## Lecture 11

# Monotone Functions, Monotone circuits, and Communication Complexity

May 6, 2008
Lecturer: Paul Beame
Notes: Widad Machmouchi

In the last lecture, we used the method of approximation to derive lower bounds on the size of any circuit that computes parity using $\mathsf{AC}[q]$ circuits. Now, we look at monotone boolean functions and derive lower bounds on the monotone complexity of the CLIQUE function using the method of approximation. We also introduce some notions from communication complexity and tie them to circuit depth and formula size.

## 11.1 Monotone functions

Let $x, y \in \{0,1\}^n$. We say $x \lesssim y$ iff $\forall i,\ x_i \leq y_i$. In other words, $x_i = 0$ whenever $y_i = 0$.

**Definition 11.1 (Monotone function)** A function $f : \{0,1\}^n \to \{0,1\}$ is *monotone* iff $x \lesssim y \Rightarrow f(x) \leq f(y)$

Every monotone Boolean function is computable in the basis $(\wedge, \vee)$. We can use monotone functions, namely *slice* functions, to compute any boolean function.

**Definition 11.2** Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function. For each $i \in [n]$, let the *slice* function $f_i(x)$ be defined as follows:

$$f_i(x) = \begin{cases} 1 & \text{if } |x| > i \\ f(x) & \text{if } |x| = i \\ 0 & \text{if } |x| < i. \end{cases}$$

Hence, $f(x) = \bigvee_{i=0}^{n} (f_i(x) \wedge (|x| = i))$.

The *slice* functions $f_i$ are monotone even if $f$ is not. Since checking whether $|x| = i$ can be done in $O(n^2)$ circuit size (actually $O(n \log n)$ size using $O(logn)$ depth networks of comparators due to Ajtai, Komlos, and Szemeredi), it's enough to find lower bounds on $size(f_i)$ to get lower bounds on $size(f)$. This helped foster the belief in the early 1980's that monotone and non-monotone complexities would be quite similar.

This belief also generated especial excitement over lower bounds on the monotone complexity $CLIQUE$ function. $CLIQUE_{n,k}$ takes $\binom{n}{2}$ inputs representing an undirected graph $G$ on $n$ nodes and outputs 1 iff there exists a clique of size $k$ in $G$. Note that $CLIQUE$ is a monotone function since adding an edge to the graph won't remove any existing cliques. We will show that $CLIQUE$ cannot be computed by polynomial-size monotone circuits.

**Theorem 11.3 (Razborov, Andre'ev, Alon-Boppana)** There exists $\epsilon > 0$ such that for all $k \leq n^{1/4}$, $size_\Omega(CLIQUE_{n,k}) > 2^{\epsilon\sqrt{k}}$, where $\Omega = (\vee, \wedge)$.

**Proof** We will use the method of approximation. Let $\ell = \sqrt{k}/10$ and $m = (100l \log n)^\ell \ell!$. Define the $(m, \ell)-$approximator $f$ as $f = \bigvee_{i=1}^m C_{S_i}$, where the *clique indicator* $C_S(G) = 1$ if $G$ has $S$ as a clique and 0 otherwise and each $S_i \subseteq [n]$ has size at most $\ell$. We will combine the approximators to form a circuit based on any monotone circuit for $CLIQUE_{n,k}$, the final function much either make a lot of errors on graphs that are just the edges of a $k$-clique or the edges of complete $(k-1)$-partite graphs but each approximation along the way will introduce relatively little error. Therefore, the circuit size must be large to accomulate such errors.

The circuit will be constructed by induction. The base case will be clique indicators for each edge. Hence if $u$ and $v$ are two nodes in $G$, $C_{\{u,v\}} = x_{uv}$. For any two $(m, \ell)-$approximators $f$ and $g$, we will approximate $f \vee g$ by an $(m, \ell)$-approximator $f \sqcup g$, and approximate $f \wedge g$ by an $(m, \ell)$-approximator $f \sqcap g$ as follows. Let $f = \bigvee_{i=1}^m C_{S_i}$ and $g = \bigvee_{j=1}^m C_{T_j}$.

