An automata network for performing combinatorial optimization

Zong-Ben Xu\textsuperscript{a}, Hui-Dong Jin\textsuperscript{b}, Kwong-Sak Leung\textsuperscript{b,}\textsuperscript{*}, Yee Leung\textsuperscript{c}, Chak-Kuen Wong\textsuperscript{b}

\textsuperscript{a}Institute for Information and System Sciences, Xi’an Jiaotong University, Xi’an, 710049, People’s Republic of China
\textsuperscript{b}Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
\textsuperscript{c}Department of Geography, CUHK, Shatin, N.T., Hong Kong

Abstract

There have been several neural network approaches to the tasks of solving combinatorial optimization problems. In this paper, a new type of neural networks—the WTA-type networks, which incorporates the Winner-Take-All mechanism into the automata networks, is proposed. Five specifications of the WTA-type networks ($N_1$–$N_5$) are presented. The theoretical foundations of the networks are developed from the standpoint of taking them as combinatorial optimization solvers. We also investigate the two key issues, reliability and efficiency, related to the application of the networks. The proposed networks and the established theories are applied to a set of combinatorial optimization benchmark problems—traveling salesman problems. The simulation results demonstrate that the proposed WTA-type networks are more effective or comparable with the Hopfield networks, the Boltzmann machine and the self-organizing feature map network. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Automata network; Winner-Take-All; Combinatorial optimization; Hopfield network; Boltzmann machine

1. Introduction

The subject of combinatorial optimization consists of a set of problems central to the disciplines of computer science and engineering. Research in this area
aims at developing efficient techniques for finding the minimum or maximum of an objective function, representing a quantitative measure of the ‘goodness’ of a complex system with many independent variables. For example, the traveling salesman problem (TSP), as a representative combinatorial problem, can be stated as a search for the shortest closed tour that visits each city once and only once. There are several different types of TSPs. The Euclidean TSP (ETSP) is perhaps the simplest one in which the cities locate on a plane with the Euclidean distances between cities [23]. If the distances between cities are symmetric, the TSP is called the symmetric TSP (STSP) [10]. Otherwise, it is called an asymmetric TSP (ATSP) [8]. The wide applications and the NP-hard or NP-complete difficulties make the combinatorial optimization problems very attractive. They, especially including TSPs, have received extensive researches in the past 20 years. Besides various heuristics designed specially for certain classes of problems [8,23], the methods of discrete linear programming [10], simulated annealing [1,19], neural networks [2,4,7,17,20,21], and genetic algorithms [14,26] have been implemented to solve the problems with various degrees of success.

Since 1985, when the Hopfield network was first applied to ETSPs [17], neural network approaches have been widely employed on performing combinatorial optimizations like the TSP [2,4,12,25,27], clustering [18], LSI module placement [3], set partitioning [22], set covering [22], and maximal independent set problems [29]. The outstanding characteristics of the neural network approaches are that they could be employed to solve problems in parallel, resulting in very robust algorithms. Furthermore, the algorithms can be sped up by building dedicated hardware [6,17].

Most of neural networks for combinatorial optimization problems are direct versions of the well-known Hopfield model [2–5,30]. Their basic idea is to transform the combinatorial optimization problem into the minimization of a well-defined energy function, which contains several adjustable penalty parameters, and then map the energy function to a neural network architecture such that any minimizer of the energy function corresponds to an equilibrium state of the network. Thus, a solution could be obtained from the equilibrium states of the network through its dynamical evolution. Though the Hopfield model has a number of successful applications, it has some drawbacks [3]. First, it is arduous to properly specify the penalty parameters in the energy function. Secondly, finding feasible solutions becomes increasingly difficult as the size of problems increases. Moreover, even a valid solution can be obtained, its quality is usually not so good. A great deal of research has been done on improving these shortcomings of the Hopfield model, like choosing appropriate forms of the energy function [2,6], incorporating some local minima escape algorithms into the network [25], hybridizing the Hopfield model with simulated annealing technique (known as Boltzmann machine (BM) [1]), and performing group updating [22]. They are all acquired more or less success.

Another kind of neural networks for combinatorial optimization problems is the competitive neural networks, where the Winner-Take-All ($WTA$) plays a crucial role in learning [16,20]. Many algorithms based on such competitive neural networks have been devised for combinatorial optimization problems and some encourag-
The Winner-Take-All (WTA) mechanism is a device that chooses an element from a candidate set based on a competition rule, and usually the one with the maximal activation value is chosen [13]. Such a mechanism has been widely implemented and plays a crucial role in competitive neural networks [16,20]. Our attempt to incorporate this mechanism into automata networks is motivated by the following observation. A large number of combinatorial optimization problems can be formulated as

\[
\text{Minimize} \quad E(V) = - \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} v_i v_j + \sum_{i=0}^{n} d_i v_i, \\
\text{Subject to} \quad V = (v_1, v_2, \ldots, v_n) \in \Omega \subset \{0, 1\}^n, \quad (1)
\]

where \( n \) is an integer, \( w_{ij} \) and \( d_i \) are real numbers (\( i,j = 1,2,\ldots,n \)) and \( \Omega \) is a well-defined domain. The domain \( \Omega \) can often be characterized by some equalities of the form

\[
\sum_{i \in S_j} v_i = 1 \quad \text{for } j = 1,2,\ldots,q. \quad (2)
\]

where \( S_j \) are subsets of \( \{1,2,\ldots,n\} \) and \( q \) is an integer. When each \( S_j \) is viewed as a candidate set, Eq. (2) then means that only one element in \( S_j \) should win the competition. This case can be explained as the WTA mechanism applied to each \( S_j \). Thus a network elaborately embodying the WTA mechanism is likely suited for solving the combinatorial optimization problems described in (1).

In order to formulate the above observation, we first describe the automata neural networks based on the theory of the automata network.
2.1. The automata neural networks

An automata network $\mathcal{A} = (G, Q, F)$ [15] is said to be an automata neural network if

- the graph $G = (I, C)$ possesses weighted and directional structure, where $I$ is a set of neurons (or vertices) and $C \subseteq I \times I$ is a connection set;
- $Q$ is the set of states any vertex may take, which is assumed to be finite in most cases;
- the transition function $F = \{ f_i : i \in I \}$ can be split into the composition of a linear function $L = \{ l_i : i \in I \}$ and a nonlinear function $H = \{ h_i : i \in I \}$, where the local transition function $f_i = h_i \cdot l_i$ with $f_i : Q_{|C_i|} \to Q$, $l_i : Q_{|C_i|} \to R_{|D_i|}$, $h_i : R_{|D_i|} \to Q$ and $C_i, D_i$ are neighborhoods of neuron $i$. Here, $R$ is an operation mode with which the state of the neural network is updated.

Let $V(t) = \{ v_i(t), i \in I \} \in Q^{|I|}$ be the states of the neurons at time $t$. Then the evolution of the automata network $\mathcal{A}$ is defined by

$$v_i(t + 1) = f_i(\{ v_j(t) : j \in C_i \}) \quad \text{for } i \in I. \quad (3)$$

Without loss of generality, we assume $I = \{1, 2, \ldots, n\}$, $C = \{(i, j) : i, j \in I\}$ (that is, $G$ is fully connected) and $w_{ij}$ is the connection weight attached to the edge $(i, j)$. Then the linear function $L$ usually takes the form

$$[l_i(V(t))]_k = \sum_{j \in C_i} w_{ij} v_j(t) - t_k, \quad \forall k \in D_i, \quad i \in I. \quad (4)$$

where $[\bullet]_k$ is the $k$th component of the vector $[\bullet]$. If $C_i = I$, this can be rewritten as

$$L(V(t)) = WV(t) - T, \quad (5)$$

where $W = (w_{ij})_{n \times n}$ and $T = (t_i)$. In terms of neural networks, $t_k$ is the threshold, and $u_k = [l_i(V)]_k$ the activation potential of neuron $k$.

Clearly, an automata neural network $\mathcal{N}$ can be characterized by a septuple $\mathcal{N} = (Q, W, T, \{ C_i \}, \{ D_i \}, H, R)$. If $Q = \{0, 1\}$, $C_i = I$, $D_i = \{i\}$ and $H = \{h_i : i \in I\}$ with $h_i$ being the signum function, then the automata neural network $\mathcal{N} = (\{0, 1\}, W, T, I, \{i\}, \text{Sgn}, R)$ is the McCulloch and Pitts model [24,16] or the Hopfield neural network presented in [17,22,25]. Now, let us introduce the WTA-type networks.

2.2. Five WTA-type networks

We define an operator $WTA : \mathbb{R}^q \to \{0, 1\}^q$ as

$$[WTA(x)]_i = \begin{cases} 1 & \text{if } x_i = \max_{j=1,\ldots,q} x_j \\ 0 & \text{otherwise} \end{cases} \quad i = 1, 2, \ldots, q. \quad (6)$$
The operator is regulated in such a way that only one component takes 1 even when more than one component in the vector $x$ takes the maximum value. A WTA-type network is an automata network incorporating the WTA nonlinearity, as defined below.

**Definition 1.** An automata neural network $\mathcal{N}$ is said to be of WTA-type if $\mathcal{N} = (\{0, 1\}, W, I, \{D\}, \{\text{WTA}\}, R)$.

