A New K-Winners-Take-All Neural Network

Shubao Liu and Jun Wang
Department of Automation and Computer-Aided Engineering
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong
{sbliu, jwang}@acae.cuhk.edu.hk

***Abstract*— In this paper, the K-Winners-Take-All (KWTA) operation is converted to an equivalent constrained convex quadratic optimization formulation. A simplified dual neural network, called KWTA network, is further developed for solving the convex quadratic programming (QP) problem. The KWTA network is shown to be globally convergent to the exact optimal solution of the QP problem. Simulation results are presented to show the effectiveness and performance of the KWTA network.**

I. INTRODUCTION

Winner-take-all (WTA) is an operation that identifies the largest value from the input signals. Such an operation has many applications including associative memories [1], cooperative models of binocular stereo [2], Fukushima's neocogniton for feature extraction, and etc [3]. In the combinatorial optimization, this operation is also needed.

As an extension of winner-take-all operation, k-winners-take-all (KWTA) selects the k largest inputs from the total n inputs. It can be considered as a generalized version of winner-take-all operation. It has recently been shown that KWTA is computationally powerful compared with standard neural network models with threshold logic gates [4][5]. Any boolean function can be computed by a single k-winners-take-all unit applied to weighted sums of input variables. Beside the applications in neural network model, the KWTA operation has important applications in machine learning, such as k-neighborhood classification, k-means clustering, etc. As the number of inputs becomes large and/or the selection process should be operated in real time, parallel hardware implementation is desirable. There have been many attempts to design very large scale integrated (VLSI) circuits to do the KWTA operation [6-13].

This paper proposes a new neural network implementation of KWTA operation based on the equivalent quadratic optimization formulation, which has the O(N) complexity. For this network, global convergence is guaranteed and time-varying signals can be tackled. The rest of this paper is organized as following. Section II derives an equivalent formulation of KWTA, which is suitable for neural network design. Section III introduces the neural network design procedure, architecture and properties. Simulation results are reported in Section IV. Section V concludes this paper.

This work was supported by the Hong Kong Research Grants Council under Grant CUHK4165/03E.

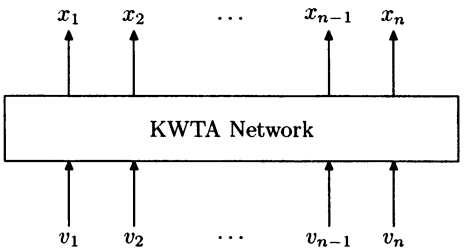

Fig. 1. The diagram of KWTA operation.

II. EQUIVALENT REFORMULATION

The optimization capability of the recurrent neural network has been widely investigated. After the seminal work of Tank and Hopfield [14][15], various neural networks have been proposed. They can be categorized as the penalty-parameter neural network [16], the Lagrange neural network [17], the deterministic annealing neural network [18], the primal-dual neural network [19][20] and the dual neural network [21][22].

Mathematically, KWTA can be formulated as a function

$$x_i = f(v_i) = \begin{cases} 1, & \text{if } v_i \in \{k \text{ largest elements of } v\}; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Fig. 1 shows the KWTA operation graphically. In this section, we will reformulate the KWTA operation as a quadratic programming problem, which is suitable for neural network design. Toward this goal, hereafter two theorems are given and proved.

Theorem 1: The solution of (1) is the same as the solution to the following discrete quadratic programming problem (2).

$$\begin{aligned} \text{minimize} \quad & ax^Tx - v^Tx \\ \text{subject to} \quad & e^Tx = k \\ & x_i \in \{0,1\}, \quad i = 1,2,\cdots,n \end{aligned} \tag{2}$$

where a is a positive constant, $v := [v_1, v_2, \cdots, v_n]^T$, $x := [x_1, x_2, \cdots, x_n]^T$, $e := [1, 1, \cdots, 1]^T$.

Proof: $e^Tx = k$ can be written as

$$\sum_{i=1}^{n} x_i = k \tag{3}$$

Because $x_i \in \{0,1\}$, we have

$$x_i^2 = x_i \tag{4}$$

From (3) and (4), we get

$$\sum_{i=1}^{n} x_i^2 = k$$

i.e.,

$$ax^T x = a\sum_{i=1}^{n} x_i^2 = ak$$

is a constant.

