An Energy Function-Based Design Method for Discrete Hopfield Associative Memory With Attractive Fixed Points

Mehmet Kerem Müezzinoglu, *Student Member, IEEE*, Cüneyt Güzeliş, and Jacek M. Zurada, *Fellow, IEEE*

***Abstract*—An energy function-based autoassociative memory design method to store a given set of unipolar binary memory vectors as attractive fixed points of an asynchronous discrete Hopfield network (DHN) is presented. The discrete quadratic energy function whose local minima correspond to the attractive fixed points of the network is constructed via solving a system of linear inequalities derived from the strict local minimality conditions. The weights and the thresholds are then calculated using this energy function. If the inequality system is infeasible, we conclude that no such asynchronous DHN exists, and extend the method to design a discrete piecewise quadratic energy function, which can be minimized by a generalized version of the conventional DHN, also proposed herein. In spite of its computational complexity, computer simulations indicate that the original method performs better than the conventional design methods in the sense that the memory can store, and provide the attractiveness for almost all memory sets whose cardinality is less than or equal to the dimension of its elements. The overall method, together with its extension, guarantees the storage of an arbitrary collection of memory vectors, which are mutually at least two Hamming distances away from each other, in the resulting network.**

***Index Terms*—Associative memory, memory storage and retrieval.**

I. Introduction

DYNAMICAL autoassociative memory (DAM) is an autonomous convergent dynamical system which produces a correction $\mathbf{p}$ of a distorted vector $\mathbf{p}^d$ along its dynamics, as with the convergence of the trajectory starting at the initial state $\mathbf{x}^0 = \mathbf{p}^d$ to the fixed point $\mathbf{x}^* = \mathbf{p}$.

Given a set M of memory vectors, the first goal of any DAM design method is to map the set of fixed points of the resulting dynamical system to M in a bijective way. This enables encoding M as system parameters and produces a content addressable memory for lossless data compression. There is a number of simple yet significant methods [1]–[3] focusing only on this condition in the design, which is, however, not sufficient to make the resulting DAM correct possible errors. The system becomes capable of correcting a distorted version of a fixed point if the basin of attraction of this fixed point is trimmed in the design to include the distorted point. This can be achieved by providing the convergence of the trajectory starting at the distorted vector to the nearest fixed point (memory vector) in a metric defined on the state–space. Note that the satisfaction of this requirement necessitates the attractiveness of each fixed point, i.e., the existence of a trajectory which starts at $\mathbf{x}^0$ and tends to the fixed point $\mathbf{x}^*$ for all $\mathbf{x}^0$ in an open neighborhood of $\mathbf{x}^*$.[1] A design method satisfying these two conditions gives an ideal DAM which maps its initial state vector $\mathbf{x}^0$ to a steady state $\mathbf{f}(\mathbf{x}^0)$ where

$$\mathbf{f}\left[\mathbf{x}^0\right] = \arg\min_{\mathbf{y}\in M} d\left(\mathbf{x}^0, \mathbf{y}\right) \tag{1}$$

is the association mapping.

This paper focuses on the design problem of discrete Hopfield network (DHN) [1] as a unipolar binary DAM to approximate (1) as steady-state solution of the network for $\mathbf{x}^0 \in \{0,1\}^n$, $M \subseteq \{0,1\}^n$ and the metric $d(\cdot,\cdot)$ chosen as the Hamming distance. Many DAM design methods have been proposed for DHN which performs a nonlinear first-order recursion of the state vector $\mathbf{x}$ in n-dimensional binary space

$$x_i[k+1] = \phi\left(\sum_{j=1}^{n} w_{ij}x_j[k] + t_i\right) \quad i \in \{1,\ldots,n\} \tag{2}$$

where $\mathbf{W} = [w_{ij}] \in \Re^{n\times n}$ is the weight matrix and $\mathbf{t} = [t_i] \in \Re^n$ is the threshold vector [4]. In unipolar binary case, the activation function $\phi(\cdot)$ is chosen as the unit-step nonlinearity. Stability analysis of DHN [5] has shown that the recursion (2) in asynchronous mode, i.e., when only one entry of the state vector is updated at each time instant, necessarily tends to a fixed point if $\mathbf{W}$ is symmetric and has nonpositive diagonal entries. The proof is based on analysis of a discrete quadratic energy function

$$E(\mathbf{x}) = \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \tag{3}$$

defined on the state–space of (2) with $\mathbf{Q} \in \Re^{n\times n}$ and $\mathbf{c} \in \Re^n$, which is nonincreasing along the asynchronous recursion. It can be shown that (2) indeed tends to a (discrete) local minimum of this function if the diagonal entries of the weight matrix are all zero, i.e., a network consisting of a single layer of neurons without self-feedback. On the other hand, as proved in

Manuscript received October 3, 2002; revised October 20, 2003. The work of M. K. Müezzinoglu was supported by the Scientific and Technical Research Council of Turkey, Ankara, Turkey. The work of J. M. Zurada was supported in part by the Systems Research Institute (IBS) of the Polish Academy of Science (PAN) 01-447 Warsaw, and in part by Wyzsza Szkola Humanistyczno-Ekonomiczna, Lodz, Poland.

M. K. Müezzinoglu and J. M. Zurada are with the Computational Intelligence Laboratory, University of Louisville, Louisville, KY 40292 USA (e-mail: mkerem@ ieee.org).

C. Güzeliş is with the Electrical and Electronics Engineering Department, Dokuz Eylül University, Izmir, 35160 Turkey.

Digital Object Identifier 10.1109/TNN.2004.841775

[1]This general definition of attractiveness reduces to the asymptotic stability in the sense of Lyapunov when the system is deterministic.

the next section, one can always find a symmetric DHN without self-feedback which has fixed points located exactly at the discrete local minima of a given quadratic form defined on the binary space. Assuming that the given memory vectors are at least 2-Hamming distance away from each other, the design of such a finite-state dynamical system as an ideal associative memory is therefore equivalent to the design of its energy function, i.e., determination of coefficients $(\mathbf{Q}, \mathbf{c})$ in (3), such that

A1) $\mathbf{x}$ is a strict local minimum of (3), i.e., $E(\mathbf{x}) < E(\mathbf{y})$ for all $\mathbf{y} \in \{0,1\}^n$ such that $d(\mathbf{x},\mathbf{y}) = 1$, if $\mathbf{x} \in M$;

A2) basins of the local minima corresponding to the memory vectors share the entire binary space in an equal way, so extraneous local minima are avoided.

In this paper, we focus on the first design criterion and show in the following section that construction of a discrete quadratic satisfying A1 is equivalent to solving a linear strict inequality system $\mathbf{A}(M)\mathbf{w} < \mathbf{0}$ for a parameter vector $\mathbf{w}$ representing coefficients $\{q_{ij}, c_i\}$ $i,j = 1,\ldots,n$. We also show that the feasibility of this system is equivalent to the existence of a symmetric DHN which possesses all memory vectors as attractive fixed points. Thus, the maximum performance of an asynchronous symmetric DHN without self-feedback as a DAM in correcting 1-b errors is revealed. Based on this equivalence, we conclude that no asychronous symmetric DHN can be designed by any method if and only if this inequality system is infeasible. In this case, we extend our original method to maintain the design by constructing a particular discrete *piecewise* quadratic energy function, and also by generalizing the conventional DHN model to minimize it. This extended method guarantees the recall of any memory vector from its distorted versions by the resulting network, without any restrictions on the memory set.

