CS6840: Advanced Complexity Theory Mar 29, 2012

Lecture 46 : Size lower bounds for AC^0 circuits computing Parity

Lecturer: Jayalal Sarma M.N. *Scribe: Dinesh K.*

THEME: Circuit Complexity
LECTURE PLAN: Proof of the one of the first separation result in circuit complexity, $Parity \notin \mathsf{AC}^0$ assuming Håstad's switching lemma

In this lecture, we shall see an important separation result in circuit complexity, $Parity \notin \mathsf{AC}^0$. That is, any circuit of constant depth having unbounded fan-in computing *Parity* must have size exponential in the number of inputs. We shall state Håstad's Switching lemma and a resulting lemma that shall be used in proving this result. The lemmas shall be proved in the subsequent lectures.

This result was first proved in 1981 by Frust-Saxe-Sipser and independently by Ajtai. This proof was later simplified by John Håstad in 1986 using conditional probabilities. We shall be seeing a simpler proof without using conditional probabilities due to Alexander Razbarov (1987). Let us start of by defining the notion of a decision tree.

1 Decision Tree

Definition 1. A decision tree is a rooted binary tree with each internal node labelled with variables and leaves labelled with one of 0 or 1.

A decision tree can also be seen as a model of computation. Consider a function $f : \{0,1\}^n \to \{0,1\}$. A decision tree can compute this function in the following manner: On input x, start with the root node v, check if the variable representing v say x_i is 0 or 1. If it is 0 go to its left child, if it is 1 go to its right child. The process is repeated till a leaf is reached and the label on the leaf will be value of $f(x)$. When we move to a node with variable x_i, we say that the variable x_i has been *queried*. An example of a decision tree computing $f(x_1, x_2, x_3) = (x_1 \vee \neg x_2 \wedge x_3)$ is shown in figure 1.

Let us now define the parameter of interest for a decision tree. One natural parameter to ask is how many queries are needed to decide the value of f for a given input x. This can be formalised as the depth of a decision tree.

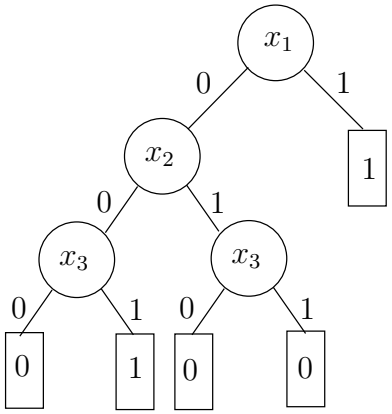

Figure 1: Decision tree computing $(x_1 \vee \neg x_2 \wedge x_3)$

1.1 Decision Tree Depth vs DNF Term Size

Definition 2. (Depth) Depth of a decision tree computing a function f is defined as the length of the longest path (in terms of the number of nodes/variables) from root to leaf. It is denoted as $|T(f)|$.

It can be observed that decision trees of boolean function f is actually representing its truth table. This leads to the following observation.

Observation 3. *Given a decision tree, a DNF (CNF) corresponding to it can be written down. Moreover the DNF (CNF) will be unique.*

Once a decision tree is given one can construct a DNF corresponding to the function in the following way: Look at the path that end in a leaf of label 1, write down the path as conjunction of terms of variables/its negation to form clauses, take an OR of all the clauses to get a DNF. Since there is a unique path between any two vertices in a tree, and in particular between root and leaf, the resultant formula will be unique. A unique CNF can also be obtained in the similar way.

What about getting a unique decision tree given a DNF of f? This is not always possible. (For example we can have two decision trees computing $(x_1 \wedge x_2) \vee (\neg x_1 \wedge x_2)$ one with x_1 appearing as the root and other with x_2 as the root.)

Hence it is necessary to define a *canonical* representation of decision trees computing f. The idea is to give a fixed ordering for the literals appearing in each clause induced by the ordering of indices of the literals. This forms a total order and is unique which enforces the uniqueness of the canonical representation.

