Linear Threshold Circuits by Vincent

Outline

1. Theory of Linear Threshold Circuits : important concepts.
2. Answer question about AND and OR (one of the "100 questions").
3. Answer question about XOR (one of the "100 questions").
4. Other areas of Research.
5. Summary.
6. Problems.

Theory of Linear Threshold Circuits

- **Boolean functions** are mappings from the hypercube $\{0,1\}^n$ to the binary set $\{0,1\}$. For a fixed dimension n there are 2^{2^n} possible functions.
- Construct bf circuits that realize those functions by combining elementary computing blocks : gates. Given a particular choice of elementary functions (and some constraints on how to combine them) we get a set of Boolean functions that it is possible to implement.
- **"Family" of functions**.
  Instead of looking at a single mapping defined for a fixed dimension, we consider a "scalable" function, that is a family of functions or more precisely a sequence of functions such that for each index $i \in N$ f_i is a Boolean function of i variables. For example AND is the family in which each function is 1 only at the all-one vertex of the hypercube. Dually, OR is the family for which each function is 0 only at the all-zero vertex of the hypercube. On the other hand NOT is a single function of one variable not a family.
- **"Function class"**.
  A class of function is a set of families of functions. For example the class AON is the set of all families of functions that can be implemented by a circuit composed of polynomially many AND, OR and NOT gates (the NOT gates are not included in the count). The key concept here is : polynomial increase; that for each family there must exist a fixed polynomial in the dimension i which is an upper bound for the number of gates used in implementing the functions in the family.
- **Relation to NP**.
  Why the polynomial constraint? Because we want to avoid circuits in which the number of components grow exponentially with the dimension of the problem. We want to study the power of circuits of polynomial size.

- **class** LT : same as AON except that the computing elements are different. An LT circuit is composed of Linear Threshold Gates. The latter output 1 only if a weighted sum of inputs exceeds a fixed threshold (neuron like behavior).

Do not worry, you don't need to know all that in order to answer Shuki's questions, but you need to be able to understand it. So let us look at the questions now.

AND, OR question

Show how to compute the Boolean functions AND and OR using a linear threshold element.
AND outputs 1 only if all inputs are 1, that is only if their sum is n. So choose the weights to be all 2 and the threshold $2n - 1$.
OR outputs 0 only if all inputs are 0 so choose the weights to be 2 and the threshold 1.

XOR question

Prove that XOR cannot be computed by a single linear threshold element.
To remind you : $XOR =$ (sum of inputs)$mod 2$. First show that XOR of two variables : XOR_2, cannot be computed by a single threshold element by writing the inequalities that weights and threshold should satisfy and showing that they are inconsistent. Next argue that any function within the XOR family can be reduced to XOR_2 by setting all except two of its inputs to 0.

Show how to compute XOR using a two-layered circuit of AND, OR and NOT gates. Is that construction optimal in size?
We have one output, therefore only one gate in the second layer. By the AND-OR duality we can assume WLOG (Without Loss Of Generality) that it is an OR gate. The next step is to note that if there is an OR gate in the first level it is possible to force the output of the circuit to 1 by setting only one of the inputs (since $(1 + x) + y = 1$ for any x and y). That will be inconsistent with the definition of the XOR function so we can safely assume that all gates in the first layer are AND gates (possibly with negated inputs but not negated outputs). We obtain the "mintern" or sum-of-products realization of a Boolean function. We know the the sum-of-products realization of XOR has 2^{n-1} terms and that it is minimal (no terms can be removed). Each term rerquires a gate plus the output gate we get $2^{n-1} + 1$ to be the size of the smallest two-layer AON circuit that implements XOR.

Other Research Areas

Above we were concerned with the size of the circuits. Another way to measure the efficiency of a computing class is the depth (number of layers or more precisely length of longest path from input to output). It has been shown that some families of functions like XOR cannot be computed by AON circuits with constant depth, while they can be computed by LT circuits of constant depth.

Another area of research is concerned with the size of the weights of LT gates. If we limit the size of the weights of the elements in an LT ciruit how much power do we loose? A common approach is to define "small" as polynomially increasing. A very recent result by Goldmann and Karpinski states that a LT circuit of depth d is less powerful than a LT circuit with small weights and depth $d+1$.

Summary

To summarize we looked at classes of Boolean functions defined by the circuits that implement them. Two examples are AON and LT, (polynomial size). Those can be further refined according to the depth or the size of the weights for LT circuits. If you have any questions or would like some more information e-mail me at vincent@cco.

Problems

1. Can you compute

   $$f(x_1, x_2, x_3, x_4) = x_1x_2 + x_3x_4 = OR(AND(x_1, x_2), AND(x_3, x_4)$$

   with a single LT element? Prove it.

2. Redefine AND :

   $$x_i \in \{0, 1, 2\}$$

   $$AND(x_1, x_2, ..., x_n) = 1 \text{ iff all } x_i > 0$$

   Can you compute it with a single LT element? Proof.

3. Design a 2-layer LT circuit that computes XOR of n variables.

4. *Open problem for 30 years :* Find the optimal design (in terms of the number of gates) for the previous problem.

5. Compare two n-bit integers using a single LT element

   $$COMPARISON(X, Y) = 1 \text{ iff } X \geq Y$$