Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 873670, 10 pages
http://dx.doi.org/10.1155/2013/873670

Research Article

Chaotic Hopfield Neural Network Swarm Optimization and Its Application

Yanxia Sun,[1] Zenghui Wang,[2] and Barend Jacobus van Wyk[1]

[1] *Department of Electrical Engineering, Tshwane University of Technology, Pretoria 0001, South Africa*
[2] *School of Engineering, University of South Africa, Florida 1710, South Africa*

Correspondence should be addressed to Zenghui Wang; wangzengh@gmail.com

Received 7 February 2013; Accepted 20 March 2013

Academic Editor: Xiaojing Yang

A new neural network based optimization algorithm is proposed. The presented model is a discrete-time, continuous-state Hopfield neural network and the states of the model are updated synchronously. The proposed algorithm combines the advantages of traditional PSO, chaos and Hopfield neural networks: particles learn from their own experience and the experiences of surrounding particles, their search behavior is ergodic, and convergence of the swarm is guaranteed. The effectiveness of the proposed approach is demonstrated using simulations and typical optimization problems.

1. Introduction

The discovery of chaos in astronomical, solar, fluid, and other systems sparked significant research in nonlinear dynamics exhibiting chaos. Chaos was found to be useful and have great potential in many disciplines such as mixing liquids with low power consumption, presenting outages in power systems, biomedical engineering applications involving signals from the brain and heart, to name just a few [1]. Chaotic systems exhibit three important properties. Firstly, a deterministic system is said to be chaotic whenever its evolution sensitively depends on the initial conditions. Secondly, there is an infinite number of unstable periodic orbits embedded in the underlying chaotic set. Thirdly, the dynamics of the chaotic attractor is ergodic, which implies that during its temporal evolution the system ergodically visits small neighborhoods around every point in each one of the unstable periodic orbits embedded within the chaotic attractor. Although it appears to be stochastic, it is generated by a deterministic nonlinear system. Lyapunov exponents characterize quantitatively stochastic properties of the dynamical systems. When the dynamical system is chaotic, there exists at least one lyapunov exponent $\lambda > 0$. It is reported that chaotic behavior also exists in biological neurons and neural networks [2, 3]. Using chaos to develop novel optimization techniques gained much attention during the last decade. For a given energy or cost function, the chaotic ergodic orbits of a chaotic dynamic system used for optimization may eventually reach the global optimum or a point close to it with high probability [4, 5].

Since Hopfield and Tank [6] applied their neural network to the travelling salesman problem, neural networks have provided a powerful approach to a wide variety of optimization problems [7, 8]. However the Hopfield neural network (HNN) often gets trapped in a local minima. A number of modifications were made to Hopfield neural networks to escape from local minima. Some modifications, based on chaotic neural networks [9] and simulated annealing [10], were proposed to solve global optimization problems [11]. In [12–14] the guaranteed convergence of Hopfield neural networks is discussed.

Particle swarm optimization (PSO), developed by Clerc and Kennedy in 2002 [15], is a stochastic global optimization method which is based on simulation of social behavior. In a particle swarm optimizer, individuals "evolve" by cooperating with other individuals over several generations. Each particle adjusts its flying according to its own flying experiences and the flying experience of its companions. Each individual is named as a particle which, in fact, represents a potential solution to a problem. Each particle is treated as a point in a D-dimensional space. However, the PSO algorithm is likely

to temporarily get stuck and may need a long period of time to escape from a local extremum [16]. It is difficult to guarantee the convergence of the swarm, especially when random parameters are used. In order to improve the dynamical behavior of PSO, one can combine chaos with PSO algorithms to enhance the performance of PSO. In [17–19] chaos were applied to the PSO to avoid the PSO getting trapped in local minima.

PSO is motivated by the behavior of organisms such as fish schooling and bird flocking [20]. During the process, future particle positions (determined by velocity) can be regarded as particle intelligence [21]. Using a chaotic intelligent swarm system to replace the original PSO might be convenient for analysis while maintaining stochastic search properties. Most importantly, the convergence of a particle swarm initialized with random weights is not guaranteed.

In this paper we propose a chaotic Hopfield neural network swarm optimization (CHNNSO) algorithm. The rest of the paper is organized as follows. In Section 2, the preliminaries of Hopfield neural networks and PSO are described. The chaotic Hopfield neural network model is developed in Section 3. In Section 4, the dynamics of the chaotic Hopfield neural network is analyzed. Section 5 provides simulation results and comparisons. The conclusion is given in Section 6.

2. Preliminaries

2.1. Basic Hopfield Neural Network Theory [22]. A Hopfield net is a recurrent neural network having a synaptic connection pattern such that there is an underlying Lyapunov energy function for the activity dynamics. Started in any initial state, the state of the system evolves to a final state that is a (local) minimum of the Lyapunov energy function. The Lyapunov energy function decreases in a monotone fashion under the dynamics and is bounded below. Because of the existence of an elementary Lyapunov energy function for the dynamics, the only possible asymptotic result is a state on an attractor.

There are two popular forms of the model: binary neurons with discrete time which is updated one at a time and continuous time graded neurons. In this paper, the second kind of model is used. The dynamics of a n-neuron continuous Hopfield neural network is described by

$$\frac{du_i}{dt} = \frac{-u_i}{\tau} + \sum_j T_{ij} x_i(t) + I_i. \tag{1}$$

Here, $u_i \in (-\infty, \infty)$ is the input of neuron i, and the output of neuron i is

$$x_i(t) = g_i\left[u_i(t)\right], \tag{2}$$

where $i = 1, 2, \ldots n$, τ is a positive constant, I_i is external inputs (e.g., sensory input or bias current) to neuron i and is sometimes called the "firing threshold" when replaced with $-I_i$. u_i is the mean internal potential of the neuron which determines the output of neuron i. T_{ij} is the strength of synaptic input from neuron i to neuron j. g is a monotone function that converts internal potential into firing rate input of the neuron. T is the matrix with elements T_{ij}. When T is symmetric, the Lyapunov energy function is given by

$$J = -\frac{1}{2}\sum_{ij} T_{ij} x_i x_j - \sum_i I_i x_i + \frac{1}{\tau}\sum_i \int_0^{x_j} g^{-1}(Z)\, dZ, \tag{3}$$

where g^{-1} is the inverse of the gain function g. There is a significant limiting case of this function when T has no diagonal elements and the input-output relation becomes a step, going from 0 to a maximum firing rate (for convenience, scaled to 1). The third term of this Lyapnouv function is then zero or infinite. With no diagonal elements in T, the minima of J are all located at corners of the hypercube $0 \le x_j \le 1$. In this limit, the states of the continuous variable system are stable.

