Multiple Threshold Neural Logic *

Vasken Bohossian Jehoshua Bruck

California Institute of Technology
Mail Code 136-93
Pasadena, CA 91125
E-mail: {vincent, bruck}@paradise.caltech.edu

May 23, 1996

Abstract

We introduce a new Boolean computing element related to the Boolean version of a neural element. Instead of the sign function in the Boolean neural element (also known as an LT element), it computes an arbitrary (with polynomialy many transitions) Boolean function of the weighted sum of its inputs. We call the new computing element an LTM element, which stands for Linear Threshold with Multiple transitions.

The paper consists of the following main contributions related to our study of LTM circuits: (i) the characterization of the computing power of LTM relative to LT circuits, (ii) a proof that the area of the VLSI layout is reduced from $O(n^2)$ in LT circuits to $O(n)$ in LTM circuits, for n inputs symmetric Boolean functions, and (iii) the creation of efficient designs of LTM circuits for the addition of a multiple number of integers and the product of two integers. In particular, we show how to compute the addition of m integers with a single layer of LTM elements.

Category : Theory, Complexity Theory.

1 Introduction

Human brains are by far superior to computers in solving hard problems like combinatorial optimization and image and speech recognition, although their basic building blocks are several orders of magnitude slower. This observation has boosted interest in the field of artificial neural networks [Hopfield 82], [Rumelhart 82]. The latter are built by interconnecting artificial neurons whose behavior is inspired by that of biological neurons. In this paper we consider the Boolean version of an artificial neuron, namely, a Linear Threshold (LT) element, which computes a neural-like Boolean function of n binary inputs [Muroga 71]. An LT element outputs the sign of a weighted sum of its Boolean inputs. The main issues in the study of networks (circuits) consisting of LT elements, called LT circuits, include the estimation of their computational capabilities and limitations and the comparison of their properties with those of traditional

*This work was supported in part by the NSF Young Investigator Award CCR-9457811 and by the Sloan Research Fellowship.

Boolean logic circuits based on AND, OR and NOT gates (called AON circuits). For example, there is a strong evidence that LT circuits are more efficient than AON circuits in implementing a number of important functions including the addition, product and division of integers [Siu 94], [Siu 93].

Motivated by our recent work on the VLSI implementation of LT elements [Bohossian 95b], we introduce in this paper a more powerful computing element, a multiple threshold neuron, which we call LTM, which stands for Linear Threshold with Multiple transitions. Instead of the sign function in the LT element it computes an arbitrary (with polynomialy many transitions) Boolean function of the weighted sum of its inputs.

The main issues in the study of LTM circuits (circuits consisting of LTM elements) include the estimation of their computational capabilities and limitations and the comparison of their properties to those of AON circuits. A natural approach in this study is first to understand the relation between LT circuits and LTM circuits. Our main contributions in this paper are:

- We characterize the computing power of LTM relative to LT circuits.
- We show that LTM circuits are more amenable in implementation than LT circuits. In particular, the area of the VLSI layout is reduced from $O(n^2)$ in LT circuits to $O(n)$ in LTM circuits, for n inputs symmetric Boolean functions.
- We demonstrate the power of LTM by deriving efficient designs of LTM circuits for the addition of m integers and the product of two integers.

Next we describe the formal definitions of LT and LTM elements.

1.1 Definitions and Examples

Let us first define a linear threshold gate.

Definition 1 *(LT)*
A linear threshold gate computes a Boolean function of its binary inputs :

$$f(X) = sgn(w_0 + \sum_{i=1}^{n} w_i x_i)$$

where the w_i are integers and $sgn(.)$ outputs 1 if its argument is greater or equal to 0, and 0 otherwise.

Here follows the formal definition of LTM.

Definition 2 *(LTM)*
A function f is in LTM if there exists a set of weights $w_i \in Z$, $1 \leq i \leq n$ and a function $h : Z \longrightarrow \{0,1\}$ such that

$$f(X) = h(\sum_{i=1}^{n} w_i x_i) \text{ for all } X \in \{0,1\}^n$$

The only constraint on h is that it undergoes polynomialy many transitions as its input scans $[-\sum_{i=1}^{n} |w_i|, \sum_{i=1}^{n} |w_i|]$.

Notice that without the constraint on the number of transitions, by setting $w_i = 2^{i-1}$, an LTM gate is capable of computing any Boolean function.

As an example of a function in LTM consider the n-variable XOR which cannot be implemented with a single LT element.

