The Hysteretic Hopfield Neural Network

Sunil Bharitkar, *Student Member, IEEE,* and Jerry M. Mendel, *Fellow, IEEE*

***Abstract*—A new neuron activation function based on a property found in physical systems—*hysteresis*—is proposed. We incorporate this neuron activation in a fully connected dynamical system to form the hysteretic Hopfield neural network (HHNN). We then present an analog implementation of this architecture and its associated dynamical equation and energy function. We proceed to prove Lyapunov stability for this new model, and then solve a combinatorial optimization problem (i.e., the N-queen problem) using this network. We demonstrate the advantages of hysteresis by showing increased frequency of convergence to a solution, when the parameters associated with the activation function are varied.**

***Index Terms*—Hysteresis, hysteretic activation function, hysteretic Hopfield neural networks, NP-complete, N-Queen problem, neural networks, optimization.**

I. Introduction

In this paper, we propose a new neuron model that is based on a phenomenon found widely in nature, namely *hysteresis*. Recall that hysteresis [31] is defined as a lagging effect due to a change of force acting on a body. Hysteresis manifests itself in the structures of many cooperative dynamical systems (formed of local interactions), and is observed in animals such as frogs [15] and crayfish [38]. Many engineering systems display hysteresis. Among these are mechanical structures subjected to acoustic or aerodynamic loads, a three-phase transformer [32], etc. Marshall *et al.* [30] give a detailed explanation of hysteresis as it occurs in ferromagnetic materials. The identification of hysteretic type nonlinearities is important in earthquake resistant designs of buildings (e.g., [4], [5]). A good background on hysteresis is provided in [44].

Taga [40] developed a network containing six coupled oscillators (that functioned as neurons) to control bipedal locomotion. During locomotion, transitions between movements (walking, running) exhibited hysteresis. Similar hysteretic behavior can also be observed in humans [3]. Hoffman and Benson [23] demonstrated that a single cell-level neuron model, based on an analogy between the immune system and the central nervous system, exhibits hysteresis.

Models of hysteresis appear, for example, in [12], [45], [46], and [8]. Hysteretic neuron models using *signum* functions have been proposed by Yanai and Sawada [49], and Keeler *et al.* [28], for associative memory. They demonstrated that their hysteretic models performed better than nonhysteretic neuron models, in terms of capacity, signal-to-noise ratio, recall ability, etc. Takefuji and Lee [42] proposed a two state (binary) hysteretic neuron model such that: 1) if the input to a neuron exceeds a threshold (upper trip point) the neuron fires (i.e., the output of the neuron is unity); 2) if the input is below a certain threshold (lower trip point), the output of the neuron is zero; and 3) if the input to the neuron is between these trip points, the output equals its previous value.

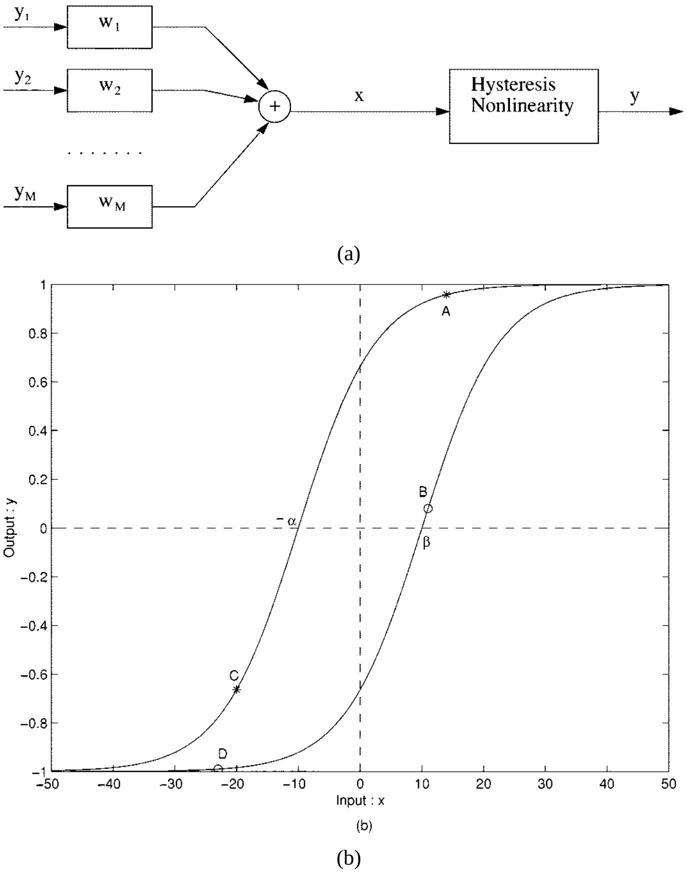

Fig. 1. (a) Hysteretic neuron and (b) hysteretic activation function.

In this paper, we describe an hysteretic neuron model that differs from those in [28], [42], and [49] in the following ways: it 1) is multivalued; 2) has memory; and 3) is adaptive.

In Section II, we present the hysteretic neuron. Section III describes the hysteretic Hopfield neural network (HHNN) along with its circuit dynamical equations. In Section IV, we briefly review the concept of Lyapunov stability for nonlinear dynamical systems We then propose a Lyapunov function and use the Lyapunov theory to prove stability of the HHNN. In Section V, we introduce a well-known combinatorial optimization application, the N-queen problem, and use the HHNN to solve it. Conclusions are drawn in Section VI.

II. Hysteretic Neuron

Our hysteretic neuron [Fig. 1(a)] is similar to other neuron models, in that it processes a linear weighted combination of

Manuscript received March 23, 1998; revised August 24, 1999.

The authors are with the Signal and Image Processing Institute, Department of Electrical Engingering-Systems, University of Southern California, Los Angeles, CA 90089-2564 USA.

Publisher Item Identifier S 1045-9227(00)05883-5.

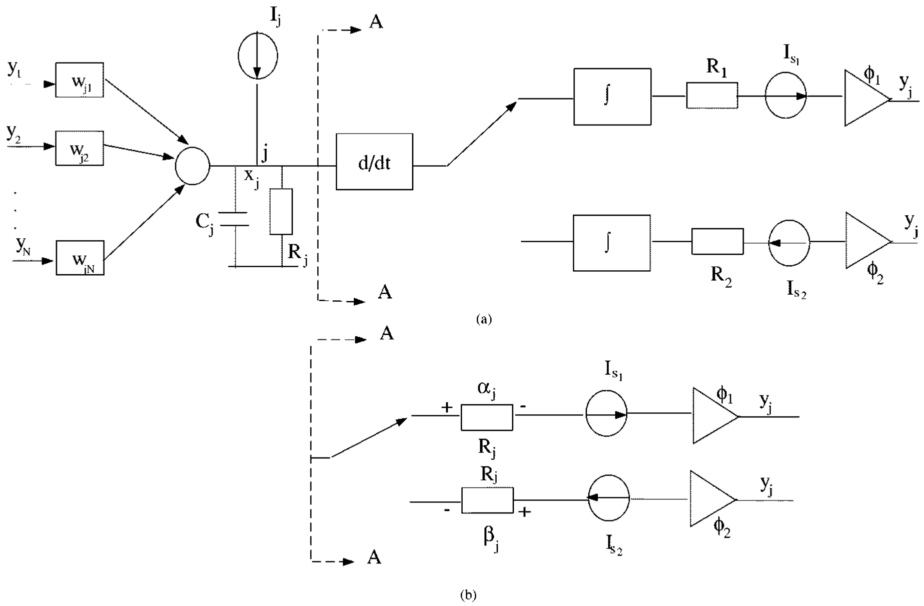

Fig. 2. (a) Hysteresis neuron circuit. (b) Differentiator-integrator pair in series and modeled as net resistance.

inputs. It differs from other neuron models in that its nonlinear gain (activation) function is the hysteresis function depicted in Fig. 1(b). Mathematically, our hysteretic neuron gain function is described as

$$y(x(t) \mid \dot{x}(t-\delta t)) = \phi[x(t) - \lambda(\dot{x}(t-\delta t))] \tag{1}$$

where $\phi(p) = \tanh[\gamma p]$ and

$$\gamma(\dot{x}(t-\delta t)) = \begin{cases} \gamma_\alpha, & \dot{x}(t-\delta t) \geq 0 \\ \gamma_\beta, & \dot{x}(t-\delta t) < 0 \end{cases} \tag{2}$$

$$\lambda(\dot{x}(t-\delta t)) = \begin{cases} -\alpha, & \dot{x}(t-\delta t) \geq 0 \\ \beta, & \dot{x}(t-\delta t) < 0. \end{cases} \tag{3}$$

$\beta > -\alpha$, and, $(\gamma_\alpha, \gamma_\beta) > 0$, and $\dot{x}(t-\delta t) \triangleq dx(t-\delta t)/dt = \lim_{\delta t \to 0}(x(t) - x(t-\delta t))/\delta t$.