To derive several useful WTA-type networks, we assume henceforth that the neuron set $I = \{(x, i)\}_{n \times m}$. Let $w_{x,i,y,j}$, $v_{x,i}$, and $t_{x,i}$ be the interconnection weight between the neurons $(x, i)$ and $(y, j)$, the state of the neuron $(x, i)$, and the threshold attached to $(x, i)$, respectively. Then we denote $W = (w_{x,i,y,j})_{nm \times nm}$, $T = (t_{x,i})_{n \times m}$, $V(t) = (v_{x,i}(t))_{n \times m}$, $N = \{1, 2, \ldots, n\}$, $M = \{1, \ldots, m\}$, and define the product between $W$ and $V$ by

$$[WV]_{x,i} = \sum_{y=1}^{n} \sum_{j=1}^{m} w_{x,i,y,j}v_{y,j}.$$ 

Thus, the activation potential of the neuron $(x, i)$ is

$$u_{x,i}(t + 1) = \sum_{y=1}^{n} \sum_{j=1}^{m} w_{x,i,y,j}v_{y,j}(t) - t_{x,i} = [WV(t)]_{x,i} - t_{x,i}. \quad (7)$$

Let $U(t) = (u_{x,i}(t))_{n \times m}$, then $U(t + 1) = L(V(t)) = WV(t) - T$.

We introduce two neighborhood structures $\{D_{(x,i)}\}$, related to how the neuron set $I$ is partitioned when the WTA mechanism is implemented, in a natural way as follows:

(i) $D_{(x,i)}^{c} = \{(y, i) \mid y \in N\}$, that is, neurons in the same column have a neighborhood relationship if the neuron set $I$ is viewed as an $n \times m$ matrix;

(ii) $D_{(x,i)}^{r} = \{(x, j) \mid j \in M\}$, which means that neurons located in the same row have a neighborhood relationship.

With these neighborhood structures, two specific WTA-type operators can be introduced. First, associated with the neighborhood structure $\{D_{(x,i)}^{c}\}$, we define a column WTA operator $WTA_{c}$ as follows:

**Definition 2.** The column $WTA$ operator $WTA_{c} : \mathbb{R}^{nm} \to \{0, 1\}^{nm}$ is defined by

$$WTA_{c}(U(t)) = (WTA(U^{(1)}(t)), \ldots, WTA(U^{(m)}(t))), \quad (8)$$

where

$$[WTA(U^{(*)}(t))]_{x} = [WTA([WV(t - 1) - T]^{(*)})]_{x}$$

$$= \begin{cases} 1 & \text{if } x \in I(i, t), \\ 0 & \text{otherwise} \end{cases} \quad (9)$$
and \([\bullet]^i\) stands for the \(i\)th column of \([\bullet]\). The index set \(I(i,t)\) is specified by

\[
I(i,t) = \{x \in N | u_{xi}(t) = u_{\max}^{(i)}(t)\}
\]

and either \(v_{xi}(t-1) = 1\) or \(x \leq \arg \max_{y \in N} \{u_{yi}(t)\}\),

where \(u_{\max}^{(i)}(t) = \max_{y \in N} \{u_{yi}(t)\}\) and \(\arg \max_{y \in N} \{u_{yi}(t)\}\) is any index \(k\) such that \(u_{ki}(t) = u_{\max}^{(i)}(t)\).

Eqs. (8) and (10) in Definition 2 imply that whenever there are more than one neuron in the \(i\)th column of \(I\) which are with the maximal activation potential, we regulate the network so that a neuron \(x\) wins the competition if either the neuron \((x, i)\) wins the competition at the previous time instant (i.e., \(v_{xi}(t-1) = 1\)) or \(x\) is the minimal index such that

\[
\begin{align*}
u_{xi}(t) &\geq u_{yi}(t) \\
& \quad \text{for any } y \in N.
\end{align*}
\]

This regulation is called the Old-Winner-Least-Index (OWLI) priority regulation since, by the definition, the old excited neuron and the neuron with the least row index will be preferred in winning the competition. Under this OWLI regulation, other neuron becomes the winner only when its activation potential is larger than that of the old winner.

Similarly, corresponding to the second neighborhood structure \(\{D^{(r)}_{(x,i)}\}\), we can define the row WTA operator \(\text{WTA}_r\).

**Definition 3.** The row WTA operator \(\text{WTA}_r : \{0,1\}^{nm} \to \{0,1\}^{nm}\) is defined by

\[
[\text{WTA}_r(U(t))]^{(x,\bullet)} = [\text{WTA}(U^{(x,\bullet)^T}(t))]^T = [\text{WTA}((WV(t-1) - T)^{(x,\bullet)^T})]^T.
\]

where \([\text{WTA}_r(U)]^{(x,\bullet)}\) and \(U^{(x,\bullet)}\) are, respectively, the \(x\)th row of \(\text{WTA}_r(U)\) and \(U\). An OWLI priority regulation is also assumed similar as in Definition 2.

Now, we associate these WTA-type operators with certain operation mode \(R\) to introduce five WTA-type networks.

**(A)** \(\mathcal{A}_1 = (\{0,1\}, W, T, I, \{D^{(c)}_{(x,i)}\}, \text{WTA}_c, S)\).

This is the WTA-type neural network with the column-neighborhood system \(\{D^{(c)}_{(x,i)}\}\) and the sequential update mode \(S\). In this network, the states are updated by column in a predefined or random order. If the \(i\)th column of the network is chosen to be updated at time \(t\), then the state of the network at time \(t+1\) is determined by

\[
[V(t + 1)]^{(\bullet,k)} =
\begin{cases}
[\text{WTA}_c(U(t + 1))]^{(\bullet,i)} & \text{if } k = i, \\
[V(t)]^{(\bullet,k)} & \text{if } k \neq i.
\end{cases}
\]

Henceforth, the network \(\mathcal{A}_1\) will be briefly denoted as \(\mathcal{A}_1 = (W, T, \text{WTA}_c, S)\).

**(B)** \(\mathcal{A}_2 = (\{0,1\}, W, T, I, \{D^{(r)}_{(x,i)}\}, \text{WTA}_r, S)\).

This WTA-type network uses the row-neighborhood system, denoted hereafter by \(\mathcal{A}_2 = (W, T, \text{WTA}_r, S)\) in brief.

**(C)** \(\mathcal{A}_3 = (W, T, \text{WTA}_c, P)\) and \(\mathcal{A}_4 = (W, T, \text{WTA}_r, P)\).
Instead of using the sequential operation mode, these WTA-type networks are obtained by applying the column or row parallel update mode in $\mathcal{N}_1$ and $\mathcal{N}_2$. Therefore, the evolution of the networks' states in $\mathcal{N}_3$ and $\mathcal{N}_4$ are governed, respectively, by

$$V(t) = WTA_c(U(t)) = WTA_c(WV(t - 1) - T), \quad t = 1, 2, \ldots$$ (13)

and

$$V(t) = WTA_r(U(t)) = WTA_r(WV(t - 1) - T), \quad t = 1, 2, \ldots$$ (14)

\(D) \quad \mathcal{N}_5 = (W, T, WTA, WTA_r, AS).

This WTA-type network is created with a new operation mode. We call this mode the Alternate Serial Mode (AS mode in brief) which means that the states of networks are alternately updated column by column via $WTA_c$ and row by row via $WTA_r$.

**Remark 1.** The significance of the WTA-type networks introduced above can be explained as follows: There are two types of constraints frequently encountered in combinatorial optimization problems such as the TSP, the shortest path problems (SPPs), the clustering problems (CPs), the graph partitioning problems (GPPs), and the task assignment problems (TAPs) [4,17]. One of such constraints is the so-called column-feasibility condition, which means that there exits one and only one neuron excited in each column of the neuron matrix. Mathematically, this requires $V = (v_{xi})_{n \times m} \in \Omega_c$ where

$$\Omega_c = \left\{ V = (v_{xi})_{n \times m}; v_{xi} \in \{0, 1\} \text{ and } \sum_{x=1}^{n} v_{xi} = 1, \text{ for } i = 1, 2, \ldots, m \right\}$$ (15)

which could be called the column-feasible set. The other type of constraints allow one and exactly one neuron excited in every row, formulated by $V \in \Omega_r$ with

$$\Omega_r = \left\{ V = (v_{xi})_{n \times m}; v_{xi} \in \{0, 1\} \text{ and } \sum_{i=1}^{m} v_{xi} = 1, \text{ for } x = 1, 2, \ldots, n \right\}$$ (16)

We call $\Omega_c$ the row-feasible set. It is obvious that with the specific neighborhood structure $\{D_{(x,i)}^{(c)}\}$ and $\{D_{(x,i)}^{(r)}\}$, the nonlinear transition functions $WTA_c$ and $WTA_r$ in $\mathcal{N}_1$ ($\mathcal{N}_3$) and $\mathcal{N}_2$ ($\mathcal{N}_4$) become the projections of $\mathbb{R}^{n \times m}$ onto $\Omega_c$ and $\Omega_r$, respectively. The states of the networks $\mathcal{N}_1$ and $\mathcal{N}_3$ ($\mathcal{N}_2$ and $\mathcal{N}_4$) therefore are always kept in the column-feasible set $\Omega_c$ (the row-feasible set $\Omega_r$, respectively). This implies that whenever applied to combinatorial optimization problems, the proposed networks $\mathcal{N}_1$ and $\mathcal{N}_3$ can naturally realize the column-feasibility constraints, and $\mathcal{N}_2$ and $\mathcal{N}_4$ the row-feasibility constraints, respectively. Thus, even the constraints involved are not embodied in the energy function, the proposed networks still guarantee us to produce feasible solutions. This reveals the intrinsic advantage of the proposed WTA-type networks over some previously known neural network models which not only penalize the violation of the given constraints but also do not guarantee the feasibility of the resultant solutions [9,17,24]. This is...
also different from the “soft” competitive networks, say, the Boltzmann machine. They normally end with a single “winner” for each column or row as the penalty parameter increases, and their performance heavily depends on the penalty increasing scheme [25]. For most combinatorial optimization problems, the solutions are cyclic, for example, tour (1,2,3,4) is identical with tour (3,4,1,2) for a 4-city TSP, and hence the order of the columns (or the rows) being selected is not important. So, the WTA dynamics can select the first few neurons in any order. This can reduce the search space of the solutions and has no impact on the quality of the final solutions.

