CS 229r Information Theory in Computer Science April 19, 2016

Lecture 23

Lecturer: Jack Murtagh *Scribe: Sitan Chen*

Today: Monotone Boolean formulas, specifically threshold functions and a lower bound on formula complexity for Th_2^n via graph entropy

1 Threshold Functions

Definition 1 *A* Boolean formula *is a Boolean function $f : \{0,1\}^n \to \{0,1\}$ which can be computed by Boolean circuits which are trees, i.e. so that every input bit is read exactly once.*

We will regard Boolean functions as maps $2^{[n]} \to \{0,1\}$, where $2^{[n]}$ denotes the *power set* of $[n]$, i.e. the set of all subsets of $[n]$.

Definition 2 *A Boolean function is* monotone *if for all $S \subseteq T$, $f(S) \le f(T)$, i.e. the circuit for f has no negations (we can furthermore assume the circuit is over the* de Morgan basis *of AND/OR's).*

One way to measure the complexity of a Boolean formula is by the size of the corresponding tree circuit.

Definition 3 *The* size *of a Boolean formula is the number of gates in the corresponding circuit, including input gates.*

In this lecture we will consider a special class of monotone functions, *threshold functions*.

Definition 4 *The* threshold function *$Th_k^n : 2^{[n]} \to \{0,1\}$ is defined to output 1 if $|S| \ge k$ and 0 otherwise.*

For example, $\mathrm{Th}_1^n(S)$ is the OR function, Th_n^n is the AND function, and $\mathrm{Th}_n^{n/2}$ is MAJORITY. For a given threshold function, what is the minimum size of a formula computing that threshold function? We deal with the $k = 2$ case. Here is a naive upper bound:

Proposition 5 *size$(Th_2^n) \le O(n^2)$.*

Proof Consider the formula which is an AND of $(x_i \vee x_j)$ over all $i \neq j$. This has size $\binom{n}{2} + n + 1 = O(n^2)$ as desired. ■

We can do slightly better with a recursive construction:

Proposition 6 *size$(Th_2^n) \le O(n \log n)$.*

Proof Input $z \in \{0,1\}^n$ is concatenation $x \circ y$ for $x, y \in \{0,1\}^{n/2}$, so $\mathrm{Th}_2^n(x) = (\mathrm{Th}_2^{n/2}(x) \wedge \mathrm{Th}_2^{n/2}(y)) \wedge (\mathrm{Th}_1^{n/2}(x) \vee \mathrm{Th}_1^{n/2}(y))$. Then the size bound S_n we want is bounded inductively by $S_n \le 2S_{n/2} + O(n)$, so this gives $S_n = O(n \log n)$, specifically $S_n = 2n\lceil \log n \rceil - 1$. ■

The main theorem that we prove today is the following lower bound that almost matches this upper bound for monotone circuits (monsize is the corresponding formula complexity for monotone formulae):

Theorem 7 (K '64, NRW '90) *monsize$(Th_2^n) \ge 2\lceil n \log n \rceil - 1$.*

The proof we give in these notes is by graph entropy.

2 Graph Entropy

Definition 8 *Given $G = (V, E)$, the* graph entropy *$H(G)$ is defined by*

$$H(G) = \min_{X,Y} I(X;Y) = \min_{Y} \left[H(X) - H(X|Y)\right],$$

where X is uniform on the vertices V, and Y is a distribution over independent sets in G which contain X.

Remark Recall that an independent set of a graph is a subset of the vertices such that no edge connecting two members of the independent set exists in the graph.

Remark Motivation for definition of graph entropy due to Korner '73: Shannon already showed that usual notion of entropy correpsonds to optimal compression of symbols in stream, and Korner wanted a relaxation of this compression problem: vertices are symbols, edges indicate that vertices do need to be distinguished in final compression. Coloring the graph partitions the vertices into independent sets, and these are sets where symbols might be confused with each other.

Example 9 *The graph entropy of a graph with no edges is zero: $H(X)$ and $H(|Y)$ are both $\log n$; i.e. in your compression, you might as well just output nothing because you don't need to distinguish any symbol from another. The graph entropy of a complete graph is $H(X)$ is $\log n$: $H(X|Y) = 0$; you need to output a compression with length equal to the entropy of X because every symbol needs to be distinguished from the other.*

Example 10 *If G is the complete bipartite set $K_{n,n}$, then*

$$H(G) \leq \log(2n) - \log(n) = 1,$$

because if we knew Y, the conditional entropy of X would be entropy of uniform distribution over n vertices. More generally, if $G = K_{m,n}$, then

$$H(G) \leq \log(m+n) - \frac{m}{m+n}\log m - \frac{n}{m+n}\log n = H\left(\frac{m}{n+m}\right)$$

because with probability $m/(n+m)$, Y lands in left half, in which case X conditioned on Y is uniform over those vertices, and with probability $n/(n+m)$, Y lands in right half, etc. In other words, graph entropy of bipartite graph should be entropy of $\frac{m}{n+m}$-biased coin flip.

