RUTCOR
RESEARCH
REPORT

On the Power of Democratic Networks

Eddy Mayoraz[a]

RRR 4-95, February 1995

RUTCOR • Rutgers Center for Operations Research • Rutgers University • P.O. Box 5062 • New Brunswick New Jersey • 08903-5062
Telephone: 908-932-3804
Telefax: 908-932-5472
Email: rrr@rutcor.rutgers.edu

[a]RUTCOR—Rutgers University's Center for Operations Research, P.O. Box 5062, New Brunswick, NJ 08903-5062, mayoraz@rutcor.rutgers.edu .

RUTCOR RESEARCH REPORT
RRR 4-95, FEBRUARY 1995

ON THE POWER OF DEMOCRATIC NETWORKS

Eddy Mayoraz

Abstract. Linear Threshold Boolean units (LTUs) are the basic processing components of artificial neural networks of Boolean activations. Quantization of their parameters is a central question in hardware implementation, when numerical technologies are used to store the configuration of the circuit. In the previous studies on the circuit complexity of feedforward neural networks, no differences had been made between a network with "small" integer weights and one composed of majority units (LTUs with weights in $\{-1, 0, +1\}$), since any connection of weight w (w integer) can be simulated by $|w|$ connections of value $\mathrm{sgn}(w)$. This paper will focus on the circuit complexity of democratic networks, *i.e.* circuits of majority units with at most one connection between each pair of units.

The main results presented are the following: any Boolean function can be computed by a depth-3 non-degenerate democratic network and can be expressed as a linear threshold function of majorities; *AT-LEAST-k* and *AT-MOST-k* are computable by a depth-2, polynomial size democratic network; the smallest sizes of depth-2 circuits computing *PARITY* are identical for a democratic network and for a usual network; the *VC*-dimension of the class of the majority functions is $n + 1$, *i.e.* equal to that of the class of any linear threshold functions.

Acknowledgements: The author gratefully acknowledge the Swiss National Science Foundation, grant 20-5637.88.

1 Introduction

A Boolean function $f(\boldsymbol{b}) : \mathbb{B}^n \to \mathbb{B}$ is a *linear threshold function* if there exists a weight vector $\boldsymbol{w} \in \mathbb{R}^n$ and a threshold $w_0 \in \mathbb{R}$ such that

$$f(\boldsymbol{b}) = \operatorname{sgn}(w_0 + \boldsymbol{b}^\top \boldsymbol{w}). \tag{1}$$

The numerical representation used in this paper for the set of Boolean values $\mathbb{B}$ will be $\{-1, +1\}$, and the sign function $\operatorname{sgn} : \mathbb{R} \to \mathbb{B}$ is defined as $\operatorname{sgn}(x) = +1$ if and only if $x > 0$. The processing unit computing a linear threshold Boolean function (LTU) is the basic component of artificial neural networks. A feedforward (*i.e.* cycle free) network $\mathcal{N}$ is characterized by its *depth* $d(\mathcal{N})$, which denotes the length of the longest oriented path in $\mathcal{N}$; and by its *size* $s(\mathcal{N})$, which will be defined, in the present study, as the number of processing units in $\mathcal{N}$. According to the notation used in [2], LT_1 denotes the set of all linear threshold Boolean functions, while $\mathcal{LT}_d$ (resp. LT_d) represents the set of all Boolean functions that can be computed by a feedforward network composed of LTUs, with a depth d and of any size (resp. with a size bounded by a polynomial in the number of inputs of $\mathcal{N}$).

Since LT_1 contains the conjunction and the disjunction of arbitrarily many arguments, the set $\mathcal{B}^n$ of every Boolean function of n arguments is clearly included in $\mathcal{LT}_2$. However, when circuits with size bounded polynomially in n are considered, many questions remain. On the one hand, since the number of linear threshold Boolean functions of at most n arguments is in $2^{\Theta(n^2)}$ (see [11, 21]), $\mathcal{B}^n \not\subset LT_d$ for any constant depth d. On the other hand, $LT_1 \subsetneq LT_2$ is the only inclusion known to be proper in the whole hierarchy $LT_1 \subset LT_2 \subset LT_3 \subset \ldots$.

The quantization of the parameters w_i $(i = 0, \ldots, n)$ of the LTUs is essential for any hardware implementation using numerical technologies to store the w_is. A famous result due to Muroga, Toda and Takasu [12] (see also [14] for a concise proof) shows that the weights of any linear threshold function of n inputs can be integers bounded from above by

$$\frac{(n+1)^{\frac{n+1}{2}}}{2^n}.$$

It is easy to see that some Boolean functions such as COMPARISON are in LT_1 but require weights of exponential size. Thus, the most important restriction of LTUs which as been considered in the literature has *small weights*, *i.e.* integer weights bounded polynomially in the fan-in [6, 15, 18]. Let $\widehat{LT}_d$ denote the set of Boolean functions computable by a depth-d polynomial size circuit composed of LTUs with small weights. The strongest relationship between LT_d and $\widehat{LT}_d$ has been obtained recently by Goldmann and Karpinski [5], who proved that $LT_d \subset \widehat{LT}_{d+1}$ $\forall d \geq 1$.