Many optimization problems can be readily represented using Hopfield nets by transforming the problem into variables such that the desired optimization corresponds to the minimization of the respective Lyapunov energy function [6]. The dynamics of the HNN converges to a local Lyapunov energy minimum. If this local minimum is also the global minimum, the solution of the desired optimization task has been carried out by the convergence of the network state.

2.2. Basic PSO Theory. Many real optimization problems can be formulated as the following functional optimization problem:

$$\begin{aligned} &\min \quad f(X_i), \quad X_i = \left[x_i^1, \ldots, x_i^n\right], \\ &\text{s.t.} \quad x_i \in \left[a_i, b_i\right], \quad i = 1, 2, \ldots, n. \end{aligned} \tag{4}$$

Here f is the objective function, and X_i is the decision vector consisting of n variables.

The original particle swarm algorithm works by iteratively searching in a region and is concerned with the best previous success of each particle, the best previous success of the particle swarm and the current position and velocity of each particle [20]. Every candidate solution of $f(X_i)$ is called a "particle." The particle searches the domain of the problem according to

$$V_i(t+1) = \omega V_i(t) + c_1 R_1\left(P_i - X_i(t)\right) + c_2 R_2\left(P_g - X_i(t)\right), \tag{5}$$

$$X_i(t+1) = X_i(t) + V_i(t+1), \tag{6}$$

where $V_i = [v_i^1, v_i^2, \ldots, v_i^n]$ is the velocity of particle i; $X_i = [x_i^1, x_i^2, \ldots, x_i^n]$ represents the position of particle i; P_i represents the best previous position of particle i (indicating the best discoveries or previous experience of particle i); P_g represents the best previous position among all particles (indicating the best discovery or previous experience of the social swarm); ω is the inertia weight that controls the impact of the previous velocity of the particle on its current velocity and is sometimes adaptive [17]; R_1 and R_2 are two random weights whose components r_1^j and r_2^j ($j = 1, 2, \ldots, n$) are chosen uniformly within the interval $[0, 1]$ which might not guarantee the convergence of the particle trajectory; c_1 and

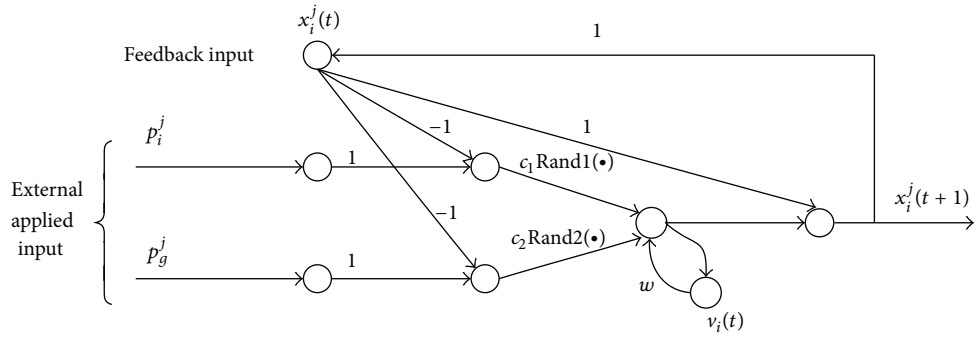

Figure 1: Particle structure.

c_2 are the positive constant parameters. Generally the value of each component in V_i should be clamped to the range $[-v_{\max}, v_{\max}]$ to control excessive roaming of particles outside the search space.

3. A Chaotic Hopfield Neural Network Model

From the introduction of basic PSO theory, every particle can be seen as the model of a single fish or a single bird. The position chosen by the particle can be regarded as a state of a neural network with a random synaptic connection. According to (5)-(6), the position components of particle i can be thought of as the output of a neural network as shown in Figure 1.

In Figure 1, Rand1(·) and Rand2(·) are two independent and uniformly distributed random variables within the range [0, 1], which refer to r_1^j and r_2^j, respectively. p_i^j and p_g^j are the components of P_i and P_g, respectively. p_g^j is the previous best value amongst all particles, and p_i^j, as an externally applied input, is the jth element of the best previous position P_i, and it is coupled with other components of P_i. The particles migrate toward a new position according to (5)-(6). This process is repeated until a defined stopping criterion is met (e.g., maximum number of iterations or a sufficiently good fitness value).

As pointed out by Clerc and Kennedy [15], the powerful optimization ability of the PSO comes from the interaction amongst the particles. The analysis of complex interaction amongst the particles in the swarm is beyond the scope of this paper which focuses on the construction of a simple particle using a neural network perspective and convergence issues. Artificial neural networks are composed of simple artificial neurons mimicking biological neurons. The HNN has the property that as each neuron in a HNN updates, an energy function is monotonically reduced until the network stabilizes [23]. One can therefore map an optimization problem to a HNN such that the cost function of the problem corresponds to the energy function of the HNN and the result of the HNN thus suggests a low cost solution to the optimization problem. The HNN might therefore be a good choice to model particle behavior.