So the cost function can be rewritten as

$$\text{maximize} \quad v^T x$$

Suppose that the solution of (2) x^* is not the solution of (1). Without loss of generality, we assume that $x_i^* = 0$, v_i in the top k largest inputs; And $x_l^* = 1$, v_l is not in the top k largest inputs.

Because v_i is in the top kth largest inputs, and v_l is not, then

$$v_i > v_l$$

Define $\bar{x}_i := 1$, $\bar{x}_l := 0$, $\bar{x}_j := x_j^*, j \neq i, l$. $\bar{x} := [\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_n]^T$ also satisfies the constraints.

$$\begin{aligned} v^T\bar{x} &= v_i\bar{x}_i + v_l\bar{v}_l + \textstyle\sum_{j\neq i,l} v_j\bar{x}_j \\ &= v_i + \textstyle\sum_{j\neq i,l} v_j x_j^* \\ &> v_k + \textstyle\sum_{j\neq i,l} v_j x_j^* \\ &= v_i x_i^* + v_l x_l^* + \textstyle\sum_{j\neq i,l} v_j x_j^* \\ &= v^T x^* \end{aligned}$$

This contradicts with the assumption that x^* is the solution of (2).

The proof is complete. □

Denote the kth largest element as $\tilde{v}_k$, $(k+1)$th largest element as $\tilde{v}_{k+1}$. Then we have the following theorem:

Theorem 2: If $\tilde{v}_k - \tilde{v}_{k+1} \geq 2a$, then the discrete quadratic programming problem (2) and the following continuous quadratic programming problem (5) have the same solution.

$$\begin{aligned} \text{minimize} \quad & ax^T x - v^T x \\ \text{subject to} \quad & e^T x = k \\ & x_i \in [0,1], \quad i = 1, 2, \cdots, n \end{aligned} \tag{5}$$

where a is a positive constant.

Proof: If we can show that the solution of the problem (5) is in the set $\{0,1\}^n$, then the theorem is proved.

From the equality constraint $e^T x = k$, we can get

$$x_l = k - \sum_{j\neq l} x_j \tag{6}$$

Substituting (6) into $ax^T x - v^T x$, we have

$$\begin{aligned} & ax^T x - v^T x \\ = \; & a(\textstyle\sum_{j=1}^n x_j^2) - \sum_{j=1}^n v_j x_j \\ = \; & a(\textstyle\sum_{j\neq l} x_j^2 + (k - \sum_{j\neq l} x_j)^2) - \\ & \textstyle\sum_{j\neq l} v_j x_j - v_l(k - \sum_{j\neq l} x_j) \\ = \; & a(x_i^2 + \cdots + x_i^2 - 2kx_i + 2x_i \textstyle\sum_{j\neq l,i} x_j + \cdots) - \\ & v_i x_i + v_l x_l + \cdots \\ = \; & 2ax_i^2 + (2a\textstyle\sum_{j\neq l,i} x_j - 2ak + v_l - v_i)x_i + \cdots \end{aligned}$$

From the above derivation, we can take $ax^T x - v^T x$ as the function of variable x_i. It can be further written as

$$ax^T x - v^T x = 2ax_i^2 + (2a\sum_{j\neq l,i} x_j - 2ak + v_l - v_i)x_i + e(x), \tag{7}$$

where $e(x)$ has nothing to do with x_i and x_l.

