A New Algorithm for RNA Secondary Structure Prediction Using Improved Transiently Chaotic Neural Network

YANQIU CHE[1], ZHENG TANG[1], QIPING CAO[2], HONGWEI DAI[1]
[1]Department of Intellectual Information Systems Engineering
Toyama University
3190 Gofuku Toyama 930-8555
JAPAN

[2]Tateyama Institute of System
Toyama 930-0004
JAPAN

Abstract: - RNA secondary structure prediction is a computationally feasible and broadly studied problem. It can be considered as the combinatorial optimization problem. In this paper, we propose an improved transiently chaotic neural network (TCNN) for RNA secondary structure prediction. In the improved model, a variable $p(t)$ called the acceptance probability of chaos is introduced into the original TCNN model. Variable $p(t)$ is used to decide if the chaos term will be calculated or not. With variable $p(t)$, the network can be speeded up to converge to a fixed point with fewer iterations. The improved TCNN is analyzed theoretically and evaluated experimentally through predicting RNA secondary structure. The simulation results show that the improved transiently chaotic neural network can reach stable state with fewer steps than the original transiently chaotic neural network.

Key-Words: - RNA secondary structure; combinatorial optimization problem; transiently chaotic neural network

1 Introduction

RNAs are molecules that are important for many processes in the cell. A molecule of RNA consists of a long chain of subunits, called ribonucleotides. Each ribonucleotides contains one of four possible bases: adenine, guanine, cytosine, or uracil (abbreviated as A, G, C, U respectively). The base pairing of RNA is generally called the secondary structure. It is the secondary structure that determines how the RNA will interact and react with other components.

Appealing computational methods for RNA secondary structure prediction from knowledge of primary structure have been developed to provide insight into functions that RNA serves. Early algorithm was made by Zuker and Stiegler [1]. The Zuker's algorithm (implemented in the programs called MFOLD [2]) is an efficient dynamic programming algorithm for identifying the globally minimal energy structure for a RNA sequence, as defined by such a thermodynamic model [3] [4]. Zuker's energy calculations have been further improved [5] [6] [7]. In 1992, Takefuji presented a Hopfield neural network for RNA secondary prediction [8]. The results showed that this neural network algorithm performed better than the previous algorithms in the aspect of calculating time and accuracy. But the major weakness of this algorithm is still due to its failure in finding the global minimum solution.

In this paper, we propose an improved transiently chaotic neural network that can reach stable state with fewer steps than the original transiently chaotic neural network. In the improved model, a variable $p(t)$ called the acceptance probability of chaos is introduced into the original TCNN model. Variable $p(t)$ is used to decide if the chaos term will be calculated or not. If the chaos term is accepted, the network will have the same chaotic dynamics as the original TCNN. With the descent of $p(t)$, the accepted chance of the chaos will be decreased and the network will be speeded up to converge to a fixed point with fewer iterations. Extensive simulations are performed, and the results verify that the proposed improved model can find satisfactory solutions on several RNA sequences. The proposed TCNN is superior to the original one in light of the time for reaching stable state.

This paper is organized as follows: the problem formulation is presented in the next section. In Section 3, the transiently chaotic neural network is briefly described. The improved TCNN model is given and compared with the original model in Section 4. The simulation results are showed in

Section 5 and finally we give some general remarks to conclude this paper.

2 Problem Formulation

In order to utilize neural network to predict RNA secondary structure, at the first all possible stack domain candidates are selected and listed for a given RNA molecule. A set of adjacent base pairs is called stack domain, as showed in Fig.1.