In order to approach P_g and P_i, the HNN model should include at least two neurons. For simplicity, the HNN model of each particle position component has two neurons whose outputs are $x_i^j(t)$ and $x_{ip}^j(t)$. In order to transform the problem into variables such that the desired optimization corresponds to the minimization of the energy function, the objective function should be determined firstly. As $x_i^j(t)$ and $x_{ip}^j(t)$ should approach p_g^j and p_i^j, respectively, $(x_i^j(t) - p_g^j)^2$ and $(x_{ip}^j(t) - p_i^j)^2$ can be chosen as two parts of the energy function. The third part of energy function $(x_i^j(t) - x_{ip}^j(t))^2$ is added to accompany $(x_{ip}^j(t) - p_i^j)^2$ to cause $x_i^j(t)$ to tend towards p_i^j. Therefore the HNN Lyapunov energy function for each particle is proposed:

$$J_i^j(t) = A\left(x_i^j(t) - p_g^j\right)^2 + B\left(x_{ip}^j(t) - p_i^j\right)^2 + C\left(x_i^j(t) - x_{ip}^j(t)\right)^2, \tag{7}$$

where A, B, and C are positive constants.

Here the neuron input-output function is chosen as a sigmoid function, given by (9) and (11). Equations (8) and (10) are the Euler approximation of (1) of the continuous Hopfield neural network [14]. The dynamics of component j of particle i is described by

$$u_i^j(t+1) = ku_i^j(t) + \alpha\frac{\partial J_i^j(t)}{\partial x_i^j(t)}, \tag{8}$$

$$x_i^j(t+1) = g_i\left(u_i^j(t+1)\right) = \frac{1}{1 - e^{\xi u_i^j(t+1)}}, \tag{9}$$

$$u_{ip}^j(t+1) = ku_{ip}^j(t) + \alpha\frac{\partial J_i^j(t)}{\partial x_{ip}^j(t)}, \tag{10}$$

$$x_{ip}^j(t+1) = g_i\left(u_{ip}^j(t+1)\right) = \frac{1}{1 - e^{\xi u_{ip}^j(t+1)}}. \tag{11}$$

According to (5)-(6) and Figure 1, the PSO uses random weights to simulate birds flocking or fish searching for food. When birds flock or fish search for food, they exhibit chaos like behavior, yet (8)–(11) do not generate chaos. Aihara et al. [9] proposed a kind of chaotic neuron, which includes relative refractoriness in the model to simulate chaos in a biological

brain. To use this result $z_i(k)(x_i^j(k) - I_0)$ and $z_i(k)(x_{i_p}^j(k) - I_0)$ are added to (8) and (10) to cause chaos. Equations (8) and (10) then become

$$u_i^j(t+1) = ku_i^j(t) + \alpha \frac{\partial J_i^j(t)}{\partial x_i^j(t)} - z_i(t)\left(x_i^j(t) - I_0\right), \quad (12)$$

$$u_{ip}^j(t+1) = ku_{ip}^j(t) + \alpha \frac{\partial J_i^j(t)}{\partial x_{ip}^j(t)} - z_i(t)\left(x_{i_p}^j(t) - I_0\right). \quad (13)$$

In order to escape from chaos as time evolves, we set

$$z_i(t+1) = (1-\beta) z_i(t). \quad (14)$$

In (8)–(14): ξ, α, and k are positive parameters; $z_i(t)$ is self-feedback connection weight (the refractory strength); β is the damping factor of the time-dependent $z_i(t)$, $(0 \le \beta \le 1)$; I_0 is a positive parameter.

All the parameters are fixed except $z(t)$ which is varied.

The combination of (9), (11)–(14) is called chaotic Hopfield neural network swarm optimization(CHNNSO) proposed by us. According to (8)–(14), the following procedure can be used for implementing the proposed CHNNSO algorithm.

(1) Initialize the swarm, assign a random position in the problem hyperspace to each particle, and calculate the fitness function which is given by the optimization problem whose variables are corresponding to the elements of particle position coordinates.

(2) Synchronously update the positions of all the particles using (9), (11)–(14) and change the two states every iteration.

(3) Evaluate the fitness function for each particle.

(4) For each individual particle, compare the particle's fitness value with its previous best fitness value. If the current value is better than the previous best value, then set this value as the p_i^j and the current particle's position, x_i^j, as p_i^j, else if the p_i is updated, then reset $z = z_0$.

(5) Identify the particle that has the best fitness value. When iterations are less than a certain value, and if the particle with the best fitness value is changed then reset $z = z_0$ to keep the particles chaotic to prevent premature convergence.

(6) Repeat steps (2)–(5) until a stopping criterion is met (e.g., maximum number of iterations or a sufficiently good fitness value).

As can be seen from (9) and (11), the particle position component x_i^j is located in the interval $[-1, 1]$. The optimization problem variable interval must therefore be mapped to $[-1, 1]$ and vice versa using

$$x^j = -1 + \frac{2\left(x_j - a_j\right)}{b_j - a_j}, \quad j = 1, 2, \ldots, n,$$
$$x_j = a_j + \frac{1}{2}\left(x^j + 1\right)\left(b_j - a_j\right), \quad j = 1, 2, \ldots, n. \quad (15)$$

Here, a_j and b_j are the lower boundary and the upper boundary of $x_j(t)$, respectively, and only one particle is analyzed for simplicity.

4. Dynamics of Chaotic Hopfield Network Swarm Optimization

In this section, the dynamics of the chaotic Hopfield network swarm optimization (CHNNSO) is analyzed. The first subsection discusses the convergence of the chaotic particle swarm. The second subsection discusses the dynamics of the simplest CHNNSO with different parameter values.

4.1. Convergence of the Particle Swarm

Theorem 1 (Wang and Smith [14]). *If one has a network of neurons with arbitrarily increasing I/O functions, there exists a sufficient stability condition for a synchronous TCNN (transiently chaotic neural network) equation* (12)*, namely,*

$$k > 1, \quad \frac{2k}{\beta_{\max}} > -T'_{\min}. \quad (16)$$

Here $1/\beta_{\max} = \min(1/g_i')$ *(g_i' denotes the derivative with respect to time t of the neural I/O function for neuron i, in this paper is the sigmoid function* (9)*. $T'_{\min}$ is the minimum eigenvalue of the connected weight matrix of the dynamics of a n-neuron continuous Hopfield neural network).*

Theorem 2. *A sufficient stability condition for the CHNNSO model is* $k > 1$.

