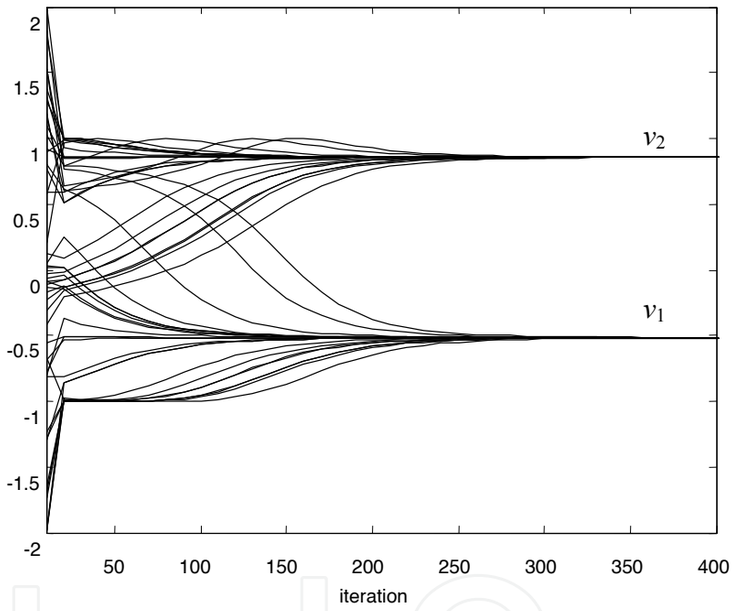


Fig. 8. Trajectories of the modified Hopfield network for 20 initial points in example 2.

It is important to observe that all trajectories starting from the inside or outside of the feasible region V converge to v^*. Thus, the proposed approach always converges to the optimal solution, independently whether the chosen initial point is located in the feasible region or not. Therefore, we can conclude that the modified Hopfield network is of high robustness.

A comparison using the SQP method and the modified Hopfield network was also made for this example. The SQP method has reached the exact solution for all simulations. Table 3 shows the number of iterations and convergence time used in each approach to reach the final solution for different initial values of the output vector v. From this table, although the method SQP obtains the final solution in less iteration, it is verified that convergence time of the modified Hopfield network is close to that required by the SQP method, where for $v^{(0)} = [-2 \;\; 2]^T$ and $v^{(0)} = [1 \;\; 0]^T$ the network converged more rapidly.

Initial Vector	Modified Hopfield Network		SQP Method	
	Iterations	Convergence time	Iterations	Convergence time
$v^{(0)} = [\,2 \;\; 2]^T$	278	3.86	34	3.72
$v^{(0)} = [-2 \;\; 2]^T$	316	2.83	24	2.87
$v^{(0)} = [-2 \;\; -2]^T$	297	3.19	24	2.81
$v^{(0)} = [\,2 \;\; -2]^T$	303	5.03	42	4.41
$v^{(0)} = [\,1 \;\; 0]^T$	359	2.94	25	3.16
$v^{(0)} = [\,0 \;\; -1]^T$	311	4.76	39	4.15

Table 3. Comparison between SQP method and modified Hopfield network in example 2.

Example 3. Consider the following constrained optimization problem proposed in Bazaraa & Shetty (1979) in page 418, which is composed by inequality and equality constraints:

$$\begin{aligned} &\text{Min } f(v) = v_1^3 + 2v_2^2 \cdot v_3 + 2v_3 \\ &\text{subject to : } \; v_1^2 + v_2 + v_3^2 = 4 \\ &\qquad v_1^2 - v_2 + 2v_3 \le 2 \\ &\qquad v_1 \, , v_2 \, , v_3 \ge 0 \end{aligned}$$

The optimal solution for this problem is given by $v^* = [0.0 \;\; 4.0 \;\; 0.0]^T$, where the minimal value of $f(v^*)$ at this point is equal to zero. Figure 9 shows the trajectories of the network variables starting from the initial point $v^0 = [1.67 \;\; 1.18 \;\; 3.37]^T$. All simulation results obtained by the modified Hopfield network using $\Delta t = 0.001$ show that the proposed architecture is globally convergent to v^*.