- $f \sqcup g$**:** This will be the approximator of $f \vee g$. Consider $h = C_{S_1} \vee C_{S_2} \vee \ldots \vee C_{S_m} \ldots \vee C_{T_1} \vee C_{T_2} \vee \ldots \vee C_{T_m}$. $h$ is not an $(m, \ell)-$approximator since it coule be the OR of more than $m$ clique approximators. We will reduce the up to $2m$ clique indicators to only $m$ using the following *sunflower* lemma:

  **Lemma 11.4 (Sunflower lemma: Erdos-Ko-Rado)** Let $U_1, U_2, \ldots, U_r$ be distinct sets of size at most $\ell$ on a universe of size $n$. Given an integer $p$, there exists $m = (p-1)^\ell \ell!$ such that every collection of at least $m$ sets contain a sunflower of size $p$; i.e. there exist $p$ sets $U_1, U_2, \ldots, U_p$ and a set $U$, called the *core* such that $U_i \cup U_j = U$ for all $i \neq j \in [p]$. (The $U_j - U$ are called the *petals* of the sunflower.)

  As long as we have more than $m$ distinct sets, we apply the sunflower lemma to replace the collection of $p$ sets by their core. Note that we don't lose anything in the case of graphs consisting of a $k$-clique and we incur only an exponentially small loss in $p$ on complete $(k-1)$-partite graphs, since any such graph must fail to satisfy one of the $p$ petals which are disjoint from each other. This will occur at most $m$ times.

- $f \sqcap g$**:** This will be the approximator of $f \wedge g$. Consider $h = C_{S_1 \cup T_1} \vee C_{S_1 \cup T_2} \vee \ldots \vee_{S_1 \cup T_m} \vee C_{S_2 \cup T_1} \vee \ldots C_{S_2 \cup T_m} \vee \ldots \vee C_{S_m \cup T_1} \vee \ldots \vee C_{S_m \cup T_m}$. Again, $h$ is not an $(m, \ell)-$approximator since it is the OR of $m^2$ approximators of sets of size possibly $> \ell$. To reduce them, we will discard all $C_Z$ where $|Z| > \ell$ and repeatedly apply the sunflower lemma to get only $m$ disjoint sets of size $\leq \ell$. Again, this involves at most $m^2$ uses of the sunflower lemma which will cause only an exponentially small loss for $(k-1)$-partite graphs. Discarding clique indicators of size $> \ell$ may also cause a loss in the $k$-clique graphs accepted since $\ell$ is small relative to $k$ (less than $\sqrt{k}$). However, only an exponentially small fraction of all $k$-cliques contain a given $\ell'$-clique $S'$ for $\ell' > \ell$. Therefore the error is small for both the positive and negative examples.

Since each approximator incurs only a small error and the whole circuit incurs a large error, we conclude that such circuit will have a large size. The rest involves doing the calculations and choosing the parameters to balance things. we omit the details. □

It is not hard to see that the argue crucially relies on the fact that there are inputs with few 1's for which the function is 1 and inputs with many 1's for which the function is 0 – this is a property completely unlike a slice function. Also $CLIQUE$ is not the only monotone function for which this method applies. In fact, it works for monotone functions that have polynomial size non-monotone circuits.

**Theorem 11.5 (Razborov)** $size_{\Omega}(BIPARTITE - MATCHING)$ is $n^{\Omega(\log n)}$.

**Theorem 11.6 (Tardos)** There exists an $\epsilon > 0$ and a monotone function (related the the Lovasz-$\theta$ function) having polynomial circuit size over the De Morgan basis such that $size(f)$ is at least $2^{n^{\epsilon}}$.

## 11.2 Communication complexity

Let $f : X \times Y \to Z$ be a function, where $X$, $Y$ and $Z$ are sets. In the communication complexity model there are two parties, Alice and Bob, who each have a part of the input: Alice has the $X$ part and Bob has the $Y$ part. They want to communicate in a way to compute $f$.