Proof. When $t \to \infty$,

$$z(t+1) = 0. \quad (17)$$

It then follows that the equilibria of (12) and (13) can be evaluated by

$$\Delta u_i^j(t+1) = (k-1) u_i^j(t) + \alpha \frac{\partial J_i^j(t)}{\partial x_i^j(t)},$$
$$\Delta u_{ip}^j(t+1) = (k-1) u_{ip}^j(t) + \alpha \frac{\partial J_i^j(t)}{\partial x_{ip}^j(t)}. \quad (18)$$

According to (7), we get

$$\Delta u_i^j(t+1) = (k-1) u_i^j(t) + 2A\alpha\left(x_i^j(t) - p_g^j\right) + 2C\alpha\left(x_i^j(t) - x_{ip}^j(t)\right) = 0, \quad (19)$$

$$\Delta u_{ip}^{j}(t+1) = (k-1)\,u_{ip}^{j}(t) + 2B\alpha\left(x_{ip}^{j}(t) - p_i^{j}\right) - 2C\alpha\left(x_i^{j}(t) - x_{ip}^{j}(t)\right) = 0, \tag{20}$$

$$\begin{aligned} T'_{\min} &= \min\left(\text{eigenvalue}\begin{pmatrix} 2A\alpha + 2C\alpha & -2C\alpha \\ -2C\alpha & 2B\alpha + 2C\alpha \end{pmatrix}\right) \\ &= (A+B+2C)\,\alpha - \alpha\sqrt{(A-B)^2 + 4C^2} \\ &= \alpha\left(A + B + 2C - \sqrt{(A-B)^2 + 4C^2}\right). \end{aligned} \tag{21}$$

In this paper,

$$\beta_{\max} = \frac{1}{\min\left(1/g'\right)} = \frac{\xi}{4}. \tag{22}$$

It is clear that $T'_{\min} > 0$ and the stability condition (16) is satisfied when $\xi > 0$. The above analysis verifies Theorem 2. □

Theorem 3. *The particles converge to the sphere with center point p_g^j and radius $R = \max\|x_{i_e}^j - p_i^j\|$ ($x_{i_e}^j$ is the final convergence equilibria, if the optimization problem is in two-dimensional plane, the particles are finally in a circle).*

It is easy to show that the particle model given by (7) and (8)–(14) has only one equilibrium as $t \to \infty$, that is, $z(t) = 0$. Hence, as $t \to \infty$, X_i belongs to the hypersphere whose origin is P_g and the radius is R. Solving (9), (11), (19), and (20) simultaneously, we get

$$\frac{(k-1)\ln\left(1 - \left(1/x_i^j\right)(t)\right)}{\xi} + 2A\alpha\left(x_i^j(t) - P_g^j\right) + 2C\alpha\left(x_i^j(t) - x_{ip}^j(t)\right) = 0, \tag{23}$$

$$\frac{(k-1)\ln\left(1 - \left(1/x_{ip}^j(t)\right)\right)}{\xi} + 2B\alpha\left(x_{ip}^j(t) - P_i^j\right) - 2C\alpha\left(x_i^j(t) - x_{ip}^j(t)\right) = 0. \tag{24}$$

With (23) and (24) satisfied, there must exist the final convergence equilibria $x_{i_e}^j$ and $x_{ip_e}^j$. So the best place the particle swarm can find is $\| p_g^j - x_{i_e}^j(t+1) \|$ and radius is $R = \max \| x_{i_e}^j - p_i^j \|$.

The above analysis therefore verifies Theorem 3.

4.2. Dynamics of the Simplest Chaotic Hopfield Neural Network Swarm. In this section, the dynamics of the simplest particle swarm model is analyzed. Equations (7) and (8)–(13) are the dynamic model of a single particle with subscript i ignored. According to (7) and Theorem 3, the parameters A, B, and C control the final convergent radius. According to trial and error, the parameters A, B, and C can be chosen in the range from 0.005 to 0.05. According to (16) and (22), $k > 1$ and $\xi > 0$. In the simulation, the results are better when k is in the neighborhood of 1.1 and ξ is in the neighborhood of 150. The parameters β and $Z(0)$ control the time of the chaotic period. If β is too big and/or $Z(0)$ is too small, the system will quickly escape from chaos and performance will be poor. The parameter $I_0 = 0.2$ is standard in the literature on chaotic neural networks. The simulation showed that the model is not sensitive to the values of parameters ξ and k, for example, $100 < \xi < 200$ and $1 < k < 1.5$ are feasible.

Then the values of the parameters in (7)–(14) are set to

$$\begin{aligned} &A = 0.02, \quad B = C = 0.01, \quad \xi = 150, \\ &\alpha = 1, \quad \beta = 0.001, \quad Z(0) = 0.3, \\ &k = 1.1, \quad I_0 = 0.2, \quad p_i = 0.5, \quad p_g = 0.2. \end{aligned} \tag{25}$$

Figure 2 shows the time evolution of $x_i(t)$, $z(t)$ and the Lyapunov exponent λ of $x_i(t)$. The Lyapunov exponent λ characterizes the rate of separation of infinitesimally close trajectories. A positive Lyapunov exponent is usually taken as an indication that the system is chaotic [1]. Here, λ is defined as

$$\lambda = \lim_{n \to +\infty} \frac{1}{m} \sum_{t=0}^{m-1} \ln\left|\frac{dx_i(t+1)}{dx_i(t)}\right|. \tag{26}$$

At about 200 steps, $z(t)$ decays to a small value and $x_i(t)$ departs from chaos which corresponds with the change of λ from positive to negative.

According to Figure 2, the convergence process of a simple particle position follows the nonlinear bifurcation making the particle converge to a stable fixed point from a strange attractor. In the following section, it is shown that the fixed point is determined by the best previous position P_g among all particles and the best position P_i of the individual particle.

Remark 4. The proposed CHNNSO model is a deterministic Chaos-Hopfield neural network swarm which is different from existing PSOs with stochastic parameters. Its search orbits exhibit an evolutionary process of inverse period bifurcation from chaos to periodic orbits then to sink. As chaos is ergodic and the particle is always in a chaotic state at the beginning (e.g., in Figure 2), the particle can escape when trapped in local extrema. This proposed CHNNSO model will therefore in general not suffer from being easily trapped in a the local optimum and will continue to search for a global optimum.

5. Numerical Simulation

To test the performance of the proposed algorithms, two famous benchmark optimization problems and an engineering optimization problem with linear and nonlinear constraints are used. The solutions to the two benchmark problems can be represented in the plane and therefore the convergence of the CHNNSO can be clearly observed. The results of the third optimization problem when compared with other algorithms are displayed in Table 1. We will compare the CHNNSO with the original PSO [20].

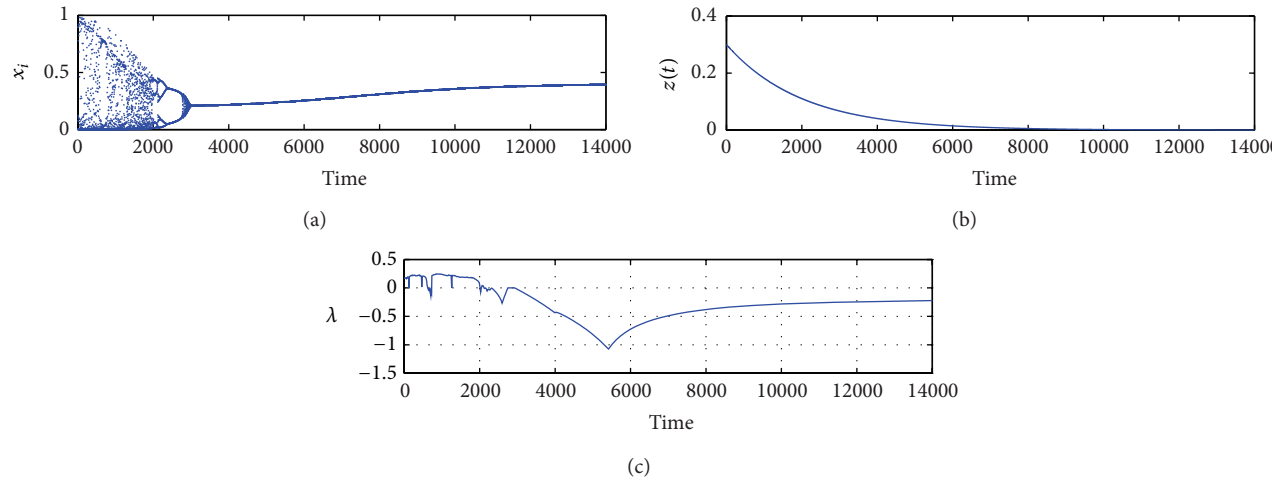

FIGURE 2: Time evolutions of $x_i(t)$, $z(t)$, and the Lyapunov exponent λ.

TABLE 1: The parameters of the Hartmann function (when $n = 3$ and $q = 4$).

i	a_{i1}	a_{i2}	a_{i3}	c_i	p_{i1}	p_{i2}	p_{i3}
1	3	10	30	1	0.3689	0.1170	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.7470
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.03815	0.5473	0.8828

5.1. The Rastrigin Function. To demonstrate the efficiency of the proposed technique, the famous Rastrigin function is chosen as a test problem. This function with two variables is given by

$$f(X) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2), \quad -1 < x_i < 1, \ i = 1, 2. \tag{27}$$

The global minimum is −2 and the minimum point is (0, 0). There are about 50 local minima arranged in a lattice configuration.

The proposed technique is applied with a population size of 20 and the maximum number of iterations is 20000. The chaotic particle swarm parameters are chosen as $A = 0.02$, $B = C = 0.01$, $\beta = 0.001$, $z(0) = 0.3$, $\alpha = 1$, $\xi = 150$, $k = 1.1$, and $I_0 = 0.2$.

The position of every particle is initialized with a random value. The time evolution of the cost of the Rastrigin function is shown in Figure 3. The global minimum at −2 is obtained by the best particle with $(x_1, x_2) = (0, 0)$.

From Figure 3, it can be seen that the proposed method gives good optimization results. Since there are two variables in the Rastrigin function, it is easy to show the final convergent particle states in the plane.

In Figure 4, the "∗"s are the best experiences of each particle. The "+"s are the final states of the particles. The global minimum (0, 0) is also included in the "∗"s and the "+"s. Most "∗"s and "+"s are overlapped at the global minimum (0, 0). According to Theorem 3, the particles will finally converge to a circle finally. For this Rastrigin problem, the particles' final states converge to the circle as shown in Figure 4, and hence the global convergence of the particles is guaranteed.

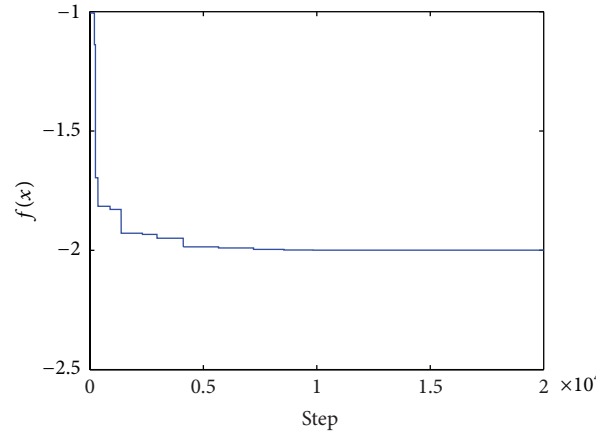

FIGURE 3: Time evolutions of Rastrigin function.

Figure 5 displays the results when the original PSO [20] was used to optimize the Rastrigin function. In the numerical simulation, the particle swarm population size is also 20 and parameters c_1 and c_2 are set to 2 and ω set to 1. $v_{\max}$ is set equal to the dynamic range of each dimension. The "∗"s in Figure 5 are the final states of all the particles corresponding to the "+"s in Figure 4. It is easy to see that the final states of the particles are ruleless even though the global minimum of −2 is obtained by the best experience of the particle swarm, that is, $(x_1, x_2) = (0, 0)$ as shown in Figure 5.

