SATELLITE FREQUENCY ASSIGNMENTS USING TRANSIENTLY CHAOTIC NEURAL NETWORKS WITH VARIABLE THRESHOLDS

Wen LIU, Lipo WANG, Haixiang SHI

College of Information Engineering, Xiangtan University,
Xiangtan, Hunan, China.
School of Electrical and Electronic Engineering,
Nanyang Technological University, Block S1, 50 Nanyang Avenue,
Singapore 639798.
E-mail: {liuw0004, elpwang, pg02782641}@ntu.edu.sg

Abstract

The objective of the satellite frequency assignment problem (FAP) is to minimize the cochannel interference between two satellite systems by rearranging the frequency assignments. This NP-complete problem is difficult to solve, especially for large-size problems, and is yet growing in importance, since we increasingly depend on satellites to fulfill our communications needs. In this paper, we propose a transiently chaotic neural network with variable thresholds (TCNN-VT) by letting the threshold of a neuron vary with the interference of the assignment which the neuron represents. We apply this new model on the FAP in satellite communications.

Keywords: Combinatorial optimization, TCNN, variable threshold, chaos, frequency assignment problem, FAP.

1 Introduction

Frequency assignment problems (FAP) [1, 2, 3] appear in various areas, including mobile telecommunications, broadcasting, military operations, and satellite communications. Among diverse formulations and objectives of the

FAP, we focus on minimization of system interference for fixed frequency assignments in satellite communications. Interference in satellite communications depends on transmitter power, channel loss, receiver sensitivity, and antenna gains. Frequency rearrangements are an effective complement alongside with the reduction of interference itself.

This paper continues to use the FAP formulation given by [4] and used in [5, 6, 7]. Mizuike and Ito [4] proposed segmentation of frequency band to avoid the treatment of the nonlinearity and presented a method based on the branch-and-bound approach. The carrier is uniformly divided to a collection of consecutive unit segments whose width can be set arbitrarily. The application of their method on both the intersystem problem and intrasystem problem showed that the method was helpful in the satellites orbit separation and noise reduction for carriers [4].

Funabiki and Nishikawa [5] solved the FAP with a gradual neural network (GNN) which they proposed, where cost optimization is achieved by a gradual expansion scheme and a binary neural network is in charge of the satisfaction of constraints. In the gradual expansion scheme, neurons are classified into P groups in the order of the cost which firing of the neuron will cause. They then activate one additional group of neurons in each phase until a solution is found or all P groups have been added. Hence, optimization is obtained by gradually increasing the number of activated neurons in the order of cost. Salcedo-Sanz *et al.* combined the Hopfield network with simulated annealing [6] and the genetic algorithm [7] to solve the FAP in satellite communications. The Hopfield network with simulated annealing (HopSA) [6] and the neural-genetic algorithm (NG) [7] are claimed to be more scalable than previous algorithms. However, as a kind of hybrid algorithms, there is an increase in the computational cost of the HopSA and the NG compared with the GNN [5, 6].

Nozawa [8, 9] modified a Hopfield network model with a negative self-feedback connection and obtained a neural network model as the globally coupled map (GCM). The solution of the traveling salesman problem (TSP) showed that chaotic search by the GCM model is efficient in approaching the global optimum or sub-optima. Chen and Aihara [10] proposed a transiently chaotic neural network (TCNN) by introducing transiently chaotic dynamics into the Hopfield neural network through a decaying negative self-feedback. The TCNN, which is also known as chaotic simulated annealing (CSA) [11], is not a problem-specific but a general method for combinatorial optimization problems [12, 13, 14]. With autonomous decreases of the self-feedback connection, TCNNs are more effective in solving combinatorial optimization problems compared to the HNN. He *et al.* [15] used the TCNN combined with a multistage self-organizing algorithm to solve the cellular channel assignment problem where the TCNN is in charge of searching for the optimal assignment.

In this paper, we further develop the TCNN by proposing a transiently chaotic neural network with variable thresholds (TCNN-VT). The thresholds are designed to minimize the largest interference after frequency rearrangements.

This paper is organized as follows. We review the TCNN and propose the TCNN-VT in Section 2. The formulation of the TCNN-VT on the FAP is described in Section 3. Instances generating, parameters settings, and simulation results are presented in Section 4. Finally, we conclude the contribution of this paper in Section 5.