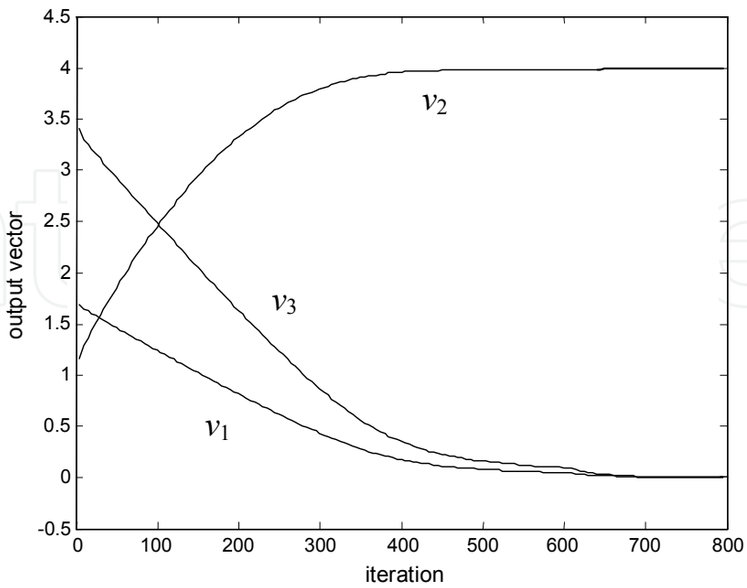


Fig. 9. Transient behavior of the modified Hopfield network in example 3.

The network has also been evaluated for different values of initial conditions. The trajectories of the objective function starting from several initial points are illustrated in Fig. 10. All trajectories lead toward the same equilibrium point. These results show the ability and efficiency of the modified Hopfield network for solving constrained nonlinear optimization when equality and inequality constraints are simultaneously included in the problem.

In comparison with results obtained by using the multilayer perceptron network proposed in Bazaraa & Shetty (1979), and starting from the same initial points, it was observed that the modified Hopfield Network not only converges more quickly, but also results in higher accuracy.

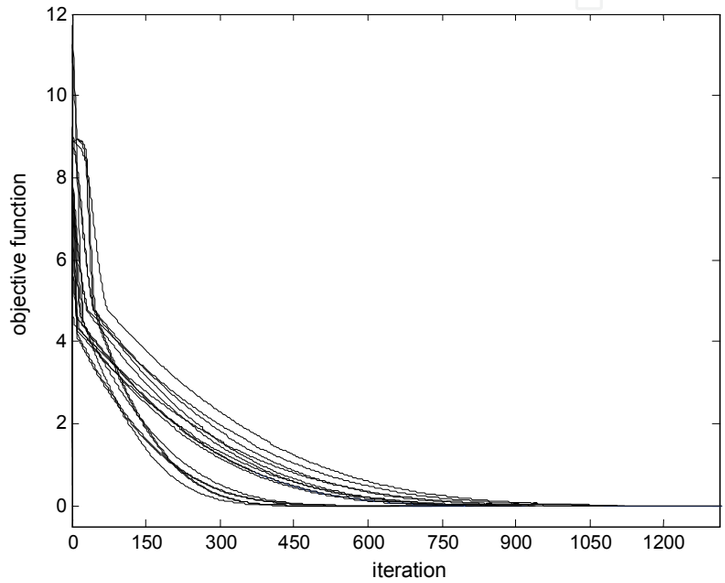


Fig. 10. Evolution of the objective function for 15 initial points in example 3.

In relation to the SQP method, the obtained solution was the same found by the modified Hopfield network. For this example, the SQP method reached the final solution using 30 iterations (3.66 seconds), whereas the modified Hopfield network needed 768 iterations (3.48 seconds). So, for this example, the modified Hopfield network has converged in less time than the SQP method.

5. Conclusions

This chapter presents an approach for solving optimization problems using artificial neural networks. More specifically, a modified Hopfield network is developed and its internal parameters are computed using the valid-subspace technique.

The developed approach allows to solve several classes of optimization problems through a unique neural network architecture. The optimization problems treated in this chapter are the combinatorial optimization problems, dynamic programming problems and nonlinear optimization problems. An energy function E^{op} was designed to conduct the network output

to the equilibrium points corresponding to a cost constraint. All structural constraints associated with the optimization problems can be grouped in E^{conf}.

The simulation results demonstrate that the network is an alternative method to specialist algorithms and has the advantage of being implementable in a neural network environment, which can be mapped in hardware for engineering applications. The internal parameters of the network were explicitly computed using the valid-subspace technique that guarantees the network convergence. All simulation results show that the proposed network is completely stable and convergent to the solutions of the optimization problems considered in this chapter. The network has also been evaluated for different values of initial conditions. All trajectories lead toward the same equilibrium point.