If the following condition is satisfied, $ax^T x - v^T x$ can only reach its minimum at its boundary, that is $x_i = 0$ or $x_i = 1$.

$$\frac{v_i - v_l + 2a(k - \sum_{j\neq l,i} x_j)}{2a} \in (-\infty, 0] \cup [1, \infty).$$

Considering $k - \sum_{j\neq l,i} x_j = x_i + x_l$, the condition is equivalent to

$$\frac{v_i - v_l + 2a(x_i + x_l)}{4a} \in (-\infty, 0] \cup [1, \infty). \tag{8}$$

If $v_i \in \{k$ largest elements of inputs$\}$, then we can choose $v_l \notin \{k$ largest elements of inputs$\}$. To let $x_i = 1$ and $x_l = 0$, the following conditions should be satisfied.

$$\frac{v_i - v_l + 2a(x_i + x_l)}{4a} \geq 1,$$

and

$$\frac{v_l - v_i + 2a(x_l + x_i)}{4a} \leq 0.$$

We can get

$$x_i - x_l \geq 2a.$$

If $v_i \notin \{k$ largest elements of inputs$\}$, we can choose $v_l \in \{k$ largest elements of inputs$\}$. In the same way, we can get

$$x_l - x_i \geq 2a.$$

So, if $\tilde{v}_k - \tilde{v}_{k+1} \geq 2a$, the solution of problem (5) is in the set $\{0,1\}^n$.

The proof is complete. □

From Theorems 1 and 2, we can easily get that if $\tilde{v}_k - \tilde{v}_{k+1} \geq 2a$, the solution to the continuous quadratic optimization problem (5) is the solution of (1).

Remark 1: Let's consider the solution of problem (5) when $\tilde{v}_k - \tilde{v}_{k+1} < 2a$. In this case

$$\frac{\tilde{v}_k - \tilde{v}_{k+1} + 2a(\tilde{x}_k + \tilde{x}_{k+1})}{4a} \in [0,1].$$

Then when

$$\tilde{x}_k = \frac{\tilde{v}_k - \tilde{v}_{k+1} + 2a(\tilde{x}_k + \tilde{x}_{k+1})}{4a},$$

the cost function reaches its minimum. That is,

$$\tilde{x}_k - \tilde{x}_{k+1} = \frac{\tilde{v}_k - \tilde{v}_{k+1}}{2a}. \tag{9}$$

If other elements can be successfully separated, then

$$\tilde{x}_k + \tilde{x}_{k+1} = 1. \tag{10}$$

From (9) and (10), we can get

$$\tilde{x}_k = 0.5 + \frac{\tilde{v}_k - \tilde{v}_{k+1}}{4a}, \quad \tilde{x}_{k+1} = 0.5 - \frac{\tilde{v}_k - \tilde{v}_{k+1}}{4a}.$$

In particular, if $\tilde{v}_k = \tilde{v}_{k+1}$, $\tilde{x}_k = \tilde{x}_{k+1} = 0.5$.

III. KWTA Network Model

In [21][22], a neural network called dual neural network is presented to solve convex quadratic problems utilizing the dual variables. In this section, we will simplify the dual neural network for solving the quadratic programming problem (5). Here we call the network as KWTA network. It reduces the architecture complexity while preserving the desirable convergence property compared with the dual neural network.

Consider the problem (5) as the primal problem P, then its dual problem D can be written as

$$\begin{array}{ll} \text{maximize} & k^T y - ax^T x - e^T w_2 \\ \text{subjectto} & aIx - v - ey - Iw_1 + Iw_2 = 0 \\ & u \geqslant 0 \end{array} \tag{11}$$

where $y \in R, w_1 \in R^n, w_2 \in R^n$ are dual variables.

Define $u = w_1 - w_2$, the equality constraints in (11) becomes

$$aIx - v - ey - Iu = 0.$$

By the Karush-Kuhn-Tucker (KKT) conditions for convex optimization [23], the following set of equations have the same solution as problem P

$$aIx - v - ey - Iu = 0 \tag{12}$$

$$e^T x = k \tag{13}$$

$$\begin{cases} x_i = 0 & \text{if } u_i > 0 \\ x_i = 1 & \text{if } u_i < 0 \\ 0 \leqslant x_i \leqslant 1 & \text{if } u_i = 0 \end{cases} \tag{14}$$

(14) can be rewritten as

$$x = g(x - u), \tag{15}$$

where

$$g(v_i) = \begin{cases} \xi_i^- & \text{if } v_i < 0 \\ v_i & \text{if } 0 \leqslant v_i \leqslant 1 \\ \xi_i^+ & \text{if } v_i > 1. \end{cases}$$