**Definition 11.7** A *2-party communication protocol* $P$ of $f$ is a binary tree with each internal node $v$ labeled by a function $a_v : X \to \{0,1\}$ or a function $b_v : Y \to \{0,1\}$. The out-edges of an internal node $v$ are labeled by 0 or 1 and each leaf is labeled by an element of $Z$. Define *cost(P)* to be the height of the tree and *leaves(P)* to be the number of leaves of the tree. Hence $leaves(P) \leq 2^{cost(P)}$.

The communication complexity of $f$ is given by optimizing over all its communication protocols.

**Definition 11.8** The deterministic communication complexity of $f$ is given by:

$$D^{cc}(f) = \min_{P,\ P \,\text{computes}\, f} cost(P)$$

. Although it is a bit non-standard, we also define

$$L^{cc}(f) = \min_{P,\ P \,\text{computes}\, f} leaves(P)$$

We will represent each function $f$ by a matrix $M_f$ with $|X|$ rows, each representing an $x \in X$, and $|Y|$ columns, each representing a $y \in Y$. Hence, $M_f(i,j) = f(x_i, y_j)$ for all $i \in [|X|]$ and $j \in [|Y|]$.

**Definition 11.9** A *combinatorial rectangle* $R$ in $X \times Y$ is a set of the form $A \times B$ for $A \subseteq X$ and $B \subseteq Y$.

We will divide $M_f$ into "monochromatic" combinatorial rectangles, i.e. rectangles over which the function is constant.

**Lemma 11.10** A protocol $P$ computing $f$ induces a partition of $X \times Y$ into $r = leaves(P)$ combinatorial rectangles $R$ on which $f$ is constant.

**Proof** For each node $v$ in $P$, we define a rectangle $R_v$. We will define the rectangles inductively. Start with the root: $R_{root} = X \times Y$. Assume that a node $v$ is labeled by the function $a_v$ and $R_v = A_v \times B_v$. Let $s$ and $t$ be the children of $v$ such that the edge $(s, v)$ is labeled 0 and the edge $(t, v)$ is labeled 1. Then

$$R_s = (A_v \cap \{x \in X \mid a_v(x) = 0\}) \times B_v$$

and

$$R_t = (A_v \cap \{x \in X \mid a_v(x) = 1\}) \times B_v.$$

Note that $R_v = R_s \dot\cup R_t$. The case when $v$ is labeled by a function $b_v$ is analogous.

At a leaf $\ell$, $R_\ell$ will the set of $(x, y)$ such that the functions labeling the nodes along the path from the root to $\ell$ output the edge labels along the path. Since at the leaves the communication between Alice and Bob has stopped and the protocol has output the label $z$ at $\ell$, then $R_\ell$ is the set of $(x, y)$ such that $f(x, y) = z$. Moreover, the paths from the root to the leaves differ at least by one edge label, hence if $\ell_1$ and $\ell_2$ are two different leaves in $P$, $R_{\ell_1} \cap R_{\ell_2} = \emptyset$ as required. $\square$

**Example 11.11** Let $X = Y = \{0, 1\}^n$. Define the function $EQ : X \times Y \to \{0, 1\}$ by $EQ(x, y) = 1$ if $x = y$ and 0 otherwise. Then $M_{EQ}$ has 1's along the diagonal and 0 in all other entries. Hence the number of monochromatic rectangles is $> 2^n$ and $D^{cc}(EQ) \geq n + 1$ since every diagonal element must be in its own rectangle.

Let $f : X \times Y \to \{0, 1\}$ be a function and $M_f$ be its corresponding matrix. Let $R = A \times B$ such that $f(R) = 1$. Consider $M_R$, the matrix over $X \times Y$ that is 1 in the elements in $R$ and 0 otherwise. Hence, $M_f = \sum_R M_R$. Now $rank(M_R) = 1$ and using the fact that rank is subadditive – that is $rank(M_1 + M_2) \leq rank(M_1) + rank(M_2)$ – we get the following corollary.

**Corollary 11.12** For any Boolean function $f : X \times Y \to \{0, 1\}$, we have $D^{cc}(f) \geq \log(rank(M_f))$.

Note that the matrix $M_{EQ}$ is the $2^n \times 2^n$ identity matrix which has full rank which yields a lower bound of $n$.