Unlike the design methods [6] employing linear inequalities derived directly from the network dynamics for allocating the fixed points, our method is concerned with the construction of energy landscape in order to ensure not only the desired fixed points but also their attractiveness, so the resulting DAM is guaranteed to correct all possible 1-b distortions of each memory vector. The method eventually establishes an encoding for an arbitrary set of n-dimensional memory vectors as $(n^2+n)/2$ weight coefficients associated to the recursion plus the number parameters of the multilayer perceptron used for separation.

A similar energy function design strategy without the extension has been recently reported in the design of complex-valued multistate Hopfield networks in [7].

We describe the design procedure in the following section. In Section III, we present the results of some computer experiments which compare the method with some conventional DAM design methods proposed in [1], [2], and [4] in terms of qualitative performance. An application of the method to restore a set of printed characters can also be found in Section III. Some concluding remarks are finally given in Section IV.

II. Discrete Quadratic Design

To simplify the notation, we first note that the $n \times n$ matrix $\mathbf{Q}$ in (3) can be considered without loss of generality as symmetric, since for an arbitrary matrix $\mathbf{P} \in \Re^{n\times n}$, there exists a symmetric counterpart $\mathbf{R} = (\mathbf{P}+\mathbf{P}^T)/2$ such that $\mathbf{x}^T\mathbf{R}\mathbf{x} = \mathbf{x}^T\mathbf{P}\mathbf{x}$ for all $\mathbf{x} \in \Re^n$. Due to the unipolarity of the discrete variable $\mathbf{x} \in \{0,1\}^n$, the linear term $\mathbf{c}^T\mathbf{x}$ can be further expressed as a quadratic term: $\mathbf{x}^T \cdot \mathrm{diag}(c_1, c_2, \ldots, c_n) \cdot \mathbf{x}$ to reformulate $E(\cdot)$ as a single quadratic term

$$E(\mathbf{x}) = \mathbf{x}^T \hat{\mathbf{Q}} \mathbf{x} \tag{4}$$

on $\{0,1\}^n$, where $\hat{\mathbf{Q}}$ is a symmetric real matrix equal to $\mathbf{Q} + \mathrm{diag}(c_1, c_2, \ldots, c_n)$. Expanding (4) as the sum $\sum_{i=1}^n \sum_{j=1}^n \hat{q}_{ij} x_i x_j$ and then using the symmetry of $\hat{\mathbf{Q}}$, provides an alternative notation

$$E(\mathbf{x}) = \mathbf{a}(\mathbf{x})^T \mathbf{w} \tag{5}$$

which is linear in the coefficient vector $\mathbf{w} = [\hat{q}_{11} \ \cdots \ \hat{q}_{1n} \vdots \hat{q}_{22} \ \cdots \ \hat{q}_{2n} \vdots \ \cdots \ \vdots \ \hat{q}_{nn}]^T \in \Re^{(n^2+n)/2}$ obtained by a lexiographic ordering of the coefficients $\hat{q}_{ij}$'s. The column vector $\mathbf{a}(\mathbf{x})$ captures the multiplicative nonlinearity of (4) in $\mathbf{x}$

$$\mathbf{a}(\mathbf{x}) := 2\left[\frac{x_1^2}{2} \vdots x_1x_2 \vdots \cdots \vdots x_1x_n :: \frac{x_2^2}{2} \vdots x_2x_3 \vdots \cdots \vdots x_2x_n :: \ \cdots :: \frac{x_n^2}{2}\right]^T. \tag{6}$$

Expressing (4) as the weighted sum of parameters as in (5) enables computation of the coefficient vector $\mathbf{w}^*$ in $\Re^{n(n+1)/2}$ under linear inequality constraints to construct a $\hat{\mathbf{Q}}$ such that (4), consequently (3), satisfies the design condition A1 for a given set of memory vectors $M \subseteq \{0,1\}^n$.

We assume throughout the paper that the condition

$$d(\mathbf{u},\mathbf{v}) > 1 \ \forall (\mathbf{u},\mathbf{v}) \in M \times M, \quad \mathbf{u} \neq \mathbf{v} \tag{7}$$

holds for a given memory set M. Then, in order to embed each memory vector as a strict local minimum of the desired quadratic (4) as suggested in A1, we obtain for each memory vector $\mathbf{p} \in M$ the set of strict linear inequalities

$$\mathbf{a}(\mathbf{p})^T\mathbf{w} < \mathbf{a}(\mathbf{y})^T\mathbf{w}, \ \ \mathbf{y} \in \mathcal{B}_1(\mathbf{p}) - \{\mathbf{p}\} \tag{8}$$

to be solved for the parameter vector $\mathbf{w}$. Here, "$-$" stands for the set difference and $\mathcal{B}_1(\mathbf{u})$ is defined as the 1-Hamming neighborhood of $\mathbf{u}$ as $\{\mathbf{x} \in \{0,1\}^n : d(\mathbf{u},\mathbf{x}) \leq 1\}$. We denote the polyhedral cone induced by these n linear inequalities by $S_{\mathbf{p}}$. Since the desired coefficient vector $\mathbf{w}^*$ lies within the intersection

$$S = \bigcap_{\mathbf{p}\in M} S_{\mathbf{p}} \tag{9}$$

its search is indeed the feasibility problem of the homogeneous linear inequalities which induce the polyhedral cone S. By rearranging (8) and incorporating all inequalities associated to all memory vectors $\{\mathbf{p}^1\}_{i=1}^{|M|}$, we obtain the homogeneous inequality system as

$$\mathbf{A}(M)\mathbf{w} < \mathbf{0} \tag{10}$$

where

$$\mathbf{A}(M) := \begin{bmatrix} \mathbf{N}(\mathbf{p}^1) \\ \vdots \\ \mathbf{N}(\mathbf{p}^{|M|}) \end{bmatrix} \tag{11}$$

and $\mathbf{N}(\mathbf{p})$ is the $n \times (n^2+n)/2$ matrix whose jth row is determined as $\mathbf{a}(\mathbf{p})^T - \mathbf{a}(\mathbf{y}^j)^T$ with $y_k^j = 1 - p_k$ for $j = k$ and $y_k^j =$

p_k otherwise. Note that the binary vectors $\mathbf{y}^j, j = 1, \ldots, n$, are 1-Hamming neighbors of $\mathbf{p}$ as pointed in (8).

A. Original Design Method

If the given memory set yields a feasible inequality system, then the coefficient matrix $\hat{\mathbf{Q}}$ of the desired quadratic (4), so the coefficients $(\mathbf{Q}, \mathbf{c})$ of (3) which satisfy A1 can be easily determined from a solution of (10). Hence, in our original design method, we look for a solution of (10) by directly applying an available linear inequality solver, such as linear programming [8] or Newton's method [9], [10]. Having determined the discrete quadratic energy function (3) which indicates the memory set as its local minima set, construction of a dynamical system, whose limit points correspond to these local minima, completes the DAM design. We describe this procedure in the following corollary.

Corollary 1: The fixed points of asynchronous recursion (2) correspond to the local minima of (3) for the weight matrix $\mathbf{W} = -2\mathbf{Q}$ and the threshold vector $\mathbf{t} = -\mathbf{c}$. Moreover, recursion (2) designed in this way is convergent, namely for any initial state it converges to one of its fixed points.