Remark 2. The network $N^5$ will be particularly suitable for solving those combinatorial optimization problems where the domain $\Omega$ is specified by $\Omega_c \cap \Omega_r$ (say, TSPs and TAPs). In fact, the operator $\text{WTA}_c$ can be viewed as a projection of $\mathbb{R}^{n \times m}$ onto $\Omega_c$ and $\text{WTA}_r$ a projection of $\mathbb{R}^{n \times m}$ onto $\Omega_r$, respectively, so the function $\text{WTA}_c \text{WTA}_r$ can be viewed as an alternative projection to $\Omega_c$ and $\Omega_r$. According to the alternative projection theory [11], the state of the network $N^5$ then may converge eventually to the intersection $\Omega_c \cap \Omega_r$.

3. Theory of the WTA-type networks

Two fundamental issues, feasibility and reliability [29], related to the applications of the proposed networks, are studied in this section.

3.1. Feasibility

When employed as a method of solving optimization problems, a neural network finds the solution through searching for certain network states known as stable states. A vector $V^* \in \Omega_{\ell}^{[J]}$ is said to be a stable state of the network if the state does not change when the network is initialized with it. The convergence to a stable state is a prerequisite for a network to be an optimization solver [28]. So we establish the convergence theorems to justify the feasibility of the proposed WTA-type networks below.

**Theorem 1.** Assume $W = (w_{x,i;y,j})_{nm \times nm}$ is symmetric, i.e., $w_{x,i;y,j} = w_{y,j;x,i}$. Let

$$C_{x,y}^{(k)} = w_{x,k;x,k} + w_{y,k;y,k} - 2w_{x,k;y,k},$$

$$R_{i,j}^{(x)} = w_{x,i;x,i} + w_{x,j;x,j} - 2w_{x,i;x,j}$$

and

$$E(V) = -\frac{1}{2} \sum_{x=1}^{n} \sum_{y=1}^{n} \sum_{i=1}^{m} \sum_{j=1}^{m} w_{x,i;y,j}v_{x,i}v_{y,j} + \sum_{x=1}^{n} \sum_{i=1}^{m} t_{x,i}v_{x,i}$$

$$= -\frac{1}{2} V^TWV + T^TV. \quad (17)$$
Then

(i) If \( C^{(k)}_{x,y} \geq 0 \) for any \( x,y \in N, k \in M \), and the initial state \( V_0 \in \Omega_c \), then the network \( \mathcal{N}_1 = (W, T, WTA_c, S) \) takes \( E(V) \) as its energy function. Furthermore, the network \( \mathcal{N}_1 \) converges to a stable state of the network for any \( V_0 \in \Omega_c \).

(ii) If \( C^{(k)}_{i,j} \geq 0 \) for any \( x \in N, i,j \in M \), and the initial state \( V_0 \in \Omega_r \), then the network \( \mathcal{N}_2 = (W, T, WTA_r, S) \) takes \( E(V) \) as its energy function. Moreover, \( \mathcal{N}_2 \) converges to its stable state for any \( V_0 \in \Omega_r \).

**Proof.** We only prove the theorem for the network \( \mathcal{N}_1 = (W, T, WTA_c, S) \). The proof for \( \mathcal{N}_2 = (W, T, WTA_r, S) \) is similar and omitted. Denote \( E(V(t)) \) as the function in Eq. (17) with the state variable \( V(t) = (v_{x,i}(t))_{n \times m} \) defined by the network \( \mathcal{N}_1 \). First we show that \( E(V(t + 1)) \leq E(V(t)) \) for any \( t \) to justify that \( E(V(T)) \) is an energy function. Assume \( V(t + 1) \) and \( V(t) \) differ only in the \( k \)th column. Since the initial state \( V_0 \in \Omega_c \), the WTA mechanism guarantees that each \( V^{(*,k)}(t) \) contains only one nonzero component. Without loss of generality, we assume that \( (b,k) \) is the unique nonzero component of \( V^{(*,k)}(t) \), and \( (a,k) \) is the unique nonzero component of \( V^{(*,k)}(t + 1) \). Thus, \( u_{ab}(t + 1) \geq u_{bk}(t + 1) \) holds, where \( u_{ab}(t + 1) = \sum_{i=1}^{n} \sum_{y=1}^{m} w_{x,i,y,j} v_{y,j}(t) - t_{sk} \) is the activation potential of the neuron \((x,k)\). We have

\[
(\Delta V)_{x,i} = (V(t + 1) - V(t))_{x,i} = \delta_{ki} \times \delta_{xi} - \delta_{ki} \times \delta_{xb},
\]

where \( \delta_{xi} \) is the sign of Kronecker delta (that is, \( \delta_{xi} = 1 \) if and only if \( x = i \)). Since \( W \) is symmetric, we have

\[
E(V(t + 1)) - E(V(t)) = -\sum_{x=1}^{n} \sum_{i=1}^{m} \left( \sum_{y=1}^{m} \sum_{j=1}^{m} w_{x,i,y,j} v_{y,j}(t) - t_{xi} \right) (\Delta V)_{x,i}
\]

\[
- \frac{1}{2} \sum_{x=1}^{n} \sum_{i=1}^{m} \sum_{y=1}^{m} \sum_{j=1}^{m} w_{x,i,y,j} (\Delta V)_{x,i} (\Delta V)_{y,j}
\]

\[
= -(u_{ab}(t + 1) - u_{bk}(t + 1)) - \frac{1}{2} C^{(k)}_{a,b} \leq 0.
\]

Therefore \( E(V) \) is an energy function of \( \mathcal{N}_1 \). We further observe that whenever \( V(t + 1) \neq V(t) \), \( E(t + 1) < E(t) \) holds because \( u_{ab}(t + 1) > u_{bk}(t + 1) \) for \( a \neq b \) due to the OWLI priority regulation.

In what follows, we use the term **one iteration** to describe one complete cycle of updates of the network \( \mathcal{N}_1 \) from the first column to the last one. To show the convergence of \( \mathcal{N}_1 \), we consider the subsequence of \( \{V(t)\} : V(0), V(m), \ldots, V(km), \ldots \), where \( V((k + 1)m) \) is the state of \( \mathcal{N}_1 \) obtained after one iteration from \( V(km) \). If \( V((k + 1)m) \neq V(km) \), then there exists a \( t \) \( (km < t \leq (k + 1)m) \) such that \( V(t - 1) \neq V(t) \). This implies from (19) that \( E((k + 1)m) \leq E(t) < E(t - 1) \leq E(km) \). Since the number of states in the column feasible set \( \Omega_c \) is finite, it follows that \( \{V(km)\} \) will not be different from one another. Let us suppose \( V((k_0 + 1)m) = V(k_0m) \). Then we obtain \( V(k_0m) = V(k_0m + 1) = \cdots = V(k_0m + m) \).
So $V(k_0 m)$ becomes a stable state of $\mathcal{N}_1$. This shows the convergence of $\mathcal{N}_1$ and completes the proof of the theorem. □

**Remark 3.** Theorem 1 is fundamental for guiding how a combinatorial optimization problem should be properly mapped to a network $\mathcal{N}_1$ (or $\mathcal{N}_2$) to guarantee the feasibility. It should be noted that the condition $C_{x,y}^{(k)} \geq 0$ or $R_{y,i}^{(k)} \geq 0$ corresponds to the nonnegative self-feedback condition for convergence of the Hopfield model [29] and it is very mild and easily justified in applications.

Analogously, we establish the following convergence theorem for $\mathcal{N}_3$ and $\mathcal{N}_4$.

**Theorem 2.** If $W$ is symmetric, then the function $E_2$ defined by

$$E_2(V(t)) = -V^T(t-1)WV(t) + T^T[V(t) + V(t-1)]$$

(20)

is an energy function of the networks $\mathcal{N}_3 = (W, T, WTA_c, P)$ and $\mathcal{N}_4 = (W, T, WTA_r, P)$. The networks $\mathcal{N}_3$ and $\mathcal{N}_4$ either converge to a stable state or converge to a limit cycle of length 2 from any initial state. Moreover, if $W$ is nonnegative definite, the networks $\mathcal{N}_3$ and $\mathcal{N}_4$ converge to a stable state.

