Brief Announcement: Simple and Fast Self-Stabilizing Dynamics for k-Winner-Take-All Computation

Vincenzo Bonifaci
Roma Tre University

Fabio Galvan
Roma Tre University

Abstract

The k-Winner Take All problem consists in identifying the k largest inputs from a collection of input values. We exhibit a simple dynamic network composed of $2n$ leaky memristor units with a continuous-time dynamics that self-stabilizes and, for any $\epsilon > 0$, falls within relative distance ϵ of the k-Winner-Take-All solution on n inputs in time $O(\epsilon^{-1} \log n)$.

2012 ACM Subject Classification Computing methodologies → Distributed algorithms; Computing methodologies → Neural networks; Theory of computation → Bio-inspired optimization

Keywords and phrases k-Winner-Take-All, network dynamics, nature-inspired distributed algorithms, electrical flows, self-stabilization, distributed optimization, neural leader election, analog computing

Digital Object Identifier 10.4230/LIPIcs.DISC.2026.63

Category Brief Announcement

Acknowledgements This research was partly supported by project ECS 0000024 of the European Commission, *Rome Technopole*, PNRR grant M4-C2-Inv. 1.5. In particular, FG's work was funded by Rome Technopole. Work of VB was carried out in association with Istituto di Analisi dei Sistemi ed Informatica, Consiglio Nazionale delle Ricerche, Italy.

1 Introduction

A paradigmatic example of a computational task that is based on competitive network dynamics is the *Winner-Take-All (WTA)* problem, together with its direct generalization, the *k-Winner-Take-All (k-WTA)* problem. In these problems, the goal is to select the maximum input, or the k largest inputs respectively, from a collection of input values. On the one hand, these modules are computationally powerful enough that together with linear units they can approximate arbitrary continuous functions or compute arbitrary boolean functions [23, 24]; at the same time, they are considered biologically plausible and are conjectured to be a fundamental primitive of certain cognitive functions in the brain [36, 7, 28, 13].

Although winner-take-all problems are trivial in a centralized setting, the solution of WTA or k-WTA in a decentralized setting or via neural networks has been the subject of much research. Previous works have studied networks and circuits that implement WTA and k-WTA using a wide range of primitives, dynamics, and network topologies [25, 15, 11, 34, 20, 37, 22, 32].

In this work, we propose and study a network dynamics for computing k-WTA that aims to strike a middle ground between hardware-oriented k-WTA implementations based on strong primitives (such as step-like activation functions [25, 34]) and k-WTA implementations that are more plausible from a biological perspective, but also much more complicated to model and analyze, such as those based on spiking neuron models [22, 32]. We propose a k-WTA dynamics that is based on *leaky memristor* units [6]: computational elements that operate

© Vincenzo Bonifaci and Fabio Galvan;
licensed under Creative Commons License CC-BY 4.0

like resistors with a short-term memory effect, much like non-spiking neurons. Specifically, we exhibit a leaky memristor network with $2n$ units that dynamically converges to a k-WTA solution for any input of size n in time $O(\log n)$. Such leaky memristor units, as well as some of our techniques, are inspired by another biologically motivated computation model, that of *Physarum computing*, which is based on the observed computational capabilities of the acellular slime mold *Physarum polycephalum* [33, 16, 17, 4, 30].

1.1 Main Contributions

Our contribution is a dynamic network model for the k-WTA problem based on leaky memristor units, and its analysis. We prove two main properties of this model, which we state here informally:

1. *Correctness and self-stabilization.* Initialized from any positive initial state, the network dynamics converges asymptotically towards the correct k-WTA solution (Theorem 5).
2. *Convergence time.* Initialized from the symmetric state, the dynamics falls within relative distance ϵ of the correct k-WTA solution in time $O(\epsilon^{-1} \log n)$, for any $\epsilon > 0$ (Theorem 6).

Conceptually, one way to summarize our results is that *k-WTA can be solved effectively by a self-stabilizing network without recourse to harsh nonlinearities* (such as step functions, thresholds, or spikes). In other words, unlike previously studied models for k-WTA, our model does not rely on high-gain activation functions, thresholds, or spikes: leaky integrators, coupled through a network mechanism, are enough to construct an effective k-WTA network.