Proof: The state vector $\mathbf{x}$ of a finite-state dynamical system converges to a local minimum of a discrete function $E(\mathbf{x})$ if every state transition provides a decrement in $E(\mathbf{x})$. The proof is based on imposing this condition on the DHN dynamics whose state–space is the binary space $\{0,1\}^n$. Taking into account that only one entry of the state vector is allowed to change at a single time step, we analyze the desired behavior of the network in two separate cases.

1) Suppose $x_i = 0$ and ith entry is updated at time instant k, then the value of this entry in the next step should be (12), shown at the bottom of the page, where $\mathbf{e}^i$ stands for the ith unit vector. Since the diagonal entries of $\mathbf{Q}$ are all zero, we rearrange (12) and formulate it as

$$x_i[k+1] = \phi\left(-2\sum_{j=1}^{n} q_{ij}x_j[k] - c_i\right) \tag{13}$$

where $\phi(\cdot)$ is the unit-step nonlinearity.

2) Suppose now that $x_i = 1$ and ith entry is updated at time instant k. Then we write (14), shown at the bottom of the page, which can also be expressed as in (13).

Comparing (13) with (2) we conclude that the desired network can be obtained by choosing $\mathbf{W} = -2\mathbf{Q}$ and $\mathbf{t} = -\mathbf{c}$. As $\mathbf{Q}$ here is considered as zero-diagonal, the convergence results from the well-known result of Bruck and Goodman in [5]. ■

The asynchronous operation of this network is characterized by updating only one neuron at a single time-step according to (2) and keeping all other states unchanged. The update order of the states is of course a parameter of the network dynamics and is usually chosen randomly, resulting in a stochastic network.

Observe from the proof of Corollary 1 that the resulting network is an energy-minimizing network, in the sense that a state transition is accepted if and only if it causes a decrement in (3). Since a point in the 1-Hamming neighborhood of a local minimum $\mathbf{x}^*$ has strictly greater energy by the construction of (3), then we conclude that each fixed point of the network, which corresponds to a memory vector, is attractive. This implies that for each $\mathbf{y}$ in the 1-Hamming neighborhood of a fixed point $\mathbf{x}^*$, there exists an index $i \in \{1, 2 \ldots, n\}$ such that the state vector necessarily converges to $\mathbf{x}^*$ in a single step when the network is initiated by $\mathbf{y}$ and ith neuron is updated first. By utilizing a random update order, one cannot ensure obviously this desired convergence, because there is no guarantee that ith neuron will be updated first in the case mentioned previously. For this purpose, we propose the following update order to be followed by the resulting network, which ensures the correction of each 1-b distortion on the memory vectors.

Attempt to update jth neuron for the current state vector for $j \in \{1, \ldots, n\}$. If any of the state transitions leads to a fixed point,[2] then accept that transition. Otherwise choose an arbitrary transition.

B. Applicability of the Original Method

It is evident by the previous derivation that the feasibility of (10), i.e., nonemptiness of S in (9), is necessary and sufficient to embed all memory vectors as attractive fixed points of DHN, hence, the success of the method is totally dependent on the given memory set M, which constitutes the only information used in the construction of the inequality system (10). Although we know that S might be an empty set for some M, we can only be sure of this by constructing (10) and then checking for its feasibility by attempting to solve it. A simple result on the feasibility of (10) is provided by the following fact:

Fact 1: Equation (10) is feasible for any memory set containing a single, yet arbitrary binary vector $\mathbf{p} \in \{0,1\}^n$.

Proof: Let us define $\mathbf{u} = \mathbf{x} - \mathbf{p}$ and observe that the positive definite quadratic $Q(\mathbf{u}) = \|\mathbf{u}\|_2^2$ on $\{0,1\}^n$ possesses a unique strict local minimum at $\mathbf{u} = \mathbf{0}$. Then, $\mathbf{p}$ is the (unique) strict local minimum of the quadratic $P(\mathbf{x}) = Q(\mathbf{x} - \mathbf{p})$. ■

[2]A point $\mathbf{x} \in \{0,1\}^n$ is a fixed point of DHN if and only if $\Phi(\mathbf{t} - \mathbf{W}\mathbf{x}) = \mathbf{x}$, where $\Phi(\cdot)$ is the diagonal transformation from $\Re^n$ to $\{0,1\}^n$ defined by $\Phi(\mathbf{u}) = [\phi(u_1) \cdots \phi(u_n)]^T$.

$$x_i[k+1] = \begin{cases} 0, & \text{if } (\mathbf{x}[k] + \mathbf{e}^i)^T\mathbf{Q}(\mathbf{x}[k] + \mathbf{e}^i) + \mathbf{c}^T(\mathbf{x}[k] + \mathbf{e}^i) < (\mathbf{x}[k])^T\mathbf{Q}\mathbf{x}[k] + \mathbf{c}^T\mathbf{x}[k] \\ 1, & \text{otherwise} \end{cases} \tag{12}$$

$$x_i[k+1] = \begin{cases} 0, & \text{if } (\mathbf{x}[k] - \mathbf{e}^i)^T\mathbf{Q}(\mathbf{x}[k] - \mathbf{e}^i) + \mathbf{c}^T(\mathbf{x}[k] - \mathbf{e}^i) < (\mathbf{x}[k])^T\mathbf{Q}\mathbf{x}[k] + \mathbf{c}^T\mathbf{x}[k] \\ 1, & \text{otherwise} \end{cases} \tag{14}$$

Since the original method described in the preceding subsection is not applicable to a memory set which yields an infeasible inequality system, we extend this method in the following subsection to carry on the design even if (10) is infeasible.

C. Extension to the Method

In this subsection, we assume that the system (10) of $n|M|$ homogeneous inequalities has no solution for a given M which satisfies (7). As no discrete quadratic energy function possessing such a memory set as strict local minima exists in this case, the best one can do instead is to construct a discrete *piecewise quadratic* function to be minimized by a modified version of DHN. For this purpose we need to partition the inequality system (10) into two feasible systems as $\mathbf{A}_1\mathbf{w} < 0$ and $\mathbf{A}_2\mathbf{w} > 0$. Such a partitioning is always possible by the following constructive proposition, since (10) contains no zero row.

Proposition 1: If $\mathbf{A} \in \Re^{p\times q}$ has no zero row, then the following algorithm provides an $\mathbf{w}^* \in \Re^q$ such that:

$$\sum_{i=1}^{q} a_{ui} w_i^* < 0, \quad \forall u \in \mathcal{I}_1$$
$$\sum_{i=1}^{q} a_{vi} w_i^* > 0, \quad \forall v \in \mathcal{I}_2 \tag{15}$$

where $\mathcal{I}_1$ and $\mathcal{I}_2$ are two disjoint integer sets with $\mathcal{I}_1 \cup \mathcal{I}_2 = \{1,\dots,p\}$.

Step 0) Choose an arbitrary $\mathbf{w}^0 \in \Re^q$. Construct three matrices $\mathbf{A_1}(0)$, $\mathbf{A_2}(0)$ and $\mathbf{A_3}(0)$ each of which consists of the rows of $\mathbf{A}$ whose inner products with $\mathbf{w}$ is negative, positive, and zero, respectively. Set $k = 0$.