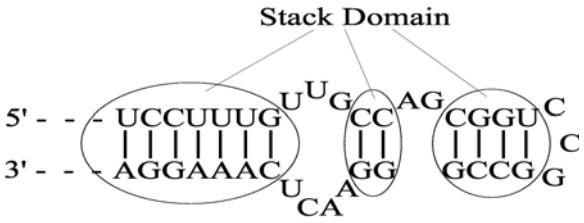

Fig.1. A simple graph shows stack domain of RNA secondary structure

The stability of RNA secondary structure is evaluated by free energy. The most stable secondary structures, those having the lowest free energy, are long chains of stack domains. Much progress has been made on the problem of assigning free energy values to substructures. The most useful free energy data have been extrapolated from experiments on particular kinds of RNA carried out by Tinoco and Uhlenbeck [9] [10]. For stack domain, the free energy is calculated according to Table 1(units is Kcal/mol).

Table 1. Free energy calculation

5'-3'\3'-5'	A-U	U-A	G-C	C-G
A-U	-1.2	-1.8	-2.1	-2.1
U-A	-1.8	-1.2	-2.1	-2.1
G-C	-2.1	-2.1	-4.8	-4.8
C-G	-2.1	-2.1	-3.0	-4.8

Some constraints on the forming of stack domain must be made here. We represent an RNA molecule as a sequence S**:** $s_1, s_2 \ldots s_m$, and $R = R_1, R_2, R_3, \ldots R_n$ is a set of stack domain candidates, and $e_1, e_2, e_3, \ldots e_n$ are free energy values of these stack domain candidates calculated according to Table 1. (i, j) is in R if and only if s_i and s_j are paired. Constrains are described as follows:

1. Only Watson-Crick pairs can be included in stack domains. That is if R contains (i, j) then s_i and s_j are either G and C, or C and G, or A and U, or U and A.
2. There is no overlap of pairs. If R contains (i, j), then it cannot contain (i, k) if $k \neq j$ or (k, j) if $k \neq i$.
3. Knots are not allowed. If $h<i<j<k$, then R cannot contains both (h, j) and (i, k).
4. There is no sharp U-turn in secondary structure. That is If R contains (i, j), then $j-i>3$.

Base pairs with the knots or overlap are called inconsistencies in this paper. According to the above analysis, RNA secondary structure prediction can be considered as the optimization problem. This optimization problem can be formulated by an objective function whose minimum value corresponds to the most stable RNA secondary structure. In a reasonable formulation, there are two components to the objective function: one is used to select stack domain candidates where the sum of free energy is the lowest; the other is used to guarantee there is no inconsistency in RNA secondary structure. Thus, this optimization problem can be mathematically formulated as follows:

$$E = \sum_{i=1}^{n} e_i V_i + \sum_{i=1}^{n} \left(|e_i| \sum_{j=1}^{n} c_{ij} V_i V_j \right) \qquad (1)$$

$$V_i = \begin{cases} 1 & if\ R_i\ is\ selected \\ 0 & otherwise \end{cases}, i \in n \qquad (2)$$

where c_{ij} is a factor that indicates there is an inconsistency or not. If both R_i and R_j are selected and there is an inconsistency between them, then $c_{ij} = 1$; If both R_i and R_j are selected and there is no inconsistency between them, then $c_{ij} = 0$.

3 The Transiently Chaotic Neural Network

Applying the Hopfield neural network (HNN) to solve combinatorial optimization problems is a popular method since Hopfield and Tank's seminal paper [11]. Although the HNN guarantees convergence to a stable equilibrium point due to its gradient descent dynamics, the major weakness is that it often suffers from the local minimum problems. Although many methods have been presented to improved it [12] [13] [14], the results are not always satisfactory.

Instead of utilizing gradient descent dynamics, many artificial neural networks with chaotic dynamics have been investigated [15] [16] [17]. Although the dynamics of the chaotic neural network has an intriguing property to move chaotically over fractal structure in the phase space, without getting

stuck at local minima [15] [16], the convergence problems of the chaotic dynamics have not been satisfactorily solved so far.