The class of linear threshold Boolean functions with integer parameters w_i bounded by a constant, constitutes naturally the next stage in this simplification of the LTUs. The simplest situation, where each w_i is either $+1, 0$ or -1, corresponds to the class of *majority Boolean functions* and is the central topic of this paper. From a circuit complexity point of view, this new subset of linear threshold Boolean functions presents no particular interest since any LTUs can be transformed into a unit computing a majority function, by replacing each connection of value w_i by $|w_i|$ connections of value $\mathrm{sgn}(w_i)$. Moreover, the polynomial size property of a network is preserved by this transformation when the functions computed by the units of the initial net are in $\widehat{LT}_1$. Therefore, in all the theoretical works on the circuit complexity of feedforward networks, the circuits composed of majority Boolean functions are only mentioned as equivalent to these based on functions in LT_1, or at least in $\widehat{LT}_1$.

However, the aim of an artificial neural network is more to be able to learn a wide family of tasks, than to achieve one particular function. Whenever a circuit architecture has to be determined to suit various tasks, it is of high interest to be able to choose, for example, between a circuit with a few bits per connections or another with some more computational units but with at most one connection of one bit between each pair of units. Also in simulation, when a neural network is used to learn a task, whatever is the training process involved, it would be desirable to know whether the loading is possible or not for a given quantization level of the parameters.

More specifically, in some other studies, we are designing training algorithms for democratic networks [7, 8, 9]. Among other approaches, we attempted to develop methods constructing the network layer by layer during the training phase [1]. In this context, it is highly important to know, for example, whether there exists a depth-2 network for any task that has to be loaded.

The present paper investigates the computational power of feedforward networks whose units realize majority functions. It tries to shed some light to the following question:

> Does the computational power of LTUs lie more in the richness of the various affine combinations of the inputs or in the non-linear function sgn ?

The remainder of this paper is divided into five sections. The model of network involved in the following sections is defined formally in section 2. Section 3 presents a couple of simple constructions for circuits computing some basic Boolean functions such as *AND*, *OR*, *AT-LEAST-k* and *PARITY*. The existence of a depth 2 universal circuit, *i.e.* able to realize any functions of $\mathcal{B}^n$, is discussed in section 4. In section 5, the VC-dimension of the class of majority Boolean functions is shown to be equal to that of all the linear threshold Boolean functions. A general discussion and some suggestions for further research constitute the concluding section.

2 Majority functions and democratic networks

The class MAJ_1 of the *majority Boolean functions* is the set of linear threshold Boolean functions with weights w_i restricted to the set $\{-1, 0, +1\}$.

> **Remark 2.1** For the sake of simplicity in the further developments, it is convenient to consider a class of functions closed under negation. The negation of a function $f \in MAJ_1$ of weights $\boldsymbol{w}$ is already in MAJ_1 when the number of arguments of f is odd, since the latter is self-dual.The negation of an 'even-majority'

function can be obtained by an inversion of the weight vector, and by simultaneously changing the convention on the value of $\text{sgn}(0)$. This effect can be obtained for example by allowing the threshold to take values in $\{-\frac{1}{2}, +\frac{1}{2}\}$. This limited flexibility of the threshold gives the choice between an 'absolute' and a 'non-absolute' majority and has no effect on odd-majority functions. Finally, observe that with the closure of MAJ_1 under negation we also get its closure under duality.

In practice, an inversion of every out-going connection of a unit computing f corresponds to a modification of f into not-f; if all the in-going connections are also inverted, f is change into its dual f^d and finally if the threshold is inverted at the same time, the new function computed by the unit is f again.

Definition 2.1 A Boolean function $f : \mathbb{B}^n \to \mathbb{B}$ is a *majority Boolean function* (*i.e.* $f \in MAJ_1$) if there exists a weight vector $\boldsymbol{w} \in \{-1, 0, +1\}^n$ and a threshold $w_0 = \pm\frac{1}{2}$ satisfying equation (1). The w_i are also called the *coefficients* of f and the dot product $\boldsymbol{w}^\top \boldsymbol{b}$ is the *potential* of f for the input $\boldsymbol{b}$.

Definition 2.2 A *democratic network* is a feedforward circuit composed of units computing majority Boolean functions with the additional property that there is at most one connection between each pair of units. $\mathcal{MAJ}_d$ (resp. MAJ_d) denotes the set of Boolean functions realizable by a democratic network of depth d and of any size (resp. of size bounded by a polynomial in its number of inputs). A network is said to be *degenerate* if it does contain at least two distinct sub-circuits computing the same Boolean function.

One observes that the closure of MAJ_1 under negation and duality implies this same property on $\mathcal{MAJ}_d$ and on MAJ_d.

3 Representation of basic functions

In the beginning of the last decade, polynomial size, constant depth circuits composed of LTUs became popular when it was first shown that there is no polynomial size, constant depth circuit computing *PARITY* with only *AND*, *OR* and *NOT* processing units [4]. A few years later it was proved that, even if we add the *PARITY* function to this previous set of basic units, there is no polynomial size, constant depth circuit able to compute *MAJORITY* [16]. In contrast, it is interesting to determine the complexity of democratic networks computing these basic functions.

Since the binary conjunction *2-AND* is a majority function, the conjunction of n arguments *n-AND* is in $MAJ_{\lceil \log_2 n \rceil}$ by decomposition into *2-AND*s. However, the conjunction of n arguments can be realized by smaller democratic networks :

Proposition 3.1 $n\text{-}AND \in MAJ_2$.