2 Transiently Chaotic Neural Networks with Variable Thresholds

The TCNN [10] model is described as follows:

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

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

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

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 is confined to the following conditions [16]:

$$\sum_{p=1,p\neq i}^{N}\sum_{q=1,q\neq j}^{M} w_{ijpq}x_{pq}(t) + I_{ij} = -\partial E/\partial x_{ij} \tag{4}$$

Furthermore, ε denotes the steepness parameter of the neuron activity function ($\varepsilon > 0$). k is the damping factor of the nerve membrane ($0 \le k \le 1$). α is the positive scaling parameter for inputs. I_{ij} is an input bias of neuron ij. $z(t)$ is the self-feedback neuronal connection weight ($z(t) \ge 0$). β is the damping factor for the time-dependent neuronal self coupling ($0 \le \beta \le 1$). E denotes the energy function, whose minimization corresponds to optimal solutions of the combinatorial optimization problem. The connection weights between neurons are derived from the energy function by Equation (4) so that the energy function will decrease monotonously as neurons update after the self-feedback interaction vanishes ($z = 0$).

In the original TCNN [10], I_0 is a constant positive bias in the self-feedback term. We propose the TCNN-VT by varying the threshold with the interference of the assignment which firing of the neuron represents and denote it as $I_{ij}^{(0)}$ ($0 \leq I_{ij}^{(0)} < 1$).

$$I_{ij}^{(0)} = 1 - \frac{d_{ij}}{d_{i,max}} \tag{5}$$

where d_{ij} is the element on row i column j of the cost matrix D, and $d_{i,max}$ is the maximum value in row i of the matrix D. Cost matrix $D = (d_{ij}, i = 1, \ldots, N; j = 1, \ldots, M)$ is obtained from the interference matrix $E^{(I)} = (e_{ij}, i = 1, \ldots, M; j = 1, \ldots, M)$ [5]. Note that the maximum value of the cost matrix does not include infinity.

Hence, the new TCNN-VT model is described as:

$$\begin{aligned} y_{ij}(t+1) &= ky_{ij}(t) + \alpha \left[\sum_{p=1,p\neq i}^{N} \sum_{q=1,q\neq j}^{M} w_{ijpq} x_{pq}(t) + I_{ij} \right] \\ &\quad - z(t) \left[x_{ij}(t) - I_{ij}^{(0)} \right] \end{aligned} \tag{6}$$

3 Problem Formulation

Detail descriptions and several illustrations on cochannel interference in satellite communications can be found in [4, 5, 6]. The objective of the FAP includes two part, i.e., minimization of the largest interference after reassignment and minimization of the total accumulated interference between systems.

We continue using the neural network formulation given by Funabiki and Nishikawa [5] as their model needs less computation resource compared with the one proposed by Kurokawa and Kozuka [17]. An N-carrier-M-segment FAP between two systems is formulated on an $N \times M$ neural network. If the neuron output $x_{ij} = 1$ at the end of the neuron update, then carrier i is assigned to segment j, and no assignments are made if $x_{ij} = 0$. The novel aspect of the TCNN-VT is that the threshold in the self-feedback term of every neuron is dependent on the interference of the frequency assignment which the neuron represents.

According to [4, 5, 18], the energy function for the TCNN-VT of the FAP is defined as:

$$E = \frac{W_1}{2}\sum_{i=1}^{N}(\sum_{j=1}^{M} x_{ij} - 1)^2 + \frac{W_2}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{\substack{p=1\\p\neq i}}^{N}\sum_{q=max(j-c_p+1,1)}^{min(j+c_i-1,M)} x_{ij}x_{pq}$$

$$+\frac{W_3}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij}(1 - x_{ij}) + \frac{W_4}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} d_{ij}x_{ij} \quad (7)$$

where $W_i, i = 1, \ldots, 4$, are weighting coefficients, whose choices are based on the rule that all terms in the energy function should be comparable in magnitude, so that none of them dominates. Function $max(x, y)$ returns the larger one between (x, y) and $min(x, y)$ finds the smaller value between (x, y).

The W_1 term forces that every segment in system 2 is assigned to one and at most one segment in system 1. The W_2 term guarantees that all the segments of one carrier in system 2 are assigned to consecutive segments in system 1 in the same order [5]. The W_3 term is used to force neuron outputs to approach a corner of the hypercube, i.e., (0 or 1) [18]. The W_4 term is needed to fulfill the optimization of the total interference after the frequency rearrangement.