1.2 Related Work

Significance of k-WTA. As mentioned in the introduction, k-WTA modules are important because they can be used as basic but relatively powerful building blocks for arbitrary boolean or continuous computations. Maass [23, 24] proved that in combination with neurons that compute weighted sums, k-WTA units have universal computational power for both digital and analog computation. For example, any boolean function can be computed by a single k-WTA unit applied to positively weighted sums of input variables [24, Theorem 2]. A 1-WTA module, in particular, can be viewed as a neural leader election unit [22]. For a general overview of k-WTA mechanisms, see Yuille and Geiger [35].

k-WTA networks based on step-like or high-gain activation functions. Much interdisciplinary work has been devoted to implementations of WTA and k-WTA [25, 19, 11, 34, 20, 37], starting with the seminal work of Majani et al. [25], which showed that k-WTA can be implemented in a continuous Hopfield neural network model – the so-called additive model [14, Chap. 13]. This line of work focused on network models that are based on step-like or high-gain activation functions and are often designed for implementation in electronic hardware; see [11, 34] and references therein. In particular, Wang [34] constructs an effective k-WTA neural network with a single state variable, based on a hard-limiting Heaviside step activation function. Li and coauthors [20, 37] propose a distributed protocol for k-WTA on an arbitrary connected graph, and prove its global asymptotic convergence under the assumption that the algebraic connectivity of the graph is sufficiently large.

k-WTA in spiking neural networks. A more recent line of work considers WTA and k-WTA implementations based on spiking neuron models [22, 32]. In this case, the focus is on biological, rather than artificial, neural network models. Lynch et al. [22] discuss the importance of self-stabilization for WTA networks and study how the number of inhibitory neurons affects the network convergence time for 1-WTA, presenting upper and (near matching) lower bounds on the number of required synchronous communication rounds.

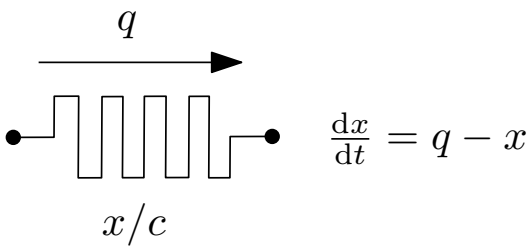


Figure 1 Circuit symbol for a leaky memristor of conductance x/c (equivalently, resistance c/x)

Su et al. [32] study a stochastic k-WTA model wherein n randomly generated input spike trains compete with each other based on their underlying firing rates, and characterize the minimum waiting time required for a given success probability to be reached.

Memristors and leaky integrators. *Memristors* are circuit components that operate like resistors with a short-term memory effect [6]. The general memristor equation [6, Eq. (4)] (see also [5, Eq. (1a-1b)], [29, Eq. (2)]) takes the form: $\frac{\mathrm{d}x}{\mathrm{d}t} = f(x,q)$, $p = R(x)q$, where x is a state variable, the dynamic variables p and q denote the electrical potential and current across the memristor, and f and R are continuous functions; the latter equation is just Ohm's law, while the former dictates the state dynamics. A special case, which we call the *leaky memristor*, is as follows:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = q - x \tag{1}$$

$$p = \frac{c}{x}q, \tag{2}$$

where c is a constant (Figure 1). Some authors have discussed memristor units as abstractions of biological neurons [5, 29]. Indeed, *leaky integrator neurons*, as defined in computational neuroscience [13, Eq. (1.5)], follow an equation that is mathematically compatible with Eq. (1):

$$\tau_m \frac{\mathrm{d}u}{\mathrm{d}t} = -(u - u_{\mathrm{rest}}) + RI, \tag{3}$$

where the dynamic variables are u, the membrane potential, and I, the current, while the constants are R, the membrane resistance, u_{rest}, the rest potential, and τ_m, the membrane time constant of the neuron. Note that setting $\tau_m = 1$, $R = 1$, $u_{\mathrm{rest}} = 0$, Eq. (3) boils down to Eq. (1) if we identify x with u and q with I.