From (12),

$$x = \frac{1}{a}(ey + Iu + v) \tag{16}$$

Substitute (16) into (13),

$$\frac{1}{a} e^T (ey + Iu + v) = k$$

y can be explicitly expressed by u.

$$y = \frac{1}{n}(ak - e^T u - e^T v) \tag{17}$$

Substitute (17) into (16),

$$x = \frac{1}{a}(I - \frac{1}{n} ee^T)u + \frac{1}{a}((-\frac{1}{n}\sum_{i=1}^{n} v_i + 1)e + v) \tag{18}$$

Based on (15, 16, 17), by projection theorem, the neural network that can solve the original problem can be designed as

$$\begin{array}{ll} \epsilon \frac{du}{dt} & = -Mu + g(Mu - u + s) + s \\ x & = Mu + s \end{array} \tag{19}$$

where $M = (I - ee^T/n)/a, s = (Mv + e)/a$.

The architecture of the KWTA network is shown in Fig 2. A circuit implementing this network consists of summers, integrators and operational amplifiers.

The properties of convergence and optimality are studied below.

Theorem 3: The KWTA network (19) is globally convergent to an equilibrium point u^* which depends on the initial state of the trajectory.

Proof: At u^*, we have the following inequality

$$(v - x^*)^T u^* \geq 0, \forall v \in \Omega. \tag{20}$$

which can be obtained by considering the following three cases:

- Case 1: If for some $i \in \{1, 2, \cdots, p\}$, $u_i^* = 0$, $0 \leq x_i^* \leq 1$, then $(v_i - x_i^*)u_i^* = 0$;
- Case 2: If for some $j \in \{1, 2, \cdots, p\}$, $u_j^* > 0$, $x_j^* = 0$ and $0 \leq v_j \leq 1$, then $v_j - x_j^* \geq 0$ and thus $(v_j - x_j^*)u_j^* \geq 0$;
- Case 3: If for some $k \in \{1, 2, \cdots, p\}$, $u_k^* < 0$, $x_k^* = 1$ and $0 \leq v_k \leq 1$, then $v_k - x_k^* \leq 0$ and thus $(v_k - x_k^*)u_k^* \geq 0$.

Therefore, it follows from (20) that

$$[g(Mu + s - u) - (Mu^* + s)]^T u^* \geq 0. \tag{21}$$

On the other hand, from projection theorem [24], it follows that, $\forall u \in R^n$

$$\begin{array}{l} [g(Mu + s - u) - (Mu^* + s)]^T \\ [(Mu + s - u) - g(Mu + s - u)] \geq 0. \end{array} \tag{22}$$

Combining (21) and (22), we have

$$\begin{array}{l} [g(Mu + s - u) - (Mu^* + s)]^T \\ [u^* + (Mu + s - u) - g(Mu + s - u)] \geq 0. \end{array} \tag{23}$$

Define $\tilde{g} := g(Mu + s - u) - (Mu + s)$, (23) becomes

$$[\tilde{g} + M(u - u^*)]^T [(u - u^*) + \tilde{g}] \leq 0. \tag{24}$$

From (24), we can get

$$\begin{array}{l} (u - u^*)^T \tilde{g} + \tilde{g}^T M(u - u^*) \leq \\ -\|\tilde{g}\|^2 - (u - u^*)^T M(u - u^*). \end{array} \tag{25}$$

Because the eigenvalues of M is either 0 or 1, M is positive semidefinite, i.e.,

$$(u - u^*)^T M(u - u^*) \geq 0. \tag{26}$$

From (25) and (26), we get

$$(u - u^*)^T \tilde{g} + \tilde{g}^T M(u - u^*) \leq 0, \tag{27}$$

and if and only if $u = u^*$, the equality holds.

Now choose the following radically unbounded Lyapunov functional candidate

$$V(u(t)) = \frac{1}{2}\|Q(u(t) - u^*)\|_2^2, \tag{28}$$

where Q is a symmetric and positive definite matrix with $Q^2 = (I + M)$.