By comparing the results obtained by the proposed CHNNSO in Figure 4 with the results of the original PSO in Figure 5, it can be seen that the final states of the particles of the proposed CHNNSO are attracted to the best experience of all the particles and that convergence is superior. The final states of CHNNSO particles are guaranteed to converge which is not the case for original PSO implications.

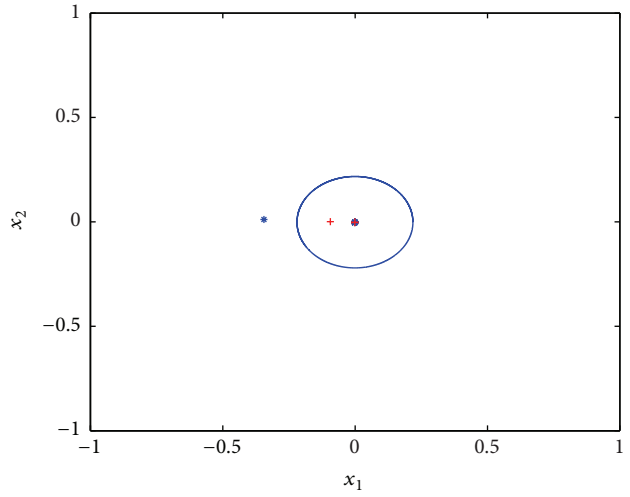

Figure 4: The final states and the best experiences of the particles achieved from the proposed CHNNSO for Rastrigin function.

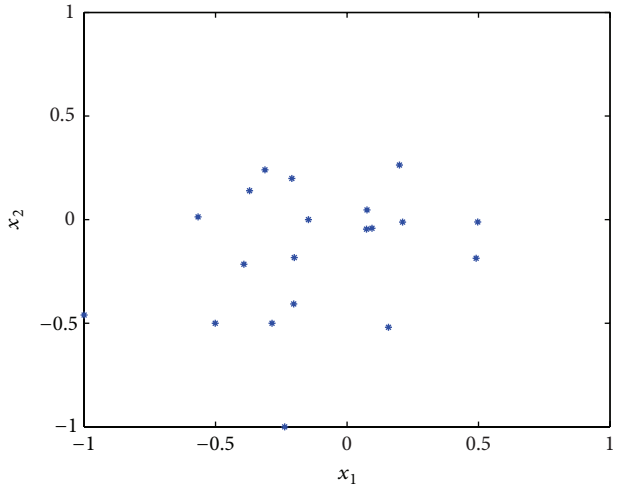

Figure 5: The final states of the particles achieved from original PSO for Rastrigin function.

When their parameters c_1 and c_2 are both set to 1.494 and ω to a value of 0.729, $v_{\max}$ is set equal to the dynamic range on each dimension. The constriction factors c_1, c_2, ω are applied to improve the convergence of the particle over time by damping the oscillations once the particle is focused on the best point in an optimal region. The main disadvantage of this method is that the particles may follow wider cycles and may not converge when the individual best performance is far from the neighborhoods best performance (two different regions) [24].

5.2. The Schaffer's F6 Function. To further investigate the performance of the CHNNSO,

$$f(X) = 0.5 + \frac{\sqrt{\sin\left(x_1^2 + x_2^2\right)^2} - 0.5}{1.0 + 0.001\left(x_1^2 + x_2^2\right)^2}, \tag{28}$$

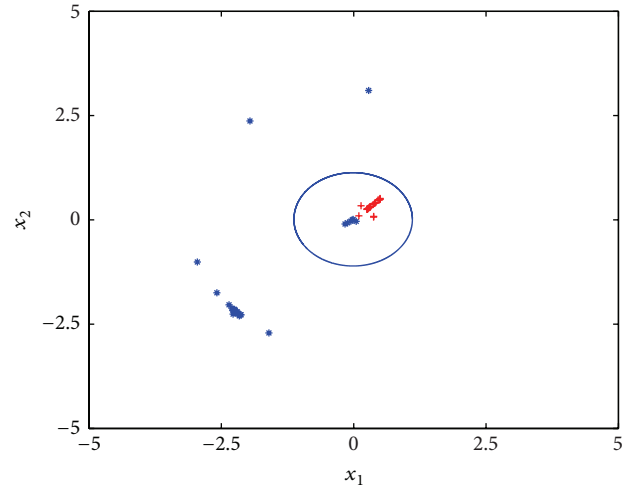

Figure 6: The final states and the best experiences of the particles achieved from the proposed CHNNSO for Schaffer' F6 function.

the Schaffer's F6 function [25] is chosen. This function has a single global optimum at $(0, 0)$ and $f_{\min}(x, y) = 0$, and a large number of local optima. The global optimum is difficult to find because the value at the best local optimum differs with only about 10^{-3} from the global minimum. The local optima crowd around the global optimum. The proposed technique is applied with a population size of 30, the iterations are 100000, and the parameters of CHNNSO are chosen as

$$\begin{aligned} A = B = C = 0.01, \quad \beta = 0.0003, \quad z(0) = 0.3, \\ \xi = 155, \quad k = 1.1, \quad I_0 = 0.2. \end{aligned} \tag{29}$$

The position of each particle is initialized with a random value. In Figure 6, the "$*$"s are the best experiences of each particle. The global minimum at 2.674×10^{-4} is obtained by the best particle with $(x_1, x_2) = (-0.00999, 0.01294)$ which is included in the "$*$"s. The "+"s are the final states of the particles. According to Theorem 3 the particles' final states converge in a circle as shown in Figure 6 which proves global convergence. From Figure 6 it is clearly seen that the particles' final states are attracted to the neighborhoods of the best experiences of all the particles and the convergence is good.

Figure 7 shows the final particle states when the original PSO [20] was used to optimize the Schaffer's F6 function. In this numerical simulation of the original PSO, the particle swarm population size is also 20 and parameters c_1 and c_2 are both set to 2 and set ω is set to a value of 1. $v_{\max}$ is set equal to the dynamic range of each dimension. The "$*$"s in Figure 7 are the final states of all the particles corresponding to the "+"s in Figure 6. It is easy to see that the final states of the particles are ruleless in Figure 7. The global minimum 6.3299×10^{-4} is obtained by the best particle with $(x_1, x_2) = (0.01889 \times 10^{-3}, -0.4665 \times 10^{-3})$. The best experience from the original PSO is not as good as the best of the proposed CHNNSO.

Comparing the results obtained from the proposed CHNNSO in Figure 6 and the original PSO in Figure 7, it is clearly seen that the particles' final states of the proposed

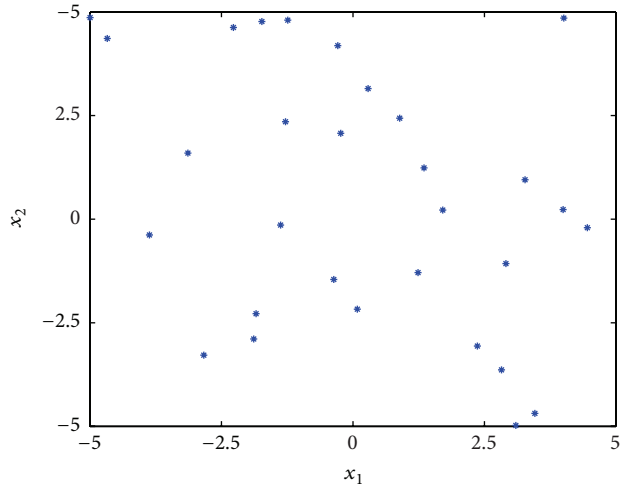

FIGURE 7: The final states of the particles achieved from original PSO for Schaffer's F6 function.

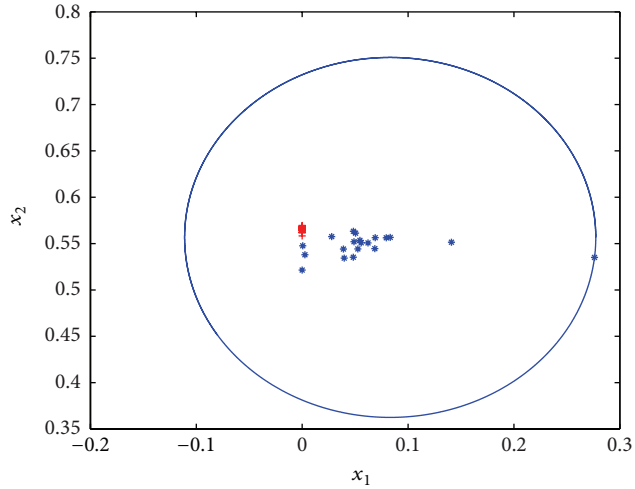

FIGURE 8: The final states of the particles achieved from original PSO for Hartmann function.

CHNNSO are finally attracted to the best experience of all the particles and the convergence is better than that of the original PSO. The CHNNSO can guarantee the convergence of the particle swarm, but the final states of the original PSO are ruleless.

5.3. The Hartmann Function. The Hartmann function when $n = 3; q = 4$ is given by

$$f(x) = -\sum_{i=1}^{q} c_i \exp\left(-\sum_{j=1}^{n} a_{ij}\left(x_j - p_{ij}\right)^2\right), \tag{30}$$

with x_j belonging to

$$S = \left\{x \in R^n \mid 0 \le x_j \le 1, 1 \le j \le n\right\}, \tag{31}$$

Table 1 shows the parameter values for the Hartmann function when $n = 3, q = 4$.

When $n = 3$, $X_{\min} = (0.114, 0.556, 0.882)$, $f(x_{\min}) = -3.86$.

The time evolution of the cost of the Hartmann function is 15000. In Figure 8, only subdimensions are pictured. In Figure 8, the "+"s are the final states of the particles and the "∗"s denote the best experiences of all particles. From Figure 8, it can be easily seen that the final states of the particles converge to the circle. The center point is $(0.0831, 0.5567, 0.8522)$ and the radius is 0.1942. The final particle states confirm Theorem 3, and the final convergency is guaranteed.

5.4. Design of a Pressure Vessel. The pressure vessel problem described in [26, 27] is an example which has linear and nonlinear constraints and has been solved by a variety of techniques. The objective of the problem is to minimize the total cost of the material needed for forming and welding a cylindrical vessel. There are four design variables: x_1 (T_s, thickness of the shell), x_2 (T_h, thickness of the head), x_3 (R, inner radius), and x_4 (L, length of the cylindrical section of the vessel). T_s and T_h are integer multiples of 0.0625 inch, which are the available thickness of rolled steel plates, and R and L are continuous. The problem can be specified as follows:

$$\begin{aligned}
\text{Minimize} \quad & f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 \\
& \quad + 3.1661x_1^2x_4 + 19.84x_1^2x_3, \\
\text{subject to} \quad & g_1(X) = -x_1 + 0.0193x_3 \le 0, \\
& g_2(X) = -x_2 + 0.00954x_3 \le 0, \\
& g_3(X) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3 + 1296000 \le 0, \\
& g_4(X) = x_4 - 240 \le 0.
\end{aligned} \tag{32}$$

The following range of the variables were used [27]:

$$\begin{aligned}
& 0 < x_1 \le 99, \quad 0 < x_2 \le 99, \quad 10 \le x_3 \le 200, \\
& 10 \le x_4 \le 200.
\end{aligned} \tag{33}$$

de Freitas Vas and de Graça Pinto Fernandes [26] proposed an algorithm to deal with the constrained optimization problems. Here this algorithm [26] is combined with CHNNSO to search for the global optimum. The proposed technique is applied with a population size of 20 and the maximum number of iterations is 20000. Then the values of the parameters in (7)–(14) are set to

$$\begin{aligned}
& A = 0.02, \quad B = C = 0.01, \quad \xi = 150, \\
& \alpha = 1, \quad \beta = 0.001, \quad Z(0) = 0.3, \\
& k = 1.1, \quad I_0 = 0.2, \\
& p_i = 0.5, \quad p_g = 0.5.
\end{aligned} \tag{34}$$

From Table 2, the best solution obtained by the CHNNSO is better than the other two solutions previously reported.

Table 2: Comparison of the results for pressure Vessel design problem.