Physarum dynamics. Networks of leaky memristors described by Eqs. (1)–(2) have been studied in the bio-inspired optimization literature, in the context of the so-called *Physarum network dynamics*. *Physarum polycephalum* is a slime mold known for its computational capabilities, which has been studied both in wet lab experiments [26, 33] and as a computational model; for the latter, several theoretical guarantees have been proved [16, 17, 4, 30]. In particular, it has been shown that a network dynamics based on a system of coupled differential equations of the form (1)–(2) converges to an optimal solution of any shortest path problem defined on the network and, more abstractly, that such a dynamics converges to an optimal solution of any positive linear optimization problem [16, 17]. The convergence time of these Physarum dynamics has also been bounded [31, 1, 3].

Self-stabilizing distributed algorithms. A *self-stabilizing* distributed system is a system that can automatically recover after the occurrence of transient faults [8]. In terms of its allowed dynamics, this means that the system should be designed to converge to a desirable

state or behavior regardless of its initial condition [21, 9]. Self-stabilizing algorithms have been designed, in particular, for the leader election problem [10, 12]. For the importance of self-stabilization in the context of WTA problems, see Lynch et al. [22].

Distributed k-selection and Top-k. A relative of k-WTA is the *distributed k-selection* problem, where, given a connected graph of diameter D consisting of n nodes in which each node holds a numeric element, the goal is to determine the k-th smallest of these elements. Kuhn, Locher, and Wattenhofer [18] discuss a randomized algorithm solving the problem in time $O(D \log_D n)$ with high probability, and prove a matching lower bound for any synchronous randomized algorithm when each message can contain solely a constant number of both node IDs and elements, and also at most $O(\log n)$ arbitrary additional bits. A related problem is the distributed top-k problem, see for instance [27] for a possible formulation.

2 Our Contributions

2.1 Problem and Notation

▶ **Definition 1.** *For an integer $k > 0$, the* k-Winner-Take-All (k-WTA) *problem is as follows.*
Given: *$a \in [-1, 1]^n$, for some positive integer n ($n > k$).*
Find: *$x^* \in \{0, 1\}^n$ such that $x^*_i = 1$ if and only if a_i is one of the k largest entries of a.*

▶ **Definition 2.** *Given an input vector $a \in [-1, 1]^n$, its* k-WTA set *is*

$$\mathrm{WTA}_k(a) \coloneqq \{i \in [n] : a_i \text{ is one of the } k \text{ largest entries of } a\}, \text{where} [n] \coloneqq \{1, 2, \ldots, n\}.$$

In this terminology, the k-WTA problem is the problem of computing the characteristic vector x^* of the k-WTA set, given an input vector a. We remark that for the k-WTA set to be well-defined, it is not essential that all the inputs be distinct: the existence of a positive gap between the k-th largest input and the $(k + 1)$-th largest input is sufficient.

▶ **Definition 3.** *Given an input vector $a \in [-1, 1]^n$, let $a^{\downarrow}_i$, for $i \in [n]$, denote the i-th largest of its coordinates, so that $a^{\downarrow}_1 \geq a^{\downarrow}_2 \geq \ldots \geq a^{\downarrow}_n$. Then the* k-WTA gap *of a is defined as $\gamma \coloneqq a^{\downarrow}_k - a^{\downarrow}_{k+1}$.*

In the following, we tacitly assume a nonzero k-WTA gap. This gap parameter arises often in the analysis of k-WTA networks: for example, Wang [34] studies a network where the gain of the activation function is assumed to be $\Omega(\gamma^{-1})$ [34, Eq. (28)].

Our convergence result is in terms of the (relative) L_1-distance between the characteristic vector x^* encoding the optimal k-WTA solution and a candidate vector x, defined as follows.