The mapping that is effected by this transformation is $y : \Re^2 \to \Re$. Note that, in the special case when $\alpha = \beta$, and $\gamma_\alpha = \gamma_\beta$, the activation function becomes the conventional sigmoid. Observe that this neuron's output not only depends on its input, x, but also on derivative information, namely, $\dot{x}$. It is the latter information that provides the neuron with memory[1] and distinguishes it from other neurons, such as the hysteretic neuron proposed by Takefuji which is memoryless. In their case, the transfer function behaves as a thresholding device. Hence from (1)–(3), if x is positive at one time point and increases in value at the next time point, the activation function remains along segment $C - A$. On the other hand, if x is positive at one time point and decreases at the next time point, then the activation function jumps from hysteretic segment $C - A$ to segment $B - D$.

Note that the hysteretic neuron's activation function has four parameters associated with it, namely, α, β, γ_α, γ_β. Usually, one does not tune a neuron's activation function because, for the most part, there are no parameters to tune (or there is, at most, one parameter, the slope of the sigmoid). The hysteretic neuron is different in this sense, and we can think about tuning all of its parameters in order to maximize its performance. So, it seems that the hysteretic neuron provides us with much more flexibility than the usual neuron.

The hysteretic neuron can be applied to many types of neural networks (multilayer, recurrent). In this paper, we apply it to the Hopfield neural network.

III. Hysteretic Hopfield Neural Network

A circuit-based noiseless dynamical model of the hysteretic neuron is depicted in Fig. 2(a). Important considerations in the design of such an analog circuit are that its individual components have a negligible (but nonzero) propagation time, and the differentiator and integrator have a unity RC-time constant. We denote the voltage at node j as x_j. In Fig. 2(b), we assume a negligible effect of the capacitance, and a high internal resistance of the operational amplifiers in the differentiator-integrator pair. The net resistance in the two branches consisting of current sources $I_{s1} \triangleq \alpha_j/R_j$ and $I_{s2} \triangleq \beta_j/R_j$ is denoted R_j (this resistance can be assumed to be introduced by the differentiator-integrator-$R_1(R_2)$ resistances in series). Accordingly, we can place a resistance of the same value (i.e., R_j) in the branch which is in parallel with the capacitor C_j, as shown in Fig. 2(a). The logic device denoted by an arrow after node j, switches to the upper branch if $\dot{x}_j(t-\delta t) \geq 0$, and to the lower branch if $\dot{x}_j(t-\delta t) < 0$ (here δt denotes the finitely small propagation time involved with respect to node j, before switching). Applying Kirchhoff's current law to node j, we obtain the following circuit equation:

$$C_j \frac{dx_j(t)}{dt} = -\frac{x_j(t)}{R_j} + \frac{\lambda_j(\dot{x}_j(t-\delta t))}{R_j} + \sum_{i=1}^{N} w_{ji} y_i + I_j \quad j = 1, 2, \ldots, N \tag{4}$$

[1] Recall that a system is said to be *memoryless* [33] if its output at any instant depends, at most, on the input at the same instant but not on any past or future values of the input.

where y_i is a shorthand notation for $y_i(x_i \,|\, \dot{x}_i(t-\delta t))$ given by (1), and $\lambda_j(\dot{x}_j(t-\delta t))$ is as defined in (3).

We refer to the interconnection of the N nonlinear equations in (4) as an HHNN. Unlike the usual Hopfield neural network (HNN), the HHNN includes memory.

A discretization of (4) can be accomplished by letting $dx_j(t)/dt = \lim_{\delta t \to 0}(x_j(t+\delta t) - x_j(t))/\delta t$, with $t = kT$, $\delta t = T$, and using a unit sampling interval. In this way, (4) may be used as a discrete update equation for the application described in Section V.

IV. Stability of the HHNN

Stability is an important consideration in the theory of nonlinear systems. Lyapunov, a Russian mathematician from the late nineteenth and early twentieth centuries, developed an approach to stability analysis that is widely used in the control theory literature, and is now known as *the direct method of Lyapunov*. Its key feature is that it leads to conclusions about stability of nonlinear systems without having to explicitly solve the system's nonlinear differential equation. It has also become an important tool for establishing the stability of HNN's [24] or more complicated nonlinear feedback neural networks [14]. We use it to study the stability of the HHNN.

To begin, we recall the definition of a Lyapunov function and stability in the sense of Lyapunov (e.g., [10]). For a function $E(y_1, y_2, \ldots, y_N)$ to be a Lyapunov function, it must satisfy the following three properties: let $\mathbf{y}^* = \mathrm{col}(y_1^*, y_2^*, \ldots, y_N^*)$ be an equilibrium point for a dynamical system; then 1) $E(y_1^*, y_2^*, \ldots, y_N^*) = 0$; 2) $E(y_1, y_2, \ldots, y_N) > 0, \mathbf{y} \neq \mathbf{y}^*$; and 3) $E(y_1, y_2, \ldots, y_N)$ should have partial derivatives with respect to all y_j. Given that $E(y_1, y_2, \ldots, y_N)$ is a Lyapunov function, and if

$$\frac{dE(y_1, y_2, \ldots, y_N)}{dt} \leq 0 \tag{5}$$

then $\mathbf{y}^*$ is stable in the sense of Lyapunov.

The stability of the HHNN can be demonstrated by either of two approaches: 1) use the Cohen–Grossberg theory [14] that is already based on Lyapunov stability theory or 2) a direct approach in which we must establish an energy function, show that it is a Lyapunov function, and then demonstrate the truth of (5) for it. Here we take the second approach (see [6] for the first approach).

We propose the following energy function for the system in (4):

$$\begin{aligned} &E(y_1, y_2, \ldots, y_N) \\ &\quad = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji} y_i y_j + \sum_{j=1}^{N} \frac{1}{R_j} \int_0^{y_j} \phi_j^{-1} \\ &\qquad \times [y_j(x_j \,|\, \dot{x}_j)]\, dy_j - \sum_{j=1}^{N} I_j y_j \end{aligned} \tag{6}$$

where $y_j(x_j \,|\, \dot{x}_j)$ is defined in (1). Details of the proof that $E(y_1, y_2, \ldots, y_N)$ in (6) is indeed a Lyapunov function can be found in [7] and [6]. Here we focus on demonstrating the truth of (5).

Note that

$$\frac{dE}{dt} = \sum_{j=1}^{N} \frac{\partial E}{\partial y_j} \frac{dy_j}{dx_j} \frac{dx_j}{dt}. \tag{7}$$

Assuming symmetric weights and zero diagonal components of the weighting matrix (i.e., $w_{ij} = w_{ji}$ and $w_{ii} = 0$), differentiating (6), and using (1), we find that

$$\begin{aligned} \frac{\partial E}{\partial y_j} &= \frac{x_j}{R_j} - \frac{\lambda_j(\dot{x}_j(t-\delta t))}{R_j} - I_j - \sum_{i=1}^{N} w_{ji} y_i \\ &= -C_j \frac{dx_j}{dt} \end{aligned} \tag{8}$$

where we have also made use of (4). Substituting (8) into (7), we find

$$\frac{dE}{dt} = -\sum_{j=1}^{N} C_j \frac{dy_j}{dx_j} \left(\frac{dx_j}{dt}\right)^2 \triangleq -\sum_{j=1}^{N} \frac{dE_j}{dt}. \tag{9}$$

Consider the jth neuron [each neuron contributes independently toward the minimization of the energy function, as can be seen from (9)]. Accordingly, from (9)

$$\frac{dE_j}{dt} = \lim_{\Delta x_j \to 0} C_j \frac{\Delta y_j}{\Delta x_j} \left(\frac{dx_j}{dt}\right)^2. \tag{10}$$

We analyze (10) by considering the following transitions [see Fig. 1(b)]: 1) Transition 1 $(A \to B)$ for which $\Delta x_j < 0 \Rightarrow \Delta y_j < 0 \Rightarrow (dE_j/dt) > 0, j = 1, 2, \ldots, N \Rightarrow (dE/dt) < 0$; 2) Transition 2 $(B \to A)$ for which $\Delta x_j > 0 \Rightarrow \Delta y_j > 0 \Rightarrow (dE_j/dt) > 0, j = 1, 2, \ldots, N \Rightarrow (dE/dt) < 0$; and 3) Transition 3 $(C \to A, B \to D)$ for which transitions occur along the sigmoid, which is a nondecreasing function i.e., for which $(dE_j/dt) > 0, j = 1, 2, \ldots, N \Rightarrow (dE/dt) < 0$.