The so-called *log-rank conjecture* tries to tie $D^{cc}(f)$ to $\log(rank(M_f))$. The conjecture is that $D^{cc}(f) = [\log(rank(M_f))]^{O(1)}$. The original form of the conjecture whas that $D^{cc}(f)$ is $O(\log(rank(M_f)))$ but this was shown to be false with the coonterexample being a function which is a recursive 3-ary tree of gates each of which if its 3 inputs are not all equal. The best upper bound is thtat $D^{cc}(f) \leq rank(M_f)$.

### 11.2.1 The Set Disjointness problem

Now, we will consider the set disjointness problem, or equivalently (and more appropriately) the set intersection problem. Let $X = Y = \{0, 1\}^n$. Define $DISJ(x, y) = \bigvee_{i=1}^n (x_i \wedge y_i)$. Looking at $x$ and $y$ as sets in $[n]$, you get that this function is 1 if and only if the sets they represent intersect. We will use the so-called *fooling set method* to prove that $D^{cc}(DISJ) \geq n$.

Let $S$ and $T$ be two different sets in $[n]$ and consider the inputs $(S, \bar{S})$, $(S, \bar{T})$, $(T, \bar{S})$ and $(T, \bar{T})$. Since $S \neq T$, either $S \not\subseteq T$ and hence $DISJ((S, \bar{T})) = 1$ , or $T \not\subseteq S$ and hence $DISJ((T, \bar{S})) = 1$.

Therefore, $(S, \bar{S})$ and $(T, \bar{T})$ cannot be in the same monochromatic rectangle or else $(S, \bar{T})$ and $(T, \bar{S})$ would also be in the same rectangle with them. Hence, you need a rectangle for each input of the form $(S, \bar{S})$, i.e there are at least $2^n$ 0-rectangles.

Now we move to computing relations in $X \times Y \times Z$. Given a function $f : \{0,1\}^n \to \{0,1\}$, define $R_f \subseteq f^{-1}(0) \times f^{-1}(1) \times [n]$ by

$$(x, y, i) \in R_f \Leftrightarrow x_i \neq y_i.$$

We can relate the communication complexity of $R_f$ to the circuit complexity of $f$.

**Lemma 11.13 (Karchmer and Wigderson)** For any Boolean function $f$, $depth(f) = D^{cc}(R_f)$ and $L(f) = L^{cc}(R_f)$.

**Proof** If you have a circuit for $f$, let Alice play the $\wedge$ gates and say which child evaluates to 0 on $x$ and let Bob do the same for the $\vee$ gates and say which child evaluates to 1. Hence we get a function of smaller depth that can differentiate between $x$ and $y$ and we can proceed by induction until we get a single bit that differentiate them.

If you have a protocol for $R_f$, you can construct by induction a circuit that computes $f$ by looking at the output of the functions that Alice and Bob compute to differentiate $x$ from $y$. $\square$

If $f$ is monotone, we modify the relation in the following way: $(x, y, i) \in R_f^m \Leftrightarrow x_i < y_i$. We get the analogous results as the lemma above.

**Lemma 11.14** Let $\Omega = \{\wedge, \vee\}$. Then $depth_\Omega(f) = D^{cc}(R_f^m)$ and $L_\Omega(f) = L^{cc}(R_f^m)$.

The following results give monotone depth complexity bounds for some monotone functions:

**Theorem 11.15 (Karchmer and Wigderson)** Let $\Omega = \{\wedge, \vee\}$. Then $depth_\Omega(PATH)$ is $\Omega(\log^2 n)$.

This proof is a direct argument.

**Theorem 11.16 (Raz and Wigderson)** Let $\Omega = \{\wedge, \vee\}$. Then $depth_\Omega(BIPARTITE-MATCHING)$ is $\Omega(n)$.

This proof goes by randomized reduction and essentially uses lower bounds on the *randomized* communication complexity of the set disjointness function. We will consider this next time.

Note how the second bound compares with the earlier results that $size_\Omega(BIPARTITE-MATCHING)$ is $n^{\Omega(\log n)}$. It is open whether the monotone size lower bound for $BIPARTITE-MATCHING$ can be improved to $2^{\Omega(n)}$.