COS597D: Information Theory in Computer Science October 10, 2011

Lecture 7

Lecturer: Mark Braverman *Scribe: Ajay Roopakalu*[*]

1 Monotone Formula Size

In the last lecture, we saw that a monotone circuit is the smallest circuit that uses only OR and AND gates. Further, we also saw that the second threshold function is defined as follows.

Definition 1. *[Second Threshold Function]* $Th_2^n(x_1, \dots, x_n) = \bigvee_{1 \le i \ne j \le n}(x_i \wedge x_j) = 1$ *if and only if at least two of the bits are set to one.*

The following powerful theorem allows us to obtain a tight lower bound on the size of the circuit that evaluates the above function.

Theorem 2. *[Krichevsky '64] [Newman, Ragde, Widgerson '90]* $size_m(Th_2^n) \ge 2\lceil n \log n \rceil - 1$.

But before we can prove this theorem, we introduce some helpful notation and definitions.

Definition 3. *Let* $f : 2^{[n]} \to \{0, 1\}$ *be any monotone function. Then, we say that* $(f)_i = \{S \subseteq [n], |S| = i \mid f(S) = 1 \text{ and } \forall T \subsetneq S, f(T) = 0\}$.

Example 4.

$$(Th_2^n)_i = \begin{cases} \emptyset & , i \ne 2 \\ \{j, k\} & , j \ne k \end{cases}$$

Definition 5. *Given a graph* $G = (V, E)$, *such that* $|V| = n$, *we define*

$$CONN(E) = \begin{cases} 1 & , (V, E) \text{ is connected} \\ 0 & , \text{otherwise} \end{cases}$$

Example 6.

$$(CONN(E))_i = \begin{cases} \emptyset & , i \ne n - 1 \\ \text{all trees} & , \text{otherwise} \end{cases}$$

Proof (of Theorem 2) We will perform the analysis on the circuit by designing an "energy function" $\mu(\cdot)$ with the following properties:

- $\mu(\text{gate}) \le \mu(\text{input}_1) + \mu(\text{input}_2)$
- $\mu(x_1) = \frac{1}{n}$
- $\mu(Th_2^n) = \log(n)$

For a monotone, boolean function f, denote by G_f the graph whose edge set is $(f)_2$. We immediately note that $G_{Th_2^n} = K_n$ and $G_{x_i} =$ empty graph on n vertices. There are two types of gates (since the formula is monotone) and we handle them case by case:

[*]Based in part on lecture notes by Anup Rao, Punyashloka Biswal and Lukas Svec.

Case 1: OR Gates
We will now handle gates of the form $f = g \vee h$.

We construct the graph $G_f \subseteq G_g \cup G_h$. The graph of f is a subset of the union of the graphs of its inputs because for a each $\{i,j\}$ in $(f)_2$:

- $f(\{i,j\}) = 1$ implies that either $g(\{i,j\}) = 1$ or $h(\{i,j\}) = 1$.
- $f(\{i\}) = 0$ implies that $g(\{i\}) = h(\{i\}) = 0$
- $f(\{j\}) = 0$ means that $g(\{j\}) = h(\{j\}) = 0$.

Taken together, this means that $\{i,j\} \in (g)_2$ or $\{i,j\} \in (h)_2$. Thus, each edge in the graph of f is an edge of either the graph of g or the graph of h. But what does this tell us about graph entropy? We realize the following:

$$H(G_f) \leq H(G_g \cup G_h) \leq H(G_g) + H(G_h)$$

where the first inequality comes from the monotonicity of graph entropy and the second inequality comes from the subadditivitity of graph entropy.

Case 2: AND Gates
We will now handle gates of the form $f = g \wedge h$.

Here, it is *not* the case that $G_f \subseteq G_g \cup G_h$. (For a simple counterexample, suppose that $f = x_1 \wedge x_2$, $g = x_1$, and $h = x_2$. Then $\{1,2\} \in G_f$ but that $\{1,2\} \notin G_g \cup G_h$). But this can only happen if the gate brings together a literal from each side.