Proof: Consider the $n-1$ pairs of inputs $(1,2),(2,3),\ldots,(n-1,n)$. For each of these pairs, introduce 2 units on the hidden layer, one with coefficients $(+1,+1)$ and the other with coefficients $(-1,-1)$. Each hidden unit has a negative threshold and is connected to the output unit by a link of weight $+1$. The contribution of each pair of hidden units to the output potential is 0 if the two corresponding inputs are identical and -2 otherwise. The total output potential will then be 0 if all the inputs are identical and negative otherwise. Adding one connection from an arbitrary input to the output will produce the desired function. Moreover, the network is clearly non-degenerate, and $d=2$, $s=2n-2$. △

Note that the latter construction uses only fan-in-2 units on the hidden layer. When the depth of the circuit is not critical, *n-AND* can be computed with a number of units bounded by a logarithm in n.

Proposition 3.2 *n-AND* can be computed by a non-degenerate democratic network, with $s \in O(\log n)$ and $d \in O(\log^* n)$.

To verify this proposition let us first prove the following lemma:

Lemma 3.3 The computation of *n-AND* can be reduced to the computation of an *AND* of $\lfloor \log_2(n+1) \rfloor$ variables, using $O(\log_2 n)$ majority units.

Proof: Consider a partition of the input set I into I_1 and I_2 with $|I_2| = \lceil \frac{n}{2} \rceil - 1$. Add a hidden unit with a negative threshold, connected to each input of I_1 with a weight $+1$ and to each input of I_2 with a weight -1. If all the inputs of I_2 are set to $+1$, the hidden neuron gives an output $+1$ only if all the inputs of I_1 are also set to $+1$. Thus, the computation of *AND* over the n inputs is equivalent to the computation of *AND* over the hidden neuron and the inputs of I_2. Applying this idea recursively to the subset I_2, one can construct one hidden layer composed of $h(n)$ neurons, where $h(n)$ is given by the following recursive equation:

$$h(n) \;=\; 1 + h(\lceil \frac{n}{2} \rceil - 1), \qquad h(0) \;=\; h(1) \;=\; 0. \tag{2}$$

The exact solution of equation (2) is $h(n) = \lfloor \log_2 \frac{2}{3}(n+1) \rfloor$ for every $n > 0$. The number $a(n)$ of arguments of the remaining *AND* is given by the same recursive relation as $h(n)$, but with initial conditions $a(0)=0$ and $a(1)=1$. By solving this equation, one obtains $a(n) = \lfloor \log_2(n+1) \rfloor$. △

Proposition 3.2 is established by recursively applying lemma 3.3.

Proof: The final size $s(n)$ of the network is given by the following recursive equation:

$$s(n) = h(n) + s(a(n)), \qquad s(0) = s(1) = 0. \tag{3}$$

It can easily be proved that the solution $s(n)$ of this equation is in $O(\log n)$. The depth of this network is given by the number of time one has to apply function a to the number n of inputs until a value smaller than 1 is reached, and this is clearly in $O(\log^* n)$. $\triangle$

Figure 1 illustrates this construction by showing the transformation of *10-AND* into a three-layered democratic network.

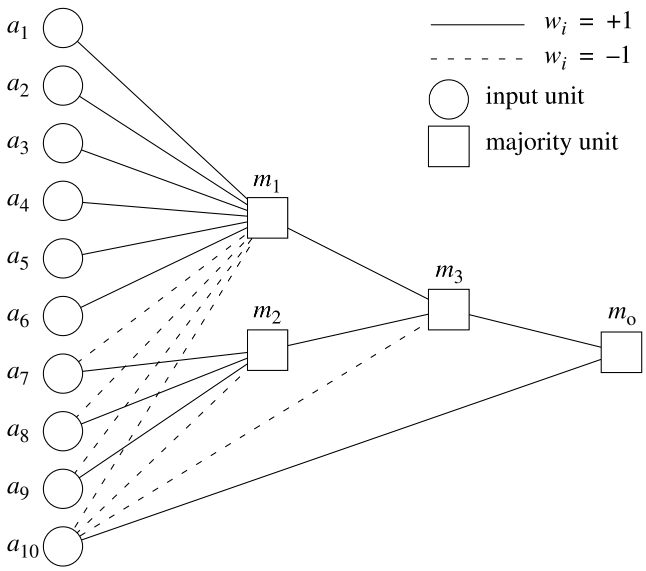

Figure 1: Multi-layer democratic circuit simulating a conjunction.
$AND(a_1, \ldots, a_{10}) = AND(m_1, a_7, \ldots, a_{10}) = AND(m_1, m_2, a_{10}) = AND(m_3, a_{10}) = m_o$.

At first glance, the fact that in our model of majority function the threshold is limited to $\{\pm\frac{1}{2}\}$ seems to be very restrictive. Indeed, with a threshold varying in the set $\{-n, ..., +n\}$, the set of basic functions would contain *n-AND* and *n-OR* and more generally for every k, *AT-LEAST-k* (resp. *AT-MOST-k*) which takes the value $+1$ if and only if there are at least (resp. at most) k positive inputs. This restriction on the threshold was maintained for uniformity with the coefficients and to satisfy the constraints given by the hardware implementation. The next proposition shows that for every k, *AT-LEAST-k* and *AT-MOST-k* are computable by small democratic networks.

Proposition 3.4 *AT-LEAST-k*, *AT-MOST-k* $\in MAJ_2 \; \forall k$.

Proof: Since MAJ_2 is closed under negation and duality, and $AT\text{-}LEAST\text{-}(\lfloor\frac{n}{2}\rfloor+1)$ is in MAJ_1, we only have to show that $AT\text{-}LEAST\text{-}k$ is in MAJ_2 $\forall\ k = \lfloor\frac{n}{2}\rfloor + 2, \ldots, n$.