**Proof.** We only prove the part for $\mathcal{N}_3 = (W, T, WTA_c, P)$. According to Eq. (20), we have

$$E_2(V(t)) - E_2(V(t - 1)) = -[V^T(t-1)W - T][V(t) - V(t - 2)]$$

$$= -\sum_{i=1}^{m} [u_{a,i}(t) - u_{b,i}(t)],$$

where $(a_i, i)$ and $(b_i, i)$ are the unique nonzero components of the $i$th column of $V(t)$ and $V(t-2)$. From Eqs. (8) and (10) we deduce $u_{a,i}(t) = \max_{x \in M} \{u_x(t)\}$ for $i = 1, 2, \ldots, n$. That implies $u_{a,i}(t) \geq u_{b,i}(t)$. So we get $E_2(V(t)) - E_2(V(t - 1)) \leq 0$.

This justifies that $E_2$ is an energy function of the network $\mathcal{N}_3$.

To see what $\mathcal{N}_3$ converges to, let’s define $\tilde{\mathcal{N}}_3 = (\{0,1\}, \tilde{T}, I, \{D^{(c)}_{(i,j)}\}, WTA_c, P)$ which differs from $\mathcal{N}_3$ only in the threshold vector term. The threshold vector $\tilde{T}$ is defined by $\tilde{T} = T + \varepsilon$ with

$$\varepsilon = \{(\varepsilon_{ai})\}_{n \times m},$$

$$\varepsilon_{ai} = e_i \times \frac{x - 1}{n}$$

and

$$e_i = \min_{1 \leq y < x \leq n} \inf_{V \in U_x} \{u_{ai}(V) - u_{yi}(V) \mid u_{ai}(V) > u_{yi}(V)\} > 0.$$

Here $u_{zi}(V) = [L(V)]_z$ for $z = x$ or $y$. $\tilde{\mathcal{N}}_3$ has an energy function of the matrix form as

$$E_3(V(t)) = -V^T(t-1)WV(t) + \tilde{T}^T[V(t) + V(t - 1)].$$

(21)
Let $\tilde{U}(t + 1) = WV(t) - \tilde{T}$. We observe that if $u_{xi}(t) > u_{yi}(t)$ and $x > y$, then $u_{xi}(t) - u_{yi}(t) > \varepsilon_i$, and hence
\[
\tilde{u}_{xi}(t) - \tilde{u}_{yi}(t) = [u_{xi}(t) - \varepsilon_i] - [u_{yi}(t) - \varepsilon_i] \\
= u_{xi}(t) - u_{yi}(t) - \frac{x - y}{n} \times \varepsilon_i \\
\geq \varepsilon_i - \frac{x - y}{n} \times \varepsilon_i > 0.
\]
In addition, if $u_{xi}(t) \leq u_{yi}(t)$ and $x > y$, then
\[
\tilde{u}_{xi}(t) - \tilde{u}_{yi}(t) = u_{xi}(t) - u_{yi}(t) - \frac{x - y}{n} \times \varepsilon_i \\
\leq -\frac{x - y}{n} \times \varepsilon_i < 0.
\]
This shows that $\tilde{u}_{xi}$ has the same order on the subscript $x$ as $u_{xi}$ with the OWLI priority regulation. So, according to the definitions of $\mathcal{N}_3$ and $\tilde{\mathcal{N}}_3$, we can conclude that $\mathcal{N}_3$ and $\tilde{\mathcal{N}}_3$ really have no difference in trajectories. The above inequalities also imply that $\tilde{u}_{xi}$ is different from each other for $x \in \mathcal{N}$. Similarly we have
\[
E_3(V(t)) - E_3(V(t - 1)) = -\sum_{i=1}^{m} [\tilde{a}_{ai}(t) - \tilde{a}_{bi}(t)]
\] (22)
and the strict inequality $E_3(V(t)) - E_3(V(t - 1)) < 0$ holds whenever $V(t) \neq V(t - 2)$. Since $\Omega_c$ contains a finite number of elements, the sequence $\{V(t)\}$ must converge to a limit cycle. We suppose $\{V(0), V(1), \ldots, V(\tau - 1), V(\tau) = V(0)\}$ is a cycle of period $\tau$. Being the energy function, $E_3(t)$ is necessarily constant on the cycle. If $\tau > 2$ we have $V(\tau) \neq V(\tau - 2)$ and then $E_3(V(\tau)) - E_3(V(\tau - 1)) < 0$ holds. This is a contradiction. Therefore the trajectory $\{V(t)\}$ of $\tilde{\mathcal{N}}_3$ converges to a stable state or a limit cycle of length 2. So does $\mathcal{N}_3$.

If $W$ is nonnegative definite, we then can show that the function defined by
\[
\tilde{E}(t) = -\frac{1}{2}V^TW(t)V + \tilde{T}^TV(t)
\]
is an energy function of $\mathcal{N}_3$ which satisfies
\[
\tilde{E}(V(t)) - \tilde{E}(V(t - 1)) = -\langle WV(t - 1) - \tilde{T}, V(t) - V(t - 1) \rangle \\
= -\frac{1}{2} \langle WV(t) - V(t - 1), V(t) - V(t - 1) \rangle \\
= -\sum_{i=0}^{m} (\tilde{a}_{ai,j}(t) - \tilde{a}_{bi,j}(t)) \\
= -\frac{1}{2} \langle (V(t) - V(t - 1))^TW(V(t) - V(t - 1)) \rangle,
\]
where $(a_{i,j})$ and $(b_{i,j})$ are the unique nonzero components on the $i$th columns of $V(t)$ and $V(t - 1)$. The first and the second terms of the right-hand side of
the above equation are nonpositive, due to $W$ being nonnegative definite. Thus we conclude that $\tilde{E}(t) \leq \tilde{E}(t - 1)$ for any $t$. Observe that if $V(t) \neq V(t - 1)$, the strict inequality $\tilde{u}_{i,a}(t) > \tilde{u}_{i,a}(t)$ holds for certain $i$. Therefore the strict inequality $\tilde{E}(t) < \tilde{E}(t - 1)$ holds whenever $V(t) \neq V(t - 1)$. This implies the convergence of $\{V(t)\}$ and completes the proof of Theorem 2.

**Remark 4.** Theorem 2 shows that the convergence of the WTA-type networks $N_3$ and $N_4$ are very much like that of Hopfield networks when updated in a parallel mode [15]. Together with Theorem 1, this then underlies the feasibility of $N_1 \sim N_4$ when used as computation models.

### 3.2. Reliability and efficiency

In this subsection, we derive the general conditions under which a WTA-type network can be both reliably and efficiently employed. Let us start with a definition of local minimizers.

**Definition 4.** A matrix $V^* \in \Omega \subset \{0, 1\}^n$ is called a local minimizer of the function $E(V)$ in (17) if, for any $V$ in the feasible domain different from $V^*$ only at one block (one column for $N_1$ or one row for $N_2$), $E(V^*) \leq E(V)$. A local minimizer $V^*$ is said to be a global minimizer if $E(V^*) \leq E(V)$ for any $V \in \Omega$.

When a combinatorial optimization problem of the form (1) is solved, it is certainly ideal to find a global minimizer of $E(V)$. This is, however, not always possible and practical, because the problem (1) is known to be NP-complete in most cases. Therefore, in practice, one often solves the problem by finding a suboptimal solution through a polynomial computation time. There is no exact definition of the meaning of a suboptimal solution. However the local minimizer in Definition 4 can safely serve as a candidate for such solutions in any case. Thus, solving the problem (1) may be based on two different purposes: either to find an optimal solution or to find a suboptimal solution. Since a WTA-type network $N$ finds such solutions through the search of its stable states, it is then required that, for the first purpose, any local minimizer of $E$ must be in the stable states of $N$ in case the optimal solution is lost, and, for the second purpose, any stable state of $N$ should be a local minimizer of $E(V)$, so that any solution generated by the network $N$ is suboptimal. This former requirement is related to the reliability of the network $N$, and the latter related to the efficiency of $N$. To further characterize these two properties, we let

- $\Omega(E)$ be the set of all local minimizers of $E(V)$, and
- $\Omega(N)$ be the set of all stable states of $N$.

According to [29], we refer the property $\Omega(E) \subset \Omega(N)$ to regularity, $\Omega(N) \subset \Omega(E)$ to normality and $\Omega(N) = \Omega(E)$ to completeness of the network $N$. Then it can be seen that the regularity of a network implies its reliability, and the
normality implies high efficiency. Therefore, the completeness of a network implies both reliability and high efficiency. If the function \( E(V) \) is an energy function of a network \( \mathcal{N} \), we will also say that the function \( E(V) \) is a regular, normal or complete energy function of \( \mathcal{N} \), provided the network \( \mathcal{N} \) is regular, normal or complete [29]. The following theorem offers the conditions for the energy function of WTA-type networks to become regular, normal or complete.

**Theorem 3.** Assume \( W = (w_{x(i,j)})_{nm\times nm} \) is symmetric, \( E(V) \), \( C_{xy}^{(k)} \) and \( R_{ij}^{(x)} \) defined as in Theorem 1.