Step 1) Let u^k be the number of rows of $\mathbf{A_3}(k)$. If $u^k = 0$, then set $\mathbf{w}^* = \mathbf{w}^k$, $\mathbf{A_1} = \mathbf{A_1}(k)$, $\mathbf{A_2} = \mathbf{A_2}(k)$. and stop. Otherwise, choose an arbitrary $i \in \{1,\dots,u^k\}$ and a sufficiently small positive ε such that $\mathbf{A_1}(k)\cdot(\mathbf{w}^k+\varepsilon\cdot\mathbf{a_3}^i(k)) > \mathbf{0}$ and $\mathbf{A_2}(k)\cdot(\mathbf{w}^k+\varepsilon\cdot\mathbf{a_3}^i(k)) < \mathbf{0}$, where $\mathbf{a_3}^i(k)$ is the transpose of the ith row of $\mathbf{A_3}(k)$. Set $\mathbf{w}^{k+1} = \mathbf{w}^k + \varepsilon\cdot\mathbf{a_3}^i(k)$.

Step 2) Augment $\mathbf{A_1}(k)$ (resp. $\mathbf{A_2}(k)$) by the rows of $\mathbf{A_3}(k)$, whose inner products with $\mathbf{w}^{k+1}$ is positive (resp. negative) to form $\mathbf{A}_1(k+1)$, (resp. $\mathbf{A}_2(k+1)$). Delete these rows from $\mathbf{A}_3$, increment k by 1 and return to Step 1.

Proof: The proof comprises two parts: i) the number of rows of $\mathbf{A_3}(k)$ decreases strictly as k increases, so $\mathbf{A_3}$ vanishes at some finite k for any choice of $\mathbf{w}^0$, and the procedure halts and ii) at each iteration k, each row of $\mathbf{A_1}(k)$, resp. $\mathbf{A_2}(k)$, has a strictly positive, resp. negative, inner-product with $\mathbf{w}^k$.

Observe from the processing done in Step 0 and 2 that, unless $\mathbf{A_3}(k)$ is empty, it consists of some rows of $\mathbf{A}$ which are orthogonal to $\mathbf{w}^k$. To show that the procedure necessarily halts for all $\mathbf{w}^0 \in \Re^q$, we choose an arbitrary row, say $(\mathbf{a}_3^i(k))^T$, then scale it by a sufficiently small positive ε, and finally add the the transpose of the product to $\mathbf{w}^k$. This gives a point which is no longer orthogonal to $\mathbf{a}_3^i$:

$$(\mathbf{a}_3^i(k))^T\cdot\left(\mathbf{w}^k+\varepsilon\cdot\mathbf{a}_3^i(k)\right) = (\mathbf{a}_3^i(k))^T\cdot\mathbf{a}_3^i(k) = \|\mathbf{a}_3^i(k)\|_2^2 > 0$$

due to the assumption that $\mathbf{A}$, consequently $\mathbf{A_3}(k)$, contains no zero row. Thus, the point $\mathbf{w}^{k+1}$ calculated in Step 1 is excluded by the hyperplane induced by $(\mathbf{a}_3^i(k))^T\cdot\mathbf{w} = 0$ and at least ith row of $\mathbf{A_3}(k)$ is transferred to $\mathbf{A_1}$ and deleted in Step 2.

The proof of the second part is by induction on k. For the base case, where $k = 0$, both $\mathbf{A_1}(k)\cdot\mathbf{w}^k > \mathbf{0}$ and $\mathbf{A_2}(k)\cdot\mathbf{w}^k < \mathbf{0}$ hold by the initialization in Step 0. Let these two strict inequalities hold for some $k > 0$. Then, for all $\mathbf{d} \in \Re^q$, there exists a sufficiently small positive ε such that $\mathbf{A_1}(k)\cdot\left(\mathbf{w}^k+\varepsilon\cdot\mathbf{d}\right) > \mathbf{0}$ and $\mathbf{A_2}(k)\cdot\left(\mathbf{w}^k+\varepsilon\cdot\mathbf{d}\right) < \mathbf{0}$. Thus, we have $\mathbf{A_1}(k)\cdot\mathbf{w}^{k+1} > \mathbf{0}$ and $\mathbf{A_2}(k)\cdot\mathbf{w}^{k+1} < \mathbf{0}$. It is obvious that the inequalities still hold after $\mathbf{A_1}(k)$ and $\mathbf{A_2}(k)$ are augmented by the rows from $\mathbf{A_3}(k)$ in the way suggested in Step 2. So, the matrices $\mathbf{A_1}(k+1)$ and $\mathbf{A_2}(k+1)$ never contain a row orthogonal to $\mathbf{w}^{k+1}$ and maintain the strict inequalities.

The rows of $\mathbf{A_1}$ and $\mathbf{A_2}$ define two partitions as given in (15) after the algorithm halts. ■

The algorithm finds a point which does not belong to any of the hyperplanes defined by the rows of $\mathbf{A}$, and is indeed an escape procedure from the null-space of the rows that are orthogonal to the initial choice $\mathbf{w}^0$. However, the decomposition $\mathbf{A_1}$ and $\mathbf{A_2}$ is not unique and the point $\mathbf{w}^*$ provided by the algorithm is dependent on the choice of $\mathbf{w}^*$. The implications of this is discussed later at the end of this section.

When the algorithm is applied to $\mathbf{A}(M)$ in (10), which is now considered as infeasible, the matrices $\mathbf{A_1}$ and $\mathbf{A_2}$ produced by the algorithm induce two disjoint feasible subsystems of the inequality system (10). Although the coefficient vector $\mathbf{w}^*$ produced by the algorithm satisfies all inequalities induced by $\mathbf{A_1}$, as $\mathbf{A_1}\mathbf{w}^* < \mathbf{0}$, the inequalities induced by $\mathbf{A_2}$ are all violated: $\mathbf{A_2}\mathbf{w}^* > \mathbf{0}$. Consequently, the former inequality system gives rise to a quadratic energy landscape coefficients $(\mathbf{Q},\mathbf{c})$ constructed by $\mathbf{w}^*$, while the latter yields $(-\mathbf{Q},-\mathbf{c}$ constructed by $-\mathbf{w}^*$.

Recall from the construction of $\mathbf{A}(M)$ described just in (11) that each inequality in (10), thus, each row of $\mathbf{A_1}$ and $\mathbf{A_2}$, indeed imposes a restriction on the energy of a specific vector $\mathbf{y} \in \mathcal{B}_1(\mathbf{x})$ to satisfy $E(\mathbf{x}) < E(\mathbf{y})$, where $\mathbf{x}$ is a memory vector. Let $\bar{D}$ denote the set of binary vectors obtained by applying the inverse of (6) to the rows of $\mathbf{A_2}$, and let $D := \{0,1\}^n - \bar{D}$. Then, we conclude by the previous discussion that each memory vector is a strict local minimum of the piecewise quadratic function

$$E_{PQ}(\mathbf{x}) = \begin{cases} \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x}, & \text{if } \mathbf{x} \in D \\ -\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{c}^T\mathbf{x}, & \text{if } \mathbf{x} \in \bar{D} \end{cases} \tag{16}$$

where the coefficients $(\mathbf{Q},\mathbf{c})$ are calculated by using $\mathbf{w}^*$ as described in the beginning of this section. Note that the points in $\bar{D}$ may contain only points which are 1-Hamming distance away from a memory vector. Since the memory vectors are restricted to be at least 2-Hamming distance away from each other by the condition (7), then they are all located in D.