Regroup the inputs into $m = \lfloor\frac{n}{2}\rfloor$ pairs $(1,2),(3,4),\ldots$. Connect each input directly to the output unit with a positive weight. For each of the m pairs of inputs, add 4 hidden units of negative threshold and with coefficients $(+1,+1)$, $(+1,-1)$, $(-1,+1)$ and $(-1,-1)$, respectively. When each of the $6m$ connections towards the output unit has the value $+1$, the contribution of each pair of inputs to the output potential is -2 times the number of negative inputs in the pair. The total contribution is then $-2l$, where l is the number of negative inputs among the $2m$ first inputs.

To realize $AT\text{-}LEAST\text{-}k$, we only need to add on the hidden layer $1+2(n-k)$ units which should have a positive answer whenever there are at least k positive inputs. These $1+2(n-k)$ units can be any of the $n+1$ units with at least $n-1$ among n coefficients equal to $+1$. As $n+1$ is bigger than $1+2(n-k)$ when $k \geq \lfloor\frac{n}{2}\rfloor + 2$, the network can always be non-degenerate. Thus, the output potential v will always be odd and

$$v \begin{cases} \geq & 1 \quad \text{if} \quad l \leq n-k \\ \leq & -1 \quad \text{if} \quad l > n-k \end{cases} .$$

In case of an odd number of arguments n, we just need to add a positive connection from the n^{th} input to the output and choose a positive threshold for the output unit. △

Definition 3.1 A Boolean function f is *symmetric* if $f(b_1,\ldots,b_n) = f(b_{\sigma(1)},\ldots,b_{\sigma(n)})$ for any permutation σ of the inputs. A well known characterization of symmetric functions is the following: f is symmetric if there exists k integers $t_1,\ldots,t_k$ such that $f(b_1,\ldots,b_n)=1$ iff $\sum_{i=1}^{n} b_i \in \{t_1,\ldots,t_k\}$ [6, 17].

Corollary 3.5 Any symmetric Boolean function is in MAJ_3.

Proof: This result is a direct consequence of proposition 3.4 and of construction presented in [6, 17]:

$$\begin{aligned} f(\boldsymbol{b}) = \mathrm{maj\_}(AT\text{-}LEAST\text{-}t_1(\boldsymbol{b}), &\quad AT\text{-}MOST\text{-}t_1(\boldsymbol{b}), \\ &\ldots, \\ AT\text{-}LEAST\text{-}t_k(\boldsymbol{b}), &\quad AT\text{-}MOST\text{-}t_k(\boldsymbol{b})) \end{aligned} \tag{4}$$

where maj_ denotes the majority function with all coefficients $+1$ and a negative threshold. △

The *n-PARITY* function is defined as the product of its inputs: $f(b_1, \ldots, b_n) = \prod_{i=1}^{n} b_i$. A well known depth-2 circuit of LTUs realizes *n-PARITY* with exactly n hidden units: each of them computes *AT-LEAST-k* for $k = 1, \ldots, n$, and the output is a function of MAJ_1 with alternating weight signs $\boldsymbol{w} = (+1, -1, +1, \ldots)$. This construction is the smallest known when no jumping connections over layers are allowed, otherwise the size s of the depth-2 circuit can be divided by 2 [10] and if depth-3 networks are considered, the size can even be reduced to $O(\sqrt{n})$ [19].

Although the above construction for *PARITY* reduces by a factor 2 the size s of the depth-3 democratic network compared to the general construction for a symmetric function (corollary 3.5), it can not be used for the development of a depth-2 circuit computing *n-PARITY*. The following proposition presents a completely different construction solving this problem with no more than n units on the single hidden layer, and without jumping connections. Note that this construction has been discovered independently by T. Grossman and is mentioned without proof in [13].

Proposition 3.6 *n-PARITY* can be computed by a depth-2 non-degenerate democratic network composed of n hidden units.

Proof: On the hypercube $\mathbb{B}^n$, let us call the point $-\boldsymbol{b}$ the *antipodal* of $\boldsymbol{b}$; *equator* of $\boldsymbol{b}$, noted $\mathrm{Eq}(\boldsymbol{b})$, denotes the set $\{\boldsymbol{e} \in \mathbb{B}^n | \boldsymbol{e}^\top \boldsymbol{b} = 0\}$. For any Boolean function f, the *characteristic subset* of $\mathbb{B}^n$ is defined as $\mathrm{C}(f) = \{\boldsymbol{b} \in \mathbb{B}^n | f(\boldsymbol{b}) = +1\}$. The two following observations are the key elements of the proof:

- The sum of the outputs of 2 majority units of positive threshold and with weights $\boldsymbol{w}$ and $-\boldsymbol{w}$ respectively is $+2$ if the input is in $\mathrm{Eq}(\boldsymbol{w})$ and 0 otherwise (obvious).
- For n even, there exist $\frac{n}{2}$ equators covering $\mathrm{C}(n\text{-}PARITY)$ without containing any other points (proved below).

With these remarks, the construction follows easily and it is illustrated in figure 2 for the case $n = 4$.