1. If \( C_{xy}^{(k)} \geq 0 \) for any \( k \in M \) and \( x, y \in \mathbb{N} \) (respectively, \( R_{ij}^{(x)} \geq 0 \) for \( i, j \in M \) and \( x \in \mathbb{N} \)), then the network \( \mathcal{N}_1 \) (respectively \( \mathcal{N}_2 \)) is regular.

2. If \( C_{xy}^{(k)} \leq 0 \) for any \( k \in M \) and \( x, y \in \mathbb{N} \) (respectively, \( R_{ij}^{(x)} \leq 0 \) for \( i, j \in M \) and \( x \in \mathbb{N} \)), then the network \( \mathcal{N}_1 \) (respectively \( \mathcal{N}_2 \)) is normal.

3. If \( C_{xy}^{(k)} = 0 \) for any \( k \in M \) and \( x, y \in \mathbb{N} \) (respectively, \( R_{ij}^{(x)} = 0 \) for \( i, j \in M \) and \( x \in \mathbb{N} \)), then the network \( \mathcal{N}_1 \) (respectively \( \mathcal{N}_2 \)) is complete.

**Proof.** Due to the similarity, we prove the part for the network \( \mathcal{N}_1 = (W, T, WTA_c, S) \).

1. If \( V^* \) is a local minimizer of \( E(V) \), and \( V = V^{(k)} \) is any state whose \( k \)th column differs from \( V^* \) and other columns are identical with those of \( V^* \), then \( (V^{(k)} - V^*)_{x} = \delta_{ab} - \delta_{tb} \) where \( a \) and \( b \) are the unique nonzero elements in the \( k \)th column of \( V^* \) and \( V \), respectively. Then we obtain

\[
\{E(V) - E(V^*)\} + \frac{1}{2} C_{ab}^{(k)} = u_{ak}^* - u_{bk}^*,
\]

where \( u_{ak}^* \) is the activation potential of the neuron \((x,k)\) associated with the state \( V^* \). By the assumption of the theorem, \( C_{ab}^{(k)} \geq 0 \) and \( E(V) \geq E(V^*) \). So the right-hand side of Eq. (23) is nonnegative. As \( b \) is arbitrary, this shows \( u_{ak}^* = \max_{b \in \mathbb{N}} \{u_{bb}^*\} \). For the operator \( WTA_c \) with the OWLI priority regulation, \( V^{(k)} = [WTA_c(L(V^*))^{(k)}] = WTA_c((WV^* - T)^{(k)}) \). Since \( k \) is arbitrary, \( V^* \) is an equilibrium state of the network \( \mathcal{N}_1 \). So any local minimizer of \( E \) is a stable state of \( \mathcal{N}_1 \). This justifies (i).

2. Under the condition of the theorem, we have \( C_{ab}^{(k)} \leq 0 \). If \( V^* \) is a stable state of \( \mathcal{N}_1 \), then \( u_{ak}^* = \max_{b \in \mathbb{N}} \{u_{bb}^*\} \). From (23) we obtain \( E(V^{(k)}) \geq E(V^*) \) for any \( k \in M \). Hence \( V^* \) is a local minimizer of \( E \). Since \( V^* \) is an arbitrary stable state of \( \mathcal{N}_1 \), we conclude that \( E(V) \) is a normal energy function of \( \mathcal{N}_1 \).

3. It directly follows from (i) and (ii). With this, the proof of Theorem 3 is complete.

**Remark 5.** Theorem 3 underlies how a combinatorial optimization problem of (1) should be properly mapped to a WTA-type network so that the problem can be solved either reliably or efficiently, or both. We will see in the next section that there are some problems which can certainly be solved in this way. It is worth observing that the regularity condition presented in Theorem 3 is identical with
that used in Theorem 1. This coincidence facilitates the practical applications of the WTA-type networks on solving combinatorial optimization problems. For a WTA-type network, the local minimizer \( V^* \) differs from its neighbor state on a set of neurons, rather than a single neuron for the case of Hopfield networks. In other words, the local minimizers of the WTA-type networks have larger neighborhood which leads to less local minimizers. This enables the WTA-type networks to find better near-optimal solutions more efficiently.

3.3. Iteration bounds

We take \( N_1 = (W, T, WTA_c, S) \) as a typical example to study the maximum number of iterations needed for a WTA-type network to converge to a stable state.

**Theorem 4.** If \( W \) is symmetric and \( C_{x,y}^{(k)} \geq 0 \) for any \( x, y \in N, k \in M \), then the network \( N_1 = (W, T, WTA_c, S) \) converges to a stable state from any initial state \( V(0) \in \Omega_c \) at most within \((\rho(W)m^2 + 2mt_M)/(e + f)\) iterations, where \( \rho(W) \) is the spectral radius of \( W \), \( t_M = \max_{x \in N, i \in M} \{|t_{ai}\}| \), \( f = \frac{1}{2} \min_{x,y \in N, k \in M} \{C_{x,y}^{(k)}\} \) and \( e = \min_{V \in \Omega_c} \min_{i \in M} \min_{x \in N} \{u_{ai}(V) - u_{ai}(V)\} = \min_{y \in N} u_{yi} \), and \( u_{ai}(V) \neq u_{ai}(V) \).

**Proof.** Assume, at time \( t \), \( V(t) \) is updated on the \( k \)th column, then
\[
E(V(t+1)) - E(V(t)) = -(u_{ab}(t+1) - u_{bk}(t+1)) - \frac{1}{2} C_{a,b}^{(k)}
\]
(as Eq. (19)). If \( \Delta V(t) = V(t+1) - V(t) \neq 0 \) (hence, \( u_{ab}(t+1) > u_{bk}(t+1) \) holds by the OWLI priority regulation on \( WTA_c \)), then from conditions of the theorem, it follows that
\[
|E(V(t+1)) - E(V(t))| = |u_{ab}(t+1) - u_{bk}(t+1)| + \frac{1}{2} |C_{a,b}^{(k)}| 
\geq e + f > 0.
\]

On the other hand, \( V(t) \in \Omega_c \) for any \( t \geq 0 \), which implies
\[
|E(V)| = \left| -\frac{1}{2} V^T WW + T^T V \right| \leq \frac{1}{2} |V^T WW| + |T^T V|
\leq \frac{1}{2} \rho(W) |V|^2 + \sum_{x=1}^{n} \sum_{i=1}^{m} t_{ai} v_{ai}
\leq \frac{1}{2} \rho(W) m^2 + m \max_{x \in N, i \in M} \{|t_{ai}|\}
= \frac{1}{2} \rho(W) m^2 + mt_M,
\]
where \(||V||_2\) is the Euclidean norm of \( V \).

Before reaching a stable state, the network \( N_1 \) changes its state at least once per iteration. The upper bounder in theorem is direct from Eqs. (24) and (25). That is, the proof of the theorem is completed.

Theorem 4 shows that the WTA-type network \( N_1 \) is always convergent in finite number of steps. Similar bounds can be obtained for other WTA-type networks.
4. Application to the TSP and simulations

Taking the TSP as an example, we demonstrate an important application of the proposed WTA-type networks to the solution of combinatorial optimization problems (1). We present numerical simulations of the networks $\mathcal{N}_1$ and $\mathcal{N}_5$, as compared the discrete Hopfield neural network (DHNN) [25], the analogue Hopfield neural network (AHNN) [2,3], Boltzmann machine (BM) [25], the self-organizing map [12] and other methods [25].

4.1. Neural network architecture

4.1.1. Energy functions

Through assigning $n$ neurons for $n$ cities and assuming that a city is visited at the $i$th order whenever the state of the $i$th neuron of the city takes value 1 (i.e., it is excited), a TSP with $n$ cities is then formulated as the minimization of the following energy function [2]:

$$E_A = \frac{A}{2} \sum_{x=1}^{n} \left( \sum_{i=1}^{n} v_{xi} - 1 \right)^2 + \frac{B}{2} \sum_{x=1}^{n} \left( \sum_{i=1}^{n} v_{xi} - 1 \right)^2 + \frac{1}{2} \sum_{x,y,i=1}^{n} v_{xi} \{ d_{xy} v_{y,i+1} + d_{yx} v_{y,i-1} \},$$  \hspace{1cm} (26)

where $v_{xi} \in \{0, 1\}$ is the state of neuron $i$ for city $x$, $d_{xy}$ is distance between city $x$ and city $y$, $A$ and $B$ are two penalty parameters.

We rewrite the above function as

$$E_{\text{TSP}} = \frac{A}{2} \sum_{x=1}^{n} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} v_{xi} v_{xj} + \frac{B}{2} \sum_{x=1}^{n} \sum_{y=1, y \neq x}^{n} \sum_{i=1}^{n} v_{xi} v_{yi} + \frac{D}{2} \sum_{x=1}^{n} \sum_{i=1}^{n} v_{xi}^2 + \frac{1}{2} \sum_{x,y,i=1}^{n} v_{xi} \{ d_{xy} v_{y,i+1} + d_{yx} v_{y,i-1} \},$$  \hspace{1cm} (27)

where $D$ is an additional penalty parameter. If $D = (A + B)/2$, Eq. (27) is identical with Eq. (26).