6. References

Aiyer, S. V. B.; Niranjan, M. & Fallside, F. (1990). A Theoretical investigation into the performance of the Hopfield network. *IEEE Transactions on Neural Networks*, Vol.1, 204-215.

Bazaraa, M. S. & Shetty, C. M. (1979). *Nonlinear Programming – Theory and Algorithms*, Wiley, New York.

Chiu, C.; Maa, C. Y. & Shanblatt, M. S. (1991). Energy function analysis of dynamic programming neural networks. *IEEE Transactions on Neural Networks*, Vol. 2, 418-426.

Dillon, J. D. & O'Malley, M. J. (2002). A Lagrangian augmented Hopfield network for mixed integer non-linear programming problems. *Neurocomputing*, Vol. 42, 323-330.

Fiacco, A. V. & McCormick, G. P. (1968). *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, Wiley, New York.

Graham, A. (1981). *Kronecher Products and Matrix Calculus*, Ellis Horwood Ltd., Chichester, UK.

Haykin, S. (1999). *Neural Networks – A Comprehensive Foundation*, Prentice-Hall Inc., Upper Saddle River, New Jersey.

Hillier, F. S. & Lieberman, G. J. (1980). *Introduction to Operations Research*, Holden Day, San Francisco, California.

Hopfield, J. J. & Tank, D. W. (1985). Neural computation of decisions in optimization problems. *Biological Cybernetics*, Vol. 52, 141-152.

Hopfield, J. J. (1984). Neurons with a graded response have collective computational properties like those of two-state neurons. *Proc. of the National Academy of Science*, Vol. 81, 3088-3092.

Kakeya, H. & Okabe, Y. (2000). Fast combinatorial optimization with parallel digital computers. *IEEE Transactions on Neural Networks*, Vol. 11, 1323-1331.

Kennedy, M. P. & Chua, L. O. (1988). Neural networks for nonlinear programming. *IEEE Transactions on Circuits and Systems*, Vol. 35, 554-562.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming*, Addison-Wesley, Reading, MA.

Papadimitriou, C. H. & Steiglitz, K. (1982). *Combinatorial Optimization - Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ.

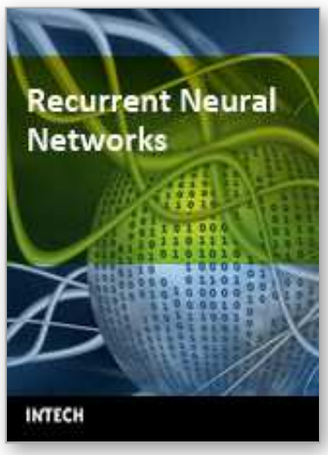
Silva, I. N.; Arruda, L. V. R. & Amaral, W. C. (1997). Robust estimation of parametric membership regions using artificial neural networks. *International Journal of Systems Science*, Vol. 28, 447-455.

Vidyasagar, M. (1993). *Nonlinear Systems Analysis*, Prentice-Hall, Englewood Cliffs, NJ.

Xia, Y.; Leung, H. & Wang, J. (2002). A projection neural network and its application to constrained optimization problems. *IEEE Trans. on Circuits and Systems*, Vol. 49, 447-458.

IntechOpen

IntechOpen



Recurrent Neural Networks

Edited by Xiaolin Hu and P. Balasubramaniam

ISBN 978-953-7619-08-4
Hard cover, 400 pages
Publisher InTech
Published online 01, September, 2008
Published in print edition September, 2008

The concept of neural network originated from neuroscience, and one of its primitive aims is to help us understand the principle of the central nerve system and related behaviors through mathematical modeling. The first part of the book is a collection of three contributions dedicated to this aim. The second part of the book consists of seven chapters, all of which are about system identification and control. The third part of the book is composed of Chapter 11 and Chapter 12, where two interesting RNNs are discussed, respectively.The fourth part of the book comprises four chapters focusing on optimization problems. Doing optimization in a way like the central nerve systems of advanced animals including humans is promising from some viewpoints.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Ivan N. da Silva, Wagner C. Amaral, Lucia V. Arruda and Rogerio A. Flauzino (2008). Recurrent Neural Approach for Solving Several Types of Optimization Problems, Recurrent Neural Networks, Xiaolin Hu and P. Balasubramaniam (Ed.), ISBN: 978-953-7619-08-4, InTech, Available from: http://www.intechopen.com/books/recurrent_neural_networks/recurrent_neural_approach_for_solving_several_types_of_optimization_problems

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2008 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen