Noisy Chaotic Neural Networks With Variable Thresholds for the Frequency Assignment Problem in Satellite Communications

Lipo Wang, *Senior Member, IEEE*, Wen Liu, *Student Member, IEEE*, and Haixiang Shi

***Abstract*—We propose a novel approach, i.e., a noisy chaotic neural network with variable thresholds (NCNN-VT), to solve the frequency assignment problem in satellite communications. The objective of this NP-complete optimization problem is to minimize cochannel interference between two satellite systems by rearranging frequency assignments. The NCNN-VT model consists of $N \times M$ noisy chaotic neurons for an N-carrier M-segment problem. The NCNN-VT facilitates the interference minimization by mapping the objective to variable thresholds (biases) of the neurons. The performance of the NCNN-VT is demonstrated by solving a set of benchmark problems and randomly generated test instances. The NCNN-VT achieves better solutions, i.e., smaller interference with much lower computation cost compared to existing algorithms.**

***Index Terms*—Chaos, combinatorial optimization, frequency assignment problem (FAP), noisy chaotic neural networks (NCNN), NP-complete, variable thresholds.**

I. Introduction

COMMUNICATIONS satellites are a billion dollar technology, with applications ranging from weather forecasting to mobile telecommunications [1]. Nowadays, there is an increasing number of satellites in geostationary orbits. In order to accommodate crowded satellites in the same orbit, optimal frequency assignments are necessary to provide high-quality transmissions. In satellite communication systems, cochannel interference is the greatest problem that seriously affects the design and operation of the system [2]. The minimization of the cochannel interference has arisen as a major issue to deal with in satellite communications.

Early efforts have focused on various analytical methods for evaluations of the cochannel interference [3], [4], rather than systematic methods to optimize frequency assignments and to reduce cochannel interference. The later work of Mizuike and Ito [2] revealed the importance of a mathematical model for the reduction of interference. They formulated the cochannel interference reduction problem as a frequency assignment problem (FAP) which minimizes the largest and the total interference among carriers.

Manuscript received April 12, 2007; revised July 11, 2007. This paper was recommended by Associate Editor S.-M. Chen.

The authors are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore (e-mail: elpwang@ntu.edu.sg; liuw0004@ntu.edu.sg; pg02782641@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCC.2007.913915

The FAP exists in many areas, such as mobile communications, broadcast, and satellite communications [5]–[8], which require optimal assignments of limited frequency resources to a number of users. In this paper, we focus on the FAP in satellite communications, where frequencies of one set of carriers are to be rearranged while keeping the other set fixed.

The calculation of interference involves nonlinear terms. To avoid the treatment of nonlinearity, Mizuike and Ito [2] proposed a segmentation method that divides the commonly shared frequency band into a number of segments. Through segmentation, nonlinear terms can be evaluated in a linear manner. The FAP is then reduced to placing carriers into an integral multiple of unit segments.

The objectives of the FAP [9] in satellite communications are to minimize: 1) the largest interference of elements selected for the assignment and 2) the sum of interferences of all the selected elements. The FAP is proven to be an NP-complete combinatorial optimization problem [2]. Due to the NP-complete nature of this assignment problem, heuristic methods, especially neural networks, are commonly adopted.

Mizuike and Ito [2] used branch-and-bound to solve several practical problems, including both intersystem and intrasystem interference optimization problems. The branch-and-bound algorithm may fail when applied to large instances [9].

Funabiki and Nishikawa [9] presented a gradual neural network (GNN) that consists of $N \times M$ binary neurons for an N-carrier M-segment system with a gradual expansion scheme of activated neurons. The objective of the FAP can be achieved by searching from the neurons with the smallest interference. However, the multiphase searching inevitably leads to heavy computation especially for large problems.

Salcedo-Sanz *et al*. [10] presented a hybrid method that combines the Hopfield neural network (HNN) and simulated annealing (HopSA) to tackle the FAP. The HNN manages constraints, and simulated annealing improves the solution quality. The HopSA separates the objective from the constraints and is more scalable compared to other methods. However, the HopSA needs more computation to find optimal or near-optimal solutions compared with other neural network methods, e.g., the GNN, due to the nature of simulated annealing.

In this paper, we propose a noisy chaotic neural network with variable thresholds (NCNN-VT), which separates the optimization term from the constraint terms in the cost function by assigning different neurons with variable thresholds. The NCNN-VT obtains better solutions compared with the GNN [9] on the benchmark examples. Compared to the HopSA [10], the

1094-6977/$25.00 © 2008 IEEE

NCNN-VT finds optimal or suboptimal solutions with less computation cost. The NCNN-VT also achieves better solutions with fewer iterations compared to transiently chaotic neural networks (TCNNs) [11] and noisy chaotic neural networks (NCNNs) [12]. In an earlier paper, Wang and Ross [13] studied the influence of a variable neuronal threshold on fixed points and convergence rates of an associative neural network in the presence of noise.

We let a neuron of the NCNN-VT to have a larger bias input (the negative of the neuronal threshold) when the neuron presents a frequency assignment with smaller interference, so that the neuron is more likely to be selected for a frequency assignment. With this mapping scheme, optimization objectives of the problem are achieved by variable thresholds of the neural network, while the energy function is in charge of only constraints. As a result, weight tuning in the energy function becomes easier.

This paper is organized as follows. We review the NCNN and propose the NCNN-VT in Section II. The NCNN-VT formulation for the FAP is described in Section III. Instance generation, parameter settings, and simulation results are presented in Section IV. Finally, we conclude the paper in Section V.

II. Noisy Chaotic Neural Networks With Variable Thresholds

A. Noisy Chaotic Neural Networks

Wang *et al.* [12], [14], [15] proposed the NCNN by adding decaying stochastic noise into the TCNN [11]. Besides the chaotic nature of the TCNN, the NCNN is also stochastic; hence, it is alternatively known as stochastic chaotic simulated annealing (SCSA). The NCNN performs stochastic searching both before and after chaos disappears, and is more likely to find optimal or suboptimal solutions compared to both the TCNN and simulated annealing [12].

The NCNN model is described as [12]

$$x_{ij}(t) = \frac{1}{1+e^{-y_{ij}(t)/\varepsilon}} \tag{1}$$

$$y_{ij}(t+1) = k\, y_{ij}(t) - z(t)\,[x_{ij}(t) - I_0] + n(t) + \alpha \left[\sum_{p=1, p\neq i}^{N} \sum_{q=1, q\neq j}^{M} w_{ijpq} x_{pq}(t) + I_{ij} \right] \tag{2}$$

$$z(t+1) = (1-\beta_1) z(t) \tag{3}$$

$$A[n(t+1)] = (1-\beta_2) A[n(t)] \tag{4}$$

where x_{ij} and y_{ij} are output and internal state of neuron ij, respectively. w_{ijpq} is the connection weight from neuron ij to neuron pq, which satisfies the following conditions:

$$w_{ijpq} = w_{pqij} \quad w_{ijij} = 0$$

$$\sum_{p=1, p\neq i}^{N} \sum_{q=1, q\neq j}^{M} w_{ijpq} x_{pq}(t) + I_{ij} = -\partial E / \partial x_{ij}. \tag{5}$$

Furthermore, ε denotes the steepness parameter of the neuron activity function ($\epsilon > 0$), k is the damping factor of the nerve membrane ($0 \leq k \leq 1$), α is the positive scaling parameter for inputs, I_{ij} is an input bias, and $z(t)$ is the self-feedback neuronal connection weight [$z(t) \geq 0$]. I_0 is a positive bias; $n(t)$ denotes the random noise in the range $(-A[n], A[n])$ with a uniform distribution and $A[n]$ is the amplitude of the noise. Hence, the random noise gradually decays with time. β_1 and β_2 are damping factors for the time-dependent neuronal self-coupling and the random noise, respectively ($0 \leq \beta_1, \beta_2 \leq 1$). E denotes the energy function, whose minima correspond to optimal solutions of the problem. The connection weights between neurons are derived from the energy function [see (5)] to ensure that the energy function decreases monotonously as neurons update after both the noise and the chaos disappear.

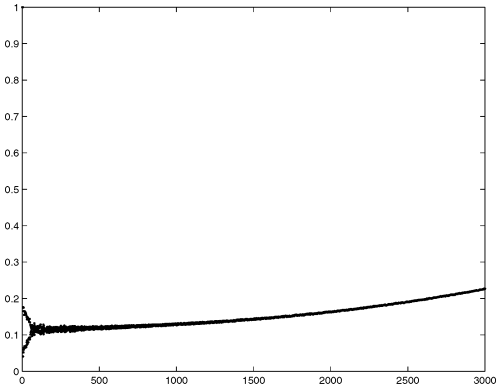
B. Objective Mapping Scheme

The dynamics of the single-neuron NCNN model [with only one neuron in (1) and (2)] for the first 3000 iterations with $I_0 = 0.1$, 0.3, 0.6, and 0.9 are shown in Fig. 1. Other parameters are set to be the same as in [12], i.e., $\epsilon = 0.004$, $k = 0.9$, $\alpha = 0.015$, $z(1) = 0.1$, $A[n(1)] = 0.02$, and $\beta_1 = \beta_2 = 0.001$. When $I_0 = 0.3$, the output of the neuron shows reversed period-doubling bifurcations to a fixed point. The bifurcation point from period 2 oscillations to fixed points is near $x(t) = 0.3$. When I_0 is set to 0.1, 0.6, and 0.9, this bifurcation point is around 0.1, 0.6, and 0.9, respectively. Equation (2) shows that when $t = +\infty$, $z = 0$, and the network output is independent of I_0. But the output at this bifurcation point depends on I_0, and as shown in Fig. 1, the network dynamics has not yet fully settled at $t = 3000$. As shown in our subsequent discussion, when we use the NCNN-VT to solve the FAP, we do not wait for the network to fully converge. Rather, we continuously check the validity of the solution and stop iterations as soon as a valid solution is found. As discussed later in Section III, neurons with output greater than the average output of the entire neural network are considered to be firing and also selected for frequency assignments. Hence, the larger the value of I_0, the more likely the neuron will be considered to be firing and selected for a frequency assignment.

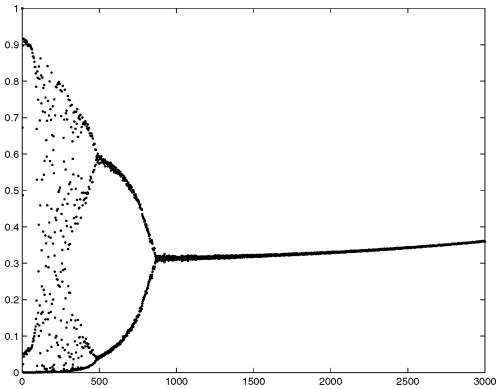
All previous methods, including the HNN [16], the TCNN [17], and the NCNN [12], combine the constraint satisfaction and the objective optimization in one energy function, which may result in poor performance in terms of solution quality and validity if weighting coefficients for various parts of the energy function are not tuned well. In particular, the balance between the constraint and the objective requires tedious trial-and-error approach. In this paper, we propose an NCNN-VT that handles only the constraint with the energy function and maps the objective to variable thresholds (biases) of the neurons.

In the NCNN-VT, we devise the bias I_0 to vary for different neurons according to the interference of the frequency assignment that the neuron represents. I_0 is then denoted as $I_{ij}^{(0)}$. The threshold affects the output of the neuron, and therefore, the likelihood that the neuron's output is above the average output of all neurons and the neuron is, thus, selected in frequency assignment. The value of $I_{ij}^{(0)}$ is calculated according to the objective of the optimization problem using a mapping function:

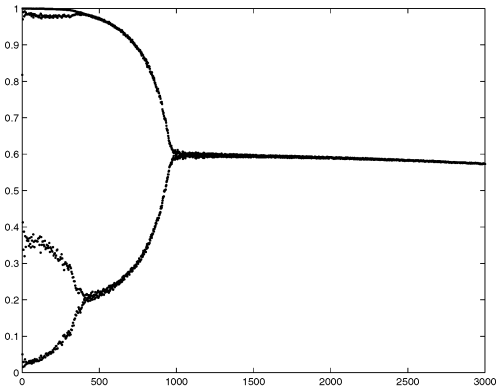
$$I_{ij}^{(0)} = f(d_{ij}), \qquad i = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, M \tag{6}$$



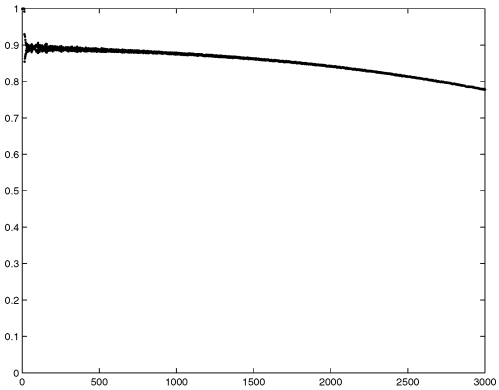
(a)



(b)

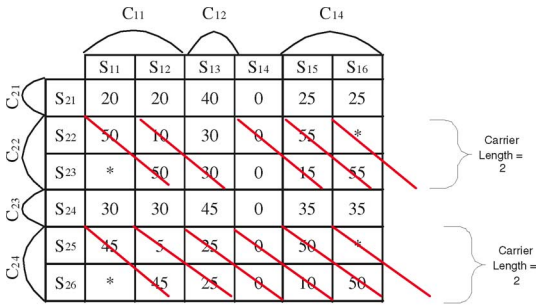


(c)

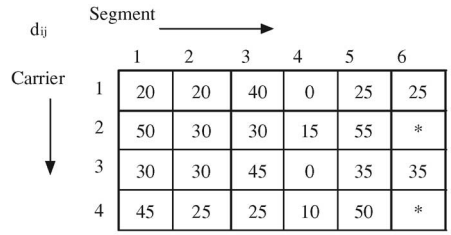


(d)

Fig. 1. Dynamics of the single-neuron NCNN with different values of I_0. The X-axis is the time t, and the Y-axis is the output of the neuron $x(t)$.



(a)



(b)

Fig. 2. Computation of cost matrix D from interference matrix E_I, extended from [9], which only showed (b). $*$ denotes infinity. (a) Interference matrix and computation method. For carriers with carrier length larger than 1, the maximum value of the crossed elements on the diagonal line represents the corresponding value in the cost matrix. (b) Cost matrix obtained from (a).

where d_{ij} is element ij in cost matrix D and represents the interference resulted by assigning carrier i to segment j.

The cost matrix $D = (d_{ij})$ is computed from the interference matrix E_I. The interference between the two systems (each with M segments) is described by an $M \times M$ interference matrix $E_I = (e_{ij})$. The (ij)th element e_{ij} represents the cochannel interference when segment i in system 2 uses a common frequency with segment j in system 1. Cost d_{ij} for neuron ij is given by the largest element in interference values $e_{lj}, e_{l+1,j+1}, \ldots, e_{l+c_i-1,j+c_i-1}$, where l is the first segment number of carrier i in the interference matrix and c_i is the length of carrier i [9]. Fig. 2 shows the way to compute the cost matrix from interference matrix E_I. If the carrier length for carrier i is 1, i.e., $c_i = 1$, then line i for carrier i in the cost matrix is the same as in the interference matrix for carrier i. If $c_i > 1$, then we choose the largest value in the diagonal line for each j, as shown in Fig. 2 [9]. C_{ij} $(i = 1, 2; j = 1, \ldots, 4)$ denotes the jth carrier in system i.

Objectives of the FAP are to minimize the largest element in the interference matrix selected for a frequency assignment, and at the same time, to minimize the sum of all selected elements of the interference matrix. Since neurons with smaller firing thresholds or greater biases are more likely to be selected in the final solution, we choose the mapping function $I_{ij}^{(0)}$ for the FAP as

$$I_{ij}^{(0)} = 1 - \frac{d_{ij} - d_{i,\min}}{d_{i,\max} - d_{i,\min}} = \frac{d_{i,\max} - d_{ij}}{d_{i,\max} - d_{i,\min}} \tag{7}$$

where $d_{i,\max}$ and $d_{i,\min}$ are the maximum and minimum values in row i of matrix D, respectively. Note that the maximum value of the cost matrix does not include infinity. Actually, the neuron corresponding to assignment with infinite interference will never fire due to its prohibitive cost.

Through the objective mapping scheme as described in (7) that maps the cost matrix to the thresholds in the neural network model, the NCNN-VT achieves optimization objectives of the FAP. Hence, the energy function needs to be concerned with only constraint terms. The separation of the objective term from the energy function will make the tuning of weighting coefficients in the energy function easier, i.e., there is no need to balance the optimization term with constraint terms. Moreover, it will improve the convergence speed compared with the NCNN, as shown in Section IV.

Thus, the dynamic equation of the NCNN-VT is

$$y_{ij}(t+1) = k y_{ij}(t) + \alpha \left[\sum_{p=1, p\neq i}^{N} \sum_{q=1, q\neq j}^{M} w_{ijpq} x_{pq}(t) + I_{ij} \right] - z(t)\left[x_{ij}(t) - I_{ij}^{(0)}\right] + n(t). \tag{8}$$

III. Application of the NCNN-VT to the FAP

A. Neural Network Formulation

As indicated in [2], each carrier in a satellite communications system can be divided into one or more consecutive unit segments. In order to reduce the cochannel interference between the two systems, the frequency assignments in system 2 are rearranged while the frequencies used in system 1 are fixed. In this way, the FAP is equivalent to assigning the carriers in system 2 to the segments in system 1.

In this paper, we use the same 2-D neural network formulation as in [9], which consists of $N \times M$ neurons for the FAP with N carriers and M segments. Fig. 3, which is extended from [9, Fig. 4], shows an example of a four-carrier six-segment problem. The output of each neuron x_{ij} will be converted into binary values x_{ij}^d that means the following:

$$\text{If } x_{ij}^d = \begin{cases} 1, & \text{carrier } i \text{ is assigned to segments } j(j + c_i - 1) \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Fig. 3(b) shows the convergence state, with black squares standing for the neurons with output $x_{ij}^d = 1$. We use the first segment in the carrier to denote the assignment $c_1 = c_3 = 1$, $c_2 = c_4 = 2$. The subsequent segments in this carrier are assigned to the consecutive segments. Fig. 3(c) shows the full assignment for each carrier by adding the consecutive assignments of subsequent segments in the same carrier. Fig. 3(d) shows the assignment of segments. Take carrier 4 in system 2 for example: it is assigned to segment 2 in system 1, as shown in Fig. 3(b). Since $c_4 = 2$, carrier 4 actually occupies two segments, as shown in Fig. 3(c). Carrier 4 is noted as segments 5 and 6 in system 2. Hence, segments 5 and 6 are assigned to segments 2 and 3 in system 1, respectively, as shown in Fig. 3(d). We can similarly expand from the convergence state Fig. 3(b) to Fig. 3(c) and (d) as the carrier length for each carrier is given. We only provide the solution format in Fig. 3(b) in this paper.

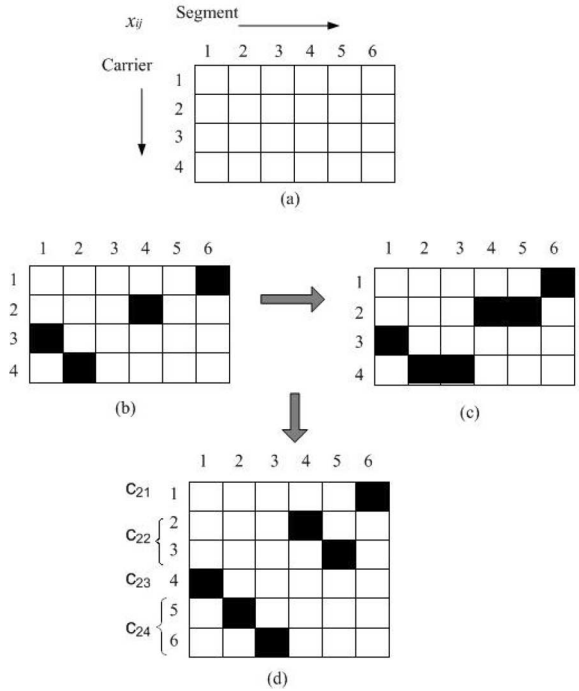


Fig. 3. Neural network formulation for the FAP [extended from [9, Fig. 4], which only consists of (a) and (b)]. (a) Twenty-four neurons for the 4-carrier-6-segment FAP. (b) Convergence state of the neural network. (c) Full assignment of the carriers. (Carrier length $c_2 = c_4 = 2$.) (d) Assignment of segments.

B. Energy Function

There are two constraints to be formulated into the energy function for the FAP.

1) Every segment in system 2 must be assigned to one and at most one segment in system 1.
2) All segments of one carrier in system 2 should be assigned to consecutive segments in system 1 in the same order.

According to Funabiki and Nishikawa [9], the energy function corresponding to above constraints is as follows:

$$E_1 = \sum_{i=1}^{N} \left(\sum_{j=1}^{M} x_{ij} - 1 \right)^2 \tag{10}$$

$$E_2 = \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{p=1, p\neq i}^{N} \sum_{q=j-c_p+1}^{j+c_i-1} x_{ij} x_{pq}. \tag{11}$$

If carrier i is assigned to segment j, any other carrier must not be assigned to segments from j to $(j + c_i - 1)$. The first segment of carrier p $(p \neq i)$ should be assigned to the segment before $(j - c_p + 1)$ or after $(j + c_i - 1)$. As $(j - c_p + 1)$ may be negative and $(j + c_i - 1)$ may exceed the total number of the segments, i.e., M, the formulation in (11) has errors and produces program bugs during simulations. We revised the second

term of the energy function as

$$E_2' = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{p=1}^{N}\sum_{p\neq i}\sum_{q=\max(j-c_p+1,1)}^{\min(j+c_i-1,M)} x_{ij}x_{pq} \qquad (12)$$

where $\max(x,y)$ is the larger value between (x,y) and $\min(x,y)$ is the smaller value between (x,y).

We add the following convergence term into the energy function to force the neuron outputs approach to 0 or 1 [18]:

$$E_3 = \sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij}(1-x_{ij}). \qquad (13)$$

The total energy function of the NCNN-VT is given by the summation of three parts E_1, E_2', and E_3 as

$$E = \frac{W_1}{2}E_1 + \frac{W_2}{2}E_2' + \frac{W_3}{2}E_3 \qquad (14)$$

where W_1, W_2, and W_3 are weighting coefficients. The choices of W_1, W_2, and W_3 are based on the rule that all terms in the energy function should be comparable in magnitude, so that none of them dominates [12]. The balance of each term in the energy function is crucial to parameter selection.

C. Neural Dynamics

From (5), (8), and (14), the dynamic equation for the NCNN-VT can be obtained

$$\begin{aligned} y_{ij}(t+1) = ky_{ij}(t) + \alpha\Bigg[& -W_1\left(\sum_{j=1}^{M} x_{ij} - 1\right) \\ & - W_2\left(\sum_{p=1}^{N}\sum_{p\neq i}\sum_{q=\max(j-c_p+1,1)}^{\min(j+c_i-1,M)} x_{pq}\right) \\ & - \frac{W_3}{2}(1-2x_{ij})\Bigg] \\ & - z(t)\left[x_{ij}(t) - I_{ij}^{(0)}\right] + n(t). \end{aligned} \qquad (15)$$

The neuron output is continuous between 0 and 1. We convert the continuous output x_{ij} of neuron ij to discrete neuron output x_{ij}^d as follows [17]:

$$x_{ij}^d = \begin{cases} 1, & \text{if } x_{ij} > \frac{1}{NM}\sum_{p=1}^{N}\sum_{q=1}^{M} x_{pq}(t) \\ 0, & \text{otherwise.} \end{cases} \qquad (16)$$

The NCNN-VT is updated cyclically and asynchronously. The new state information of a neuron is immediately available for the other neurons in the next iteration. The iteration is terminated once a feasible assignment is obtained or the computation step exceeds the predefined maximum number of iteration steps (15 000 in our simulations).

D. Computational Cost of the NCNN-VT

For an N-carrier-M-segment FAP, each neuron updates once in an iteration step. Hence, our algorithm has the worst time cost of $O(NM)$ in one iteration. However, it is difficult to determine the exact number of iterations required for different problem instances with different problem sizes. For large-size problems, the number of iterations to obtain a feasible solution will be greater than for small-size problems. This is also true for computation time. We will include the computation time for each instance simulated in this paper in Section IV.

TABLE I
SPECIFICATIONS OF THE FAP INSTANCES USED IN THE SIMULATION

Instance	Number of carriers N	Number of segments M	Range of carrier length	Range of interference
BM 1	4	6	1 - 2	5 - 55
BM 2	4	6	1 - 2	1 - 9
BM 3	10	32	1 - 8	1 - 10
BM 4	10	32	1 - 8	1 - 100
BM 5	10	32	1 - 8	1 - 1000
BM 6	18	60	1 - 10	1 - 100
BM 7	30	100	1 - 10	1 - 100
BM 8	15	50	1 - 8	1 - 1000
Case 9	50	200	1 - 10	1 - 10
Case 10	50	200	1 - 10	1 - 100
Case 11	50	200	1 - 10	1 - 1000
Case 12	80	200	1 - 5	1 - 100
Case 13	80	300	1 - 8	1 - 100
Case 14	80	400	1 - 10	1 - 100
Case 15	80	450	1 - 12	1 - 100
Case 16	80	500	1 - 14	1 - 100
Case 17	80	600	1 - 16	1 - 100
Case 18	100	500	1 - 10	1 - 100
Case 19	150	400	1 - 5	1 - 100
Case 20	200	300	1 - 2	1 - 100

IV. SIMULATION RESULTS AND DISCUSSIONS

We implement the NCNN-VT algorithm in C language and simulate on a 16-node dual Xeon 3.06-GHz (Intel IA32) Linux cluster with a parallel C/C++ compiler and a toolkit NPACI ROCKS v3.0.

The specifications of the 20 instances are listed in Table I. Benchmarks BM 1 to BM 5 are from [9], where these were called instances 1–5, respectively. Benchmarks BM 6 to BM 8 are from [10], where these were called problems 3–5, respectively.

Cases 9–20 are newly randomly generated in this paper to evaluate the performance of the NCNN-VT in large-size FAP problems and are generated in the following steps. First, we choose the number of carriers N and the number of segments M for the instance. Next, we select the values of the range of carrier length and the range of interference between the two systems. In this paper, in order to test the scalability of the NCNN-VT, we gradually increased these two parameters. And finally, we generate a set of carrier lengths c_i $(i = 1, \ldots, N)$ and interference matrix E_I $(M \times M)$ using uniformly distributed random values in the range that we have defined. We generate three groups of random instances, as shown in Table I. Group 1 (cases 9–11) is generated to observe the influence of the magnitude of the interference by varying the interference while the number of carriers, the number of segments, and the range of carrier length are fixed. Group 2 (cases 12–17) is designed to show the effects of the carrier length. And group 3 (cases 18–20) is used to show the ability of the proposed NCNN-VT to deal with a large number of carriers.

TABLE II
PARAMETER SETTING OF THE NCNN-VT APPROACH FOR DIFFERENT INSTANCES, WITH $k = 0.9$, $\epsilon = 1/250$, $\alpha = 0.015$, $z(0) = 0.08$, $\beta_1 = 0.001$, AND $W_1 = 1.0$

Instance	W_2	W_3	$A[n(0)]$	β_2
1	1.0	0.7	0.02	0.001
2	1.0	0.7	0.02	0.001
3	0.3	0.7	0.02	0.001
4	0.3	0.7	0.02	0.001
5	0.4	0.7	0.02	0.0001
6	0.2	0.6	0.02	0.0001
7	0.2	0.6	0.01	0.0001
8	0.1	0.3	0.02	0.0001
9	0.2	0.4	0.02	0.0001
10	0.2	0.6	0.02	0.0001
11	0.2	0.6	0.01	0.0001
12	0.2	0.4	0.01	0.0001
13	0.2	0.4	0.01	0.0001
14	0.1	0.4	0.01	0.0001
15	0.1	0.5	0.01	0.0001
16	0.1	0.5	0.01	0.0001
17	0.1	0.5	0.01	0.0001
18	0.1	0.6	0.02	0.0001
19	0.1	0.6	0.02	0.0001
20	0.2	0.4	0.02	0.0001

TABLE III
PERFORMANCE OF THE NCNN-VT ON 20 INSTANCES

Instance	Largest interference			Total interference	Iterations mean±SD
	Best	Opt rate %	Average error		
1	30	100	0	100	378±16.8
2	4	100	0	13	478±22.6
3	7	100	0	85	1901±96.5
4	64	88.3	1.74	880	2157±252
5	640	37.1	37.3	7246	1819±156
6	35	25.8	4.78	1000	2016±484
7	61	10.4	18.1	2779	2891±465
8	695	26.6	182	15373	3769±402
9	9	73.8	0.8	618	4019±359
10	85	64.8	5.9	5187	4245±363
11	797	38.4	113.9	48951	4281±357
12	70	45.5	5.8	3418	4192±303
13	73	63.7	3.4	6183	4315±369
14	93	49.7	4.2	10261	4126±311
15	89	67.2	3.3	12475	4655±327
16	81	29.8	6.8	14272	4558±310
17	91	42.6	8.6	20731	4820±329
18	87	72.5	9.2	14629	5671±485
19	85	46.4	13.9	9836	5281±461
20	36	76.3	6.2	14264	5649±716

"Opt rate" stands for the percentage at which the NCNN-VT reached the optimum in the convergent runs. "SD" stands for "standard deviation."

A. Parameter Selection

The NCNN-VT approach has two types of parameters to tune: parameters for the NCNN-VT model and weighting coefficients in the energy function. We choose parameters for the NCNN-VT model such that the neural network produces rich and flexible neurodynamics. The choices of these parameters are similar to those used in other optimization problems [14], [15].

The selection of weighting coefficients is based on the rule that all terms in the energy function should be comparable in magnitude. We list the parameters used for each problem in this paper in Table II. $I_{ij}^{(0)}$ is computed from (7).

Parameters listed here are empirical, and tuning these parameters is necessary for different problems. Our experiences show that the neural network model parameters do not vary much with problems, whereas the weighting coefficients (W_1, W_2, W_3) can be slightly more sensitive to problems.

B. Simulation Results

We run the NCNN-VT on each instance 1000 times with different randomly generated initial neuron states. We end the neuron update when a valid solution is found, i.e., when E_1 and E_2 vanish. Table III shows results, including the best largest interference, the percentage to reach the optimum (Opt rate), the average error from the optimal result, and the total interference when the optimum of the largest interference is found. The average numbers of iteration steps and standard deviations are also shown in this table. The percentage at which the NCNN-VT reaches the optimum is evaluated only in the runs in which valid solutions are found. Furthermore, as shown in this table, the NCNN-VT finds a feasible solution in nearly a constant number of iteration steps.

Table IV compares results on the largest and total interference obtained by the NCNN-VT, the GNN [9], and the HopSA [10]. Results of the GNN are from [9] and [10]. Results of the HopSA on benchmarks BM 1 to BM 8 are also from [10]. Since the average value of the largest and total interference was not presented in [10], only the best value is included in Table IV. For instances 9–20, we rerun the HopSA algorithm from [10]. For benchmark problems from BM 1 to BM 8, the NCNN-VT matches or improves results of the GNN and HopSA. Compared with the GNN, the NCNN-VT obtains the same optimal results in BM 1 to BM 4, but the NCNN-VT is more capable to reach the optima; hence, it has better average performance over the GNN. For problems BM 5 to BM 8, the NCNN-VT obtains better results both in terms of the largest interference and the total interference. Compared with the HopSA, the NCNN-VT obtains smaller largest interference than the HopSA [10] did in all the benchmarks while maintaining comparable total interference. Between the two optimization objectives of the FAP, the minimization of the largest interference is critical and has a higher priority over the minimization of the total interference [9].

We further compare the HopSA with the NCNN-VT on randomly generated instances 9–20. We simulate the HopSA algorithm under the same simulation environment as the NCNN-VT. Results are summarized in Table V and VI, which show that the HopSA needs several hours of computation for a 50-carrier 200-segment problem. The HopSA fails to obtain solutions in instances with larger size due to the excessive computation time. "N/A" in Table V represents situations that the algorithm cannot find solutions for in one month. In comparison, our proposed NCNN-VT finds a feasible solution in less than 1 h of CPU time on all the instances.

We listed the number of iteration steps and computation time of the NCNN-VT and the HopSA [10] on all the instances in Table VI together with the convergence rate (the ratio at which the neural network finds a feasible solution). We consider the algorithm to be nonconvergent if a solution is not found in 15 000

TABLE IV
COMPARISON OF THE LARGEST INTERFERENCE AND TOTAL INTERFERENCE OBTAINED BY THE NCNN-VT, GNN, AND HOPSA FOR BENCHMARKS BM 1 TO BM 8

Instance	GNN [9]		HopSA [10]		NCNN-VT	
	Largest (Best/ Ave)	Total (Best/ Ave)	Largest Best	Total Best	Largest (Best/ Ave / SD)	Total (Best/ Ave / SD)
BM 1	30/ 31.5	100/ 100.8	30	100	30 / 30 / 0	100 / 100 / 0
BM 2	4/ 4.9	13/ 15.4	4	13	4 / 4 / 0	13 / 13.7 / 1.2
BM 3	7/ 8.1	85/ 99.4	7	85	7 / 7 / 0	85 / 89.9 / 3.8
BM 4	64/ 77.1	880/ 982.0	64	888	64 / 65.7 / 2.6	880 / 903.6 / 35.1
BM 5	640/ 766.8	8693/ 9413.9	817	6910	640 / 677 / 71.1	7246 / 8445 / 656.7
BM 6	49	1218	45	1080	35 / 39.8 / 4.8	1000 / 1068 / 40.5
BM 7	100	4633	98	3396	61 / 79.9 / 7.8	2779 / 2995 / 124
BM 8	919	16192	741	13178	695 / 877 / 64.8	15373 / 18034 / 318

Results are shown as the best, average values, and standard deviation (Best/Ave/SD).

TABLE V
COMPARISONS OF THE LARGEST INTERFERENCE AND TOTAL INTERFERENCE ON RANDOMLY GENERATED INSTANCES BETWEEN THE HOPSA [10] AND THE NCNN-VT

#	HopSA		NCNN-VT	
	Largest	Total	Largest	Total
9	10.5±0.6	658.3±15.3	9.8±0.4	627.3±12.9
10	124.3±10.2	6533±123.5	90.9±5.8	5374±60.4
11	989.2±73.5	49982±150.5	910.9±64.4	49388±120.2
12	N/A	N/A	75.8±2.5	3437±74.3
13	N/A	N/A	76.0±11.4	6295±151.8
14	N/A	N/A	97.2±0.8	10827±351.2
15	N/A	N/A	92.3±6.4	12644±273.7
16	N/A	N/A	87.0±8.4	14385±283.8
17	N/A	N/A	99.7±0.7	21894±776.4
18	N/A	N/A	96.9±3.1	14841±628.9
19	N/A	N/A	98.8±1.8	9924±387.9
20	N/A	N/A	42.9±22.3	14488±434.3

Results are displayed as mean ± SD (standard deviation).

TABLE VI
COMPARISON BETWEEN THE HOPSA AND THE NCNN-VT ON COMPUTATION TIME IN SECONDS AND CONVERGENCE RATE

#	HopSA		NCNN-VT	
	Time (Sec.) mean±SD	η (%)	Time (Sec.) mean±SD	η (%)
1	1.0±0.0	100	0.02±0.1	100
2	1.0±0.0	100	0.02±0.1	100
3	33.0±0.0	100	0.31±0.04	100
4	33.8±0.8	100	0.36±0.05	100
5	33.0±0.5	100	0.37±0.04	100
6	1529±156	100	11.8±0.6	93.6
7	2845±528	100	78.3±13.3	86.0
8	3738±615	75	143.2±12.5	80.8
9	5389±823	80	133.6±2.3	96.6
10	8916±1529	70	172.7±5.8	100
11	15544±2117	70	187.5±7.3	100
12	N/A	N/A	271.8±27.1	100
13	N/A	N/A	1034.0±56.1	93
14	N/A	N/A	1655±207.2	91
15	N/A	N/A	1852±353.9	100
16	N/A	N/A	1959±118.1	79
17	N/A	N/A	2621±118.4	78
18	N/A	N/A	2989±200.7	80
19	N/A	N/A	3550±233.3	85
20	N/A	N/A	2477±14.0	88

η stands for the convergence rate.

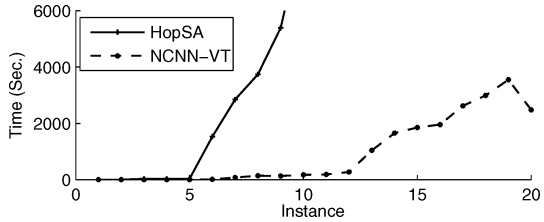


Fig. 4. Computation time of the HopSA and NCNN-VT for the 20 instances.

steps. One iteration step in the NCNN-VT means one loop that all neurons are updated. The computation time is measured in seconds. Also, the computation time is calculated only for runs in which feasible solutions are found. Tables VI shows that our NCNN-VT achieves at least a 78% convergence rate. Parameter selection and instance complexity affect the convergence rate. We attempted to adjust the parameters so as to improve the convergence rates; however, it was not possible to achieve 100% for some large problems (see Table VI). This may be due to the high complexity in some large problems.

We plot the computation time of the HopSA and the NCNN-VT in Fig. 4, which shows that compared to the HopSA, the computation time of the NCNN-VT increases more slowly as the problem size increases. As the HopSA needs several hours to find a solution for instances 9–11, the NCNN-VT needs no more than 200 s.

We have also solved the same FAPs using the TCNN and NCNN, with the energy function that consists of both the constraint terms and the objective term [2]

$$E_4 = W_4 \sum_{i=1}^{N} \sum_{j=1}^{M} d_{ij} x_{ij}. \tag{17}$$

Table VII shows the comparison between the TCNN [17], NCNN [12], and NCNN-VT on benchmarks BM 1 to BM 8. For each algorithm on each instance, we list the convergence rate η, the average number of iteration steps T, and the best and average values of the largest and total interference, respectively. It is shown that the NCNN-VT needs fewer iterations compared with the TCNN and NCNN. In addition, the NCNN-VT achieved better solutions, i.e., smaller largest interference and total interference. Furthermore, the NCNN-VT has higher convergence rate compared to the NCNN and TCNN. The advantage of the NCNN-VT grows as the problem size increases.

The energy functions of the NCNN and TCNN have an additional term, i.e., optimization term E_4, which is needed to fulfill the optimization objective of the FAP. With this optimization term in the energy function, the tuning of weighting coefficients

TABLE VII
COMPARISON OF THE TCNN, NCNN, AND NCNN-VT ON BENCHMARKS BM 1 TO BM 8 IN 1000 RUNS

#	TCNN				NCNN				NCNN-VT			
	η (%)	T	I_L (Best/ Ave)	I_T (Best/ Ave)	η (%)	T	I_L (Best/ Ave)	I_T (Best/ Ave)	η (%)	T	I_L (Best/ Ave)	I_T (Best/ Ave)
1	100	426.5	30/ 35.4	105/ 112.6	100	393.2	30/ 32.0	100/ 100.0	100	378.3	30/ 30.0	100/ 100.0
2	100	829.4	6/ 7.6	17/ 21.7	100	766.4	6/ 6.8	17 / 18.2	100	478.1	4/ 4.0	13/ 13.7
3	92.4	2485	8/ 10.9	112/ 142.2	98.8	2242	9/ 9.9	114/ 163	100	1901	7/ 7.0	85/ 89.9
4	87.5	2383	84/ 96.2	971/ 1145	97.3	2069	78/ 97.1	1025 /1444	100	2157	64/ 65.7	880/ 903.6
5	61.0	2342	697/ 871	7574/ 9927	92.6	1994	674/ 945	9279/ 11613	100	1819	640/ 677	7246/ 8445
6	53.7	2651	42/ 46.8	1154/ 1491	80.0	2115	47/ 49.1	1057/ 1231	93.6	2016	35/ 39.8	1000/ 1068
7	67.1	3716	91/ 97.3	3296/ 3826	85.9	3225	88/ 95.2	3173/ 3501	86.0	2891	61/ 79.9	2779/ 2995
8	52.2	4839	836/ 869	17659/ 20493	70.1	4021	800/ 835	16321/ 17342	80.8	3769	695/ 877	15373/ 18034

# denotes the instance number, η is the convergence rate, T is the average number of iteration steps, I_L is the largest interference, and I_T is the total interference. The interference is shown as the best and average values (Best/Ave).

will be more difficult because E_1–E_3 are much smaller compared to E_4 [see (17)], which makes the final solution very sensitive to the choice of W_4. The magnitude of E_4 is related to cost matrix D that is problem-specific. Hence, E_4 may vary greatly when the range of interference in the simulation changes from 1–100 to 1–1000. On the contrary, if we separate the optimization term from constraint terms as in our NCNN-VT approach, the balance among the remaining terms in the energy function and the selection of weighting coefficients become simpler and easier.

V. CONCLUSION

In this paper, we have proposed a novel approach, i.e., *NCNN-VT*, to solve the FAP in satellite communications. The NCNN-VT consisted of $N \times M$ neurons for an N-carrier M-segment system. We used a linear mapping of thresholds to separate the optimization objectives from constraints, thereby simplifying the formulation of the energy function and the selection of weighting coefficients. The objective mapping scheme achieves the objectives of the FAP, and the update rule for neurons helps to find a feasible solution satisfying constraints through the complex neurodynamics (stochastic noise and flexible chaos) of the NCNN-VT model. Simulation results on the 20 instances show that the NCNN-VT obtains better solutions with low computational cost compared to the previous methods.

ACKNOWLEDGMENT

The authors sincerely thank the editors and reviewers for carefully reading the paper and for many constructive comments that have helped to significantly improve the paper. They also thank Salcedo-Sanz *et al*. for giving the source code of the HopSA.

REFERENCES

[1] D. J. Whalen. Communications satellites: Making the global village possible (2007, Jul.) [Online]. Available: http://www.hq.nasa.gov/office/pao/History/satcomhistory.html

[2] T. Mizuike and Y. Ito, "Optimization of frequency assignment," *IEEE Trans. Commun.*, vol. 37, no. 10, pp. 1031–1041, Oct. 1989.

[3] B. Pontano, J. Fuenzalida, and N. Chitre, "Interference into angle-modulated systems carrying multichannel telephony signals," *IEEE Trans. Commun.*, vol. COM-21, no. 6, pp. 714–727, Jun. 1973.

[4] M. Jeruchim, "A survey of interference problems and applications to geostationary satellite networks," *Proc. IEEE*, vol. WC-65, no. 3, pp. 317–331, Mar. 1977.

[5] R. A. Murphey, P. M. Pardalos, and M. G. C. Resende, *Frequency Assignment Problems*, suppl. vol. A. Norwell, MA: Kluwer, 1999.

[6] R. Montemanni, J. N. Moon, and D. H. Smith, "An improved tabu search algorithm for the fixed-spectrum frequency-assignment problem," *IEEE Trans. Veh. Technol.*, vol. 52, no. 4, pp. 891–901, May 2003.

[7] S. Salcedo-Sanz and C. B. Calzón, "A hybrid neural-genetic algorithm for the frequency assignment problem in satellite communications," *Appl. Intell.*, vol. 22, pp. 207–217, May 2005.

[8] H. Shi and L. P. Wang, "An adaptive noisy chaotic neural network approach for frequency assignment in satellite communications systems," in *Proc. 12th Int. Conf. Neural Inf. Process. (ICONIP 2005)*, Taipei, Taiwan, Oct. 30–Nov. 2, pp. 150–153.

[9] N. Funabiki and S. Nishikawa, "A gradual neural-network approach for frequency assignment in satellite communication systems," *IEEE Trans. Neural Netw.*, vol. 8, no. 6, pp. 1359–1370, Nov. 1997.

[10] S. Salcedo-Sanz, R. Santiago-Mozos, and C. B. Calzón, "A hybrid hopfield network-simulated annealing approach for frequency assignment in satellite communications systems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 1108–1116, Apr. 2004.

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

[12] L. P. Wang, S. Li, F. Y. Tian, and X. J. Fu, "A noisy chaotic neural network for solving combinatorial optimization problems: Stochastic chaotic simulated annealing," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 5, pp. 2119–2125, Oct. 2004.

[13] L. P. Wang and J. Ross, "A variable threshold as a model of selective attention, (de)sensitization, and anesthesia in associative neural networks," *Biol. Cybern.*, vol. 64, pp. 231–241, 1991.

[14] L. P. Wang and K. Smith, "On chaotic simulated annealing," *IEEE Trans. Neural Netw.*, vol. 9, no. 4, pp. 716–718, Jul. 1998.

[15] L. P. Wang and F. Y. Tian, "Noisy chaotic neural networks for solving combinatorial optimization problems," in *Proc. Int. Joint Conf. Neural Netw.*, Como, Italy, Jul. 24–27, 2000, vol. 4, pp. 37–40.

[16] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Natl. Acad. Sci. USA*, vol. 81, pp. 3088–3092, 1984.

[17] M. Yamguti Ed., *Transient Chaotic Neural Networks and Chaotic Simulated Annealing*. Amsterdam, The Netherlands: Elsevier, 1994.

[18] H. E. Rauch and T. Winarske, "Neural networks for routing communication traffic," *IEEE Control Syst. Mag.*, vol. 8, no. 2, pp. 26–31, Apr. 1988.

Lipo Wang (M'97–A'97–SM'98) received the B.S. degree from the National University of Defense Technology, Changsha, China, in 1983, and the Ph.D. degree from Louisiana State University, Baton Rouge, in 1988.

He is currently with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His current research interests include computational intelligence, with applications to data mining, bioinformatics, and optimization. He is the author or coauthor of more than 170 papers published in journals and conferences. He holds a U.S. patent in neural networks. He has authored two monographs and has edited 20 books. Since 2002, he has been an Area Editor of the *Soft Computing* journal. He has also been a member of the Editorial Boards of ten international journals and three other

journals. Since 1998, he has been an Associate Editor for the journal *Knowledge and Information Systems*.

Dr. Wang was a keynote/plenary/panel speaker for several international conferences. Since 2002, he has been an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS; since 2003, for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION; and since 2005, for the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. He is the Vice President—Technical Activities of the IEEE Computational Intelligence Society (2006–2007) and has served as the Chair of the Emergent Technologies Technical Committee (2004–2005). He has been on the Governing Board of the Asia-Pacific Neural Network Assembly since 1999 and served as its President during 2002–2003. He was the Founding Chair of both the IEEE Engineering in Medicine and Biology Chapter Singapore and the IEEE Computational Intelligence Chapter Singapore. He also serves/served as the General/Program Chair for 11 international conferences and as a member of the steering/advisory/organizing/program committees of over 130 international conferences.

Wen Liu (S'07) received the B.S. degree in electronic engineering from Peking University, Beijing, China, in 2004. She is currently working toward the Ph.D. degree at the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

Her research interests include combinatorial optimization in communications using the neural network.

Haixiang Shi received the B.S. degree in electronic engineering from Xidian University, Xi'an, China, in 1998. He is currently working toward the Ph.D. degree at the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

His research interests include combinatorial optimization using computational intelligence, with applications to routing, assignment, and scheduling problems in wireless *ad hoc* and sensor networks.