The network with n hidden units and 1 output unit is such that each hidden unit i has a weight vector $\boldsymbol{w}^i$, a positive threshold and a positive connection with the output. The $\boldsymbol{w}^i$s are defined by

$$\boldsymbol{w}^1 = (\underbrace{+1, \ldots, +1}_{\lceil \frac{n}{2} \rceil}, \underbrace{-1, \ldots, -1}_{\lfloor \frac{n}{2} \rfloor}),$$

and $\boldsymbol{w}^2, \ldots, \boldsymbol{w}^n$ are the cyclic permutations of the n components of $\boldsymbol{w}^1$, *i.e.* $w_{j\oplus 1}^{i+1} = w_j^i,\ i, j \in \{1, \ldots, n\}$, where $j \oplus 1$ denotes the cyclic increment of j ($n \oplus 1 = 1$).

As a consequence of this definition of the $\boldsymbol{w}^i$ we note that $\forall \boldsymbol{b} \in \mathbb{B}^n, \forall i \in \{1, \ldots, n\}$,

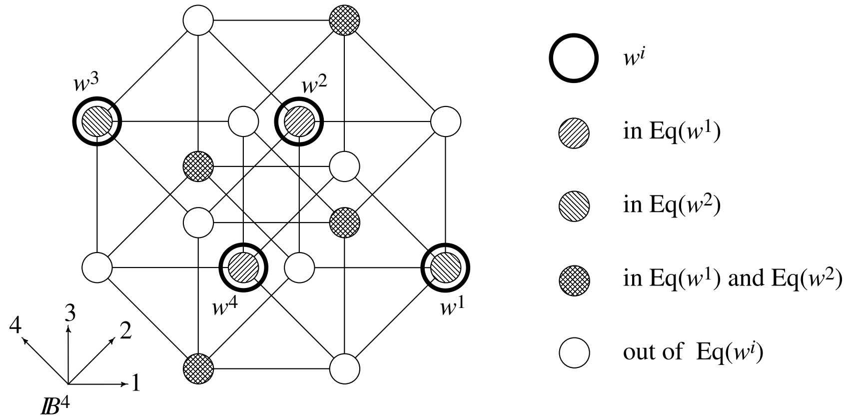

Figure 2: Construction of 4-*PARITY* using 4 majority units.
Four majority units, whose weight vectors $\boldsymbol{w}^i$ are equal to the vertices of the hypercube indicated with a thick circle, are used to cover all the vertices of positive parity without covering any other vertex.

$$\boldsymbol{b}^\top \boldsymbol{w}^{i\oplus 1} - \boldsymbol{b}^\top \boldsymbol{w}^i \in \{-4, 0, +4\}. \tag{5}$$

Moreover, if b^+ (resp. b^-) denotes the number of positive (resp. negative) components in the point $\boldsymbol{b}$,

$$\boldsymbol{b}^\top \boldsymbol{w}^i \bmod 4 = \begin{cases} 0 & \text{if} \quad b^+, b^- \text{ are even} & (i.e.\ n \text{ is even}) \\ 1 & \text{if} \quad b^+ \text{ is odd and } b^- \text{ is even} & (i.e.\ n \text{ is odd}) \\ 2 & \text{if} \quad b^+, b^- \text{ are odd} & (i.e.\ n \text{ is even}) \\ 3 & \text{if} \quad b^+ \text{ is even and } b^- \text{ is odd} & (i.e.\ n \text{ is odd}) \end{cases} . \tag{6}$$

- When n is even, $\boldsymbol{w}^i$ and $\boldsymbol{w}^{i\pm\frac{n}{2}}$ are antipodal and thus the contribution of the units i and $i \pm \frac{n}{2}$ to the output potential is 0 for every point of $\mathbb{B}^n$ but these in Eq($\boldsymbol{w}^i$) for which is it +2. By equation (6), if a point $\boldsymbol{b}$ is in Eq($\boldsymbol{w}^i$), b^+ and b^- are both even and thus $\boldsymbol{b} \in$ C(n-*PARITY*). To complete the proof it remains to show that any point of C(n-*PARITY*) is in Eq($\boldsymbol{w}^i$) for at least one i. Let $\boldsymbol{b}$ be an arbitrary point in C(n-*PARITY*), *i.e.* b^- is even and by equation (6), $\boldsymbol{b}^\top \boldsymbol{w}^i \bmod 4 = 0$, and say $\boldsymbol{b}^\top \boldsymbol{w}^1 = 4\alpha = -\boldsymbol{b}^\top \boldsymbol{w}^{\frac{n}{2}+1}$. By property (5), the sequence $4\alpha = \boldsymbol{b}^\top \boldsymbol{w}^1, \ldots, \boldsymbol{b}^\top \boldsymbol{w}^{\frac{n}{2}+1} = -4\alpha$ has to be 0 for at least one $i \in \{1, \ldots, \frac{n}{2}\}$, and thus $\boldsymbol{b} \in$ Eq($\boldsymbol{w}^i$).
- When n is odd (say $n = 2m+1$), one observes that $\boldsymbol{b}^\top \boldsymbol{w}^i = -\boldsymbol{b}^\top \boldsymbol{w}^{i\pm m} \pm 2\ \forall i \in \{1, \ldots, n\}$, for any possible input vector $\boldsymbol{b}$. This implies that these two dot products are of the same sign only if they are both +1 or both −1. As n is odd, there is at least one couple $(\boldsymbol{b}^\top \boldsymbol{w}^i, \boldsymbol{b}^\top \boldsymbol{w}^{i\pm m})$ with both elements of the same sign and by property (6), the existence of couples of type $(+1, +1)$

and of type $(-1, -1)$ are mutually exclusive. The sign of the output is thus completely determined by the type of these particular couples appearing in the sequence of the potentials of all the hidden units. Property (6) concludes the proof, since couples of type $(+1, +1)$ correspond to the case of b^- even, *i.e.* $\boldsymbol{b} \in \mathrm{C}(n\text{-}PARITY)$.