Because $dE/dt < 0$ for all possible transitions, we have shown that the equilibrium point for the HHNN, described by (1)–(4), is stable in the sense of Lyapunov. □

V. The N-Queen Problem

In this section we describe an application of the HHNN to a well-known combinatorial optimization problem, the N-queen problem. According to Wirth [47], "…the problem of the eight queens is a well known example of the use of trial and error methods and of backtracking algorithms. It was investigated by C. F. Gauss in 1850, but he did not completely solve it. The characteristic property of these problems is that they defy analytic solution. Such algorithms have therefore gained relevance almost exclusively through the automatic computer, which possesses these properties to a much higher degree than people, even geniuses, do." This quote typifies the prevalent view of the eight queen problem, which was first posed in 1848, and investigated by several 19th century mathematicians.

A general N-queen problem is defined by the following constraints on an $N \times N$ grid: 1) only one queen can be placed in any row; 2) only one queen can be placed in any column; 3) only one queen can be placed on any diagonal; and 4) exactly N-queens must be placed on the grid. This problem was explored in the 1950's by Yaglom and Yaglom [48]. Since then, there have been diverse approaches taken in the study of this problem, including algorithmic design, program development, parallel and distributed computing, and artificial intelligence ([1], [25], [26], [19], [16], [37], [47], [17], [18], [20], [21], [35], [9], [39], [27], [34], [42], [29], [2]). The eminent mathematician Pólya [36] examined a solution to a constrained N-queen problem. Chandra [11] developed the theory of independent permutations to characterize the family of solutions to the N-superqueen problem. This widespread interest in the N-queen problem is due in part to an aspect that often characterizes difficult problems, namely a set of global constraints. For example, the very moment that one queen is placed on an $N \times N$ grid, the number of positions the next queen can be placed on is significantly reduced.

A. Mathematical Model

The following mathematical model for the N-queen problem was introduced by Takefuji [41]. Let us denote the state of the existence of a queen at the ijth location on an $N \times N$ grid as y_{ij}. If a queen exists at the ijth location, then $y_{ij} = 1$, otherwise $y_{ij} = 0$. The nonlinear state equations for the N-Queen problem are

$$\begin{aligned}\frac{dx_{ij}}{dt} = &-\left(\sum_{k=1}^{N} y_{ik} - 1\right) - \left(\sum_{k=1}^{N} y_{kj} - 1\right) + \varrho\left(\sum_{k=1}^{N} y_{ik}\right)\\ &+ \varrho\left(\sum_{k=1}^{N} y_{kj}\right) - \left(\sum_{\substack{1\le i-k;\ j-k\le N\\ k\neq 0}} y_{i-k,j-k}\right)\\ &- \left(\sum_{\substack{1\le i-k; j+k\le N\\ k\neq 0}} y_{i-k,j+k}\right);\quad i,j=1,2,\ldots,N\end{aligned} \tag{11}$$

where $y_{ij} \equiv 0.5\,\mathrm{sgn}x_{ij} + 0.5$, x_{ij} is the input to the ijth neuron, and $\varrho(\cdot)$ is the Kronecker-delta function, i.e., $\varrho(\tau) = \{1, \tau = 0;\ 0, \tau \neq 0\}$. In (11), the first term is a row constraint that forces only one queen to be placed in a row; the second term is a column constraint that forces only one queen per column; the third and fourth terms are perturbation (hill-climbing) terms that have been included so that the state of the system can escape local minima (when no neuron is fired in the ith row or the jth column); and, the last two terms correspond to diagonal constraints, i.e., no pair of queens can diagonally command each other.

We have modified this model to one in which y_{ij} is given by the following hysteretic function: Note that this hysteretic function differs from the one in (1), because y_{ij} must, in this application, be zero or one. Now, in (11), y_{ij} is short for $y_{ij}(x_{ij} \,|\, \dot{x}_{ij})$ (where, $\dot{x}_{ij} = \lim_{\delta t \to 0}(x_{ij}(t) - x_{ij}(t - \delta t)/\delta t)$), as is clear from (12), shown at the bottom of the page.

We must show that (11) reduces to the correct form for the hysteretic Hopfield circuit dynamical equation, so that it inherits the Lyapunov stability property, proven in the previous section.

For our application involving hard-limiting functions (i.e., two state neurons) as given in (12), the second term of (6) becomes negligibly small [22], [13] due to the large gain in the neuron's activation function. The corresponding energy (Lyapunov) function reduces to

$$\begin{aligned}&E(y_1, y_2, \ldots, y_N)\\ &\quad = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji} y_i y_j - \sum_{j=1}^{N} I_j y_j\end{aligned} \tag{13}$$

and (4) can be expressed as

$$\begin{aligned}\frac{dx_j}{dt} = &\underbrace{-\frac{1}{R_j C_j}[x_j - \lambda_j(\dot{x}_j(t - \delta t))]}_{\text{Term I}}\\ &\underbrace{-\frac{1}{C_j}\frac{\partial E(y_1, y_2, \ldots, y_N)}{\partial y_j}}_{\text{Term II}} \quad j = 1, 2, \ldots, N.\end{aligned} \tag{14}$$

Term I in (14) is the decay term (with $\tau_j = R_j C_j$ time constant); it may, as we explain next, cause an increase in the time derivative of the energy function. Recall, that

$$\frac{dE}{dt} = \sum_{j=1}^{N} \frac{\partial E}{\partial y_j}\frac{dy_j}{dx_j}\frac{dx_j}{dt}. \tag{15}$$

Solving (14) for Term II and substituting this into (15), we see that

$$\begin{aligned}\frac{dE}{dt} = &\underbrace{-\sum_{j=1}^{N}\frac{1}{R_j}[x_j - \lambda_j(\dot{x}_j)]\frac{dy_j}{dx_j}\frac{dx_j}{dt}}_{\text{Term A}}\\ &\underbrace{-\sum_{j=1}^{N} C_j \frac{dy_j}{dx_j}\left(\frac{dx_j}{dt}\right)^2}_{\text{Term B}}.\end{aligned} \tag{16}$$

In Section IV, we showed that *Term B* in (16) is always positive [by analyzing the finite differences in the input–output signals of a neuron (10)]; however, *Term A* can be positive or negative, so that dE/dt may not be negative (in which case the proposed energy function is no longer a valid Lyapunov function). One solution to this, that has been considered in [43], is to eliminate *Term I* in (14), which eliminates *Term A* in (16). This can be accomplished by using a sufficiently large resistance and capacitance (e.g., $R_j = 100$ Mohm, $C_j = 1$ F). Doing this, reduces (16) to

$$\frac{dE}{dt} \approx -\sum_{j=1}^{N} C_j \frac{dy_j}{dx_j}\left(\frac{dx_j}{dt}\right)^2 < 0 \tag{17}$$

$$y_{ij} = \begin{cases} 0.5\,\mathrm{sgn}(x_{ij} + \alpha_{ij}) + 0.5 & \dot{x}_{ij} \geq 0\\ 0.5\,\mathrm{sgn}(x_{ij} - \beta_{ij}) + 0.5 & \dot{x}_{ij} < 0\end{cases};\quad i, j = 1, \ldots, N. \tag{12}$$

thereby demonstrating Lyapunov stability for (14). In this case, (14) (using (13)) reduces to

$$C_j \frac{dx_j}{dt} = \sum_{i=1}^{N} w_{ji} y_i + I_j \quad j = 1, 2, \ldots, N. \tag{18}$$