The best performance a conventional asynchronous symmetric DHN can achieve as a binary associative memory is indeed provided by the original design method described in Section II-A. It was shown in [5] that (2) necessarily tends to a local minimum of a discrete quadratic for a symmetric weight matrix $\mathbf{W}$. However, in the present case, no regular

quadratic form (3) satisfying A1 exists, thus, there exists no asynchronous symmetric DHN having attractive fixed points located at the memory vectors. A modification of (2) becomes, thus, necessary to minimize the discrete piecewise quadratic energy function (16), which we have constructed instead of (3) in this case.

The idea of choosing the weights and thresholds of a continuous recurrent network to minimize a given continuous piecewise quadratic, dependent on the state vector is not new [11]. But, to our knowledge, no asynchronous recursion has been proposed for minimizing a discrete piecewise quadratic form yet. To minimize the discrete piecewise quadratic function (16) constructed in the way previously explained, we propose a generalized version of (2) with state-dependent weights and thresholds, and investigate its qualitative performance in the following.

Definition 1: The generalized version of (2) as

$$\hat{x}_i[k] = \phi\left(h\left[\mathbf{x}\left[k\right]\right]\left(\sum_{i=1}^{n} w_{ij}x_j[k] + t_i\right)\right) \tag{17}$$

$$x_i[k+1] = \frac{1 - h\left[\hat{\mathbf{x}}[k]\right]}{2}x_i[k] + \frac{1 + h\left[\hat{\mathbf{x}}[k]\right]}{2}\hat{x}_i[k] \tag{18}$$

where $h[\cdot] : \{0,1\}^n \rightarrow \{-1,1\}$ is a discrete function that separates a subset $D \subseteq \{0,1\}^n$ from its complement $\bar{D}$ as

$$h[\mathbf{u}] = \begin{cases} 1, & \text{if } \mathbf{u} \in D \\ -1, & \text{if } \mathbf{u} \in \bar{D} \end{cases} \tag{19}$$

is called the constrained one-nested discrete Hopfield network (CON-DHN).

We use the terms *constrained* and resp. *one-nested* here to point out the additional constraint imposed by (18) on the original recursion (2), and resp. the parameter control mechanism $h[\cdot]$, which can be realized as a discrete multilayer perceptron [12], bringing an additional nonlinearity nested in the activation function $\phi(\cdot)$ as in (17).

Corollary 2: The asynchronous recursion (17)–(18) is convergent and has fixed points located at the local minima of (16) for the weight matrix $\mathbf{W} = -2\mathbf{Q}$ and the threshold vector $\mathbf{t} = -\mathbf{c}$. Moreover, these fixed points are all attractive.

Proof: We first prove that the recursion leaves $\bar{D}$ following a single state transition, whenever it is initiated by a point in $\bar{D}$. Let $\mathbf{x}[0] \in \bar{D}$. Then $h[\mathbf{x}[0]] = -1$ and the right-hand side of (17) enables a state transition which would cause a decrement in $-E(\mathbf{x})$, where $E(\cdot)$ is the quadratic form with parameters $(\mathbf{Q}, \mathbf{c})$. If the state candidate $\hat{\mathbf{x}}[1]$ is in D, then $\mathbf{x}[1]$ assigned by (18) is in D. If $\hat{\mathbf{x}}[1] \in \bar{D}$, then the candidate is rejected (not assigned to $\mathbf{x}[1]$ by (18)) and $\mathbf{x}[1] = \mathbf{x}[0]$. Then, (17) produces another candidate in the next time step. This procedure is repeated until a point in D is obtained and assigned to $\mathbf{x}[1]$. Such a point is necessarily produced by (17) at some time step because, by construction of $\bar{D}$, for each $\bar{\mathbf{z}}$ in $\bar{D}$, there exists a point $\mathbf{z}$ in D, which is 1-Hamming distance away from $\bar{\mathbf{z}}$, such that $-E(\mathbf{z}) < -E(\bar{\mathbf{z}})$.

Suppose now that the state vector $\mathbf{x}[k]$ of the network, whose parameters are determined as $(\mathbf{W}, \mathbf{t}) = (-2 \cdot \mathbf{Q}, \mathbf{c})$, is in D at any time instant k. In this case, $h[\mathbf{x}[k]] = 1$ and the right-hand side of (17) is the same as that of (2). If $\hat{\mathbf{x}}[k] \in \bar{D}$, i.e., $h[\hat{\mathbf{x}}[k]] = -1$, then (18) imposes $\mathbf{x}[k+1] = \mathbf{x}[k]$ and the state vector is not admitted into $\bar{D}$. If $\hat{\mathbf{x}}[k] \in D$, then, due to (18), $\mathbf{x}[k+1] = \hat{\mathbf{x}}[k]$, so either $\mathbf{x}[k+1]$ is equal to $\mathbf{x}[k]$, or it has strictly lower energy than that of $\mathbf{x}[k]$ by Corollary 1. In other words, the network operates exactly the same as the conventional DHN (2) in this case. As a result, $\mathbf{x}[k] \in D$ implies $\mathbf{x}[k+1] \in D$ and, together with the result in the first step, we have $E_{PQ}(\mathbf{x}[k+1]) < E_{PQ}(\mathbf{x}[k])$, if $\mathbf{x}[k+1] \neq \mathbf{x}[k]$. Since D is finite, the recurrence (17)–(18) is convergent and it converges to a point in D.

Finally, let $\mathbf{x}^*$ be a local minimum of E_{PQ}. The point $\hat{\mathbf{x}}$ calculated by (17) for $\mathbf{x}[k] = \mathbf{x}^*$ cannot be any point in D other than $\mathbf{x}^*$, because $\mathbf{x}^*$ is necessarily in D and each 1-Hamming neighbor $\mathbf{u}$ in D has $E(\mathbf{u}) > E(\mathbf{x}^*)$. (See Corollary 1). The only possibility for $\hat{\mathbf{x}}$, other than $\mathbf{x}^*$, is a point in $\bar{D}$, which would cause the state vector keep its previous state by (18). Thus, $\mathbf{x}^*$ is a fixed point of (17)–(18). Let $\mathbf{x}[k]$ be a binary vector in $\mathcal{B}_1(\mathbf{x}^*) \cap D$ differing from $\mathbf{x}^*$ by the ith element. Then updating ith element according to (17) gives an $\hat{\mathbf{x}}$ equal to $\mathbf{x}^*$ by Corollary 1. Since this vector is necessarily in D, $\mathbf{x}[k+1]$ is assigned to $\mathbf{x}^*$ in (18). Now suppose $\mathbf{x}[k]$ is a binary vector in $\mathcal{B}_1(\mathbf{x}^*) \cap \bar{D}$ differing from $\mathbf{x}^*$ by the ith element. As the update of this element according to (17) would necessarily cause a decrement in the quadratic $-E(\mathbf{x})$ by Corollary 1, $\hat{\mathbf{x}}$ is indeed equal to $\mathbf{x}^*$ and consequently to $\mathbf{x}[k+1]$. This establishes that $\mathbf{x}^*$ is attractive. ■

Since the recursion (17)–(18) designed this way is also an energy minimizing network, which converges to a local minimum of (16), we can conclude that the resulting CON-DHN corrects all possible 1-b distortions of the original memory vectors, with the update order of the neurons chosen as proposed at the end of Section II-A. The summary of the overall design method proposed in this section is illustrated as a flowchart in Fig. 1.