1.2 Canonical Decision Tree

Definition 4. Canonical decision tree of a boolean function $f : \{0,1\}^n \to \{0,1\}$ can be defined inductively as follows.

Let $c_1 \vee c_2 \vee \ldots \vee c_m$ be the DNF representation of f. Let the first non-trivial clause be c_i. Let $x_{i_1}, x_{i_2}, \ldots, x_{i_r}$, where $r \leq$ term size of DNF and $i_1 < i_2 < \ldots i_r$ (the canonical ordering). Construct a full binary tree (figure 2) with all nodes at level j getting the label x_{i_j} for $1 \leq j \leq r$.

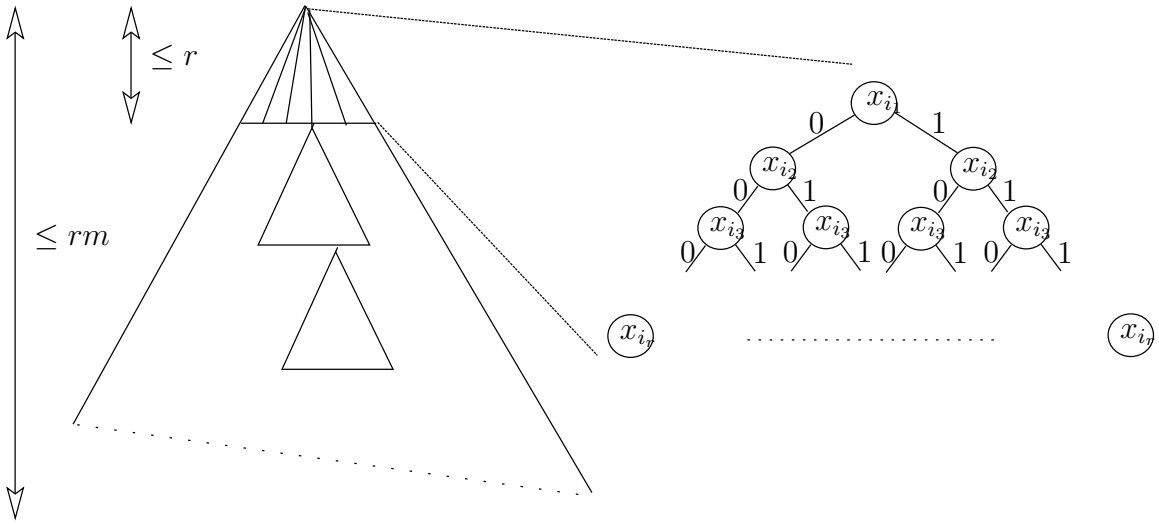

Figure 2: Canonical Decision tree

Note that some setting of the variable might have caused it it evaluate to 1. In that case, the assignment to $x_{i_1}, x_{i_2}, \ldots, x_{i_r}$ is said to have *trivialised* the function. It can also be that this assignment simplifies some other clauses also. Now for all the assignments ρ that has not trivialised f, we look at $f|_\rho$ and apply the same process.

For the base case which is 0 or 1, just attach a leaf node with appropriate label.

Since each clause can have term size at most r and since there are m clauses, depth of the canonical decision tree can be at most rm.

1.3 Decision trees for *Parity*

Let us now ask the question

What is the decision tree depth required for *Parity* ?

Claim 5. *Any decision tree computing $Parity_n$ must have a depth of n.*[1]

Proof. (Adversarial argument) Suppose there exists a decision tree of depth $< n$ computing $Parity_n$. Hence there must be a variable not being queried in every path. Hence on inputs where the variable is flipped, output will be unaffected. But for *Parity* we know that every flip of the input variable must affect the output. Hence the decision tree is not correctly computing $Parity_n$ contradicting our assumption. □

2 Exponential Size Lower Bounds for *Parity*

Here is the statement of the theorem $Parity \notin \mathsf{AC}^0$.