In order to take advantages of both the convergent dynamics and the chaotic dynamics, Chen and Aihara have proposed a transiently chaotic neural network (TCNN) by modifying the chaotic neural network [18]. The model is described as:

$$x_i(t) = \frac{1}{1+e^{-y_i(t)/\varepsilon}} \tag{3}$$

$$y_i(t+1) = ky_i(t) + \alpha\left(\sum_{j=1, j\neq i}^{n} w_{ij}x_j(t) + I_i\right) - z_i(t)\left(x_i(t) - I_0\right) \tag{4}$$

$$z_i(t+1) = (1-\beta)z_i(t) \tag{5}$$

where $w_{ij} = w_{ji}$, $w_{ii} = 0$; $\sum_{j=1, j\neq i}^{n} w_{ij}x_j + I_i = -\partial E / \partial x_i$

y_i : internal state of neuron i,

x_i : output of neuron i,

w_{ij}: connection weight form neuron j to neuron i,

I_i : input bias of neuron i,

k : damping factor of nerve membrane ($0 \le k \le 1$),

α : positive scaling parameter for inputs;

z_i(t): self-feedback connection weight or refractory strength ($z_i(t) \ge 0$),

β : damping factor of the time-dependent z_i(t) ($0 \le \beta \le 1$),

I_0 : positive parameter,

ε : steepness parameter of the output function ($\varepsilon > 0$).

This neural network has actually transiently chaotic dynamics which eventually converges to a stable equilibrium point through successive bifurcations like a route of reversed period-doubling bifurcations, with the temporal evolution of a new variable z_i(t) according Eq.(5). The variable z_i(t) corresponds to the temperature in usual stochastic annealing process. Thus, Eq.(5) represents an exponential cooling schedule for the annealing.

4 The Improved Transiently Chaotic Neural Network

We use a single neuron model to show the dynamics of the TCNN. The single neuron model is as follows:

$$x(t) = \frac{1}{1+e^{-y(t)/\varepsilon}} \tag{6}$$

$$y(t+1) = ky(t) + \gamma - z(t)(x(t) - I_0) \tag{7}$$

$$z(t+1) = (1-\beta)z(t) \tag{8}$$

where $\gamma = \alpha I_i$. The value of the parameters in Eqs.(6)-(8) are set as follows:

$\varepsilon = 1/250; k = 0.9; \gamma = 0.004; I_0 = 0.65; z(0) = 0.08;$

$\beta = 0.002.$

Fig.2 shows the time evolutions of $x(t)$ with the initial condition $y(0)=1$. From Fig.2, we can see that with exponential damping of $z(t)$, the neuron output $x(t)$ gradually transits from chaotic behavior to a fixed point through period-doubling bifurcations; that is $x(t)$ behaves erratically and unpredictably during the first 400 iterations and eventually converges to a stable fixed point. But the neuron can not reach stable state even after 3000 iterations.

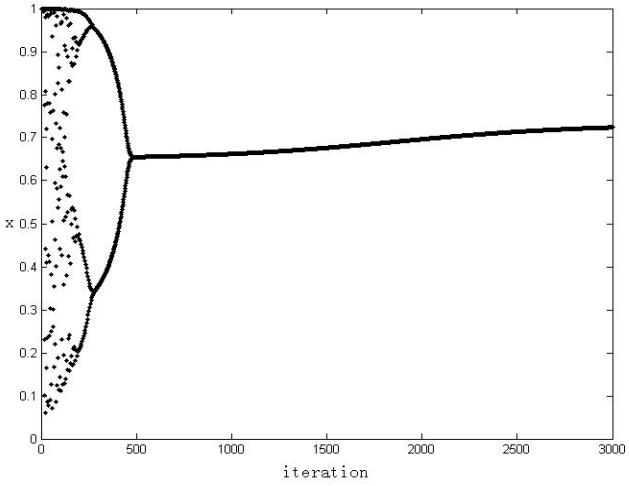

Fig.2. The time evolution of output x in the single model of TCNN

In order to overcome this disadvantage we introduce a variable $p(t)$, the acceptance probability of chaos into the equations of the TCNN. The new model of the TCNN can be described as follows:

$$x_i(t) = \frac{1}{1+e^{-y_i(t)/\varepsilon}} \tag{9}$$

$$y_i'(t+1) = ky(t) + \alpha\left(\sum_{j=1, j\neq i}^{n} \omega_{ij}x_j(t) + I_i\right) \tag{10}$$

$$y_i(t+1) = \begin{cases} y_i'(t+1) - z_i(t)(x_i(t) - I_0) & if\ \delta < p_i(t) \\ y_i'(t+1) & others \end{cases} \tag{11}$$

$$z_i(t+1) = (1-\beta)z_i(t) \tag{12}$$

$$p_i(t+1) = (1-\lambda)p_i(t) \tag{13}$$

where δ is random number($0< \delta <1$) and λ is damping factor of $p(t)$.

The difference between the improved TCNN and the original TCNN is the acceptance probability of chaos $p(t)$. Variable $p(t)$ is used to decide if the chaos term $-z_i(t)(x_i(t)-I_0)$ will be calculated or not. If the chaos term is accepted, the network will have the same chaotic dynamics as the original TCNN. With the descent of $p(t)$, the accepted chance of the chaos will be decreased and the network will be speeded up to converge to a fixed point with fewer iterations. We examine these characteristics using a single model based on Eqs.(9)-(13). $p(0)$ and λ are set to 1 and 0.005, respectively and other parameters are set as same as the original TCNN. Fig.3 shows the time evolution of output x in the single model of the improved TCNN.

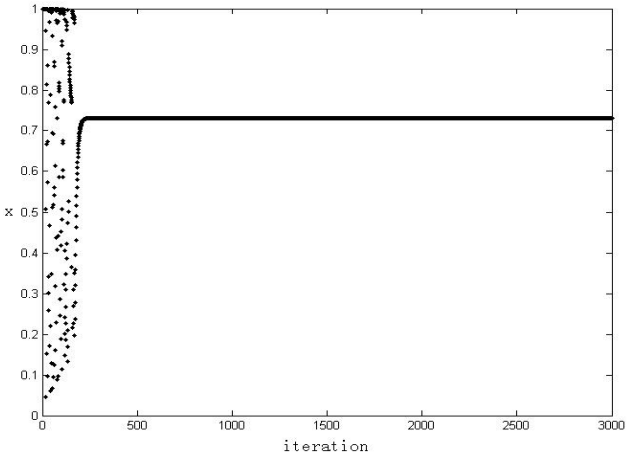

Fig.3. The time evolution of output x in the single model of the improved TCNN

Fig.3 shows that the proposed TCNN uses less iteration to reach stable state in comparison with the original one. The proposed model uses only 200 iterations to reach stable state. The results verify that the proposed TCNN model outperforms than the original one in respects to convergence speed.

5 Simulation Results

Based on Eqs.(9)-(13), the improved TCNN can find its secondary structure for a RNA sequence. Consider the folding of a short RNA sequence with 31 bases (5' and 3' represent the start and the end of a RNA sequence):

5'---ACCCCCUCCUUCCUUGGAUCAAGGGGCUCAA---3'

The values of the parameters in Eqs.(9)-(13) are set as follows:

ε=1/250; k=0.9; α=0.001; I_0=0.65;

z(0)=0.08; β=0.02; $p(0)$=1.0; λ= 0.001

Fig.4 shows the secondary structure predicted by the improved TCNN.

In order to verify the effectiveness of the proposed algorithm for RNA secondary prediction, extensive simulations has been carried out on three RNA sequences. All the simulations were implemented in C++ on PC (CPU:1.7GHz). The values of parameters are set as the same as RNA sequence with 31 bases. For each of algorithms, the simulation program ran 100 times. The results that we recorded for each RNA sequence are the lowest energy and the steps for reaching stable state. The comparisons were arranged in Table 2. The column "n" represents the number of the stack domain candidates.