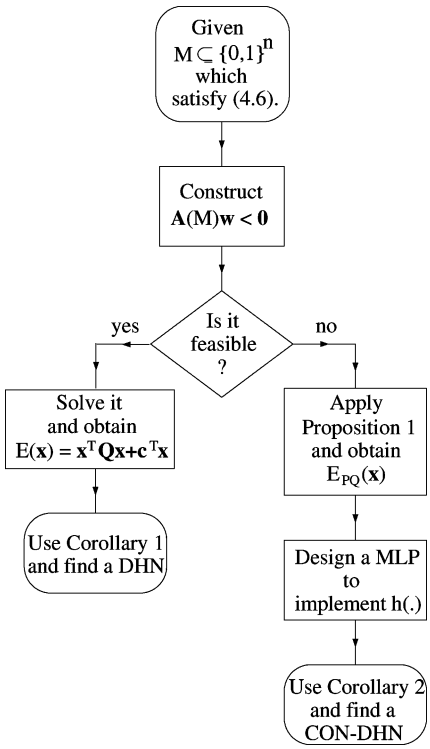


Fig. 1. Flowchart of the overall design method.

The necessity of algebraic computations $h[\mathbf{x}[k]]$ and $h[\hat{\mathbf{x}}[k]]$, which are used in the update of the state vector by (17)–(18) can be justified as follows: CON-DHN performs a more complicated task than that of the conventional DHN, as it can be designed to recall each element of an *arbitrary* M from its distorted versions, even when M does not comply with the restriction (10). This relaxation costs a multilayer perceptron (MLP) to perform $h[\cdot]$ in addition to the computations required by conventional DHN recursion. From the information storage point of view, the weights $\hat{w}_{ij}$ and the threshold $\hat{t}_j$ of this MLP in addition to the parameters $\mathbf{W}$ and $\mathbf{t}$ of the conventional recurrence (which have been determined by Corollary 2) are needed to be known.

In order to minimize globally the amount of data required to characterize MLP, one should adjust the coefficient vector $\mathbf{w}$ in the energy landscape design such that the number of hyperplanes which separate D from $\bar{D}$ is minimum. In other words, the algorithm presented in Proposition 1 should be modified to provide an optimal $\mathbf{w}^*$ in the sense that the points obtained by applying the inverse of (6) to the rows of resulting $\mathbf{A_2}$ can be separated from other points in $\{0,1\}^n$ by using the minimum number of hyperplanes.

We note that an approximate solution to this problem is to minimize the cardinality of $\bar{D}$ in the design, i.e., finding a $\mathbf{w}^*$ which minimizes the number of rows of $\bar{A}_2$. This obviously involves a discrete optimization scheme. However, finding a least squares solution to the infeasible inequality system (10) by Han's method [13], or by its variations [9], [10] can be still considered as an approximation to minimizing the cardinality of $\bar{D}$, consequently to minimizing the number of parameters of the MLP.

An exact solution to the following problem would constitute another approximation to the problem at hand, alternative to modifying the algorithm in Proposition 1, so would make our design more efficient.

Problem 1: Given $D \subseteq \{0,1\}^n$, find a set of hyperplanes with minimum cardinality which separates D from $\bar{D} = \{0,1\}^n - D$.

We leave this problem open and proceed with the simulation results.

III. Computer Experiments

A. Applicability and Capacity of the Original Design Method

The original design method proposed in Section II-A is applicable for a memory set which satisfies (7) and only when this set yields a linear homogeneous strict inequality system (10) derived from the local minimality conditions. Hence, the probability that a given memory set M yields a feasible inequality system is a measure of the performance of the design method if M satisfies (7). To quantify the applicability of the original method, we have investigated the mildness of these restrictions.

From pattern reconstruction point of view, our assumption (7) actually does not impose a significant restriction, since two prototype binary patterns that differ by a single bit is a rare instance especially in high dimensions. On the other hand, if they coexist in any given memory set of a high-dimension n, one can consider them too similar patterns. Thus, ignoring their difference in this case and excluding either of them from the memory set do not cause a significant error in association. We have not considered the design problems in which distinguishing two such binary patterns is essential, so proposed the design procedure in terms of strict inequalities (10). However, it is straightforward from the motivation that equalities could be incorporated to (10) as a generalization to ensure two neighboring binary points be (nonstrict) minima. The network parameters would then be determined exactly in the same way using a common solution to both the inequality and equality systems, if they exist. In short, (7) has a minor effect on the performance of the design method, and the limitations are mainly relevant to the feasibility condition as shown next.

TABLE I
Percentages of Memory Sets That Yielded Feasible Inequality Systems

n	$\lvert M\rvert$	$P\%$	n	$\lvert M\rvert$	$P\%$
10	5	100	50	25	100
	10	90		50	100
	15	9		75	78
	20	0		100	5
20	10	100	100	50	100
	20	100		100	100
	30	62		150	86
	40	0		200	8

We have generated 100 binary memory sets containing $|M|$ unipolar binary vectors of dimension n randomly, all satisfying (7), for some $|M|$, n values and constructed the homogeneous inequality system associated to each set as described by (10). The percentages ($P\%$) of memory sets which resulted in a feasible inequality system are averaged over 100 trials and presented in Table I.

It can be observed from Table I, that almost all memory sets with ratio $|M|/n$ less than 1 give a feasible inequality system, so our original method is applicable for such sets. Moreover, as n increases, this critical ratio also increases. This means that the method has better performance for large n values.

$|M| < n$ can be considered as the memory capacity of the original method, and also the capacity that any design method for DHN can achieve.

The experimentally derived bound 1 on ratio $|M|/n$, under which our method almost always ensures the desired DAM, is much greater than that of the conventional outer product rule [1] which ensures the storage of only $0.14 \cdot n$ arbitrarily chosen memory vectors [14] as fixed points (without ensuring their attractiveness). Assuming that the memory vectors are mutually orthogonal, projection learning rule [2] is capable of embedding up to n binary vectors as attractive fixed points to a DHN. However, this bound is not comparable to ours since orthogonality is a rather strict restriction on the memory vectors. In other words, the projection learning rule has an acceptable performance when applied to some specific memory sets among all memory sets with $|M|/n < 1$. The eigenstructure method [4], [6], which is probably the most effective design method yet, can store an arbitrary binary memory set as attractive fixed points. This method

has been proposed for the design of a continuous Hopfield network whose state–space is the n-dimensional hypercube $[0,1]^n$, including its interior region. The attractiveness of a fixed point is defined on this space but not on $\{0,1\}^n$. The method does not guarantee the correction of errors caused by a bit reversal of the memory vectors despite its providing attractiveness in the continuous sense.

The drawback of the proposed design procedure may be its high-computational requirements, especially when compared to the outer-product and the projection learning rules. Outer product method requires summing up simply $|M|$ outer products, so the time required for such design is in the order of $|M|$. Projection learning rule requires a single pseudoinversion and then multiplication, which is in the order of n^2. The eigen-structure method requires singular value decomposition and many matrix multiplications; the time required for its performance is also in n^2. Our method on the other hand requires a solution to a system of $n \cdot |M|$ linear inequalities involving $(n^2+n)/2$ real parameters. This can be performed in the order of n^3 multiplications.

With a relatively high-computational cost, the proposed method exploits the best performance that an asynchronous DHN can ever achieve. The following design example is developed for demonstrating the superiority of the the method to the former methods.