Next, we proceed to show that (11) can be expressed in the form of a two-dimensional version of (18), as

$$C_{ij} \frac{dx_{ij}}{dt} = \sum_{k=1}^{N} \sum_{l=1}^{N} w_{ij,kl} y_{kl} + I_{ij}. \tag{19}$$

Comparing (11) and (19), the two are equivalent when

$$\begin{aligned} C_{ij} &= 1; \quad i, j = 1, 2, \ldots, N \\ w_{ij,kl} &= -\delta_{i,k} - \delta_{j,l} - (\delta_{i+j,k+l} + \delta_{i-j,k-l}) \\ &\quad \times (1 - [\delta_{i,k} + \delta_{j,l} - \delta_{i,k}\delta_{j,l}]) \\ I_{ij} &= \varrho \left(\sum_{k=1}^{N} y_{ik} \right) + \varrho \left(\sum_{k=1}^{N} y_{kj} \right) + 2, \\ &\qquad i, j = 1, 2, \ldots, N \end{aligned} \tag{20}$$

where $\delta_{r,s}$ is a two-dimensional Kronecker delta, defined as follows:

$$\delta_{r,s} = \begin{cases} 1, & r = s \\ 0, & r \neq s \end{cases} \tag{21}$$

In the equation for $w_{ij,kl}$, the first two terms represent the row and column constraints respectively, whereas the third term contains the two diagonal constraints. It can also be easily demonstrated that $w_{ij,kl} = w_{kl,ij}$. So, (11), which is the basis for a HHNN solution of the N-queen problem, reduces to a HHNN.

The equilibrium point for the nonlinear system in (11) occurs when $dx_{ij}/dt = 0$. A stable and valid equilibrium point occurs when all the constraints are satisfied simultaneously, i.e., when terms 1, 2, 5, and 6 in (11) equal zero (i.e., when one neuron is fired per row, per column, and per queen diagonal). This in turn implies that the hill-climbing terms (3 and 4) will be zero, since terms 1 and 2 are zero. At the equilibrium point, the output of the HHNN (i.e., y_{ij}) will be either a one or a zero.

B. Tuning of HHNN Parameters

For an N-queen problem that uses the hysteretic activation function, there is a high degree of flexibility in deciding the choice for the $2N^2$ network parameters (i.e., $\{\alpha_{ij}, \beta_{ij}\} \in \Re$; $i, j = 1, 2, \ldots, N$). Instead of performing an exhaustive search over all of these parameters, which would be prohibitive, we used a gradient descent procedure to optimize the $(\alpha_{ij}, \beta_{ij})$ pairs. Using the notation from (3), our gradient descent algorithm has the following form:

$$\begin{aligned} \lambda_{ij}^{l+1}(\dot{x}_{ij}) &= \lambda_{ij}^{l}(\dot{x}_{ij}) + \Delta\lambda_{ij}^{l}(\dot{x}_{ij}) \\ &= \lambda_{ij}^{l}(\dot{x}_{ij}) + \eta \frac{\partial E}{\partial \lambda_{ij}^{l}(\dot{x}_{ij})} \\ &= \lambda_{ij}^{l}(\dot{x}_{ij}) + \eta \frac{\partial E}{\partial y_{ij}} \frac{dy_{ij}}{d\lambda_{ij}^{l}(\dot{x}_{ij})} \end{aligned} \tag{22}$$

where E is the energy function in (13). Substituting the right-hand portion of (8) into (22) (replacing the single subscripted quantities in (8) by their appropriate double-subscripted counterparts, and, assuming $C_{ij} = 1, \forall i, j$) we have

$$\lambda_{ij}^{l+1}(\dot{x}_{ij}) = \lambda_{ij}^{l}(\dot{x}_{ij}) - \eta \frac{dx_{ij}}{dt} \frac{dy_{ij}}{d\lambda_{ij}^{l}(\dot{x}_{ij})}. \tag{23}$$

Unfortunately, we cannot compute $dy_{ij}/d\lambda_{ij}^{l}(\dot{x}_{ij})$ using (12), because of the discontinuity of the step function. Instead, we use the continuous version of the step function, viz., the sigmoid, since the step function is just the limiting form of the sigmoid when the slope of the sigmoid $\gamma \rightarrow \infty$ [i.e., $y = 1/(1 + e^{-\gamma x})$, and $y = 0.5$ when $x = 0$]. We, therefore, use sigmoids instead of the signum functions in (11), but only to approximate the derivatives that are needed in (23), i.e.,

$$y_{ij} \approx \begin{cases} 0.5 \tanh\left[\gamma_{ij}^{\alpha}(x_{ij} + \alpha_{ij})\right] + 0.5 & \dot{x}_{ij} \geq 0 \\ 0.5 \tanh\left[\gamma_{ij}^{\beta}(x_{ij} - \beta_{ij})\right] + 0.5 & \dot{x}_{ij} < 0. \end{cases} \tag{24}$$

It is straightforward to show, using (24) and (23), that

$$\begin{aligned} &\lambda_{ij}^{l+1}(\dot{x}_{ij}) \\ &\quad = \lambda_{ij}^{l}(\dot{x}_{ij}) - \eta \frac{dx_{ij}}{dt} \frac{\gamma_{ij}(\dot{x}_{ij})}{\cosh^2[\gamma_{ij}(x_{ij} - \lambda_{ij}(\dot{x}_{ij}))]} \end{aligned} \tag{25}$$

where $\gamma_{ij}(\dot{x}_{ij})$ is defined in (2).

The steady-state solution of (25) is achieved when the system in (11) converges to a solution i.e., when $dx_{ij}/dt = 0, \forall i, j$.

Using (2) and (3), (25) can be expressed in terms of the parameters α_{ij} and β_{ij}, as

$$\begin{aligned} \alpha_{ij}(l+1) &= \alpha_{ij}(l) + \eta_1 \left| \frac{dx_{ij}}{dt} \right| \\ &\quad \times \frac{\gamma_{ij}^{\alpha}(l)}{\cosh^2\left[\gamma_{ij}^{\alpha}(l)(x_{ij} + \alpha_{ij}(l))\right]}; \quad \frac{dx_{ij}}{dt} \geq 0 \\ \beta_{ij}(l+1) &= \beta_{ij}(l) + \eta_2 \left| \frac{dx_{ij}}{dt} \right| \\ &\quad \times \frac{\gamma_{ij}^{\beta}(l)}{\cosh^2\left[\gamma_{ij}^{\beta}(l)(x_{ij} - \beta_{ij}(l))\right]}; \quad \frac{dx_{ij}}{dt} < 0 \\ &\qquad i, j = 1, 2, \ldots, N. \end{aligned} \tag{26}$$