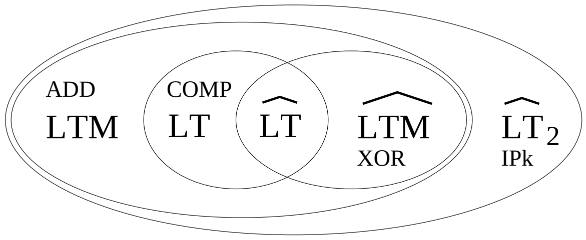

Figure 1: Relationship between Classes

Example 1 $(XOR \in LTM)$
$XOR(X)$ outputs 1 if $|X|$, the number of 1's in X, is odd. Otherwise it outputs 0. To implement it choose $w_i = 1$ and $h(k) = \frac{1}{2}(1 - (-1)^k)$ for $0 \leq k \leq n$. Note that $h(k)$ needs not be defined for $k < 0$ and $k > n$, and has polynomialy many transitions.

Another example is $ADD(X, Y)$, the sum of two n-bit integers X and Y.

Example 2 $(ADD \in LTM)$
To implement addition we set

$$f_l(X, Y) = h_l(\sum_{i=1}^{l} 2^i(x_i + y_i))$$

where $h_l(k) = 1$ for $k \in [2^l, 2 \times 2^l - 1] \cup [3 \times 2^l, +\infty)$. Defined thus, f_l computes the m-th bit of $X + Y$.

We use a hat to indicate small (polynomialy growing) weights, e.g. $\widehat{LT}$, $\widehat{LTM}$ [Bohossian 95a], [Siu 91], and a subscript to indicate the depth (number of layers) of the circuit of more than a single layers. All the circuits we consider in this paper are of polynomial size (number of elements) in n (number of inputs). For example, the class $\widehat{LT}_2$ consists of those Boolean functions that can be implemented by a depth-2 polynomial size circuit of $\widehat{LT}$ elements.

Figure 1 depicts the membership relations between five classes of Boolean functions, including, LT, $\widehat{LT}$, LTM, $\widehat{LTM}$ and $\widehat{LT}_2$, along with the functions used to establish the separations.

1.2 Organization

The paper is organized as follows. In Section 2, we prove the characterization results of LTM, including, the inclusion relations, in particular $LTM \subseteq \widehat{LT}_2$. In addition, we indicate which inclusions are proper and exhibit functions to demonstrate the separations. In Section 3, we study a number of applications as well as the VLSI implementations of LTM circuits. In particular, we show how to compute the addition of m integers with a single layer of LTM elements.

2 Classification of LTM

In this section we will prove the relations illustrated by Figure 1. We first show the inclusion relations. Then, we provide functions that demonstrate the separation between classes.

2.1 Inclusions

Most inclusion relations follow from the definitions : $\widehat{LT} \subseteq LT \subseteq LTM$ and $\widehat{LT} \subseteq \widehat{LTM} \subseteq LTM$. Only one requires a proof :

$\underline{LTM \subseteq \widehat{LT}_2}$

To show the above statement we use a result from [Goldman 93] : a single LT gate with arbitrary weights can be realized by an $\widehat{LT}_2$ circuit. Furthermore the non-linearity in the second layer can be removed without affecting the output of the circuit (a property called "1-approximability", [Hofmeister 96]). So, given $f \in LT$, $f(X) = \sum_{i_1}^{p} w_i f_i(X)$ where p is polynomial in n and $f_i \in \widehat{LT}$ for all i.

Now, consider the LT_2 implementation of a function in LTM. It consists of a layer of identical LT gates followed by a single gate with 1 and -1 weights and a -1 threshold. We substitute each LT gate of the first layer by its equivalent layer of $\widehat{LT}$ gates and weighted sum. We combine the weighted sums, i.e. collapse the second and the third level. The resulting circuit is in $\widehat{LT}_2$.

2.2 Separation

In Example 1 we saw that $XOR \in \widehat{LTM}$ and it is well known that $XOR \notin LT$. On the other hand $COMP(X,Y)$, the comparison of two n-bit integers is in LT [Siu 91].

$$COMP(X,Y) = sgn(\sum_{i=1}^{n} 2^i (x_i - y_i)) = \begin{cases} 1 & \text{if } Y \leq X \\ 0 & \text{otherwise} \end{cases}$$

Let us show that $COMP \notin \widehat{LTM}$. For that we introduce the notion of *entropy* of a Boolean function. An equivalent definition based on communication complexity is developed in [Szegedy 89].

Definition 3 *(Entropy)*
Given a n-variable Boolean function, S a subset of those variables and $s \in \{0,1\}^{|S|}$, we call $f_s(x_1,..,x_{n-|S|})$ the function obtained by assigning the value s to S in f. The entropy of f is defined as :

$$E[f] = \max_S |\{f_s : s \in \{0,1\}^{|S|}\}|$$

In words, the entropy is the maximum number of subfunctions over $n - |S|$ variables one can produce by assigning to a set S of its n variables all possible $2^{|S|}$ values. The maximum is taken over S.

Lemma 4 *(Exponential Entropy implies Exponential Weights)*
Given a function f such that $E[f]$ is exponential in n, its LTM implementation requires exponential weights, i.e. $\sum_1^n |w_i|$ exponential in n.