Comparing $E_{\text{TSP}}$ with (17), we yields the following specification of the WTA-type networks:

$$t_{xi} = 0$$  \hspace{1cm} (28)

and

$$w_{x,i;x,y,j} = - \{ A \delta_{xy} (1 - \delta_{ij}) + B \delta_{ij} (1 - \delta_{xy}) + D \delta_{ij} \delta_{xy} + d_{xy} \delta_{j,i+1} + d_{yx} \delta_{j,i-1} \}.$$  \hspace{1cm} (29)

It is seen from (28) that $W = (w_{x,i;x,y,j})_{nm\times nm}$ is symmetric. This shows that with the energy function (27), the deduced WTA-type networks always have symmetric connections.
4.1.2. Specifications of the penalty parameters

We derive some advice to help appropriately specify the penalty parameters \( A, B, D \) in (27) from the theory of the WTA-type networks.

First of all, the network \( \mathcal{N}_1 \) has been designed for realizing the column-feasibility constraints, thus, the second column-feasibility constraint term in Eq. (26) can be ignored. Thus we can take \( B = 0 \) in implementation of \( \mathcal{N}_1 \). So does \( A = 0 \) for \( \mathcal{N}_2 \).

Secondly, by Theorems 1 and 2, \( C_x^{(k)} = 0 \) is the sufficient condition for the network \( \mathcal{N}_1 \) to be feasible, reliable and efficient as a solver of combinatorial optimization problems. From Eq. (28), this implies \( C_x^{(k)} = w_{k,y,i} + w_{y,k,i} - 2w_{x,k,y,i} = 2D - 2B = 0. \) That is, \( D = B \) should be satisfied. By specifying \( R_x^{(i)} = 0 \) in (28), we can similarly get the condition \( D = A \) for the network \( \mathcal{N}_2 \) to be both reliable and efficient. Thus, we can set \( D = B = A \) whenever \( E_{\text{TSP}} \) is used as the energy of the network \( \mathcal{N}_5 \).

The following theorem further provides the foundation of the choice of these parameters.

Theorem 5. For any given TSP,

(i) if \( B = D \) and \( A > d_c = \max_{x_1 \neq x_2, y_1, y_2 \in N} \{d_{y_1,x_1} + d_{x_1,x_2} - d_{y_1,x_2} - d_{y_2,x_2}\} \) in \( E_{\text{TSP}} \), then \( V = (v_{xi})_{n \times n} \) is a valid tour of the TSP iff \( V \) is a stable state of \( \mathcal{N}_1 \). In this case, \( \mathcal{N}_1 \) converges to a stable state within one iteration from any initial state \( V_0 \in \Omega_c \).

(ii) if \( A = D \) and \( B > d_r = \max_{x \in N} \{\sum_{y=1}^{n} \max \{d_{xy}, d_{yx}\}\} \) in \( E_{\text{TSP}} \), then \( V = (v_{xi})_{n \times n} \) is a valid tour of the TSP iff \( V \) is a stable state of \( \mathcal{N}_2 \). In this case, \( \mathcal{N}_2 \) converges to a stable state within one iteration from any initial state \( V_0 \in \Omega_r \).

Proof. We only prove the part for \( \mathcal{N}_1 = (W, T, WTA_c, S) \). Due to its projection property of \( WAT_c \) to the column-feasible domain \( \Omega_c \), we only consider the column-feasible states below.

According to Eqs. (7) and (28), when \( B = C \), the activation potential of the neuron \((x,i)\) located in the \( i \)th column of a column-feasible state has the form

\[
    u_{xi} = -A \sum_{j \neq i} v_{xj} - d_{y_1,x_1} - d_{x_1,y_1} - B,
\]

where \( x, y_1, y_2 \in N \), and the \((i-1)\)th and \((i+1)\)th column states have nonzero components on \((y_1, i-1)\) and \((y_2, i+1)\), respectively.

Consider a state corresponding to a valid tour, i.e., every row and every column have exactly one excited neuron. The potential of the excited neuron in a column then has the form \(-d_{y_1,x_1} - d_{x_1,y_2} - B\), and the activation potentials of its neighbors have the form \(-A - d_{y_1,x_2} - d_{x_2,y_2} - B\). The former is clearly larger than the latter whenever \( A > d_c \). So the excited neuron still wins the competition again when the WTA operation is applied to this column. Consequently, this state is a stable state of \( \mathcal{N}_1 \). Therefore every state corresponding to a valid tour is a stable state of the network \( \mathcal{N}_1 \).
Inversely, consider a state in $\Omega_c$ which does not correspond to any valid tour. Then there is at least one row, say $x$th, which has no excited neurons and there is one row, say $y$th, which has more than one excited neurons. Without loss of generality, we assume $(y,i)$ is one of the excited neurons, i.e., $v_{yi} = 1$. Then according to Eq. (30), we get

$$u_{yi} = -A \sum_{j \neq i} v_{yj} - d_{yi,x} - d_{yi,y_2} - B < -A - d_{yi,x} - d_{y_1,y_2}$$

and $u_{xi} = -d_{yi,x} - d_{x,y_2} - B$ where $y_1, y_2 \in \mathbb{N}$, and the $(i-1)$th and $(i+1)$th column states are assumed to take the unique nonzero component on $(y_1,i-1)$ and $(y_2,i+1)$, respectively. $u_{xi} > u_{yi}$ holds whenever $A > d_c$. So the winner is not located at the older winner neuron $(y,i)$, that is, this state is not a stable state. Thus, every stable state must correspond to a valid tour.

From the above argument, we see that the winner can only be located at the row without any excited neuron. So after operating $WTA_c$ on each column exactly once, that is, an iteration of $\mathcal{N}_1$, the state of the network becomes a permutation matrix. A permutation matrix corresponds to a valid tour. Therefore $\mathcal{N}_1$ will converge to a stable state after one iteration. This completes the proof of the theorem.

Theorem 5 says that as long as $A > d_c$, there is a one-to-one correspondence between the stable states of $\mathcal{N}_1$ and the valid tours of the TSP. This highlights the importance of the bound $d_c$. But, the bound $d_c$ may be overestimated. In the implementation, we therefore use $A = p \times d_c$ with an adjustable factor $p$ in $[0,1]$. Similar cases happen for $\mathcal{N}_2$ and $\mathcal{N}_5$.

4.2. Simulations and comparisons

4.2.1. Implementation

With the above specified $WTA$-type neural network architecture, we have applied them to compute a set of TSPs. To assess the quality of an approximate solution and facilitate a comparison of the $WTA$-type networks with other algorithms, several commonly used TSPs with known optimal tours were used. Hopf10 is the 10-city ETSP used by Hopfield and Tank [17], whose optimal tour length is known to be 2.69. Hopf30, a 30-city ETSP, also originally in [17], has the optimal tour length of 4.15. The other three TSPs were taken from the TSP data library collected by Reinet (FTP:softlib.rice.edu). Fri26, a 26-city STSP, has the optimal tour length of 937. Eil51, a 51-city ETSP, has the optimal tour length of 426.0. Eil101, a 101-city ETSP, has the optimal tour length of 629.0.

Owing to the similar feature of $\mathcal{N}_2$ with $\mathcal{N}_1$, we have only implemented the $WTA$-type networks $\mathcal{N}_1$ and $\mathcal{N}_5$. In the implementation of these networks, the factor $p$ in the penalty parameter $A = p \times d_c$ is taken differently as the problem size varies. Besides such a “fixed-setting” scheme of $p$, we have also implemented the networks with the factor $p$ in a dynamically increasing manner. A $WTA$-type network with such “dynamically update-setting of $p$” scheme will be referred to as the modified version of the network, say, the modified $\mathcal{N}_1$. 
Table 1
Comparison of $\mathcal{N}_1$ and DHNN when applied to TSPs

<table>
<thead>
<tr>
<th>TSP (optimal length)</th>
<th>Network model (penalty $p$)</th>
<th>Simulation runs</th>
<th>Shortest tour length</th>
<th>Average tour length</th>
<th>Longest tour length</th>
<th>Average run time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hopf10 (2.69)</td>
<td>$\mathcal{N}_1(0.40)$</td>
<td>100</td>
<td>2.69</td>
<td>3.23</td>
<td>3.76</td>
<td>0.47</td>
</tr>
<tr>
<td></td>
<td>DHNN</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eil51 (426.0)</td>
<td>$\mathcal{N}_1(0.36)$</td>
<td>10</td>
<td>814.1</td>
<td>848.0</td>
<td>884.5</td>
<td>43.4</td>
</tr>
<tr>
<td></td>
<td>DHNN</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eil101 (629.0)</td>
<td>$\mathcal{N}_1(0.36)$</td>
<td>10</td>
<td>1599.5</td>
<td>1809.8</td>
<td>1975.5</td>
<td>341.8</td>
</tr>
<tr>
<td></td>
<td>DHNN</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In our simulations, the distance $d_{xx}$ was all assumed to be zero, and all initial states were taken randomly in the column-feasible set $\Omega_c$ for both networks $\mathcal{N}_1$ and $\mathcal{N}_3$. For each TSP, the performance of the algorithm is measured by the average tour length, the shortest tour length, the longest tour length and the average computation time based on a total of 10 or 100 runs. All our simulations were executed on a Pentium 100MHZ personal computer.