B. Design Example

Consider that a memory set consisting of the following four vectors is to be stored as attractive fixed points of a recurrent neural network

$$\mathbf{x}^1 = [0\,1\,0\,0\,1]^T \quad \mathbf{x}^2 = [0\,1\,1\,1\,1]^T$$
$$\mathbf{x}^3 = [1\,0\,1\,0\,1]^T \quad \mathbf{x}^4 = [1\,1\,0\,1\,1]^T.$$

Note that these memory vectors satisfy (7). By applying the proposed method and making use of linear programming for the solution of the homogenous linear inequalities, we have obtained the weight matrix and the threshold vector of the asynchronous DHN (2) as

$$\mathbf{W}=\begin{bmatrix} 0 & -15.2 & -7.5 & 7.6 & 5 \\ -15.2 & 0 & -15.2 & 15.1 & 9.2 \\ -7.5 & -15.2 & 0 & 7.6 & 5 \\ 7.6 & 15.1 & 7.6 & 0 & -6.1 \\ 5 & 9.2 & 5 & -6.1 & 0 \end{bmatrix} \quad \mathbf{t}=\begin{bmatrix} -6.3 \\ -3.3 \\ -6.3 \\ 12.8 \\ -1.3 \end{bmatrix}.$$

It can be verified that each memory vector is an attractive fixed point of this DAM. The projection learning rule and the eigenstructure method (for design parameter $\tau = 0.5$) could also store each vector as a fixed point of the recurrent networks of their concern, while the outer product rule could not store any of these vectors at all. By injecting each of the 32 binary vectors of dimension 5 to the networks obtained by these methods, we have also checked their performance in terms of creating spurious states. This simulation has shown that our method caused no spurious memory while the three extraneous binary vectors $[0\,0\,1\,1\,0]^T$, $[1\,0\,0\,1\,0]^T$, $[1\,1\,1\,1\,1]^T$ were stored as fixed points in the network obtained by the projection learning rule. The outer product rule also stored two spurious memories $[0\,1\,0\,1\,1]^T$ and $[1\,0\,1\,0\,0]^T$. For the design parameter $\tau = 0.5$, the eigenstructure method created four spurious memories, namely $[0\,0\,1\,0\,1]^T$, $[0\,1\,0\,1\,1]^T$, $[1\,0\,0\,0\,1]^T$, $[1\,0\,1\,1\,1]^T$ but they could be avoided by increasing τ. However, this effect also prevented some desired memory vectors from being stored. As an example, for $\tau = 1$, no binary fixed point could be stored as a fixed point to the network by this method.

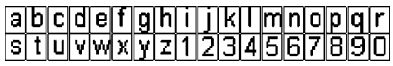


Fig. 2. Set of characters embedded as memory vectors to DHN.

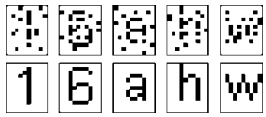


Fig. 3. Sample corrections performed by the proposed DHN.

C. Character Reconstruction

We applied the design procedure for the set of characters given in Fig. 2. The lexiographic orderings of these 13×10 black-white characters, where 1 and 0 denote a black and a white pixel, respectively, have been considered as the given memory vectors. It has been observed that this memory set satisfies (7). These 130-dimensional vectors have resulted in a feasible inequality system, so we have generated a regular quadratic energy function by solving it. The fixed points of DHN obtained as stated by Corollary 1 were identical with at the original memory vectors. It can be verified that DHNs designed by the outer product method and projection learning rule cannot store this information without any modification on the original characters. The network designed by the eigenstructure method stores all memory vectors but it is incapable of correcting most of the errors caused by 1-b reversals on the original characters. Moreover, the convergence of the state vector of this network to some nonbinary fixed points in $[0,1]^n$ was observed for some initial conditions. This, of course, cannot be considered as a correct behavior.

Although the original method ensures only the correction of 1-b errors, many 10-b distortions, even some 20-b distortions, of the memory vectors can be corrected by the resulting DHN, i.e., the basin of attraction of some memory vectors includes even some of its 20-Hamming neighborhood. Some of these corrections are illustrated in Fig. 3. Interestingly, no spurious memory was detected during the simulations of the DHN designed for this memory set, despite the fact that our method does not devise any procedure to avoid spurious memories.

D. Classification Application

We have also tested the performance of DAM as a classifier. The classification network in this experiment consists of a preprocessing network cascaded to a DAM designed by our method for the lexiographic orderings of the three 7×7 memory patterns in Fig. 4. The preprocessing network is used here to scan the input image with a 7×7 window and then to obtain the lexicographic ordering of each window. A distorted version of a map, shown in Fig. 5(a), which contains three types of patterns, is presented to the classification network. The network was able

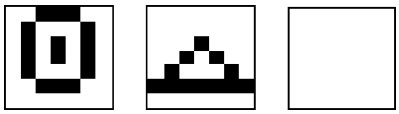
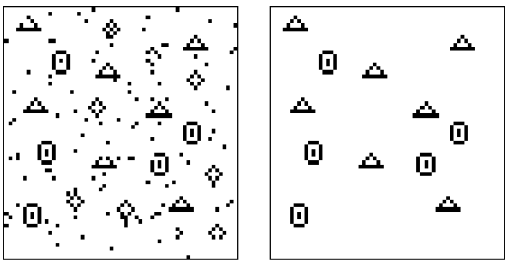


Fig. 4. Three memory patterns used in the classification experiment.



(a) (b)

Fig. 5. (a) Input map. (b) Classification result.

to classify the two recognized patterns in the map [see Fig. 5(b)] and the other patterns were associated to the "blank," which had also been introduced to the network as a memory pattern in the design phase. This example has shown that the classification task can be performed by the proposed DAM, even in noisy environment, besides its general usage in pattern recognition applications.

E. Application of the Extended Method

Finally, we present the results of another simple example to demonstrate the extension of the method described in Section II-C.

Consider a memory set consisting of the following vectors:

$$\begin{aligned}\mathbf{x}^1 &= [0\,0\,0\,0\,0]^T \quad \mathbf{x}^2 = [0\,0\,1\,1\,1]^T\\ \mathbf{x}^3 &= [0\,1\,0\,1\,1]^T \quad \mathbf{x}^4 = [0\,1\,1\,0\,0]^T\\ \mathbf{x}^5 &= [1\,0\,0\,1\,1]^T \quad \mathbf{x}^6 = [1\,1\,0\,0\,0]^T\\ \mathbf{x}^7 &= [1\,1\,1\,0\,1]^T \quad \mathbf{x}^8 = [1\,1\,1\,1\,0]^T.\end{aligned}$$

The original method cannot be applied for this memory set, because the inequality system (10) is infeasible and, thus, there exists no quadratic form (3) that has strict local minima located at these vectors. By applying Proposition 1, we obtain a coefficient vector $\mathbf{w}^*$ which partitions the design inequalities as in (15). From this coefficient vector, we then construct the piecewise quadratic form (16) with

$$\mathbf{Q} = \begin{bmatrix} 0 & -4.7 & 0.9 & -0.9 & -0.9\\ -4.7 & 0 & -4.7 & 2.9 & 2.9\\ 0.9 & -4.7 & 0 & -0.8 & -0.8\\ -0.9 & 2.9 & -0.8 & 0 & -7.5\\ -0.9 & 2.9 & -0.8 & -7.5 & 0 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} 7.6\\ -6.2\\ 7.7\\ 4.6\\ 4.6 \end{bmatrix}.$$

and

$$\begin{aligned}\bar{D} = \{&[1\,1\,1\,0\,0]^T [1\,1\,1\,1\,1]^T, [0\,1\,1\,1\,1]^T\\ &[0\,0\,0\,1\,1]^T [1\,1\,0\,1\,1]^T, [0\,1\,0\,0\,0]^T\}.\end{aligned}$$

It can be easily verified that each memory vector is a strict local minimum of this discrete function. The weight matrix and the threshold vector of CON-DHN are determined as $\mathbf{W} = -2\mathbf{Q}$ and $\mathbf{t} = -\mathbf{c}$, respectively, according to Corollary 2. The separating function $h[\cdot]$ is realized by a discrete multilayer perceptron, which responds to a vector in D as 1, and -1, otherwise. It has then been observed that each memory vector has been stored as an attractive fixed point. By initiating the network with each of all possible 32 binary vectors we have observed that no fixed point other than the memory vectors occurred in the resulting network.

