TOPIC 4
Monotone Circuits With Real-Valued Gates

(Related sections: 10.6)

Summary: Proof that the lower bounds criterion for monotone *boolean* circuits (Theorem 10.16) holds also for monotone circuits with *real-valued* gates.

In this lecture we consider monotone circuits where, besides boolean AND and OR gates, one may use arbitrary non-decreasing *real-valued* functions $\varphi : \mathbb{R}^2 \to \mathbb{R}$ as gates. Such a function φ is *non-decreasing* if $\varphi(x_1, x_2) \le \varphi(y_1, y_2)$ whenever $x_1 \le y_1$ and $x_2 \le y_2$. The corresponding circuits are called *monotone real circuit.*

As in boolean circuits, inputs for such circuits also are binary strings $x \in \{0,1\}^n$; the output is also a binary bit 0 and 1. But each intermediate gate may compute any monotone function $f : \{0,1\}^n \to \mathbb{R}$. Hence, unlike in boolean case, here we have uncountable number possible gates $\varphi : \mathbb{R}^2 \to \mathbb{R}$, and one may expect that at least some monotone boolean functions can be computed much more efficiently by such circuits. Exercise 10.15 of the book shows that this intuition is correct: every slice function can be computed by a very small monotone circuit with real-valued gates, but easy counting shows that most of slice functions cannot be computed by boolean circuits of polynomial size (even if NOT gates are allowed!).

It is therefore somewhat surprising that the criterion, stated (in the book) for boolean circuits remains true also for circuits with real-valued gates. The only difference from the boolean case is that now in the definition of t-simplicity we allow larger CNFs and DNFs.

Definition A monotone boolean function f is *weakly t-simple* if for every pair of integers $2 \le s, r \le n/2$ there exist a $(c+1)$-CNF C consisting of at most $t \cdot (2d)^{c+1}$ clauses, a $(d+1)$-DNF D consisting of at most $t \cdot (2c)^{d+1}$ monomials, and a subset $I \subseteq \{1, \ldots, n\}$ of $|I| \le c$ indices such that either $C \le f$ or $f \le D \vee \left(\bigvee_{i \in I} x_i\right)$ (or both) hold.

Theorem 4.1 (The Criterion for Real Circuits)**.** *Let f be a monotone boolean function. If f can be computed by a monotone real circuit of size t then f is weakly t-simple.*

The criterion was proved in [**15**]. Here we will give another, more direct proof.

Proof. The idea (due to Avi Wigderson) is to consider thresholds of real gates and approximate the thresholded values. For a real-valued function $f : \{0,1\}^n \to \mathbb{R}$ and a real number t, let $f^{(t)}$ denote the boolean function that outputs 1 if $f(x) \ge t$, and outputs 0 otherwise.

For a gate $f = \varphi(g, h)$ with two input gates g and h the threshold function $f^{(t)}$ can be determined from the thresholds of its input gates as

$$f^{(t)} = \bigvee_{\varphi(s,r)\geq t} g^{(s)} \wedge h^{(r)} \tag{4.2}$$

or

$$f^{(t)} = \bigwedge_{\varphi(s,r)<t} g^{(s)} \vee h^{(r)}. \tag{4.3}$$

Proof of (4.2). We have to show that, for every $x \in \{0,1\}^n$,

$$\varphi(g(x), h(x)) \geq t \iff \exists s, r : \ g(s) \geq s, \ h(x) \geq r \ \text{ and } \ \varphi(s,r) \geq t.$$

The $\Rightarrow$ direction is trivial: just take $s = g(x)$ and $r = h(x)$. For the other ($\Leftarrow$) direction we use the monotonicity of φ: $\varphi(g(x), h(x)) \geq \varphi(s, r) \geq t$. □

Ex. 4.4. Prove (4.3).

It is convenient to think these expressions as an infinite AND and an infinite OR, respectively. However, since the number of settings $x \in \{0,1\}^n$ for input variables is finite, the real gates take only finite number of possible values, and therefore, we really need finite expressions.

As before, every threshold $f^{(t)}$ is approximated by two functions: a c-And-Or $f_0^{(t)}$ and a d-Or-And $f_1^{(t)}$. The approximators for the thresholds of the input variables are 0, 1, or the variable itself, depending on the value of the threshold; they can always represented by at most one literal and thus newer fail.