▶ **Definition 4.** *The* L_1-distance *between two vectors $x, x^* \in \mathbb{R}^n$ is $L_1(x, x^*) \coloneqq \sum_{i \in [n]} |x_i - x^*_i|$. The* relative L_1-distance *between x and x^* is $\ell_1(x, x^*) \coloneqq L_1(x, x^*) / \sum_{i \in [n]} |x^*_i|$.*

2.2 The Dynamics and its Properties

Our dynamic network to solve the k-WTA problem is schematically depicted in Figure 2. For inputs of size n, the network consists of $n + 2$ nodes and $2n$ edges; the nodes are sources or sinks of current flow, while each edge corresponds to a leaky memristor. The signed incidence matrix of the network is the $(n + 2) \times 2n$ matrix

$$B = \begin{pmatrix} I & I \\ -\mathbf{1}^\top & 0^\top \\ 0^\top & -\mathbf{1}^\top \end{pmatrix},$$

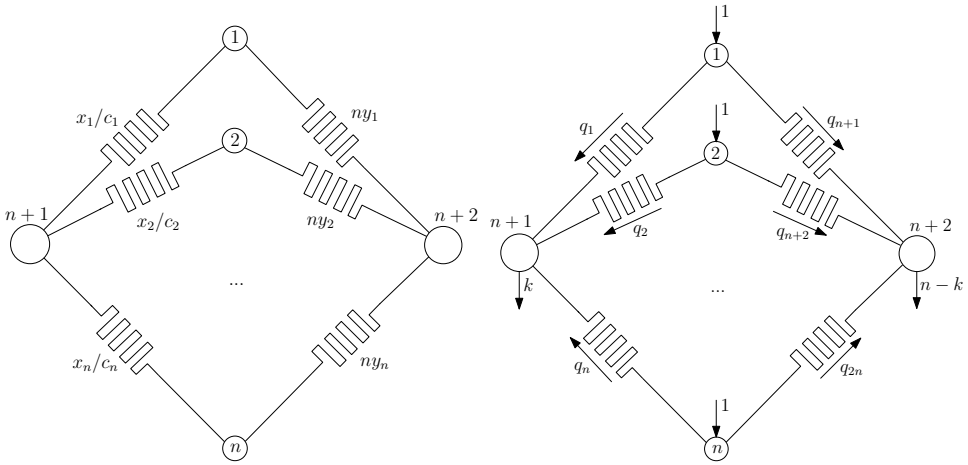


Figure 2 A dynamic network for k-WTA: conductances (left) and current flows (right)

where rows correspond to nodes and columns correspond to edges, namely $B_{ij} = 1$ (respectively, -1) if the j-th edge leaves (resp., enters) the i-th node, and $B_{ij} = 0$ otherwise. Nodes $n+1$ and $n+2$ have a prescribed current in-flow equal to k and $n-k$ respectively, while all other nodes have a prescribed current out-flow equal to 1. Thus, the vector of currents $q = (q_1, \ldots, q_{2n})$ obeys Kirchhoff's current rule [2] $Bq = b'$, where $b'_i = 1$ for $i = 1, \ldots, n$, $b'_{n+1} = -k$, $b'_{n+2} = -(n-k)$. The $(n+2)$-th equation is linearly dependent on the others and can be discarded, resulting in the equivalent system $Aq = b$, where

$$A = \begin{pmatrix} I & I \\ -\mathbf{1}^\top & 0^\top \end{pmatrix}, \qquad b = (1, \cdots, 1, -k)^\top$$

and A has full row rank. Since the network elements are leaky memristors, according to Eqs. (1)–(2) the conductances in the network are dynamic and proportional to a state vector $z = (x, y) \in \mathbb{R}^n \times \mathbb{R}^n$. Namely, the conductance of the j-th edge is defined as x_j/c_j for $j \le n$ and $y_{j-n}/(1/n)$ for $j > n$ (right side of Figure 2), where $c_j \coloneqq 2 - a_j$ for each $j \in [n]$.[1] Let $G = G(z) \in \mathbb{R}^{2n \times 2n}$ be the diagonal matrix with the conductances $x_1/c_1, \ldots, x_n/c_n, ny_1, \ldots, ny_n$ on its diagonal. By Ohm's law, $q = GA^\top p$ where $p \in \mathbb{R}^{n+1}$ is the vector of electrical potentials at nodes $1, \ldots, n+1$.[2] Therefore, the electrical potentials satisfy $AGA^\top p = b$; note that the matrix $AGA^\top$ is symmetric and positive definite (since A has full rank), and one can recover $p = (AGA^\top)^{-1}b$. Finally, the memristor states follow a leaky integration dynamics as in Eq. (1):