$\triangle$

4 Universal democratic networks

Although *AND*s and *OR*s are not very adequate to simulate, within a compact size circuit, most of the Boolean functions, they remain interesting since every function can be represented in a depth-2 circuit, using the CNF or DNF forms (conjunctive or disjunctive normal forms). This section will address the question of what is the shortest democratic network able to compute any Boolean function.

Using the CNF or the DNF form, an obvious corollary of proposition 3.1 is that $\mathcal{B}^n \in \mathcal{MAJ}_4$. The following proposition shows how $\mathcal{B}^n \in \mathcal{MAJ}_3$ by the simultaneous use of both, conjunctive and disjunctive forms.

Proposition 4.1 $\mathcal{MAJ}_3 \ = \ \mathcal{B}^n$.

Proof: For a given function f, consider a CNF decomposition $AND(d_1, \cdots, d_k)$ and a DNF decomposition $OR(c_1, \cdots, c_l)$, where d_i and c_i are disjunctions and conjunctions, respectively, over the set of inputs and their negations. Let us assume that $k \geq l$; if it is not the case, one can replace f by not-f and exchange the roles of the d_i and the c_i. Then we claim that:

$$f = \mathrm{maj}(c_1, \cdots, c_l, d_1, \cdots, d_{l-1}).$$

This majority is composed of $2l - 1$ terms. If $f(\boldsymbol{b}) = +1$, then $d_i(\boldsymbol{b}) = +1 \ \forall i = 1, \cdots, k$ and $c_i(\boldsymbol{b}) = +1$ for at least one $i = 1, \cdots, l$, so at least l terms among the $2l - 1$ will be $+1$. On the other hand, if $f(\boldsymbol{b}) = -1$, $c_i(\boldsymbol{b}) = -1 \ \forall i = 1, \cdots, l$ and thus at least l of the $2l - 1$ hidden units answer -1 for $\boldsymbol{b}$. $\triangle$

The size of the network obtained by this construction is in $O(min\{k, l\})$, but of course, for almost all interesting Boolean functions, this size of the most compact normal form can be quite large, *i.e.* exponential in n (*e.g.* *PARITY*). Moreover, there is very little hope to be able to save one more layer of a three-layered democratic network simulating an arbitrary Boolean function, when the construction is based on CNFs or DNFs of the functions, as suggested in proposition 4.1.

An alternative way for expressing Boolean functions is based on the well known fact that every Boolean function can be expressed as a polynomial, on the field of rational, in the variables $b_1, \ldots, b_n$, when the numerical representation -1 and $+1$ is used for the Boolean values True and False, respectively. For every Boolean function $f : \mathbb{B}^n \rightarrow \mathbb{B}$, there is a unique vector of coefficients $\boldsymbol{c} \in \mathbb{R}^{2^n}$, such that

$$f(\boldsymbol{b}) = \sum_{\alpha \in \{0,1\}^n} c_\alpha \prod_{i=1}^{n} b_i^{\alpha_i}, \tag{7}$$

A term of this polynomial, indexed by α, is simply a parity function over the subset of variables whose characteristic vector is α. Thus, using expression (7), an arbitrary Boolean function can be expressed as a linear threshold function of parities defined over some of the n inputs. Since the coefficients c_α of the polynomial are rational, the linear threshold function can be simulated by a majority function, assuming a duplication of the parity functions. Using proposition 3.6, this polynomial form provides a depth-3 degenerate circuit of majority units.

In the last part of this section, we are going to show how any Boolean function can be simulated by a democratic network of depth 2. This issue is of high interest when we consider incremental training algorithms that build a depth-2 democratic network by adding hidden units iteratively.

In [2], the polynomial representation (7) has been used to simulate any Boolean function by a network with LTUs, and the author shows how one can get rid of the second layer, in order to get a depth-2 network of LTUs computing an arbitrary Boolean function (see theorem 2.1 in [2]). However, the construction proposed uses *AT-LEAST-k* and *AT-MOST-k* functions in the first layer, and so it can not be exploited for the construction of a depth-2 universal democratic network, unless n constant inputs are artificial added to the n original inputs of the network.

Proposition 4.2 Any Boolean function can be computed by a depth-2 democratic network.

Proof: Since non-degeneracy is not required in this result, it is sufficient to show that any Boolean function can be computed by a depth-2 circuit, with majority units on the hidden layer, and a single LTU as output unit. The latter can then be simulated by a majority unit and an appropriate duplication of the hidden units. For this purpose, we are going to start from the depth-3 democratic network mentioned above and based on the polynomial representation of equation (7), and using remark 4.1, we will show how one can get rid of the second layer computing the parity functions.

Remark 4.1 An intermediate unit can be suppressed from a network composed of LTUs if the absolute value of its potential is a non-zero constant α for every possible input of the network. After dropping such a unit of coefficients $\boldsymbol{w}$ and of output connection value o, for each of its in-connection of value w_i, a connection of value $\frac{ow_i}{\alpha}$ should be introduced from the i^{th} predecessor of the unit to its successor.