IV. Conclusion

A DAM design method which employs homogeneous linear inequalities derived from the local minimality conditions is presented. A solution to this inequality system yields the coefficients $(\mathbf{Q}, \mathbf{c})$ of the discrete quadratic energy function of the DHN, so the weight matrix and the threshold vector of the network can be directly determined. Simulations have shown that almost all memory sets with cardinality less than n, where n is the dimension of the memory vectors, can be completely stored into the dynamical network and so perfectly recalled. The method eventually establishes an encoding for an arbitrary set of n-dimensional memory vectors as $(n^2 + n)/2$ weight and threshold coefficients associated to the recursion.

Throughout this paper, we have assumed that the given memory vectors were restricted to satisfy the condition (7), which constrains the memory vectors to be located at least 2-Hamming distance away from each other. Although it is a much milder condition than the ones imposed by many other design methods, the design method proposed here may still be needed to be extended to handle memory sets which do not satisfy (7), as a further step of this work. This improvement is not detailed in this paper, but it can be simply achieved by introducing some equality constraints in addition to the design inequalities (10), which impose the 1-Hamming neighbor memory vectors possessing equal energy values in the resulting landscape.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci.*, vol. 79, pp. 2554–2558, 1982.

[2] L. Personnaz, I. Guyon, and G. Dreyfus, "Collective computational properties of neural networks: New learning mechanisms," *Phys. Rev. A*, vol. 34, pp. 4217–4228, 1986.

[3] M. E. Savran and Ö. Morgül, "On the design of dynamic associative neural memories," *IEEE Trans. Neural Netw.*, vol. 5, no. 3, pp. 489–492, May 1994.

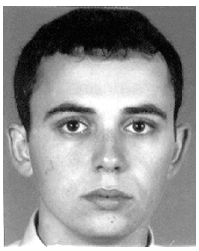
[4] A. N. Michel and J. A. Farrell, "Associative memories via artificial neural networks," *IEEE Control Syst. Mag.*, vol. 10, no. 3, pp. 6–17, Apr. 1990.

[5] J. Bruck and J. W. Goodman, "A generalized convergence theorem for neural networks," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 1089–1092, Sep. 1988.

[6] A. N. Michel and D. Liu, *Qualitative Analysis and Synthesis of Recurrent Neural Networks*. New York: Marcel Dekker, 2002.

[7] M. K. Müezzinoglu, C. Güzeliş, and J. M. Zurada, "A new design method for the complex-valued multistate Hopfield associative memory," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 891–899, Jul. 2003.

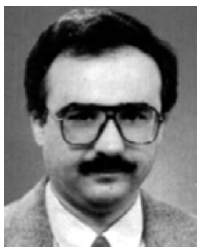
[8] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1973.
[9] R. Bramley and N. Winnicka, "Solving linear inequalities in a least squares sense," *SIAM J. Sci. Comput.*, vol. 17, pp. 275–286, 1996.
[10] M. C. Pinar and B. T. Chen, "$l(1)$ solution of linear inequalities," *IMA J. Numer. Anal.*, vol. 19, pp. 19–37, 1999.
[11] J. H. Park, Y. S. Kim, I. K. Eom, and K. Y. Lee, "Economic load dispatch for piecewise quadratic cost function using Hopfield neural network," *IEEE Trans. Power Syst.*, vol. 8, no. 3, pp. 1030–1038, Aug. 1993.
[12] J. M. Zurada, *Introduction to Artificial Neural Systems*. St. Paul, MN: West, 1992.
[13] S.-P. Han, "Least squares solution of linear inequalities," Math. Res. Center, Univ. Wisconsin, Madison, WI, TR-2141, 1980.
[14] A. Dembo, "On the capacity of associative memories with linear threshold functions," *IEEE Trans. Inf. Theory*, vol. 35, no. 4, pp. 709–720, Jul. 1989.



Mehmet Kerem Müezzinoglu (S'01) was born in Ankara, Turkey, in 1976. He received the B.Sc. and M.Sc. degrees from Istanbul Technical University, Istanbul, Turkey (TUBITAK), in 1998 and 2000, respectively, and the Ph.D. degree from Dokuz Eylül University, Izmir, Turkey, in 2003.

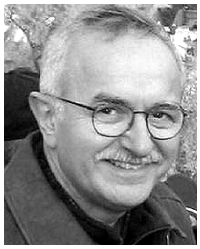
He has been pursuing his studies on recurrent neural networks and decision making under uncertainty at the Computational Intelligence Laboratory, University of Louisville, Louisville, KY, since 2003.

Dr. Müezzinoglu was awarded a Ph.D. scholarship by the Scientific and Technical Research Council of Turkey, Münir Birsel Foundation.



Cüneyt Güzeliş received the B.Sc., M.Sc., and Ph.D. degrees from Istanbul Technical University, Istanbul, Turkey, in 1981, 1984, and 1989, respectively.

He was a Visiting Researcher and Lecturer in the Department of Electrical Engineering and Computer Science, University of California, Berkeley, from 1989 to 1991, and was a Visiting Professor at the Information Laboratory, University of Paris-Nord, Paris, France, in 1986 and 1997. He has been a Full Professor at Dokuz Eylül University, Izmir, Turkey, since 1999. His research interests include nonlinear circuits and systems, neural networks, and their signal processing applications.



Jacek M. Zurada (M'82–SM'83–F'96) is the Samuel T. Fife Alumni Professor, and Acting Chairman of Electrical and Computer Engineering Department at the University of Louisville, Louisville, KY. He was the Coeditor of *Knowledge-Based Neurocomputing* (Cambridge, MA: MIT Press, 2000), *Computational Intelligence: Imitating Life* (Piscataway, NJ: IEEE Press), and *Introduction to Artificial Neural Systems* (Boston, MA: PWS-Kent, 1992), and was a contributor to *Progress in Neural Networks* (Norwood, NJ: Ablex, 1995). He is the author or coauthor of more than 200 journal and conference papers in the area of neural networks, data mining, image processing, and VLSI circuits. He is an Associate Editor of *Neurocomputing*.

Dr. Zurada was an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. From 2001 to 2003, he was a Member of the Editorial Board of the PROCEEDINGS OF THE IEEE. From 1998 to 2003, he was the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS. He has received a number of awards for distinction in research and teaching, including the 1993 Presidential Award for Research, Scholarship and Creative Activity. In 2001, he received the University of Louisville President's Distinguished Service Award for Service to the Profession. He has delivered numerous invited plenary conference presentations and seminars throughout the world. In 2003, He was conferred the Title of the Professor by the President of Poland, Aleksander Kwasniewski, and the Honorary Professorship of Hebei University, China. He is serving as the President of the IEEE Computational Intelligence Society, of which he is also a Distinguished Speaker.