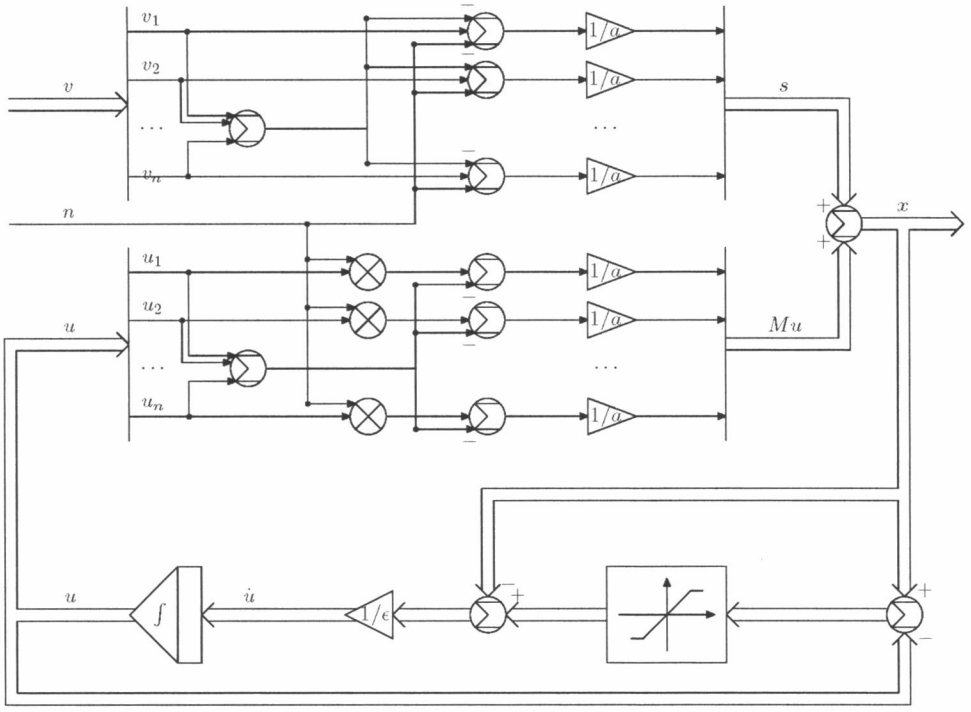

Fig. 2. Architecture of the KWTA network.

Then from (27), we get

$$\begin{aligned}\frac{dV}{dt} &= (u-u^*)^T Q^2 \dot{u} \\ &= (u-u^*)^T (I+M)\tilde{g} \\ &= (u-u^*)^T \tilde{g} + \tilde{g}^T M (u-u^*) \\ &\leq 0.\end{aligned} \tag{29}$$

By the Lyapunov stability theorem, the simplified dual neural network is globally stable. □

Remark 2: The convergence speed of the neural network is determined by the eigenvalues of M which are independent of n and inversely proportional to a.

Theorem 4: $x^* = Mu^* + s$ is an optimal solution to the quadratic programming problem (5), where u^* is an equilibrium point of the dynamic equation (19).

This theorem can be verified by substituting x^* into the KKT conditions (12,13,14). The three equations are satisfied, which means that x^* is the optimal solution to the quadratic programming problem (5).

From the Theorems 3 and 4, we can conclude that the KWTA network is globally convergent to the exact solution of the problem (5). Further, from Theorems 1 and 2, the KWTA network is globally convergent to the exact solution of the KWTA operation. As a comparison, the KWTA circuit in [9] will oscillate under some conditions.

IV. Simulation Results

In this simulation, the inputs are $v_i = i$ $(i = 1, 2, \cdots, 5)$ and $k = 2$. When $\epsilon = 10^{-8}$ and $a = 0.5$, the transient behaviors of u and v are shown in Figs. 3 and 4. In Fig. 4, the curves from bottom to top correspond respectively v_1, v_2, $\cdots$, v_5. It can be seen that the outputs are [0 0 1 1 1]. The 2 largest elements are successfully selected within 1.0×10^{-7} second.