Proof : A subfunction can be written as

$$f_s(x_1,..,x_{n-|S|}) = f(X, S = s) = h(\sum_{i \in X-S} w_i x_i + W_s)$$

where $W_s = \sum_{i \in S} w_i s_i$. By the pigeonhole principle, and given that W_s is an integer, $|\{W_s : s\}|$ must be greater than $E[f]$. If it is not, there will not be enough distinct values of W_s to map to all $E[f]$ distinct subfunctions. That in turn implies

$$E[f] \leq \sum_{i \in S} |w_i| \leq \sum_{i=1}^{n} |w_i|$$

$\square$

$\underline{COMP \notin \widehat{LTM}}$

Proof : We show that $E[COMP]$ is exponential and use Lemma 4. Let

$$f_s(x_1, .., x_n) = COMP(X, Y = s)$$

There are 2^n such functions, let us show that they are all distinct. Given two distinct integers s_1 and s_2 choose X_0 such that $s_1 \leq X_0 < s_2$ then $f_{s_1}(X_0) \neq f_{s_2}(X_0)$. $\square$

$\underline{ADD \in LTM \text{ but } ADD \notin LT \cup \widehat{LTM}}$

Proof : We already saw that $ADD \in LTM$. The least significant bit of the sum is XOR which is not in LT. On the other hand, $E[ADD]$ is exponential by a proof similar to the one for $COMP$, implying that $ADD \notin \widehat{LTM}$. $\square$

$\underline{IP_k \in \widehat{LT}_2 \text{ but } IP_k \notin LTM}$

Proof : Let $IP(X, Y) = \sum_1^n x_i y_i$. Define the function $IP_k(X, Y) = 1$ iff $IP \geq k$, else $IP_k = 0$. We claim that $IP_k \notin LTM$. Indeed, if IP_k was in LTM then it could be implemented by a layer of $\widehat{LT}$ gates followed by a weighted sum [Goldman 93]. We could then combine the circuits for $k = 1..n$ to implement $IP2$ (Inner Product mod 2) in $\widehat{LT}_2$ which is known to be false [Hajnal 94]. $\square$

What remains to be shown in order to complete the classification picture is $\widehat{LT} = LT \cap \widehat{LTM}$. We conjecture that this is true and we are in the process of completing the proof.

3 Applications

The theoretical results about LTM can be applied to the VLSI implementation of Boolean functions. The idea of a gate with multiple thresholds came to us as we were looking for an efficient VLSI implementation of symmetric Boolean functions. Even though a single LT gate is not powerful enough to implement any symmetric function, a 2-layer LT circuit is, Figure 2. Furthermore, it is well known that such a circuit performs much better than the traditional logic circuit based on AND, OR and NOT gates. The latter has exponential size (or unbounded depth) [Wegener 91]. Implementing a generalized symmetric function in LT_2 requires up to n LT gates in the first layer. Those have the same weights w_i except for the threshold w_0. Instead of laying out n times the same linear sum $\sum_1^n w_i x_i$ we do it once and compare the result to n different thresholds The resulting circuit corresponds to a single LTM gate. Figure 2 shows the advantage of LTM over LT for the implementation of a generalized symmetric function. Indeed, the LT_2 layout is redundant, it has n copies of each weight, requiring area of at least $O(n^2)$. On the other hand, LTM performs a single weighted sum, its area requirement is $O(n)$.

We have fabricated a programmable generalized symmetric function on a 2μ, analog chip using the model described above. Floating gate technology is used to program the weights. We store a weight on a single transistor by injecting and tunneling electrons on the floating gate [Mead 95].

A single LTM gate can compute the addition of m n-bit integers $MADD$. The only constraint is that m be polynomial in n.

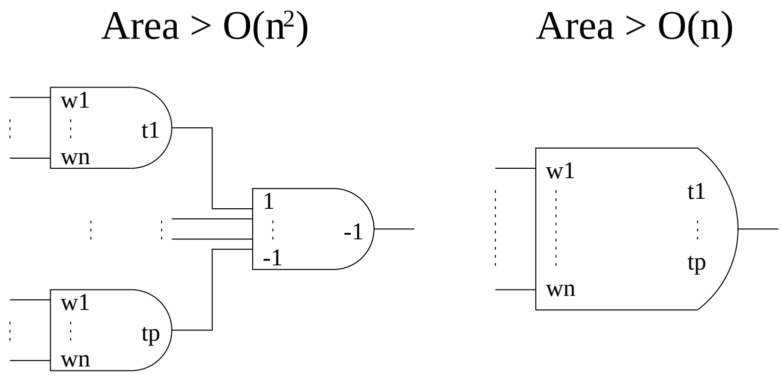

Figure 2: VLSI layout with LT_2 (left), with LTM (right).