4.2.2. Comparison of $\mathcal{N}_1$ with DHNN

The performance of $\mathcal{N}_1$ and DHNN applied to Hopf10, Eil51 and Eil101 is shown in Table 1. The simulation results of DHNN is quoted from [25]. The network $\mathcal{N}_1$ (with $p = 0.40$) find the optimal tour with the length of 2.69 five times in a total of 100 runs for Hopf10. The average tour length is 3.23, 20.1% longer than the optimal tour. From Table 1, it is seen that for all three TSPs, the longest tour obtained by the network $\mathcal{N}_1$ is even much better than the average tour length obtained by DHNN. We may conclude that $\mathcal{N}_1$ outperforms DHNN in the quality of solutions.

BM, which incorporates simulated annealing into DHNN, has been also reported to solve the TSPs in [25]. Both BM and $\mathcal{N}_1$ can find the optimal tour of Hopf10. The solution quality of $\mathcal{N}_1$ is slightly worse than BM on average. The simulation results of BM on Eil51 are as follows: For a total of 5 runs the shortest tour length is found to be 805, the longest 893, and the average 859.6. The result are not significantly different from the result by the network $\mathcal{N}_1$. But according to [25], it took 16 h for BM to get a tour on a HP7000 workstation, while it took 43.4 s for $\mathcal{N}_1$ to yield a tour of Eil51 on our PC. This shows much higher computational efficiency of the network $\mathcal{N}_1$ over BM. No exact comparison can be provided for the TSP Eil101, since, according to [25], more than 10 of days should be needed for BM to obtain a solution. On the contrary, our $\mathcal{N}_1$ spends only 341.8 s. Thus, the $\mathcal{N}_1$ can generate tours comparable to BM with much less computation time.
Table 2
Comparison among $\mathcal{N}_5$, AHNN and NPA–LME algorithm when applied to TSPs

<table>
<thead>
<tr>
<th>TSP (optimal length)</th>
<th>Network model (penalty $p$)</th>
<th>Simulation run</th>
<th>Shortest tour length</th>
<th>Average tour length</th>
<th>Longest tour length</th>
<th>Average runtime (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hopf10 (2.69)</td>
<td>$\mathcal{N}_5(0.16)$</td>
<td>1000</td>
<td>2.69</td>
<td>2.90</td>
<td>2.93</td>
<td>9.0</td>
</tr>
<tr>
<td></td>
<td>AHNNc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\mathcal{N}_5(0.16)$</td>
<td>1000</td>
<td>2.69</td>
<td>2.79</td>
<td>2.93</td>
<td>9.0</td>
</tr>
<tr>
<td>Hopf30 (4.15)</td>
<td>$\mathcal{N}_5(0.16)$</td>
<td>100</td>
<td>4.63</td>
<td>5.97</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AHNNc</td>
<td>1000</td>
<td>4.49</td>
<td>4.88</td>
<td>5.41</td>
<td>187.0</td>
</tr>
<tr>
<td></td>
<td>$\mathcal{N}_5(0.16)$</td>
<td>100</td>
<td>4.31</td>
<td>4.45</td>
<td>5.8</td>
<td></td>
</tr>
<tr>
<td>Eil51 (426.0)</td>
<td>NPA–LME</td>
<td>10</td>
<td>496</td>
<td>533.1</td>
<td>562</td>
<td>4.5 h</td>
</tr>
<tr>
<td></td>
<td>$\mathcal{N}_5(0.11)$</td>
<td>10</td>
<td>472.5</td>
<td>514.4</td>
<td>545.5</td>
<td>2232.8</td>
</tr>
<tr>
<td>Eil101 (629.0)</td>
<td>NPA–LME</td>
<td>10</td>
<td>865</td>
<td>900.8</td>
<td>924</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\mathcal{N}_5(0.11)$</td>
<td>10</td>
<td>844.8</td>
<td>894.5</td>
<td>943.1</td>
<td>7259.0</td>
</tr>
</tbody>
</table>

4.2.3. Comparison of $\mathcal{N}_5$ with AHNN and NPA–LME algorithm

Table 2 shows the comparison results of the network $\mathcal{N}_5$, the AHNN and the NPA–LME algorithm, when applied to TSPs, where the NPA–LME algorithm is the latest modification of Hopfield model provided in [25]. These two versions of AHNNs in Abe [2] and Abe [3] are denoted by AHNNc and AHNNd in Table 2, respectively, and the best results are used for comparison.

From Table 2, it is seen that among a total of 1000 runs, the average tour length for Hopf10 obtained by $\mathcal{N}_5$ is 2.79, only 3.7% longer than the optimal tour, and the average computation time is 9.0 s. The average length for Hopf30 by $\mathcal{N}_5$ is 4.88, 17.6% longer than the optimal tour, and the average computation time 187.0 s. It can be seen that the average tour lengths based on 1000 or 100 runs found by $\mathcal{N}_5$ was shorter than those by AHNNc, while they are longer than those by AHNNd for Hopf10 and Hopf30. The performance of $\mathcal{N}_5$ is comparable to AHNNs.

To evaluate the effectiveness of $\mathcal{N}_5$ for larger-scale TSPs, we have compared the $\mathcal{N}_5$ with the network partitioning algorithm (NPA) proposed in [25] (note that no simulation results were reported for the AHNN or its variants to solve larger size TSPs). Through partitioning the large-scale Hopfield neural network into many subnetworks, NPA solves a large-scale TSP by using the subnetworks, which can be well solved by certain algorithms, say, the local minima escape algorithm (LME) [25]. We hereafter call the combination of the NPA technique and the LME algorithm the NPA–LME algorithm. The simulation results for the NPA–LME algorithm in Table 2 are from [25], running on a HP7000 workstation.

In our simulations, the average tour length for Eil51 obtained by $\mathcal{N}_5$ is 514.4, 20.7% longer than the optimal tour. It is shorter than the average tour length 533.1 by the NPA–LME algorithm. The average computation time of $\mathcal{N}_5$ is about 0.62 h on our PC, which is only 13.7% of 4.5 h, the average computation time of the NPA–LME algorithm on a HP7000 workstation. For the 101-city problem Eil101,
Table 3
Comparison of the modified $\mathcal{N}_1$ and the Favata and Walker algorithm (FW algorithm) for TSPs.

<table>
<thead>
<tr>
<th>TSP (optimal length)</th>
<th>Network models</th>
<th>Shortest tour length</th>
<th>Average tour length</th>
<th>Solution quality</th>
<th>Longest tour length</th>
<th>Average run time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hopf10 (2.69)</td>
<td>Modified $\mathcal{N}_1$</td>
<td>2.69</td>
<td>2.781</td>
<td>1.033</td>
<td>2.85</td>
<td>3.1</td>
</tr>
<tr>
<td></td>
<td>FW Algorithm</td>
<td>2.69</td>
<td>2.850</td>
<td>1.059</td>
<td>3.21</td>
<td>2.3</td>
</tr>
<tr>
<td>Hopf30 (4.15)</td>
<td>Modified $\mathcal{N}_1$</td>
<td>4.37</td>
<td>4.800</td>
<td>1.157</td>
<td>5.27</td>
<td>34.4</td>
</tr>
<tr>
<td></td>
<td>FW Algorithm</td>
<td>4.67</td>
<td>5.145</td>
<td>1.242</td>
<td>5.59</td>
<td>10.2</td>
</tr>
<tr>
<td>Eil51 (426.0)</td>
<td>Modified $\mathcal{N}_1$</td>
<td>449.6</td>
<td>473.0</td>
<td>1.110</td>
<td>509.4</td>
<td>276.7</td>
</tr>
<tr>
<td></td>
<td>FW Algorithm</td>
<td>455.3</td>
<td>496.2</td>
<td>1.165</td>
<td>533.7</td>
<td>23.1</td>
</tr>
<tr>
<td>Eil101 (629.0)</td>
<td>Modified $\mathcal{N}_1$</td>
<td>685.4</td>
<td>714.3</td>
<td>1.136</td>
<td>758.3</td>
<td>1032.8</td>
</tr>
<tr>
<td></td>
<td>FW Algorithm</td>
<td>714.2</td>
<td>728.5</td>
<td>1.158</td>
<td>754.3</td>
<td>97.6</td>
</tr>
</tbody>
</table>

the average tour length found by $\mathcal{N}_5$ is 894.5, 42.2% longer than the optimal tour, and it is still less than the average tour length found by the NPA–LME algorithm, 900.8. Thus we can conclude that $\mathcal{N}_5$ is comparable to the NPA–LME algorithm in the solution quality with less computation time.

4.2.4. Comparison of the Modified $\mathcal{N}_1$ with the Favata and Walker algorithm

In our simulations, the factor $p$ in the modified WTA-type network $\mathcal{N}_1$ was initialized with 0 and then updated iteration by iteration with increment $\Delta p > 0$. Furthermore, to speed up the convergence, once the network’s states become stable in two consecutive iterations, the new value is set according to $p_{\text{new}} = \max\{p_{\text{old}} + \Delta p, (d_{yx} + d_{xz})/d_c\}$ where $x$, $y$, and $z$ are the row indices such that

$$(x, y, z) = \arg\left\{\min_{y\neq z} \left\{d_{yx} + d_{xz} \left| \sum_{i=0}^{n} v_{xi} = 0 \text{ and } v_{yk} = v_{z,k+1} = 1 \text{ for } k \in N\right.\right\}\right\}.$$ 

All these strategies assure that the penalty factor $p$ is monotonously increased. Thus, by Theorem 5, the network then can yield a tour when $p$ gets large enough, say, $p > 1$.