From Equations (2), (4) and (7), the dynamic equation for the TCNN-VT can be obtained:

$$y_{ij}(t+1) = ky_{ij}(t) + \alpha[-W_1(\sum_{j=1}^{M} x_{ij} - 1) - W_2(\sum_{\substack{p=1\\p\neq i}}^{N}\sum_{q=max(j-c_p+1,1)}^{min(j+c_i-1,M)} x_{pq})$$

$$-\frac{W_3}{2}(1 - 2x_{ij}) - \frac{W_4}{2}d_{ij}] - z(t)\left[x_{ij}(t) - I_{ij}^{(0)}\right] \quad (8)$$

Different with Salcedo-Sanz *et al.* [6, 7], who use binary Hopfield networks with neurons outputting only 0 or 1, we use a continuous neural network, i.e., the neuron output is continuous between 0 and 1. We convert the continuous output x_{ij} of neuron (i, j) to discrete neuron output x_{ij}^b as follows [19]:

$$x_{ij}^b = \begin{cases} 1, & \text{if} x_{ij} > \frac{1}{NM}\sum_{p=1}^{N}\sum_{q=1}^{M} x_{pq}(t); \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

4 Simulation Results and Discussions

An iteration is terminated once a feasible assignment is obtained or the number of iteration steps exceeds a predefined maximum number (15000 in our simulation).

Table 1. 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
1	4	6	1 - 2	5 - 55
2	4	6	1 - 2	1 - 9
3	10	32	1 - 8	1 - 10
4	10	32	1 - 8	1 - 100
5	10	32	1 - 8	1 - 1000
6	30	100	1 - 10	1 - 100
7	50	200	1 - 10	1 - 100
8	80	200	1 - 5	1 - 100

The specifications of the 8 instances are listed in Table 1. Instances 1 to 5 are from [5], where these are called instances 1-5, respectively. Using the instance generation algorithm in Appendix I, we generated instances 6 to 8 to show the performance of the TCNN-VT on large-size problems. The interference matrices $E^{(I)}$ for instances 6 to 8 are randomly generated from $[1, 100]$.

Parameters for the neural network are chosen as follows [10]: $\varepsilon = 0.004$, $k = 0.99$, $\alpha = 0.0015$, $\beta = 0.001$, and $z(0) = 0.1$. Initial inputs of neural networks $y_{ij}(0)$, $(i = 1, \ldots, N, j = 1, \ldots, M)$ are randomly generated from $[-1, 1]$.

The weight coefficients of the energy function W_i, $i = 1, \ldots, 4$, for all the 8 instances are listed in Table 2. The tuning of these weight coefficients is necessary to obtain better performance. From our experience, along with the growing problem size, the total aggregate interference the W_4 term increases, so does the differences between numerical values of the W_1 term and W_2, W_3 terms. Hence, we slightly decrease W_2 and W_3, but increase W_4 as the problem size grows.

We run the TCNN-VT on each instance 1000 times with different randomly generated initial neuron states. Table 3 shows results for every instance, including the best largest interference I_L, the rate to reach the optimum (Opt rate), the average error from the optimal result, the convergence rate η (the ratio at which the neural network finds a feasible solution in 1000 runs), and the total interference I_T when the optimum of the largest interference is found. The

Table 2. Weight coefficients $W_i, i = 1, \ldots, 4$, of the energy function for all the 8 instances. W_1 is fixed at 1.0.

#	W_2	W_3	W_4
1	1.0	0.7	0.00015
2	1.0	0.7	0.00015
3	0.4	0.7	0.00015
4	0.4	0.7	0.0002
5	0.35	0.7	0.0002
6	0.2	0.6	0.0002
7	0.2	0.6	0.0002
8	0.2	0.6	0.0002

average iteration steps T and standard deviations are also shown in this Table. The convergence rate denotes the ratio that the neural network finds a feasible solution at the end of iterations. The results show that the TCNN-VT is effective in reducing the largest interference and total interference by rearranging the frequency assignment.

Table 4 shows results obtained by the TCNN-VT and the comparison with the GNN [5] and the HopSA [6]. Results of the GNN are from references [5, 6], hence we only have the GNN results on instances 1-5. Results of the HopSA on instances 1-5 are also from the reference [6]. As authors in [6] did not publish the average value of the largest and total interference which they found, only the best value is included in Table 4. For instances 6-8, we rerun the HopSA algorithm from the authors of [6].