$$\dot{z}_j(t) = q_j(t) - z_j(t), \qquad \text{for } j = 1, 2, \ldots, 2n, \tag{4}$$

in the open domain $\Omega = \mathbb{R}^{2n}_{>0}$, where we used the shorthand $\dot{z}_j = (\mathrm{d}/\mathrm{d}t)z_j(t)$. Any initial condition $z(0) \in \Omega$ is allowed. Substituting Ohm's law in (4), the dynamics can then be summarized in vector notation as

$$\dot{z} = GA^\top(AGA^\top)^{-1}b - z. \tag{5}$$

[1] We use the definition $c_j \coloneqq 2 - a_j$ for concreteness, but other monotone decreasing functions of the a_j can also be used here, as long as the resulting c_j are positive and fall in a bounded range.

[2] The potential at node $n+2$ is implicitly zero due to the removal of the $(n+2)$-th equation.

We show the correctness of the dynamics – that is, that the state variables of our dynamic network converge to the correct solution of the k-WTA problem. Moreover, the dynamics is *self-stabilizing* in the sense that the initial condition is irrelevant.

▶ **Theorem 5.** *The network dynamics* (5) *solves the k-WTA problem in the sense that, as $t \to \infty$, $x_i(t) \to 1$ if $i \in \mathrm{WTA}_k(a)$ and $x_i(t) \to 0$ if $i \notin \mathrm{WTA}_k(a)$, for any $i \in [n]$, and for any initial condition $z(0) > 0$.*

We also analyze the convergence time in terms of the L_1-distance between $x(t)$ and the optimal k-WTA solution x^* when the dynamics is initialized from the symmetric (and non-informative) configuration $x^{(0)} = (k/n)\mathbf{1}$.

▶ **Theorem 6.** *For any $\epsilon' \in (0,1)$, for all $t \geq t_0$ with $t_0 = O({\epsilon'}^{-1}\gamma^{-1}\log n)$, $\ell_1(x(t), x^*) \leq \epsilon'$.*

References

1. R. Becker, V. Bonifaci, A. Karrenbauer, P. Kolev, and K. Mehlhorn. Two results on slime mold computations. *Theoretical Comput. Sci.*, 773:79–106, 2019.
2. B. Bollobás. *Modern Graph Theory*. Springer, 2002.
3. V. Bonifaci. On the convergence time of a natural dynamics for linear programming. *Algorithmica*, 82(2):300–315, 2019.
4. V. Bonifaci, K. Mehlhorn, and G. Varma. Physarum can compute shortest paths. In Y. Rabani, editor, *Proc. of the 23rd ACM-SIAM Symposium on Discrete Algorithms*, pages 233–240, Philadelphia, PA, 2012. SIAM.
5. L. Chua. Memristor, Hodgkin-Huxley, and edge of chaos. In L. Chua, G. C. Sirakoulis, and A. Adamatzky, editors, *Handbook of Memristor Networks*, pages 287–314. Springer, 2019.
6. L. Chua and S. M. Kang. Memristive devices and systems. *Proceedings of the IEEE*, 64(2):209–223, Feb. 1976.
7. R. L. Coultrip, R. H. Granger, and G. Lynch. A cortical model of winner-take-all competition via lateral inhibition. *Neural Networks*, 5(1):47–54, 1992.
8. E. W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, 1974.
9. S. Dolev. *Self-Stabilization*. MIT Press, 2000.
10. S. Dolev, A. Israeli, and S. Moran. Uniform dynamic self-stabilizing leader election. *IEEE Trans. Parallel Distributed Syst.*, 8(4):424–440, 1997.
11. L. Ferreira, E. Kaszkurewicz, and A. Bhaya. Synthesis of a k-winners-take-all neural network using linear programming with bounded variables. In *Proceedings of the International Joint Conference on Neural Networks*, volume 3, pages 2360–2365, 2003.
12. M. J. Fischer and H. Jiang. Self-stabilizing leader election in networks of finite-state anonymous agents. In A. A. Shvartsman, editor, *Principles of Distributed Systems, 10th International Conference, OPODIS 2006, Bordeaux, France, December 12-15, 2006, Proceedings*, Lecture Notes in Computer Science, pages 395–409. Springer, 2006.
13. W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski. *Neuronal Dynamics*. Cambridge University Press, 2014.
14. S. Haykin. *Neural Networks and Learning Machines*. Pearson, 3rd edition, 2009.
15. J. A. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation*. CRC Press, 1991.
16. K. Ito, A. Johansson, T. Nakagaki, and A. Tero. Convergence properties for the Physarum solver. arXiv 1101.5249, 2011.
17. A. Johannson and J. Y. Zou. A slime mold solver for linear programming problems. In S. B. Cooper, A. Dawar, and B. Löwe, editors, *How the World Computes - Turing Centenary Conference and 8th Conference on Computability in Europe, CiE 2012, Cambridge, UK, June 18-23, 2012. Proceedings*, volume 7318 of *Lecture Notes in Computer Science*, pages 344–354. Springer, 2012.