Example 3 $(MADD \in LTM)$
$MADD$ returns an integer of at most $n + \log m$ bits. We need one LTM gate per bit. The least significant bit is computed by a simple m-bit XOR. For all other bits we use

$$f_l(X^{(1)}, .., X^{(m)}) = h_l(\sum_{i=1}^{l} 2^i \sum_{j=1}^{m} x_i^{(j)})$$

to compute the l-th bit of the sum.

Example 4 $(PRODUCT \in PTM)$ By analogy with PT_1, defined in [Bruck 90], in PTM_1 (or simply PTM) we allow a polynomial rather than a linear sum :

$$f(X) = h(w_1x_1 + ... + w_nx_n + w_{(1,2)}x_1x_2 + ...)$$

However we restrict the sum to have polynomialy many terms (else, any Boolean function could be realized with a single gate). The product of two n-bit integers X and Y can be written as $PRODUCT(X,Y) = \sum_{i=1}^{n} x_iY$. We use the construction of $MADD$ in order to implement $PRODUCT$.

$$PRODUCT(X,Y) = MADD(x_1Y, x_2Y, ..., x_nY)$$

$$f_l(X,Y) = h_l(\sum_{j=1}^{n} \sum_{i=1}^{l} 2^i x_j y_i)$$

f_l outputs the l-th bit of the product.

4 Conclusions

Our original goal was to use theoretical results in order to efficiently lay out a generalized symmetric function. During that process we came to the conclusion that the LT_2 implementation is partially redundant, which lead to the definition of LTM, a new, more powerful computing element. We characterized the power of LTM relative to LT. We showed how it can be used to reduce the area of VLSI layouts from $O(n^2)$ to $O(n)$ and derive efficient designs for multiple addition and product. Interesting directions for future investigation are (i) to prove the conjecture : $\widehat{LT} = LT \cap \widehat{LTM}$, (ii) to apply spectral techniques ([Bruck 90]) to the analysis of LTM, in particular show how PTM fits into the classification picture (Figure 1).

References

[Bohossian 95a] V. Bohossian and J. Bruck. On Neural Networks with Minimal Weights. In *Proc. of Neural Information Processing Systems 8*, 1995.

[Bohossian 95b] V. Bohossian, P. Hasler and J. Bruck. Programmable Neural Logic. *Unpublished manuscript*, Nov. 1995.

[Bruck 90] J. Bruck. Harmonic Analysis of Polynomial Threshold Functions *SIAM J. Disc. Math*, Vol. 3(No. 2)pp. 168–177, May 1990.

[Goldman 93] M. Goldmann and M. Karpinski. Simulating threshold circuits by majority circuits. In *Proc. 25th ACM STOC*, pages pp. 551–560, 1993.

[Hajnal 94] A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, G. Turan. Threshold Circuits of Bounded Depth. *Journal of Computer and System Sciences*, Vol. 46(No. 2):pp. 129–154, April 1993.

[Mead 95] P. Hasler, C. Diorio, B.A. Minch, C.A. Mead. Single Transistor Learning Synapses. *Unpublished manuscript.*

[Hastad 94] J. Hastad. On the size of weights for threshold gates. *SIAM. J. Disc. Math.*, 7:484–492, 1994.

[Hofmeister 96] T. Hofmeister. A Note on the Simulation of Exponential Threshold Weights. *1996 CONCOON conference.*

[Hopfield 82] J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. of the USA National Academy of Sciences*, 79:2554–2558, 1982.

[Muroga 71] M. Muroga. *Threshold Logic and its Applications.* Wiley-Interscience, 1971.

[Rumelhart 82] D. Rumelhart and J. McClelland. Parallel distributed processing: Explorations in the microstructure of cognition. *MIT Press*, 1982.

[Siu 91] K. Siu and J. Bruck. On the power of threshold circuits with small weights. *SIAM J. Disc. Math.*, Vol. 4(No. 3):pp. 423–435, August 1991.

[Siu 93] K. Siu, J. Bruck, T. Kailath, and T. Hofmeister. Depth Efficient Neural Networks for Division and Related Problems. *IEEE Trans. on Information Theory*, Vol. 39(No. 3), May 1993.

[Siu 94] K. Siu and V.P. Roychowdhury-VP. On Optimal Depth Threshold Circuits for Multiplication and Related Problems. *SIAM J. Disc. Math.*, Vol. 7(No. 2):pp. 284–292, May 94.

[Szegedy 89] M. Szegedy. Algebraic Methods in Lower Bounds for Computational Models with Limited Communication. *PhD Thesis*, Dep. Computer Science, Chicago Univ., December 1989.

[Wegener 91] D. Wegener. The complexity of the parity function in unbounded fan-in unbounded depth circuits. In *Theoretical Computer Science*, Vol. 85, pp. 155–170, 1991.