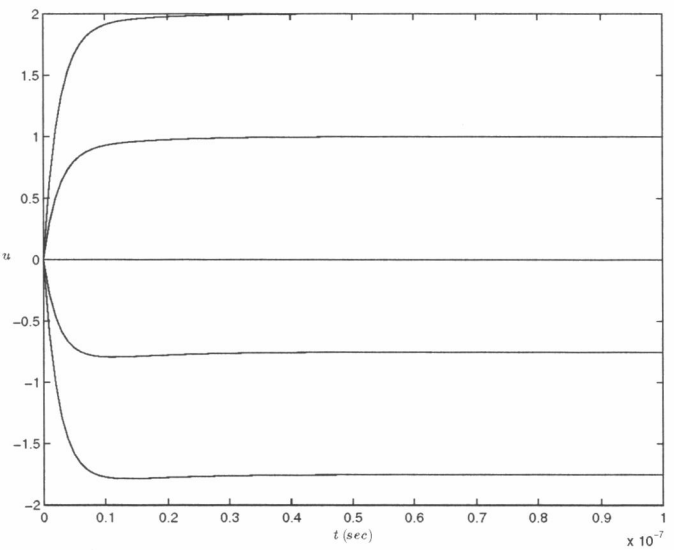

Fig. 3. The transient behavior of u.

Next, consider a set of 5 sinusoidal input signals with the following instantaneous values $v_p(t) = 10\sin[2\pi(1000t - 0.25\pi(p-1)](p = 1, 2, 3, 4)$ and $k = 2$. Fig. 5 illustrates the 5 input signals and the transient outputs of the KWTA network with $\epsilon = 10^{-6}$ and $a = 10^{-3}$. The simulation results show that the KWTA network can effectively determine the two largest signals from the time-varying signals in real time.

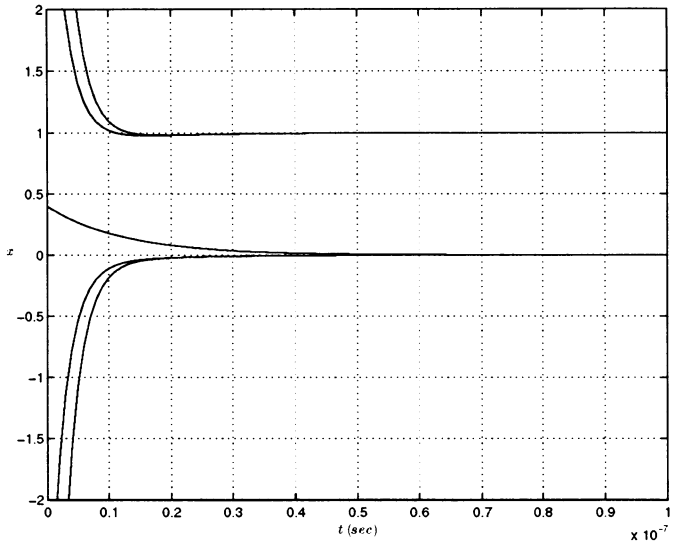

Fig. 4. The transient behavior of x.

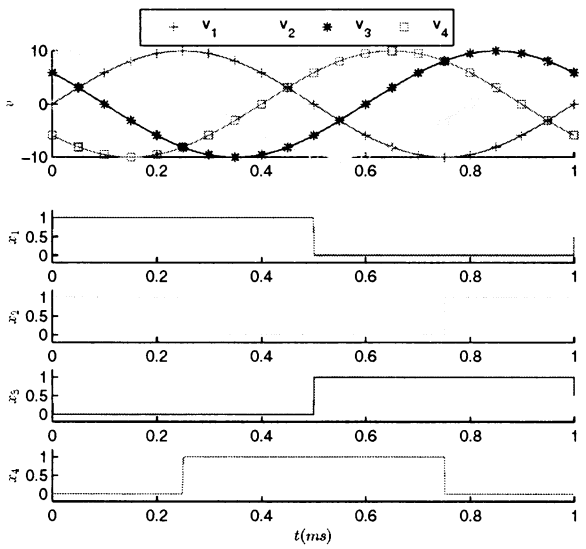

Fig. 5. Inputs and outputs of the KWTA network.