The approximation in (24) introduces two positive slope parameters, γ_{ij}^{α} and γ_{ij}^{β}, which can also be tuned. In order to perform gradient descent on these parameters, we introduce an intermediate parameter τ so as to prevent them from becoming negative. With $\tau = \sqrt{\gamma}$, our gradient descent algorithm for the slopes can be written, as

$$\begin{aligned} \tau_{ij}^{l+1}(\dot{x}_{ij}) &= \tau_{ij}^{l}(\dot{x}_{ij}) + \Delta\tau_{ij}^{l}(\dot{x}_{ij}) \\ &= \tau_{ij}^{l}(\dot{x}_{ij}) - \eta \frac{\partial E}{\partial y_{ij}} \frac{dy_{ij}}{d\tau_{ij}^{l}(\dot{x}_{ij})} \\ &= \tau_{ij}^{l}(\dot{x}_{ij}) + \eta \frac{dx_{ij}}{dt} \\ &\quad \times \frac{\tau_{ij}^{l}(\dot{x}_{ij})\left(x_{ij} - \lambda_{ij}^{l}(\dot{x}_{ij})\right)}{\cosh^2\left[\left(\tau_{ij}^{l}(\dot{x}_{ij})\right)^2\left(x_{ij} - \lambda_{ij}^{l}(\dot{x}_{ij})\right)\right]} \end{aligned} \tag{27}$$

thus

$$\begin{aligned}
&\tau_{ij}^{\alpha}(l+1)\\
&\quad = \tau_{ij}^{\alpha}(l) + \eta_3 \frac{dx_{ij}}{dt}\\
&\quad\times \frac{\tau_{ij}^{\alpha}(l)(x_{ij}+\alpha_{ij}(l))}{\cosh^2\left[\left(\tau_{ij}^{\alpha}(l)\right)^2 (x_{ij}+\alpha_{ij}(l))\right]}; \quad \frac{dx_{ij}}{dt} \geq 0\\
&\tau_{ij}^{\beta}(l+1)\\
&\quad = \tau_{ij}^{\beta}(l) + \eta_4 \frac{dx_{ij}}{dt}\\
&\quad\times \frac{\tau_{ij}^{\beta}(l)(x_{ij}-\beta_{ij}(l))}{\cosh^2\left[\left(\tau_{ij}^{\beta}(l)\right)^2 (x_{ij}-\beta_{ij}(l))\right]}; \quad \frac{dx_{ij}}{dt} < 0\\
&\qquad\qquad\qquad\qquad j = 1, 2, \ldots, N. \qquad (28)
\end{aligned}$$

In summary, we compute $\alpha_{ij}, \beta_{ij}, \tau_{ij}^{\alpha}$ and τ_{ij}^{β} using (26) and (28). We substitute the α_{ij} and β_{ij} values into (12), and the resulting quantity is used in (11) to compute dx_{ij}/dt. Note that we use (12) and not the approximation in (24) in this step, since the constraints for (11) are satisfied only at the 2^{N^2} vertices of a hypercube, i.e., for $y_{ij} = 1$ or $y_{ij} = 0$. We use (24) only to derive (26) and (28), and τ_{ij}^{α} and τ_{ij}^{β} are in turn only used to update α_{ij} and β_{ij} in (26).

C. Results

Associated with (11) is an energy function as defined in (13), but with the weights and current sources appropriately defined in (20). We refer to (13) as a "modified" energy function, because it accounts for all six terms on the right-hand side of (11), including terms 3 and 4, which were added in by Takefuji et al. [41] for the reason given below (11). The energy function that is associated with the version of (11) that only contains terms 1, 2, 5, and 6 on its right-hand side is referred to by us as a "standard" energy function. We have compared the performance of our HHNN with that of the HNN for both the standard as well as modified energy functions. In addition, we have performed many experiments with the HHNN by tuning its available parameters.

Tables I and II provide a summary of our simulation results. The traditional HNN does not have any tunable parameters for either energy function. The initializations for the HNN and the HHNN model were the same, i.e., the initial values for the inputs to all neurons, for the $N = 30$-queen and the $N = 50$-queen problems, were $x_{ij}(0) \in [-1, 1]$ (uniform), $i, j = 1, 2, \ldots, N$

The differential equations corresponding to the modified energy function and the standard energy function, were discretized. We then performed a Monte Carlo simulation with 100 randomized initializations on the inputs to the neurons. Each number within the tables indicates the percentage of solutions obtained from 100 random initializations, i.e., the percentage of convergence to a solution. Our stopping rule was: terminate if a solution is obtained or if 1000 iterations have occurred, where an iteration refers to one time-step for the implementation of the discretized version of (11), or its standard energy function counterpart.

TABLE I
SUMMARY OF EXPERIMENTAL RESULTS FOR $N = 30$-QUEEN PROBLEM. ANN: ARTIFICIAL NEURAL NETWORK; HNN: HOPFIELD NEURAL NETWORK; HHNN: HYSTERETIC HOPFIELD NEURAL NETWORK

ANN	Standard energy function		Modified energy function	
	No tuning	Tuning	No tuning	Tuning
HNN	45%	☒	66%	☒
HHNN	56%	58%*	55%	73%*
		55%**		68%**

* All parameters were tuned using gradient descent.
** Only (α,β) parameters were tuned using gradient descent.

TABLE II
SUMMARY OF EXPERIMENTAL RESULTS FOR $N = 50$-QUEEN PROBLEM. ANN: ARTIFICIAL NEURAL NETWORK; HNN: HOPFIELD NEURAL NETWORK; HHNN: HYSTERETIC HOPFIELD NEURAL NETWORK

ANN	Standard energy function		Modified energy function	
	No tuning	Tuning	No tuning	Tuning
HNN	61%	☒	83%	☒
HHNN	71%	72%*	78%	90%*
		63%**		82%**

* All parameters were tuned using gradient descent.
** Only (α,β) parameters were tuned using gradient descent.

Note that, when the HHNN is not tuned, y_{ij} are computed using (12), so that the HHNN is characterized just by the α_{ij} and β_{ij} parameters. For the HHNN results related to "No tuning" (Standard or Modified energy functions), we scanned over the α_{ij}, β_{ij} parameters (constraining $\beta_{ij} > -\alpha_{ij}$) looking for the values that yielded the best convergence percentage, as follows.

1) We chose $\alpha_{ij} = \omega_1 \in [-4, 4]$ and $\beta_{ij} = \omega_2 \in [-4, 4]$, $\forall i, j = 1, 2, \ldots, N$, using only integer values. All possible integer-valued pairs were chosen subject to $\omega_2 > -\omega_1$ (a total of 36 pairs).
2) 100 random initializations of $x_{ij}(0)$ were used for each (ω_1, ω_2).
3) For each (ω_1, ω_2), we recorded the convergence percentage averaged over the 100 random initializations of $x_{ij}(0)$.
4) Each result shown in Tables I and II, for the HHNN "No tuning" cases, is the maximum average convergence percentage over all (ω_1, ω_2).

Each neuron in the "tuned" HHNN has four tunable parameters $(\alpha_{ij}, \beta_{ij}, \tau_{ij}^{\alpha}, \tau_{ij}^{\beta})$, as described in the previous section.

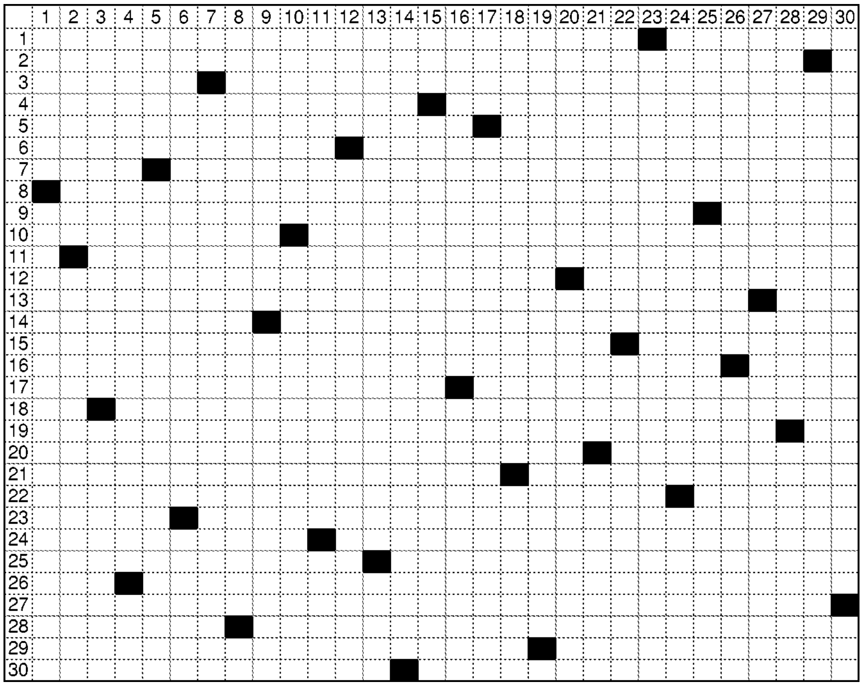

Fig. 3. Solution for $N = 30$, obtained at iteration $= 33$.