In Table 4, we show that the TCNN-VT is comparable with the GNN in terms of the largest interference and outperforms the GNN in terms of the total interference. Compared with the HopSA, the TCNN-VT is more efficient, especially on large-size problems. For instance 7 which is a 50-carrier-200-segment FAP, the best largest interference and total interference obtained by the TCNN-VT are 117 and 5839, respectively, whereas for the HopSA, they are 124 and 6533, respectively. The HopSA fails to obtain a solution for instance

Table 3. The performance of the TCNN-VT on 8 instances. # denotes the instance number. 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). "Opt rate" stands for the rate that the TCNN-VT reached the optimum in the 1000 runs. T is the average number of iteration steps. η is the convergence rate. "SD" stands for "standard deviation".

#	I_L			I_T	T	η
	(Best/ Ave)	Opt rate %	Average error	(Best/ Ave)	mean±SD	(%)
1	30/ 35.4	64.0	7.7	100/ 112.6	1191 ± 273	100
2	4/ 4.8	59.4	0.81	13/ 15.4	1799 ± 317	100
3	7/ 8.4	29.3	1.46	96/ 130.6	2485 ± 96.5	92.4
4	70/ 94.1	15.8	14.7	828/ 1145	2383 ± 252	87.5
5	661/ 849	23.7	121	6910/ 9527	3075 ± 268	86.6
6	91/ 97.3	22.8	11.6	3296/ 3826	3716 ± 484	77.1
7	117/ 137	28.1	20.4	5839/ 5722	3834 ± 397	78.4
8	134/ 150	21.5	27.5	6008/ 6340	4319 ± 430	73.3

8 due to the excessive computation time. "N/A" represents the situation that the algorithm cannot find solutions in one month. In comparison, our proposed TCNN-VT finds a feasible solution in less than one minute of CPU time on all the instances.

We listed the computation times of the TCNN-VT and the HopSA [6] on all 8 instances in Table 5. The computation time is measured in seconds (Sec.). Also the computation time is calculated only based on convergence runs. As shown in the Table, the TCNN-VT is much more computational efficient compared with the HopSA. Also, the advantage of the TCNN-VT is more and more distinctive as the problem size grows.

5 Conclusion

We propose a *transiently chaotic neural network with variable thresholds (TCNN-VT)* by varying the threshold in the self-feedback term of the TCNN model. With rich and complex dynamics, the new TCNN-VT maintains the advantage of the TCNN.

We solve the FAP in satellite communications with the TCNN-VT. To min-

Table 4. Comparison of simulation results (largest interference and total interference) obtained by the TCNN-VT, GNN and HopSA for instances 1 to 8.

#	GNN [5]		HopSA [6]		TCNN-VT	
	Largest (Best/Ave)	Total (Best/Ave)	Largest	Total	Largest (Best/Ave)	Total (Best/Ave)
1	30/ 31.5	100/ 100.8	30	100	30 / 35.4	100 / 112.6
2	4/ 4.9	13/ 15.4	4	13	4 / 4.8	13 / 15.4
3	7/ 8.1	85/ 99.4	7	85	7 / 8.4	96 / 130.6
4	64/ 77.1	880/ 982.0	84	886	70 / 94.1	828 / 1145
5	640/ 766.8	8693/ 9413.9	817	6851	661 / 849	6910 / 9527
6	N/A	N/A	98	3396	91 / 97.3	3296 / 3826
7	N/A	N/A	124	6533	117 / 137	5839 / 5722
8	N/A	N/A	N/A	N/A	134 / 150	6008 / 6340

Table 5. Comparison between the HopSA and the TCNN-VT on computation time (Sec.). Results are displayed as mean±SD (standard deviation).

#	HopSA	TCNN-VT
1	1.0±0.0	0.026±0.1
2	1.0±0.0	0.038±0.1
3	33.0±0.0	0.53±0.49
4	33.8±0.8	0.46±0.45
5	33.0±0.5	0.47±0.42
6	2845±528	16.2±2.6
7	8916±1529	49.0±3.4
8	N/A	65.9±4.1

imize the largest interference, the neuron is designed to have a larger threshold if this neuron presents an assignment with smaller interference. The simulation results on 8 benchmark instances demonstrate that the TCNN-VT can find better solutions than existing algorithms.

Appendix I

The following procedure describes the instance generation algorithm for a N-carrier-M-segment FAP.

- Choose the number of carriers N and the number of segments M for the instance.
- Select the values of the range of carrier length and the range of interference between the two systems.
- Generate the set of carrier lengths c_i, $(i = 1, \ldots, N)$ and the interference matrix $E^{(I)}$ $(M \times M)$ using uniformly distributed random values in the range that we defined, respectively.