Proposition 11 *Graph entropy satisfies:*

1. ***Subadditivity**: For $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, if $G = G_1 \cup G_2$, then*

$$H(G) \leq H(G_1) + H(G_2).$$

2. ***Monotonicity**: For $G = (V, E)$ and $G' = (V, E')$ such that $E \subseteq E'$, $H(G) \leq H(G')$, i.e. adding edges cannot decrease entropy.*

3. ***Disjoint union**: For $G_1, ..., G_k$ connected components of G and $\rho_i = |V(G_i)|/|V|$, where $V(G_i)$ denotes the vertices in G_i, then*

$$H(G) = \sum_{i=1}^{k} \rho_i H(G_i).$$

3 Applying Graph Entropy

Let $f : 2^{[n]} \to \{0,1\}$ be monotone. Define

$$(f)_i = \{S \subseteq [n], |S| = i \mid f(S) = 1 \text{ and } \forall T \subsetneq S, \quad f(T) = 0\}.$$

Example 12 *$(Th_2^n)_5 = \emptyset$¡ while $(Th_2^n)_2 = \{\{i,j\} \mid i \neq j\}$.*

To f, we associate the graph $G = (V, E) = ([n], (f)_2)$. Consider the graphs for each gate in a monotone formula computing f. Certainly $G_{\mathrm{Th}_2^n}$ is just the complete graph on n vertices. G_{x_i} is just the singleton graph.

Let's look at the behavior of graph entropy as we go up the gates of the formula. First, an easy observation:

Observation 13 *If $f = g \wedge h$, then $G_f \subseteq G_g \cup G_h$.*

Proof Let $e = \{i,j\}$ be some edge in G_f. This belongs in G_g and/or G_h. We know $f(\{i,j\}) = g(\{i,j\}) \wedge h(\{i,j\}) = 1$; WLOG suppose $g(\{i,j\}) = 1$. Suppose $e \notin G_g$. This would only be the case if there is some proper subset $T \subsetneq \{i,j\}$ such that $g(T) = 1$. But then $f(T) = g(T) \wedge h(T) = 1$ as well, a contradiction to the fact that $e \in G_f$. ■

Corollary 14 *Suppose $f = g \wedge h$. Then by subadditivity, $H(G_f) \leq H(G_g) + H(G_h)$.*

It's less clear what we can hope for in the case of AND gates. Suppose $\{i,j\} \in G_f$. Then $\{i,j\} \in G_f \backslash G_g \cup G_h$. In fact:

Claim 15

$$\{i,j\} \in ((g)_1 \backslash (h)_1) \times ((h)_1 \backslash (g)_1).$$

Define the set on the right-hand side in the claim to be T_{gh} and we get:

Observation 16 *If $f = g \vee h$,*

$$G_f \subseteq G_g \cup G_h \cup T_{gh},$$

Corollary 17 *Suppose $f = g \vee h$. T_{gh} is bipartite, so $H(T_{gh}) \leq 1$, and thus*

$$H(G_f) \leq H(G_g) + H(G_h) + 1.$$

We conclude that every formula for Th_2^n uses at least $\lceil \log n \rceil$ AND gates. This isn't enough, so instead of graph entropy, we'll use a different potential function. For gate f, define

$$\lambda(f) = H(G_f) + \frac{|(f)_1|}{n}.$$

For example, at the leaves, $\lambda(x_i) = 1/n$, whereas at the root, $\lambda(\mathrm{Th}_2^n) = \log n$. With this, we'll be able to show that $\lambda(f)$ is actually additive not just in the OR gates, but in the AND gates as well! This gives a bound on the number of leaves in this tree of $\lceil n \log n \rceil$. Just to connect all the leaves, we need at least $\lceil n \log n \rceil - 1$ gates, giving the main theorem.

One last pair of observations before we accomplish all this:

Observation 18 *For any gate f in a minimal monotone formula computing Th_2^n, $(f)_1 = \{\{i\} \mid f(\{i\}) = 1\}$.*

Proof If this weren't the case, $f(\emptyset) = 1$, meaning f is the constant function and we can ignore the gate f and get a smaller formula for Th_2^n, a contradiction. ■

Observation 19 $(f)_1 \subseteq (g)_1 \cup (h)_1$.

Lemma 20 *If $f = g \wedge h$, then $\lambda(f) \leq \lambda(g) + \lambda(h)$.*

Proof A straightforward calculation gives:

$$\begin{aligned}\lambda(f) &= H(G_f) + \frac{|(f)_1|}{n} \\ &\leq H(G_g) + H(G_h) + \frac{|(g)_1 \cup (h)_1|}{n} \\ &\leq H(G_g) + H(G_h) + \frac{|(g)_1|}{n} + \frac{|(h)_1|}{n} \\ &= \lambda(g) + \lambda(h).\end{aligned}$$

■

Lemma 21 *If $f = g \vee h$, then $\lambda(f) \leq \lambda(g) + \lambda(h)$.*

Proof We know that $\lambda(f) \leq H(G_g) + H(G_n) + H(T_{gh}) + \frac{|(f)_1|}{n}$. But $V(T_{gh}) = \frac{|(g)_1 cup (h)_1 \backslash (g)_1 \cap (h)_1|}{n}$, and combining this with $(f)_1 \subseteq (g)_1 \cap (h)_1$ we get

$$\lambda(f) \leq H(G_g) + H(G_h) + \frac{|(g)_1 \cup (h)_1 \backslash (g)_1 \cap (h)_1}{n} + \frac{|(g)_1 \cap (h)_1|}{n} \leq \lambda(g) + \lambda(h)$$

as desired. ■