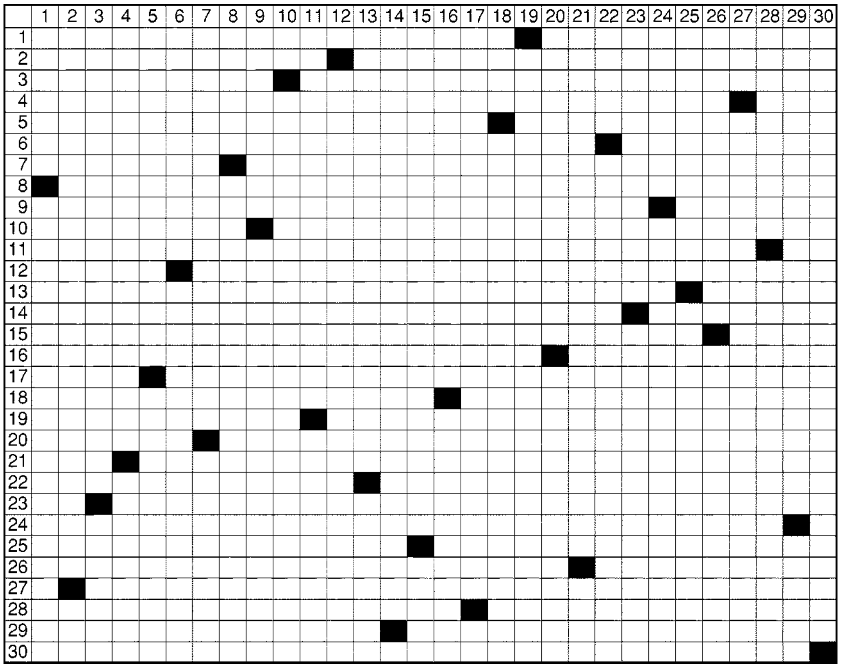

Fig. 4. Solution for $N = 30$, obtained at iteration $= 29$.

These had to be initialized. Rather than randomly initializing these parameters (which would add further complexity to the Monte Carlo simulations), we decided to start the HHNN as an HNN; thus, we chose: $\alpha_{ij}(0) = \beta_{ij}(0) = 0, i, j = 1, 2, \ldots, N$; and $\tau_{ij}^{\alpha}(0) = \tau_{ij}^{\beta}(0) = \vartheta_{ij}$, $\vartheta_{ij} \in (0, 1)$ (uniform).

The gradient descent algorithms in (23) and (25) also required learning rates $\eta_1, \eta_2, \eta_3, \eta_4$, which we chose as: $\{\eta_1, \eta_2, \eta_3, \eta_4\} \in (0, 0.5)$. Examples of typical learning rates that we used for the modified energy function, and $N = 50$ are: $\eta_1 = 0.02$, $\eta_2 = 0.03$, $\eta_3 = 0.008$, $\eta_4 = 0.001$. We noticed that the convergence performance was more sensitive to the learning rates for the slopes than to the learning rates of the crossover points $(\alpha_{ij}, \beta_{ij})$. We believe that this is due to the direct proportionality of the change of the slope to the state of the neuron [see (27), where $\Delta\tau_{ij}(l + 1) \propto x_{ij}$].

There are many possible solutions to an N-queen problem. We were only interested in converging to any one of them. Two possible solutions that we converged to, for $N = 30$ and $N = 50$, are displayed in Figs. 3–6. They were obtained for the fully tuned HHNN.

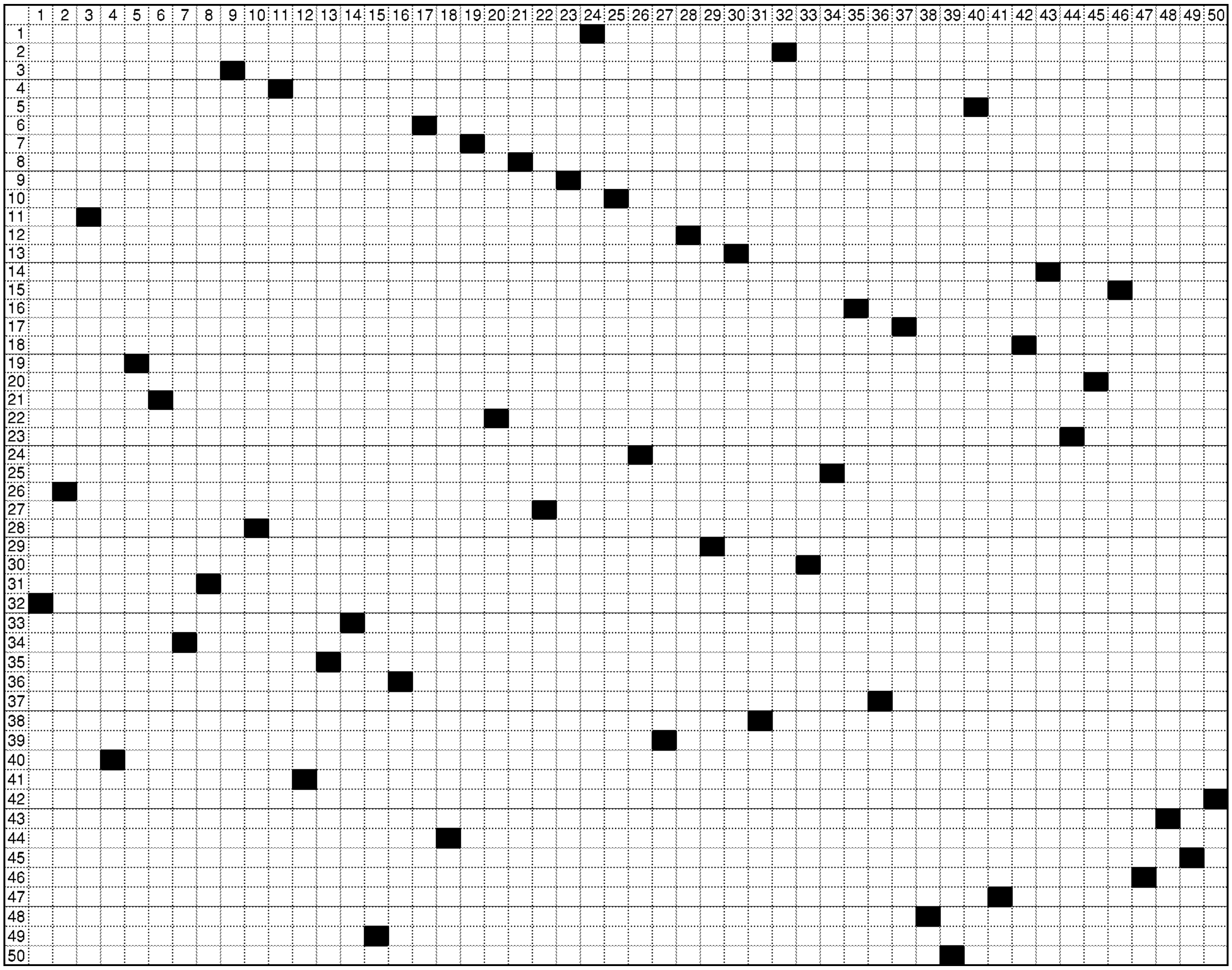

Fig. 5. Solution for $N = 50$, obtained at iteration $= 62$.

The following observations are made from Tables I and II.

1) We reconfirmed the results in [41], that much better results are obtained for the HNN when one uses the dynamical equation of the modified energy function [for $N = 30$, 66% versus 45%, and, for $N = 50$, 83% versus 61%].

2) Tuning of the HHNN on all its parameters for the modified energy function gives the overall best results [for $N = 30$, 73%, and for $N = 50$, 90%].

3) Tuning of the HHNN just on the $(\alpha_{ij}, \beta_{ij})$ parameters for the modified energy function may not necessarily give better results than those obtained using the HNN [e.g., for $N = 50$, 82% versus 83%].

4) The untuned HHNN outperformed the HNN for the standard energy function [for $N = 30$, 56% versus 45%, and, for $N = 50$, 71% versus 61%].

5) The HNN outperformed the untuned HHNN for the modified energy function [for $N = 30$, 66% versus 55%, and, for $N = 50$, 83% versus 78%]. This may be due to the way we chose the α_{ij} and β_{ij} parameters for the HHNN, as described above.