V. Conclusions

A KWTA network is developed for K-winners-take-all operation based on the equivalent quadratic programming formulation. The KWTA network is shown to be stable and can implement the KWTA operation in real time. Compared with the dual neural network, KWTA network has lower architecture complexity. The KWTA network is shown to be effective by analysis and simulation.

References

[1] A. Krogh J. Hertz and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, CA, 1991.

[2] D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science*, vol. 195, pp. 283–328, 1977.

[3] A. L. Yuille and D. Geiger, *The Handbook of Brain Theory and Neural Networks*, chapter Winner-Take-All Networks, pp. 1228–1231, The MIT Press, 2 edition, 2002.

[4] W. Maass, "Neural computation with winner-take-all as the only nonlinear operation," *Advances in Neural Information Processing Systems*, vol. 12, 2000.

[5] W. Maass, "On the computational power of winner-take-all," *Neural Computation*, vol. 12, pp. 2519–2535, 2000.

[6] W. J. Wolfe, D. Mathis, and et al, "K-winner networks," *IEEE Transactions on Neural Networks*, vol. 2, pp. 310–315, March 1991.

[7] J. Wang, "Analogue winner-take-all neural networks for determining maximum and minimum signals," *Int. J. Electronics*, vol. 77, no. 3, pp. 355–367, 1994.

[8] K. Urahama and T. Nagao, "K-winners-take-all circuit with O(N) complexity," *IEEE Transactions on Neural Networks*, vol. 6, pp. 776–778, May 1995.

[9] B. Sekerkiran and U. Cilingiroglu, "A CMOS k-winners-take-all circuit with O(N) complexity," *IEEE Transactions on Circuits and Systems*, vol. 46, no. 1, January 1999.

[10] Jayadeva and S. A. Rahman, "A neural network with O(N) neurons for ranking N numbers in O(1/N) times," *IEEE Transactions on Circuits and Systems I: in press*, 2004.

[11] J. Yen, J. Guo, and H. Chen, "A new k-winners-take-all neural network and its array architecture," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 901–912, Sept. 1998.

[12] B.D. Calvert and C.A. Marinov, "Another k-winners-take-all analog neural network," *IEEE Transactions on Neural Networks*, vol. 11, no. 4, pp. 829–838, July 2000.

[13] C.A. Marinov and B.D. Calvert, "Performance analysis for a k-winners-take-all analog neural network: basic theory," *IEEE Transactions on Neural Networks*, vol. 14, no. 4, pp. 766–780, July 2003.

[14] D. W. Tank and J. J. Hopfield, "Simple neural optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Transactions on Circuits and Systems*, vol. 33, no. 5, pp. 533–541, 1986.

[15] J. J. Hopfield and D. W. Tank, "Computing with neural circuits: a model," *Science*, vol. 233, pp. 625–633, 1986.

[16] M. P. Kennedy and L. O. Chua, "Neural networks for nonlinear programming," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 5, pp. 554–562, May 1988.

[17] S. Zhang and A. G. Constantinides, "Lagrange programming neural networks," *IEEE Transactions on Circuits and Systems II*, vol. 39, no. 7, pp. 441–452, July 1992.

[18] J. Wang, "A deterministic annealing neural network for convex programming," *Neural Networks*, vol. 5, no. 4, pp. 962–971, 1994.

[19] Y. Xia, "A new neural network for solving linear and quadratic programming problems," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, pp. 1544–1547, November 1996.

[20] Y. Xia, G. Feng, and J. Wang, "A primal-dual neural network for online resolving constrained kinematic redundancy in robot motion control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 35, no. 1, pp. 54–64, Feb. 2005.

[21] Y. Xia and J. Wang, "A dual neural network for kinematic control of redundant robot manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 31, no. 1, pp. 147–154, February 2001.

[22] Y. Zhang and J. Wang, "A dual neural network for convex quadratic programming subject to linear equality and inequality constraints," *Physics Letters A*, pp. 271–278, June 2002.

[23] S. Boyd and L. Vandenbeghe, *Convex Optimization*, Cambridge Uniersity Press, Cambridge, UK, 2004.

[24] D. Kinderlehrer and G. Stampacchia, *An Introduction to Variational Inequalities and Their Applications*, Academic Press, New York, 1980.