The Favata and Walker algorithm, a self-organizing neural network technique developed in [12] for ETSPs, can generate tours which are slightly longer than those produced by simulated annealing but spends much shorter computation time. In our implementation, all the learning parameters were set according to those advised in [12].

Table 3 shows the simulation results of the modified $\mathcal{N}_1$ and the Favata and Walker algorithm, in which the solution quality is defined as the ratio of the average tour length to the optimal tour length. It is seen that for Hopf10 and Hopf30, both algorithms generate very high quality solutions of the TSPs, and their best tours in total 10 runs are within a few percentage over the optimum. The modified $\mathcal{N}_1$ generates slightly better tours than the Favata and Walker algorithm. For the two larger-scale TSPs, Eil51 and Eil101, the shortest tour lengths obtained by the modified $\mathcal{N}_1$ are very near the optimal ones. The tours yielded by the Favata
and Walker algorithm are also near the optimal tours but slightly longer than those obtained by the modified $N_1$. Fig. 1 shows typical tours obtained by two algorithm for EIL101. From these simulation results, we conclude that the modified $N_1$ can yield high quality solutions of TSPs comparable with the Favata and Walker algorithm. The solution quality yielded by the modified $N_1$ for these TSPs are less than 1.2. This shows that the solution quality generated by the modified $N_1$ for TSPs is fairly steady.

On the other hand, as Table 3 shows, this is at the cost of longer computation time by the modified $N_1$. But, it should be noted that the WAT-type networks, similar to the Hopfield model, can likely solve problems quickly by using dedicated hardware. Furthermore, the WTA-type networks can perform well on STSPs as well. We also applied the modified $N_1$ to the STSP Fri26. Among a total of 10 simulation runs, our modified network $N_1$ finds the tours with the shortest tour length 1021, the average tour length 1050.6, and the longest tour length 1086. The average tour length is 12.1% longer than the optimal one. However, the Favata and Walker algorithm and other self-organizing map networks are not valid for STSP in general.

The series of simulations show that the WTA-type networks are effective for yielding some good solutions of TSPs. We also believe that the WTA-type networks are effective on solving other combinatorial optimization problems like set partitioning [22].

5. Conclusion

Through incorporating the Winner-Take-All mechanism of the competitive neural networks into automata networks, we have introduced a new type of neural networks—the WTA-type networks. Among these networks, the WTA mechanism is used to handle some constraints automatically. The theoretical foundation of the proposed networks has been developed from the viewpoint of taking them as com-
binatorial optimization solvers. Two key issues, feasibility and reliability, related to the use of networks have been studied. It has been shown that any combinatorial optimization problems whose feasible solutions are permutation matrices (or, column, row-permutation matrices) can be solved both reliably and efficiently by the proposed WTA-type networks. We have demonstrated the power of these networks in solving a class of combinatorial optimization problems. A series of numerical simulations with TSPs has shown that the network $\mathcal{N}_1$ generally performs better than the Boltzmann machine, the network $\mathcal{N}_5$ performs as well as the NPA–LME algorithm and the modified $\mathcal{N}_1$ is comparable with one competitive neural network—the Favata and Walker algorithm, in terms of the solution quality, at least for not so large scale ETSPs. The results indicate that the WTA-type networks are effective and efficient to some degrees.

The paper concentrates on studying the WTA-type network for performing optimization tasks. The networks developed, however, is by no means only significant for the computation use. In a forthcoming paper, we will examine in detail the associate memory aspects of the networks in terms of their capacity and error-correcting capability.

Acknowledgements

This research was partially supported by the earmarked grant CUHK 352/96E of the Hong Kong Research Grant Council, the Hi-Tech R & D (863–306) Project and the Natural Science Foundation of China. The authors thank the anonymous reviewers for their valuable comments to strengthen the paper.

References


Zong-Ben Xu received the MS degree in mathematics in 1981 and the Ph.D. degree in applied mathematics in 1987 from Xi’an Jiaotong University, China. In 1988, he was a postdoctoral researcher in the Department of Mathematics, the University of Strathclyde, United Kingdom. He worked as a research fellow in the Information Engineering Department from February 1992 to March 1994, the Center for Environment Studies from April 1995 to August 1995, and the Mechanical Engineering and Automation Department from September 1996 to October 1996, at the Chinese University of Hong Kong. From January 1995 to April 1995, he was a research fellow in the Department of Computing at the Hong Kong Polytechnic University. He has been with the faculty of Science and Research Center for Applied Mathematics at Xi’an Jiaotong University since 1982, where he was promoted to associate professor in 1987 and full professor in 1991, and now serves as an authorized Ph.D. supervisor in mathematics, dean of the Faculty of Science, and director of the Institute for Information and System Sciences. He has published two monographs and more than 80
academic papers on nonlinear functional analysis, numerical analysis, optimization techniques, neural networks, and genetic algorithms, most of which are in international journals. His current research interests include neural networks, evolutionary computation, and multiple objective decision making theory. Dr. Xu holds the title “Owner of Chinese Ph.D. Degree Having Outstanding Achievements” awarded by the Chinese State Education Commission and the Academic Degree Commission of the Chinese Council in 1991. He is a member of the New York Academy of Sciences and International Mathematicians Union (IMU).

Hui-Dong Jin received his B.Sc. degree from the Department of Applied Mathematics in 1995, and his M.Sc. degree from the Institute of Information and System Sciences in 1998, both from Xi’an Jiaotong University, China. He currently is a Ph.D. candidate in the Computer Science and Engineering Department at the Chinese University of Hong Kong, Hong Kong. His research interests include data mining, evolutionary computation, and neural networks. He is a student member of International Society for Genetic and Evolutionary Computation.

Kwong-Sak Leung (M’77-SM’89) received his B.Sc. (Eng.) and Ph.D. degrees in 1977 and 1980, respectively, from the University of London, Queen Mary College. He worked as a senior engineer on contract R&D at ERA Technology and later joined the Central Electricity Generating Board to work on nuclear power station simulators in England. He joined the Computer Science and Engineering Department at the Chinese University of Hong Kong in 1985, where he is currently professor and chairman of the Department.

Dr. Leung’s research interests are in soft computing including evolutionary computation, neural computation, probabilistic search, information fusion and data mining, fuzzy data and knowledge engineering. He has published over 140 papers and 2 books in fuzzy logic and evolutionary computation. He has been chair and member of many program and organizing committees of international conferences, and is in the Editorial Board of Fuzzy Sets and Systems. He is a senior member of the IEEE, a chartered engineer, a member of IEE and ACM and a fellow of HKCS.

Yee Leung received the B.S.Sc. degree in geography from The Chinese University of Hong Kong in 1972, the M.A. and Ph.D. degrees in geography and the M.S. degree in engineering from The University of Cororado, USA in 1974, 1977, and 1977, respectively.

He is currently Professor of Geography and Chairman of The Department of Geography; Research Fellow of The Center for Environmental Policy and Resource Management and Deputy Academic Director of The Joint Laboratory for Geoinformation Science at The Chinese University of Hong Kong. He has published four monographs and about 100 articles in international journals and book chapters. His areas of specialization covers the development and application of intelligent spatial decision support systems, spatial optimization, fuzzy sets and logic, neural networks, and evolutionary computation. Dr. Leung serves in the editorial boards of several international journals and is council member of several Chinese professional organizations.
Chak-Kuen Wong (M’71-SM’78-F’85) received the B.A. degree (First Class Honors) in mathematics from University of Hong Kong in 1965, and the M.A. and Ph.D. degrees in mathematics from Columbia University in 1966 and 1970, respectively. He joined the IBM T. J. Watson Research Center in 1969 as a Research Staff Member and was Manager of the VLSI Design Algorithms group from 1985 to 1995. He was Visiting Associate Professor of Computer Science at University of Illinois, Urbana in 1972–73 and Visiting Professor of Computer Science at Columbia University in 1978–79. Since 1995, he has been Professor of Computer Science and Engineering at The Chinese University of Hong Kong and served as Chairman of the Department of Computer Science and Engineering from 1995–1997.

His most recent research is on VLSI design algorithms. He holds four US patents and has published more than 200 papers. He has also published two books: “Algorithmic Studies in Mass Storage Systems”, Computer Science Press, 1983, and “An Introduction to VLSI Physical Design”, McGraw-Hill, 1996. He received an Outstanding Invention Award, an Outstanding Technical Achievement Award and four Invention Achievement Awards from IBM. At the 1995 IEEE International Conference on Computer Design: VLSI in Computers & Processors (ICCD’ 95), he received a best paper award for his work on FPGA design.


He is also the founding Editor-in-Chief of the international journal “Algorithmica” since 1986, and an Advisory Board Member of “ACM Journal of Experimental Algorithmics”, since 1995.