To complete the proof, we will show that there is a depth-2 democratic network computing the n-*PARITY* in such a way that the absolute value of the output potential is α, where α is a function of n. This network is obtained by putting on the hidden layer the 2^n units with positive threshold and coefficients $\boldsymbol{w} \in \mathbb{B}^{2^n}$. The output connection o of a given hidden unit is fixed by the following rule:

$$\text{if } n \text{ is even,} \quad o = \begin{cases} +1 & \text{if} \quad \mathrm{Eq}(\boldsymbol{w}) \subset \mathrm{C}(n\text{-}PARITY) \\ -1 & \text{if} \quad \mathrm{Eq}(\boldsymbol{w}) \subset \mathbb{B}^n - \mathrm{C}(n\text{-}PARITY) \end{cases} ,$$

$$\text{if } n \text{ is odd,} \quad o = \begin{cases} +1 & \text{if} \quad \exists\ \boldsymbol{x} \in \mathrm{C}(n\text{-}PARITY) \quad \text{s. t.} \quad \boldsymbol{x}^\top \boldsymbol{w} = +1 \\ -1 & \text{if} \quad \exists\ \boldsymbol{x} \in \mathrm{C}(n\text{-}PARITY) \quad \text{s. t.} \quad \boldsymbol{x}^\top \boldsymbol{w} = -1 \end{cases} .$$

This particular choice for the value o ensures that the total contribution to the potential of the output unit is strictly positive if the input is in $\mathrm{C}(n\text{-}PARITY)$ and strictly negative otherwise. Moreover, the property assumed in remark 4.1 is a consequence of the symmetry due to consideration on the hidden layer of all the complete majorities (*i.e.* without 0 weights) over the n inputs. △

5 The VC-dimension of MAJ_1

A characterization of the computational power of a class of functions is given by the Vapnik-Chervonenkis dimension [20]. In order to formalize this notion, we first introduce some preliminary definitions. A *dichotomy* of p points $\omega_1, \ldots, \omega_p$ in some space Ω is a partition of these points into two disjoint classes. It can be considered as a function $d : \{\omega_1, \ldots, \omega_n\} \to \mathbb{B}$ and it will be denoted by a Boolean vector $\boldsymbol{d} \in \mathbb{B}^p$. Let F be a set of Boolean-valued functions defined on Ω; a subset $\Gamma \subseteq \Omega$ is said to be *shattered* by F if each of the $2^{|\Gamma|}$ possible dichotomies of Γ corresponds to at least one function of F restricted to Γ. The *Vapnik-Chervonenkis dimension* of F, noted VC-dim(F), is the size of the largest shattered subset $\Gamma \in \Omega$.

When $\Omega = \mathbb{R}^n$, all dichotomies of $n+1$ points are linearly separable if and only if the $n+1$ points are not contained in a $n-1$ dimensional hyperplane of $\mathbb{R}^n$ [3]. This result proves that the VC-dimension of the set of all linearly separable Boolean functions of n arguments is $n+1$. The following proposition shows that the VC-dimension does not change when the set of Boolean functions is restricted from LT_1 to MAJ_1.

Proposition 5.1 $VC\text{-dim}(MAJ_1 \cap \mathcal{B}^n) = n + 1$.

Proof: Since $MAJ_1 \subset LT_1$, the VC-dimension of $MAJ_1 \cap \mathcal{B}^n$ is bounded from above by $n+1$. To prove the proposition, we will show that the following set of $n+1$ points of $\mathbb{B}^n$

$$\underbrace{(-1,-1,\ldots,-1)}_{\boldsymbol{b}^0},\ \underbrace{(+1,-1,\ldots,-1)}_{\boldsymbol{b}^1},\ \underbrace{(+1,+1,\ldots,-1)}_{\boldsymbol{b}^2},\ \ldots\ ,\underbrace{(+1,+1,\ldots,+1)}_{\boldsymbol{b}^n}$$

is shattered by MAJ_1.

Let an arbitrary dichotomy of these $n+1$ points be given by $\boldsymbol{d} = (d_0, d_1, \ldots, d_n) \in \mathbb{B}^{n+1}$. Since MAJ_1 is closed under negation, we can assume without loss of the generality that $d_0 = +1$.

Consider the following definition of the threshold and of the weights:

$$w_0 = \frac{1}{2}, \quad w_i = \frac{d_i - d_{i-1}}{2} \; \forall i = 1, \ldots, n. \tag{8}$$

With these choices, equation (1) gives $f(\boldsymbol{b}^0) = +1 = d_0$, since $w_0 + \boldsymbol{w}^\top \boldsymbol{b}^0 = w_0 + \frac{1-d_n}{2} = 1 - \frac{d_n}{2}$, which is either $\frac{1}{2}$ or $\frac{3}{2}$. Let call α this quantity $1 - \frac{d_n}{2}$. Finally observe that for all $i = 1, \ldots, n$, $w_0 + \boldsymbol{w}^\top \boldsymbol{b}^i = w_0 + \boldsymbol{w}^\top \boldsymbol{b}^0 + \boldsymbol{w}^\top (\boldsymbol{b}^i - \boldsymbol{b}^{\mathrm{o}}) = \alpha + \frac{1}{2} \sum_{j=1}^{n} (d_j - d_{j-1})(b_j^i - b_j^0) = \alpha + \sum_{j=1}^{i} (d_j - d_{j-1}) = \alpha + d_i - 1$, which is α (*i.e.* positive) if $d_i = 1$, and $\alpha - 2$ (*i.e.* negative) if $d_i = -1$. Thus, the choice of the parameters proposed in (8) solves the dichotomy $\boldsymbol{d}$.