Precisely, suppose we had an edge $e = \{i,j\} \in G_f \backslash (G_g \cup G_h)$. Consider that we restrict f, g, h to only $\{i,j\}$ (just set all other coordinates to 0). Then, it follows necessarily that $f(x_i, x_j) = x_i \wedge x_j$. (Further, we know that $g(x_i, x_j) \neq x_i \wedge x_j$ and $h(x_i, x_j) \neq x_i \wedge x_j$ since otherwise the edge e would be in the respective graphs. This also means that we cannot have that either g or h are equal to constants only. Finally, we cannot have OR gates in either of the inputs.). Two options therefore remain:

$$\begin{aligned} g(x_i, x_j) = x_i \quad &\Longrightarrow \quad h(x_i, x_j) = x_j \\ g(x_i, x_j) = x_j \quad &\Longrightarrow \quad h(x_i, x_j) = x_i \end{aligned}$$

Therefore, this means that $e \in ((g)_1 - (h)_1) \times ((h)_1 - (g)_1)$. Call the induced subgraph of these edges $T_{g,h}$. What can we say about $H(T_{g,h})$? With some observation, we see that $T_{g,h}$ is bipartite! From this, we are able to observe that, using the properties of graph entropy and the fact that the graph entropy is bounded above by 1, we simplify:

$$H(G_f) \leq H(G_g \cup G_h \cup T_{g,h}) \leq H(G_g) + H(G_h) + H(T_{g,h}) \leq H(G_g) + H(G_h) + 1$$

As a side note, we also see that this is the *only* way that we are increasing the potential function. Thus, we have the following theorem as a direct consequence of the argument above:

Theorem 7. *Any monotone formula for Th_2^n must have at least $\lceil \log(n) \rceil$ AND gates.*

We can however make the bound for $H(T_{g,h})$ tighter with the observation that there are many singleton components that we haven't taken into account. Using the disjoint decomposition of graph components for entropy, we can derive the tighter bound $H(T_{g,h}) \leq \frac{|(g)_1 \cup (h)_1 - (g)_1 \cap (h)_1|}{n}$. This is the weighted average.

From this, we are able to derive the actual energy function as

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

We can again observe that $\mu(x_i) = \frac{1}{n}$ and $\mu(Th_2^n) = \log(n) + \frac{0}{n} = \log(n)$, as desired. We now need to go back to the cases and ensure that the first property ($\mu(f) \leq \mu(g) + \mu(h)$) still holds.

Case 1 (Revisited): OR Gates
This is the easy case. As noted before, we have the useful property that $G_f \subseteq G_g \cup G_h$, so that $H(G_f) \leq H(G_g) + H(G_h)$. Further, we also know that $(f)_1 \subseteq (g)_1 \cup (h)_1$ by the argumentation above. Further, we also note that f *cannot* be a constant. This is simply because we have assumed that our formula is minimal; a constant in a formula can simply be omitted to yield a smaller formula, which is a contradiction. Thus,

$$\mu(f) = H(G_f) + \frac{|(f)_1|}{n} \leq H(G_g) + H(G_h) + \frac{|(g)_1|}{n} + \frac{|(h)_1|}{n} = \mu(g) + \mu(h)$$

Case 2 (Revisted): AND Gates
We can now compute the energy function for this type of gate:

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

Thus, we conclude that we have successfully shown that this energy formula works in all cases. Therefore, we note that we need at least $\lceil n \log(n) \rceil$ leaves and therefore at least $\lceil n \log(n) \rceil - 1$ gates to a total size of at least $2\lceil n \log(n) \rceil - 1$. ■

Definition 8. *[Hamming Cube] The graph H_n with vertex set $V = \{0,1\}^n$ and edges between vertices that differ in exactly one binary location is called the Hamming Cube.*

Example 9. *The following are examples of canonical Hamming Cubes:*

- *H_1 consists of the graph of a single line between a vertex labeled 0 and a vertex labeled 1.*
- *H_2 is the graph of a square of vertices labeled $\{00, 01, 11, 10\}$ in clockwise order.*
- *H_3 is a cube vertices labeled 000 through 111.*

Through a simple counting argument, we can see that the number of edges in H_n is $\frac{2^n n}{2} = 2^{n-1}n$.