Best results were obtained by using gradient descent on all the parameters of the HHNN, because of the availability of additional parameters in the activation function of the neuron. This improvement in performance occurs at the expense of an increase in computation. Note that all the quantities needed for computing (26) and (28) are available from (11); moreover, (11) has the highest computational cost, compared to (26) and (28). Interestingly enough, we observed experimentally, that performing a full-blown gradient descent for the HHNN resulted in a faster convergence to a solution than that obtained for the HNN. So, even though computation for each iteration of a fully tuned HHNN is greater than that of a HNN, tuning frequently gets the HHNN to a solution in fewer iterations than it takes for the HNN to reach a solution.

Finally, an associated experimental observation for the current application is that, system performance (in terms of convergence percentage) improved for hysteretic band sizes less than three; however larger band sizes had a detrimental effect on performance. A possible explanation for this can be seen with the aid of Fig. 1(b) [in which $\gamma^{\alpha} = \gamma^{\beta} = \infty$, as is the case for the activation functions in (12)]. Consider the input to a neuron to be in the saturation region of the top function, i.e., near region A. A change in direction to the input of a neuron at this point causes the output to switch from A to the other saturation region marked D. This rapid switching in the neuron's output

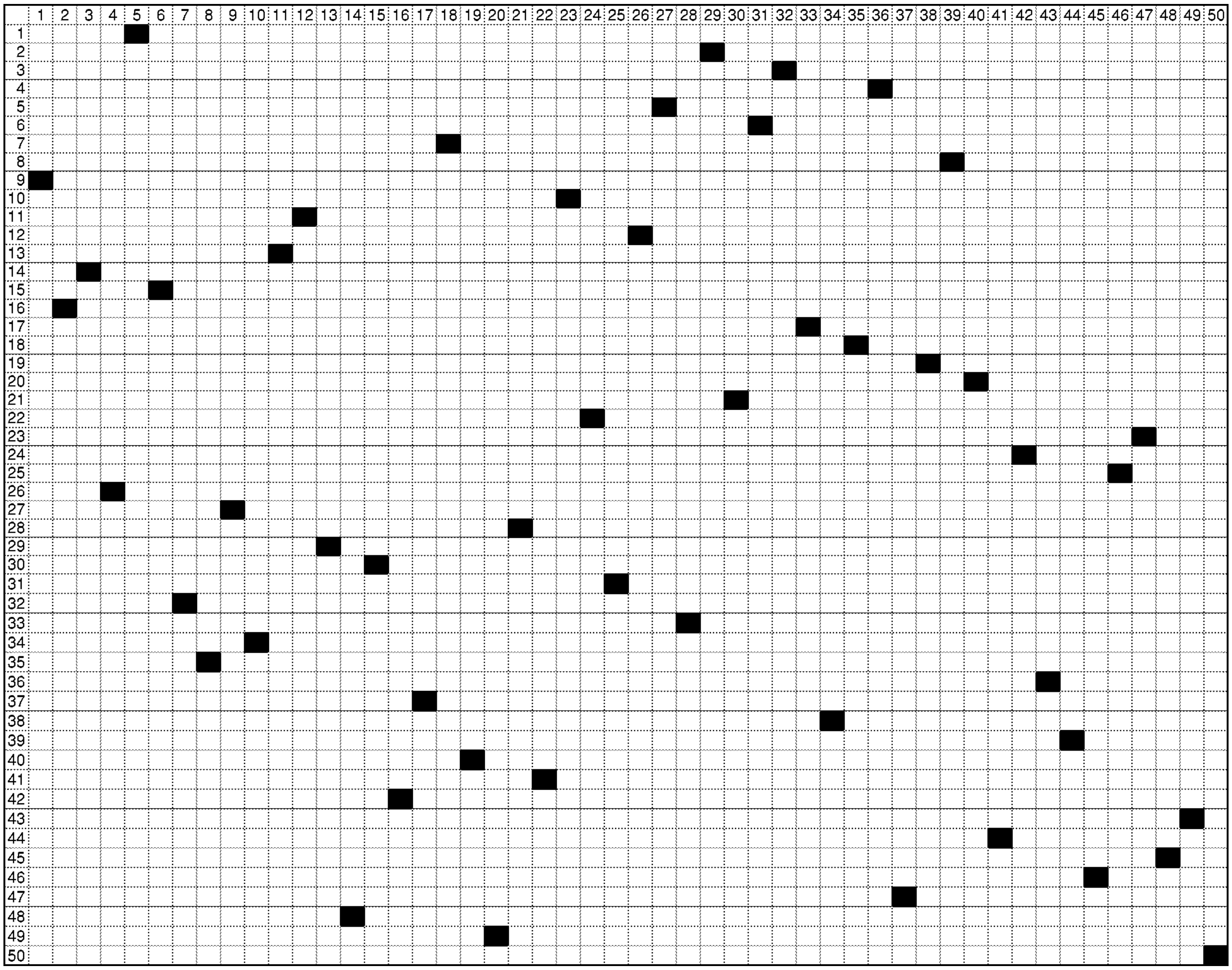

Fig. 6. Another solution for $N = 50$, obtained at iteration $= 47$.

could very well cause the system to jump over local minima. Larger hysteretic bands cause relatively larger gradients in (28), which accentuate this effect.

VI. Conclusion

In this paper, we have introduced the hysteretic activation function which is multivalued, has memory, and is adaptive. We have also introduced the HHNN and its associated circuit model, proved Lyapunov stability for the HHNN, and used the HHNN to solve the N-queen problem. Usually, one does not tune a HNN. It is straightforward to tune a HHNN on its free parameters $\alpha_{ij}, \beta_{ij}, \gamma_{ij}^{\alpha}, \gamma_{ij}^{\beta}$.

The usual sigmoid activation function has a very small gradient in its nonlinear saturation portion, as compared to its linear portion. This may cause a system using the usual sigmoid to get stuck in minima if some or all the sigmoids prematurely saturate. On the other hand, the hysteretic activation function has a tendency to overcome local minima. This can be seen with the aid of Fig. 1(b). Due to a change in the direction of the input, a system can pull itself out of a saturated region by jumping from one segment of the hysteretic activation function to the other segment. Finally, we wish to reemphasize the fact that our hysteretic neuron can be used in other types of neural networks, such as a feedforward or recurrent network. Our present work, in which we do this, is directed at identification of systems that contain hysteresis (e.g., mechanical structures).

References

[1] B. Abramson and M. Yung, "Divide and conquer under global constraints: A solution to the N-queens problem," *J. Parallel Distrib. Comput.*, vol. 6, pp. 649–662, 1989.

[2] Y. Akiyama, A. Yamashita, M. Kajiura, and H. Aiso, "Combinatorial optimization with Gaussian machines," in *Proc. Int. Joint Conf. Neural Networks*, May 1989.

[3] R. M. Alexander, "Optimization and gaits in the locomotion of vertebrates," *Phys. Rev.*, vol. 69, pp. 1199–1227, 1989.

[4] A. M. Andronikou, G. A. Bekey, and F. Y. Hadaegh, "Identifiability of nonlinear systems with hysteretic elements," *J. Dyn. Syst., Measurement, Contr.*, vol. 105, pp. 209–214, 1983.

[5] T. Balendra, *Vibration of Buildings to Wind and Earthquake Loads*. New York: Springer-Verlag, 1993.

[6] S. Bharitkar and J. M. Mendel, "Theoretical Observations of the Hysteretic Hopfield Neural Network,", SIPI (USC-EE) Tech. Rep. 322, 1998.

[7] ——, "The hysteretic hopfield neural network," presented at the Proc. IEEE World Congr. Comput. Intell. '98, Anchorage, AK, May 1998.

[8] G. Biorci and A. Ferro, "Hysteresis losses along open transformations," *J. Phys. Radium*, vol. 20, pp. 237–240, 1959.

[9] J. Bitner and E. M. Reingold, "Backtrack programming techniques," *Commun. ACM*, vol. 18, pp. 651–655, 1975.

[10] W. L. Brogan, *Modern Control Theory*. New York: Quantum, 1974.