Let now $f = \varphi(g, h)$ be an intermediate gate with two input gates g and h, and suppose that, for all (finitely many!) reals s, r, the approximators for threshold functions $g^{(s)}$ and $h^{(r)}$ of its input gates are already constructed.

To construct the approximator $f_0^{(t)}$ from the approximators $g_0^{(s)}$ and $h_0^{(r)}$ of its two input gates g and h, we first consider the representation

$$f^{(t)} = \bigvee_{\varphi(s,r)\geq t} g_1^{(s)} \wedge h_1^{(r)}.$$

Since the monomials in the DNFs $g_1^{(s)}$ and $h_1^{(r)}$ have length at most d, all the subexpressions $g_1^{(s)} \wedge h_1^{(r)}$ can be turned into a single $2d$-Or-And D_t. After that we use the same procedure as before to convert this Or-And into a c-And-Or $f_0^{(t)}$ (Claim 10.17 from the book).

When forming $f_0^{(t)}$, we introduce errors as we throw away clauses that become longer than c. We want to count the number of inputs $x \in \{0,1\}^n$ such that $f^{(t)}(x) = 0$ while $f_0^{(t)}(x) = 1$ for some t, i.e., the union over t of the errors introduced in a gate by $f_0^{(t)}$. To do this, let us list in the increasing order $t_1 \leq t_2 \leq \ldots \leq t_N$ all the $N \leq 2^n$ possible values $f(x)$ the gate f can output when the input vector x ranges over $\{0,1\}^n$. Hence,

$$D := D_{t_1} \vee D_{t_2} \vee \cdots \vee D_{t_N}$$

is a $2d$-Or-And, and this Or-And makes no error on x, i.e., $D(x) = f(x)$.

By the definition,

$$D_{t_1} \geq D_{t_2} \geq \cdots \geq D_{t_N}.$$

Hence, if $\mathrm{tr}\,(D)$ denotes the family of all transversals of D (i.e., the family of all subsets of variables, each of which intersects all the monomials of D) then

$$\mathrm{tr}\,(D_{t_1}) \subseteq \mathrm{tr}\,(D_{t_2}) \subseteq \cdots \subseteq \mathrm{tr}\,(D_{t_N})\,,$$

implying that

$$\mathrm{tr}\,(D) = \mathrm{tr}\,(D_{t_N})\,.$$

This means that all the clauses (=transversals), which we throw away (because they are longer than c) when forming a c-And-Or f_0 from the Or-And D, are precisely those clauses, which we would throw away when converting the $2d$-Or-And D_{t_N} into a c-And-Or. Thus, by Claim 10.17 from the book, all the errors that may appear during the construction of the left approximator f_0, can be corrected by a $(c+1)$-CNF C of size $|C| \leq (2d)^{c+1}$. That is, for every input x such that $f(x) = 0$ but $f_0(x) = 1$, we have that $C(x) = 0$.

A dual argument can be used to bound the number of errors introduced when constructing the right approximator f_1. In this case we use the expression (4.3) instead of (4.2), and all the introduced errors can be corrected by a $(d+1)$-DNF D of size $|D| \leq (2c)^{d+1}$.

The rest of the proof is the same as that of Theorem 10.16 from the book. $\square$

Using Theorem 4.1 we can, for example, immediately extend the exponential lower bound of Alon and Boppana (see Theorem 10.20 in the book) to the case of monotone real circuits.

Theorem 4.5. *If $s \leq \sqrt{q/2\ln q}$ then any monotone circuit computing* $\mathrm{POLY}(q,s)$ *has size at least $q^{\Omega(s)}$.*

Remark. As we already mentioned at the beginning, there *exists* (not explicitly given) monotone boolean functions which can be computed by a small monotone circuits with real gates, but requires boolean (even non-monotone!) circuits of exponential size. On the other hand, in [**15**] an explicit monotone boolean function is exhibited which has a small non-monotone boolean circuit, but requires monotone real circuits of exponential size. The overall conclusion is that non-monotone boolean circuits and monotone real circuits are incomparable in their power.