$\triangle$

6 Discussion

Throughout this paper, we established various results suggesting that many Boolean functions can be represented efficiently by multi-layered networks composed of majority units, even if multiple connections between two units are not allowed. This suggests that a usual network of fixed architecture has no intrinsic limitations when its parameters are limited to $+1$, 0 and -1.

The constructions developed in the first part of section 2 for the computation of *AND* and *AT-MOST-k* are quite simple and their results are not surprising. On the other hand, the number n of hidden units used for the computation of n-*PARITY* with a depth-2 non-degenerate democratic network without jumping connections, is probably a tight bound of the minimum, since this value is also the best known one when general LTUs compose the network.

Many open questions are related to universal democratic networks. Is any Boolean function f computable by a depth-2 non-degenerate democratic network? This question has been solved positively by computer for $n \leq 4$, but is still open for larger numbers of arguments. The polynomial (7) is unique and gives the exact value $+1$ or -1 of the function for any input, and so the sgn function is not necessary. Another open question we attempted to solve without success is the following:

> Can the coefficients c_α of this polynomial be restricted to $-1, 0, +1$ when only the sign of the polynomial is required to match with the output of function f ?

This issue goes beyond the neural network field, since it will provide a general way of expressing any Boolean function into a majority of distinct parities. We also checked this question by computer, and it was found to be true for all functions with up to 5 arguments, but the general question remains open.

Acknowledgment: This work has been done while the author was working at the Swiss Federal Institute of Technology, supported by grant 20-5637.88 of the Swiss National Science Foundation. Thanks are expressed to Frederic Aviolat for his valuable participation in the proof of proposition 5.1 and for his research on the question quoted in the discussion section. I am also grateful to an anonymous referee, for his pertinent remarks and especially for his suggestion of a much simpler form of the proof of proposition 5.1.

References

[1] F. AVIOLAT AND E. MAYORAZ, *A constructive training algorithm for feedforward neural networks with ternary weights*, in Proceedings of ESANN 94, F. Blayo and M. Verleysen, eds., 1994, pp. 123–128.

[2] J. BRUCK, *Harmonic analysis of polynomial threshold functions*, SIAM J. Disc. Math., 3 (1990), pp. 168–177.

[3] T. M. COVER, *Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition*, IEEE Trans. on Electronic Computers, 14 (1965), pp. 326–334.

[4] M. FURST, J. B. SAXE, AND M. SIPSER, *Circuits and the polynomial-time hierarchy*, in Proceedings of 22^{nd} IEEE FOCS Symposium, 1981, pp. 260–270.

[5] M. GOLDMANN AND M. KARPINSKI, *Simulating threshold circuits by majority circuits*, in Proceedings of the Twenty Fifth Annual ACM Symposium on Theory of Computating, 1993, pp. 551–560.

[6] A. HAJNAL, W. MAASS, P. PUDLÁK, M. SZEGEDY, AND G. TURÁN, *Threshold circuits of bounded depth*, in Proceedings of 28^{th} IEEE FOCS Symposium, 1987, pp. 99–110.

[7] E. MAYORAZ, *Maximizing the stability of a majority perceptron using tabu search*, in Proceedings of IJCNN'92 Baltimore, 1992, pp. II254–II259.

[8] ———, *Feedforward Boolean Networks with Discrete Weights: Computational Power and Training*, PhD thesis, Swiss Federal Institute of Technology, Department of Mathematics, 1993.

[9] E. MAYORAZ AND V. ROBERT, *Maximizing the robustness of a linear threshold classifier with discrete weights*, Network: Computation in Neural Systems, 5 (1994), pp. 299–315.

[10] R. C. MINNICK, *Linear-input logic*, IEEE Trans. on Electronic Computers, 10 (1961).

[11] S. MUROGA, *Threshold Logic and Its Applications*, John Wiley & Sons, New York, 1971.

[12] S. MUROGA, I. TODA, AND S. TAKASU, *Theory of majority decision elements*, J. Franklin Inst., 271 (1961), pp. 376–418.

[13] D. NABUTOVSKY, T. GROSSMAN, AND E. DOMANY, *Learning by chir without storing internal representations*, Complex Systems, 4 (1990), pp. 519–541.

[14] I. PARBERRY, *Circuit complexity and neural networks*, Tech. Rep. CRPDC-91-9, University of North Texas, Department of Computer Sciences, 1991.

[15] I. PARBERRY AND G. SCHNITGER, *Parallel computation with threshold functions*, J. Comp. System Sci., 36 (1988), pp. 278–302.

[16] A. A. RAZBOROV, *Lower bounds on the size of bounded depth circuits over a complete basis with logical addition*, Math. Notes, 41 (1987), pp. 333–338.

[17] K.-Y. SIU AND J. BRUCK, *Neural computation of arithmetic functions*, Proceedings of the IEEE, 78 (1990), pp. 1669–1675.

[18] ——, *On the power of threshold circuits with small weights*, SIAM J. Disc. Math., 4 (1991), pp. 423–435.

[19] K.-Y. SIU, V. P. ROYCHOWDHURY, AND T. KAILATH, *Depth-size tradeoffs for neural computation*, IEEE Trans. on Computers, Special Issue on Neural Networks, (1991), pp. 1402–1412.

[20] V. N. VAPNIK, *Estimation of Dependences Based on Empirical Data*, Springer-Verlag, New York, 1982.

[21] Y. A. ZUEV, *Asymptotics of the logarithm of the number of threshold functions of the algebra of logic*, Soviet Mathematics - Doklady, 39 (1989), pp. 512–513.