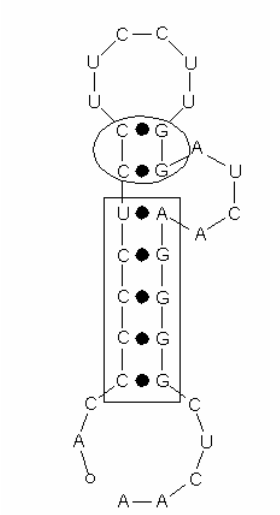

Fig.4. Secondary structure of RNA sequence with 31 bases predicted by the improved TCNN

Example 1: 61 bases of RNA sequence

5'---ACAGGAGUAAUUCCCGCCGAAACAGGGUUUUCACCCUUCCUUUCUUCGGGUGUCCUUCCUC---3'

Example 2: 77 bases of RNA sequence

5'---AGGCUUGUAGCUCAGGUGGUUAGAGCGCACCCCUGAUAAGGGUGAGGUCGGUGGUUCAAGUCCACUCAGGCCUACCA---3'

Example 3: 120 bases of RNA sequence

5'---UGCCUGGCGGCCUUAGCGCGGUGGUCCCACCUGACCCCAUGCCGAACUCAGAAG

UGAAACGCCGUAGCGCCGAUGGUAGUGUGGGG
UCUCCCCAUGCGAGAGUAGGGAACUGCCAGGC
AU---3'

From the comparisons in Table 2, we can see that the proposed TCNN performed better than the original TCNN with respect to the steps for reaching stable state. Although the original TCNN found the optimal folding for some examples, it used more than 2000 steps for all the three examples. The proposed model used less that 300 steps to reach stable states for all examples. That is to say the proposed TCNN has the ability to converge to a fixed point with fewer steps.

Table 2. Simulation results on RNA sequence

Example	n	Original	Proposed
Example1	16	-39.90(2347)	-30.90(264)
	51	-39.90(2489)	-39.90(261)
Example2	22	-43.50(2531)	-43.50(261)
	80	-49.80(2515)	-49.80(269)
Example3	56	-68.70(2129)	-69.90(272)
	177	-73.80(2743)	-82.20(270)

Takefuji algorithm was also executed for comparison. Hopfield neural network is used in his algorithm to predict RNA secondary structure. Table 3 shows the comparison of the lowest energy calculated by the Takefuji algorithm and the improved TCNN model. From Table 3, we can see that the improved model found the same value as the Takefuji algorithm on Example 1. But, on Example 2 and Example 3, the improved model found better solutions than the Takefuji algorithm.

Table 3. Comparison between Takefuji algorithm and the improved TCNN model

Example	n	Takefuji	Proposed
Example1	16	-39.90	-30.90
	51	-39.90	-39.90
Example2	22	-43.50	-43.50
	80	-43.50	-49.80
Example3	56	-65.10	-69.90
	177	-76.20	-82.20

6 Conclusion

This paper presents an improved transiently chaotic neural network for RNA secondary structure prediction. In the proposed model, a variable $p(t)$ called the acceptance probability of chaos was introduced into the original TCNN model. Variable $p(t)$ was used to decide if the chaos term would be calculated or not. If the chaos term was accepted, the network would have the same chaotic dynamics as the original TCNN. With the descent of $p(t)$, the accepted chance of the chaos would be decreased and the network would be speeded up to converge to a fixed point with fewer iterations. This algorithm has been experimentally compared with other methods on several RNA sequences. The results gave evidence of its effectiveness in light of the computation steps and the capability of escaping from local minimum. Moreover, this improved model can also be extended to other combinatorial optimization problems.

References:

[1] M. Zuker, P. Stiegler, Optimal computer folding of lager RNA sequences using thermodynamic and auxiliary information, *Nucleic Acids Re*s., Vol.9, 1981, pp. 133-148.