Appendix II

The following procedure describes the TCNN-VT algorithm on the FAP.

For $RUN = 1, \ldots, 1000$:

- STEP 1: Initialization. $T = 1$. Read in the instance information. Set up the parameters. Compute the threshold value for each neuron. and initialize the neural network status.
- STEP 2: Do neuron updating.
- STEP 3: Validate outputs of neurons. $T{+}{+}$.
  - IF a valid solution is found, i.e., E_1 and E_2 equal to zero, then store the result.
  - Otherwise IF the number of iterations T reaches T_MAX, then the algorithm does not converge.
  - Otherwise return to STEP 2

IF *Run* < 1000, *Run*++. Go back to STEP 1.
Otherwise go to STEP 4

- STEP 4 :Calculate the statistical result for the performance of the TCNN-VT, including the convergence rate and the best, average, and standard deviation of interference results.

References

[1] Murphey R. A., Pardalos P. M., Resende M. G. C., *Frequency assignment problems*, 1999, In: Du D.-Z., Pardalos P. M. (Eds.), Handbook of combinatorial optimization, Kluwer Academic Publishers, vol. Supplement Volume A.

[2] Montemanni R., Moon J. N., Smith D. H., 2003, *An improved tabu search algorithm for the fixed-spectrum frequency-assignment problem*, IEEE Transactions on vehicular technology, vol. 52, no. 3, pp. 891–901.

[3] Wang L. P., Li S., Tian F. Y., Fu X. J., 2004, *A noisy chaotic neural network for solving combinatorial optimization problems: stochastic chaotic simulated annealing*, IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics, vol. 34, no. 5, pp. 2119–2125.

[4] Mizuike T., Ito Y., 1989, *Optimization of frequency assignment*, IEEE Trans. Communications, vol. 37, no. 10, pp. 1031–1041.

[5] Funabiki N., Nishikawa S., 1997, *A gradual neural-network approach for frequency assignment in satellite communication systems*, IEEE Transactions on Neural Networks, vol. 8, no. 6, pp. 1359–1370.

[6] Salcedo-Sanz S., Santiago-Mozos R., B-Calzón C., 2004, *A hybrid hopfield network-simulated annealing approach for frequency assignment in satellite communications systems*, IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, vol. 34, no. 2, pp. 1108–1116.

[7] Salcedo-Sanz S., B-Calzón C., 2005, *A hybrid neural-genetic algorithm for the frequency assignment problem in satellite communications*, Applied Intelligence, vol. 22, no. 3, pp. 207–217.

[8] Nozawa H., 1992, *A neural-network model as a globally coupled map and applications based on chaos*, Chaos, vol. 2, no. 3, pp. 377–386.

[9] ——, 1994, *Solution of the optimization problem using the neural-network model as a globally coupled map*, Physica D, vol. 75, pp. 179–189.

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

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

[12] Ding Z., Leung H., Zhu Z., 2002, *A study of the transiently chaotic neural network for combinatorial optimization*, Mathematical and computer modelling, vol. 36, no. 9, pp. 1007–1020.

[13] Xu X., Tang Z., Wang J., 2005, *A method to improve the transiently chaotic neural network*, Neurocomputing, vol. 67, pp. 456–463.

[14] He Y., 2002, *Chaotic simulated annealing with decaying chaotic noise*, IEEE Transactions on Neural Networks, vol. 13, no. 6, pp. 1526–1531.

[15] He Z., Zhang Y., Wei C., Wang J., 2002, *A multistage self-organizing algorithm combined transiently chaotic neural network for cellular channel assignment*, IEEE Transactions on Vehicular Technology, vol. 51, no. 6, pp. 1386–1396.

[16] Hopfield J. J., 1984, *Neurons with graded response have collective computational properties like those of two-state neurons*, Proc. Nat. Academy Science, vol. 81, pp. 3088–3092.

[17] Kurokawa T., Kozuka S., 1993, *A proposal of neural networks for the optimum frequency assignment problem*, Trans. IEICE, vol. J76-B-II, no. 10, pp. 811–819.

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

[19] Chen L., Aihara K., *Transient chaotic neural networks and chaotic simulated annealing*, 1994, In: Yamguti M. (Ed.), Towards the Harnessing of chaos, Amsterdam: Elsevier Science Publishers, pp. 347–352.