18 F. Kuhn, T. Locher, and R. Wattenhofer. Distributed selection: a missing piece of data aggregation. *Commun. ACM*, 51(9):93–99, 2008.

19 J. Lazzaro, S. Ryckebusch, M. Mahowald, and C. Mead. Winner-take-all networks of $O(n)$ complexity. In *Advances in Neural Information Processing Systems*, pages 703–711, 1989.

20 S. Li, M. Zhou, X. Luo, and Z. You. Distributed winner-take-all in dynamic networks. *IEEE Trans. Autom. Control.*, 62(2):577–589, 2017.

21 N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

22 N. A. Lynch, C. Musco, and M. Parter. Computational tradeoffs in biological neural networks: Self-stabilizing winner-take-all networks. In C. H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 15:1–15:44. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

23 W. Maass. Neural computation with winner-take-all as the only nonlinear operation. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 293–299. The MIT Press, 1999.

24 W. Maass. On the computational power of winner-take-all. *Neural Comput.*, 12(11):2519–2535, 2000.

25 E. Majani, R. Erlanson, and Y. S. Abu-Mostafa. On the K-Winners-Take-All Network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1, Denver, Colorado, USA, 1988*, pages 634–642. Morgan Kaufmann, 1988.

26 T. Nakagaki, H. Yamada, and Á. Tóth. Maze-solving by an amoeboid organism. *Nature*, 407:470, 2000.

27 B. Patt-Shamir and A. Shafrir. Approximate distributed top- k queries. *Distributed Comput.*, 21(1):1–22, 2008.

28 M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–1025, 1999.

29 M. P. Sah, H. Kim, and L. Chua. Brains are made of memristors. In L. Chua, G. C. Sirakoulis, and A. Adamatzky, editors, *Handbook of Memristor Networks*, pages 315–350. Springer, 2019.

30 D. Straszak and N. K. Vishnoi. Natural algorithms for flow problems. In R. Krauthgamer, editor, *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms*, pages 1868–1883. SIAM, 2016.

31 D. Straszak and N. K. Vishnoi. On a natural dynamics for linear programming. In M. Sudan, editor, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, page 291. ACM, 2016.

32 L. Su, C. Chang, and N. A. Lynch. Spike-based winner-take-all computation: Fundamental limits and order-optimal circuits. *Neural Comput.*, 31(12):2523–2561, 2019.

33 A. Tero, S. Takagi, T. Saigusa, K. Ito, D. P. Bebber, M. D. Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki. Rules for biologically inspired adaptive network design. *Science*, 327:439–442, 2010.

34 J. Wang. Analysis and design of a k-winners-take-all model with a single state variable and the heaviside step activation function. *IEEE Trans. Neural Networks*, 21(9):1496–1506, 2010.

35 A. L. Yuille and D. Geiger. Winner-take-all networks. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 1228–1231. MIT Press, 2nd edition, 2003.

36 A. L. Yuille and N. M. Grzywacz. A winner-take-all mechanism based on presynaptic inhibition feedback. *Neural Comput.*, 1(3):334–347, 1989.

37 Y. Zhang, S. Li, B. Xu, and Y. Yang. Analysis and design of a distributed k-winners-take-all model. *Automatica*, 115:108868, 2020.