[2] M. Zuker, Mfold web server for nucleic acid folding and hybridization prediction, *Nucleic Acids Res.*, Vol.31, No.13, 2003, pp. 3406-3415.

[3] M. Zuker, D. Sankoff, RNA secondary structure and prediction, *Bull. Math. Biol.*, Vol.46, 1984, pp. 591-621.

[4] M. Zuker, D.H. Mathews, D.H. Turner, Algorithms and Thermodynamics for RNA Secondary Structure Prediction: A Practical Guide, In *RNA Biochemistry and Biotechnology*, J. Barciszewski & B.F.C. Clark, eds., NATO ASI Series, Kluwer Academic Publishers, Dordrecht, NL, 1999.

[5] A.E. Walter, D.H. Turner, J. Kim, M.H. Lyttle, P. Müller, D.H. Mathews, M. Zuker, Coaxial stacking of helixes enhances binding of oligoribonucleotides and improves predictions of RNA folding, Proc. *Natl. Acad. Sci.*, Vol.91, 1994, pp. 9218-9222.

[6] D. Mathews, H.J. Sabina, M. Zuker, D.H. Turner, Expanded Sequence Dependence of Thermodynamic Parameters Improves Prediction of RNA Secondary Structure, *J. Mol. Biol.*, Vol. 288, 1999, pp. 911-940.

[7] A. Akutsu, Dynemic programming algorithms for RNA secondary structure prediction with pseudoknots, *Discrete Applied Mathematics*, Vol.104, No.1-3, 2000, pp. 45-62.

[8] Y. Takefuji, D. Ben-Alon, A. Zaritsky, Neural computing in discovering RNA interactions, *BioSystems*, Vol.27, No.2, 1992, pp. 85-96.

[9] I. Tinoco, O.C. Uhlenbeck, M.D. Levine, Estimation of secondary structure in ribonucleic acids, *Nature*, Vol.230, 1971, pp. 362-367.

[10] I. Tinoco, P.N. Borer, B. Dengler, M.D. Levine, O.C. Uhlenbeck, D.M. Crothers, J. Gralla, Improved estimation of secondary structure in ribonucleic acids, *Nature New Biology*, Vol.246, 1973, pp. 40-41.

[11] J.J. Hopfield, D.W. Tank, Neural computation of decisions in optimization problems, *Biological Cybernetics*, Vol.52, 1985, pp. 141- 152.

[12] P.W. Protzel, D.L. Palumb, M.K. Arras, Performance and fault-tolerance of Neural networks for optimization, *IEEE Transactions on Neural Networks*, Vol.4, 1993, pp. 600-614.

[13] S.Z. Li, Improving convergence and solution quality of Hopfield-type neural networks with augmented Lagrange Multipliers, *IEEE Transactions on Neural Network*, Vol.7, No.6, 1996, pp. 1507-1516.

[14] R.L. Wang., Z. Tang, Q.P. Cao, A parallel algorithm for maximum cut problem using gradient ascent learning of Hopfield neural networks, *IEEJ Trans. EIS*, Vol.2, No.11, 2002, pp. 1986-1994.

[15] H. Nozawa, A neural network model as a globally coupled map and applications based on chaos, *Chaos*, Vol.2, 1992, pp. 377-386.

[16] L. Chen, K. Aihara, Chaotic simulated annealing for combinatorial optimization, in: *Dynamical Systems and Chaos*, ed. N. Aoki, K. Shiraiwa and Y. Takahashi, 1(World Scientific, Singapore), 1995, pp. 319-322.

[17] M. Ohta, Chaotic neural networks with reinforced self-feedbacks and its application to N-Queen problem, *Mathematics and Computers in Simulation*, Vol.59, 2002, pp. 305-317.

[18] L. Chen, K. Aihara, Chaotic simulated annealing by a neural network model with transient chaos, *Neural Networks*. Vol.8, No.6, 1995, pp. 915- 930.