Claim 10. *For each subset of vertices in H_n, the number of edges in $S \subseteq \{0,1\}^n$ is at most $\frac{|S|\log(|S|)}{2}$.*

Proof We start by letting $X_1 X_2 \cdots X_n$ be a random element in S. Then we observe that

$$H(X_1|X_2 = x_2, X_3 = x_3, \ldots, X_n = x_n) = \begin{cases} 1 & , \exists \text{ edge in direction 1 at } x_2, \ldots, x_n \\ 0 & , \text{otherwise} \end{cases}$$

Further, we can generalize this to any edge i in a natural way:

$$H(X_i|X_1 = x_1, \ldots, X_{i-1} = x_{i-1}, X_{i+1} = x_{i+1}, \ldots, X_n = x_n)$$

It then follows that the total number of edges in direction i is given by $\frac{|S|}{2} H(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$, since each edge has a $\frac{2}{|S|}$ chance of being picked (there are two directions along which we can count). Therefore, we can also compute the total number of edges by:

$$\begin{aligned} T &= \frac{|S|}{2} \sum_{i=1}^{n} H(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n) \\ &\leq \frac{|S|}{2} \sum_{i=1}^{n} H(X_i|X_1, \ldots, X_{i-1}) \\ &= \frac{|S|}{2} H(X_1, X_2, \ldots, X_n) = \frac{|S|\log(|S|)}{2} \end{aligned}$$

where we used the fact that the entropy of a uniform element of S is $\log|S|$. ■

2 Locally Decodable Codes

In this formulation, we want to store an n-bit string X so that each X_i can be decoded using as few queries as possible.

Example 11. *The Hadamard Code is an example of such an encoding. Here the scheme used is that for $\forall x \in \mathbb{F}_2^n$, we store $\langle a, x\rangle$ for each $a \in \mathbb{F}_2^n$, where $\mathbb{F}_2^n$ denotes a field of two values and vectors of length n and $\langle \cdot, \cdot \rangle$ is the inner product.*

To decode x_i: Pick a random $y \in \mathbb{F}_2^n$. Query $\langle y, x\rangle$ and $\langle y + e_i, x\rangle$, where e_i is the vector of zeros and only a 1 in the i^{th} place. We then return the differences of these two values. This is sufficient because $\langle y + e_i, x\rangle - \langle y, x\rangle = \langle y, x\rangle + \langle e_i, x\rangle - \langle y, x\rangle = \langle e_i, x\rangle = x_i$.

The values are correlated, but each is random. That is, if <1% of the storage is corrupted, then we can still return the correct value of x_i with probability >98% (this can be shown via union bound argument).

Theorem 12. *[Samorodnitsky] Every linear 2-query locally decodable code must store $> 2^{\Omega(n)}$ bits.*

Proof Without loss of generality, to recover X_i, we must query a^j and a^k such that $a^j + a^k = e_i$. To tolerate 1% error, at least $\frac{m}{200}$ queries can be made with $a^j + a^k = e_i$. Let $S = \{a^1, a^2, \ldots, a^m\} \subseteq \{0,1\}^n$. This means that S contains $\geq \frac{m}{200}$ edges in direction i. Therefore, the total number of edges is $\geq \frac{m \cdot n}{200}$. But by the properties of the Hamming Cube, we must have that the total number of edges is $\leq \frac{m \log(m)}{2}$. It therefore follows that $\log(m) \geq \frac{n}{100} \implies m \geq 2^{\frac{n}{100}}$, as required. ■