Design variables	Best solutions found		
	This paper	Hu et al. [27]	Coello [28]
$x_1(T_s)$	0.8125	0.8125	0.8125
$x_2(T_h)$	0.4375	0.4375	0.4375
$x_3(R)$	42.09845	42.09845	40.3239
$x_4(L)$	176.6366	176.6366	200.0
$g_1(X)$	0	0	0.03424
$g_2(X)$	−0.03588	−0.03588	−0.052847
$g_3(X)$	$-1.1814 * 10^{-11}$	$-5.8208 * 10^{-11}$	−27.105845
$g_4(X)$	−63.3634	−63.3634	−40.0
$f(X)$	**6059.1313**	**6059.1313**	**6288.7445**

6. Conclusion

This paper proposed a chaotic neural networks swarm optimization algorithm. It incorporates the particle swarm searching structure having global optimization capability into Hopfield neural networks which guarantee convergence. In addition, by adding chaos generator terms into Hopfield neural networks, the ergodic searching capability is greatly improved in the proposed algorithm. The decay factor introduced in the chaos terms ensures that the searching evolves to convergence to global optimum after globally chaotic optimization. The experiment results of three classic benchmark functions showed that the proposed algorithm can guarantee the convergence of the particle swarm searching and can escape from local extremum. Therefore, the proposed algorithm improves the practicality of particle swarm optimization. As this is a general particle model, some techniques such as the local best version algorithm proposed in [29] can be used together with the new model. This will be explored in future work.

Acknowledgments

This work was supported by China/South Africa Research Cooperation Programme (nos. 78673 and CS06-L02), South African National Research Foundation Incentive Grant (no. 81705), SDUST Research Fund (no. 2010KYTD101), and Key scientific support program of Qingdao City (no. 11-2-3-51-nsh).

References

[1] J. C. Sprott, *Chaos and Time-Series Analysis*, Oxford University Press, New York, NY, USA, 2004.

[2] K. Aihara and G. Matsumoto, *Chaos in Biological Systems*, Plenum Press, New York, NY, USA, 1987.

[3] C. A. Skarda and W. J. Freeman, "How brains make chaos in order to make sense of the world," *Behavioral and Brain Sciences*, vol. 10, pp. 161–165, 1987.

[4] L. Wang, D. Z. Zheng, and Q. S. Lin, "Survey on chaotic optimization methods," *Computation Technology Automation*, vol. 20, pp. 1–5, 2001.

[5] B. Li and W. S. Jiang, "Optimizing complex functions by chaos search," *Cybernetics and Systems*, vol. 29, no. 4, pp. 409–419, 1998.

[6] J. J. Hopfield and D. W. Tank, "'Neural' computation of decisons in optimization problems," *Biological Cybernetics*, vol. 52, no. 3, pp. 141–152, 1985.

[7] Z. Wang, Y. Liu, K. Fraser, and X. Liu, "Stochastic stability of uncertain Hopfield neural networks with discrete and distributed delays," *Physics Letters A*, vol. 354, no. 4, pp. 288–297, 2006.

[8] T. Tanaka and E. Hiura, "Computational abilities of a chaotic neural network," *Physics Letters A*, vol. 315, no. 3-4, pp. 225–230, 2003.

[9] K. Aihara, T. Takabe, and M. Toyoda, "Chaotic neural networks," *Physics Letters A*, vol. 144, no. 6-7, pp. 333–340, 1990.

[10] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[11] L. Chen and K. Aihara, "Chaotic simulated annealing by a neural network model with transient chaos," *Neural Networks*, vol. 8, no. 6, pp. 915–930, 1995.

[12] L. Chen and K. Aihara, "Chaos and asymptotical stability in discrete-time neural networks," *Physica D*, vol. 104, no. 3-4, pp. 286–325, 1997.

[13] L. Wang, "On competitive learning," *IEEE Transactions on Neural Networks*, vol. 8, no. 5, pp. 1214–1217, 1997.

[14] L. Wang and K. Smith, "On chaotic simulated annealing," *IEEE Transactions on Neural Networks*, vol. 9, no. 4, pp. 716–718, 1998.

[15] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[16] J. Ke, J. X. Qian, and Y. Z. Qiao, "A modified particle swarm optimization algorithm,," *Journal of Circuits and Systems*, vol. 10, pp. 87–91, 2003.

[17] B. Liu, L. Wang, Y. H. Jin, F. Tang, and D. X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons and Fractals*, vol. 25, no. 5, pp. 1261–1271, 2005.

[18] T. Xiang, X. Liao, and K. W. Wong, "An improved particle swarm optimization algorithm combined with piecewise linear chaotic map," *Applied Mathematics and Computation*, vol. 190, no. 2, pp. 1637–1645, 2007.

[19] C. Fan and G. Jiang, "A simple particle swarm optimization combined with chaotic search," in *Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA '08)*, pp. 593–598, Chongqing, China, June 2008.

[20] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.

[21] S. H. Chen, A. J. Jakeman, and J. P. Norton, "Artificial Intelligence techniques: an introduction to their use for modelling environmental systems," *Mathematics and Computers in Simulation*, vol. 78, no. 2-3, pp. 379–400, 2008.

[22] J. J. Hopfield, "Hopfield network," *Scholarpedia*, vol. 2, article 1977, 2007.

[23] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 81, no. 10, pp. 2554–2558, 1984.

[24] Y. del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J. C. Hernandez, and R. G. Harley, "Particle swarm optimization:

basic concepts, variants and applications in power systems," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 171–195, 2008.

[25] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das, "A study of control parameters affectiong online performance of genetic algorithms for function optimization," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, pp. 51–60, 1989.

[26] A. I. de Freitas Vaz and E. M. da Graça Pinto Fernandes, "Optimization of nonlinear constrained particle swarm," *Technological and Economic Development of Economy*, vol. 12, no. 1, pp. 30–36, 2006.

[27] X. Hu, R. C. Eberhart, and Y. Shi, "Engineering optimization with particle swarm," in *Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 53–57, 2003.

[28] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245–1287, 2002.

[29] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," *Neural Networks*, vol. 18, pp. 205–217, 1997.