[11] A. K. Chandra, "Independent permutations, as related to the problem of Moser and a theorem of Pólya," *J. Combin. Theory*, vol. 16, no. 1, pp. 111–120, Jan. 1974.

[12] L. O. Chua and S. C. Bass, "A generalized hysteresis model," *IEEE Trans. Circuit Theory*, vol. CT-19, no. 1, pp. 36–48, 1972.

[13] A. Cichocki and R. Unbehauen, *Neural Networks for Optimization and Signal Processing*. Berlin, Germany: Springer-Verlag, 1995.

[14] M. A. Cohen and S. Grossberg, "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, pp. 815–825, 1983.

[15] R. L. Didday, "A model of visuomotor mechanisms in the frog optic tectum," *Math. Biosci.*, vol. 30, pp. 169–180, 1976.

[16] E. W. Dijkstra, *Notes on Structerd Programming*, O. J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, Eds. New York: Academic, 1972.

[17] R. E. Filman and D. P. Friedman, *Coordinated Computing: Tools and Techniques for Distributed Software*. New York: McGraw-Hill, 1984.

[18] R. Finkel and U. Manber, "DIB-A distributed implementation of backtracking," *ACM Trans. Program. Lang. Syst.*, vol. 9, pp. 235–256, 1987.

[19] R. W. Floyd, "Nondeterministic algorithms," *J. Assoc. Comput. Machinery*, vol. 14, pp. 636–644, 1967.

[20] J. Gaschnig, "Performance Measurement and Analysis of Certain Search Algorithms," Ph.D. dissertation, Carnegie Mellon Univ., Pittsburge, PA, May 1979.

[21] R. M. Haralick and G. L. Elliot, "Increasing tree search efficiency for constraint satisfaction problems," *Artif. Intell.*, vol. 14, pp. 263–313, Oct. 1980.

[22] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ: Macmillan, 1994.

[23] G. W. Hoffman and M. W. Benson, "Neurons with hysteresis form a network that can learn without any changes in synaptic strengths," in *Proc. Amer. Inst. Phys. Conf. Neural Networks Comput.*, J. S. Denker, Ed., 1986, pp. 219–226.

[24] J. J. Hopfield, "Neurons, dynamics, and computations," *Phys. Today*, vol. 47, no. 2, pp. 40–46, Feb. 1994.

[25] E. Horowitz and S. Sahni, *Fundamentals of Computer Algorithms*. Rockville, MD: Computer Science, 1978.

[26] E. Horowitz and A. Zorat, "Divide and conquer for parallel processing," *IEEE Trans. Comput.*, vol. C-32, pp. 582–585, June 1983.

[27] L. V. Kale, "An almost perfect heuristic for the N nonattacking queens problem," *Inform. Process. Lett.*, vol. 34, pp. 173–178, 1990.

[28] J. D. Keeler, E. E. Pichler, and J. Ross, "Noise in neural networks: Thresholds, hysteresis and neuromodulation of signal-noise," *Proc. Nat. Acad. Sci. USA*, vol. 86, pp. 1712–1716, 1989.

[29] J. Mańdziuk and B. Macukow, "A neural network designed to solve the N-queens problem," *Biol. Cybern.*, vol. 66, pp. 375–379, 1992.

[30] S. V. Marshall, R. E. DuBroff, and G. G. Skitek, *Electromagnetic Concepts and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[31] *Webster's Ninth New Collegiate Dictionary*. Springfield, MA: Merriam-Webster, Inc., 1990.

[32] E. Napieralska, J. R. Grzybowski, and J. F. Brudny, "Modeling of losses due to eddy currents and histeresis in converter transformer cores during failure," *IEEE Trans. Magn.*, vol. 31, pp. 1718–1721, 1995.

[33] A. Oppenheim and R. Schaefer, *Discrete-Time Signal Processing*. Englewood-Cliffs, NJ: Prentice-Hall, 1989.

[34] E. Page and G. A. Tagliarini, "Solving constraint satisfaction problems with neural networks," in *Proc. IEEE Int. Conf. Neural Networks*, vol. III, 1987, pp. 741–747.

[35] J. Pearl, *Heuristics*. Reading, MA: Addison-Wesley, 1984.

[36] G. Pólya, "Uber die 'doppelt-periodischen' Losungen des n-Damen-Problems," in *Mathematische Unterhaltungen und Spiele*, W. Ahrens, Ed. Leipzig, Germany: Teubner, 1918, pp. 364–374.

[37] J. T. Schwartz, R. B. K. Dewar, E. Dubiner, and E. Schonberg, *Programming with Sets: An Introduction to Set 1*. New York: Springer-Verlag, 1986.

[38] J. P. Segundo and O. D. Martinez, "Dynamic and static hysteresis in crayfish receptors," *Biol. Cybern.*, vol. 52, pp. 291–296, 1985.

[39] H. Stone and J. Stone, "Efficient search techniques—An empirical study of the N-queens problem," *IBM J. Res. Develop.*, vol. 31, no. 4, 1987.

[40] G. Taga, Y. Yamaguchi, and H. Shimizu, "Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment," *Biol. Cybern.*, vol. 65, pp. 147–159, 1991.

[41] Y. Takefuji, *Neural Network Parallel Computing*. Boston, MA: Kluwer, 1992.

[42] Y. Takefuji and K. C. Lee, "An hysteresis binary neuron: A model suppressing the oscillatory behavior of neural dynamics," *Biol. Cybern.*, vol. 64, pp. 353–356, 1991.

[43] ——, "Artificial neural network for four-coloring map problems and K-colorability problem," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 326–333, 1991.

[44] A. Visintin, *Differential Models of Hysteresis*. New York: Springer-Verlag, 1994.

[45] V. Volterra, *Theory of Functionals and of Integral and Integro-Differential Equations*. New York: Dover, 1959.

[46] J. R. Whiteman, "A mathematical model depicting the stress-strain diagram and the hysteresis loop," *Trans. ASME*, vol. 26, 1959.

[47] N. Wirth, *Algorithms + Data Structures = Programs*. Englewood-Cliffs, NJ: Prentice-Hall, 1976.

[48] A. M. Yaglom and I. M. Yaglom, *Challenging Mathematical Problems with Elementary Solutions*. San Francisco, CA: Holden-Day, 1964.

[49] H. Yanai and Y. Sawada, "Associative memory network composed of neurons with hysteretic property," *Neural Networks*, vol. 3, pp. 223–228, 1990.

Sunil Bharitkar (S'00) was born in Pune, India, on April 20, 1969. He received the B.E. degree in electronics and communications engineering from University of Pune, India, in 1990, and the M.S. degree in electrical engineering from Case Western Reserve University, Cleveland, OH, in 1995. He is currently pursuing the Ph.D. degree at the Signal and Image Processing Institute, Department of Electrical Engineering-Systems, University of Southern California, Los Angeles.

He was a Visiting Researcher to Keio University, Japan in the summer of 1994, through Hitachi Corp. During 1995, he was a Senior Engineer at Intelligent Machines, Inc., Sunnyvale, CA. He was employed at the Advanced Manufacturing Technology Division at Ford Motor Company, Detroit, MI, during 1998 and 1999, where he developed methods for detecting signal stability for ultrasonic sensors in thin film paint measurements on automobiles. He has published in different journals and conference proceedings, and has a patent pending. His research interests include, acoustics/audio and speech processing, control systems, neural networks, and communications.

Jerry M. Mendel (S'59–M'61–SM'72–F'78) received the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, NY.

Currently, he is Professor of Electrical Engineering and Associate Director for Education of the Integrated Media Systems Center at the University of Southern California, Los Angeles, where he has been since 1974. He has published more than 380 technical papers and is author and/or editor of seven books. His present research interests include type-2 fuzzy logic systems, higher-order statistics, and neural networks, and their applications to a wide range of signal processing problems.

Dr. Mendel is a Distinguished Member of the IEEE Control Systems Society. He was President of the IEEE Control Systems Society in 1986. Among his awards are the 1983 Best Transactions Paper Award of the IEEE Geoscience and Remote Sensing Society, the 1992 Signal Processing Society Paper Award, a 1984 IEEE Centennial Medal and an IEEE Third Millenium Medal.