Theorem 6. *If a circuit C with unbounded fan-in and constant depth d is computing $Parity_n$ then, size of C must be $2^{\Omega(n^{1/(d-1)})}$. Since all AC^0 circuits are polynomial sized, $Parity \notin \mathsf{AC}^0$.*

Let us look at a high level picture of the theorem and its proof. How could one show that $Parity \notin \mathsf{AC}^0$? One way is to identify a property that *Parity* has which no circuit in AC^0 has. Just now, we saw that *Parity* has long decision trees. But this property can easily be verified to be true for $\wedge_n$ and $\vee_n$ functions by the same argument. So this property will not work directly.

But if we look at a random restriction[2] $\rho \in R_n^l$ getting applied to $\wedge_n$ or $\vee_n$ it can be seen that the chances that they do not get killed is really less ($\frac{1}{2^{n-l}}$). But *Parity* function after a restriction still remains as parity or its negation on the remaining input variables. It is this property that will help in separating *Parity* from AC^0. Let us now look at the following lemma by Håstad.

Lemma 7. ***(Håstad's Switching lemma)*** *Let F be a DNF having term size $\leq r$. Let $l = pn$, $p \leq \frac{1}{7}$. Let $\rho \in R_n^l$. Denote $T(F|_\rho)$ as the canonical tree of $F|_\rho$. Then for any $s > 0$,*

$$\mathsf{Pr}_{\rho \in R_n^l}\left[|T(F|_\rho)| \geq s\right] < (7pr)^s$$

The switching lemma captures how does the application of a random restriction simplify a DNF. Note that the above theorem also holds for F expressed in CNF. The following lemma follows from Håstad's switching lemma.

[1] Note that the class of functions that require every decision tree computing f to have depth $\geq n$ are called *Evasive functions*. Watch out for more about these functions in our course presentations!

[2] Recall $R_n^l = \{\rho : [n] \to \{0, 1, *\} \mid \text{No of variables left unassigned} = l\}$

Lemma 8. *Let $f : \{0,1\} \to \{0,1\}$, C be an AC^0 circuit computing f of size S of depth d. Define height of a gate g in C to be the maximum number of AND or OR gates from g to any input. Now, for $0 \leq i \leq d-1$, let*

$$n_{i+1} = \frac{n}{14(14 \log S)^i}$$

If $n_i \geq \log S$ for $1 \leq i \leq d$, then there exists a $\rho \in R_n^{n_i}$ such that for every gate g below height i,

$$|T(g|_\rho)| \leq \log S$$

That is, every gate below i can be computed by a decision tree of depth $\log S$.

2.1 How does this shows that $Parity \notin \mathsf{AC}^0$?

By lemma 8, we can get a proof for $Parity \notin \mathsf{AC}^0$. Here is the idea. We know that given a circuit C in AC^0, a random restriction is likely to simplify the circuit. By "simplify" we mean that the circuit after a restriction can be computed by a small depth decision tree.

What lemma 8 guarantees is the existence of such a restriction ρ for level[3] i that sets all but n_i variables. Here is how. Håstad's lemma after plugging in appropriate parameters guarantee that for a given gate g at level $\leq i$,

$$\mathsf{Pr}_\rho[\text{Depth of } T(g|_\rho) \geq \log S] < \frac{1}{S}$$

Hence by union bound,

$$\mathsf{Pr}_\rho[\exists g : \text{Depth of } T(g|_\rho) \geq \log S] < 1$$

Equivalently,

$$\mathsf{Pr}_\rho[\forall g : \text{Depth of } T(g|_\rho) < \log S] > 0$$

Thus by a probabilistic method argument, there indeed exists a ρ such that for every gate at level $\leq i$, $g|_\rho$ has depth $\leq \log S$. Now, by repeated application of the such restrictions at each level, we simplify the unbounded fan-in gates by replacing them with small depth circuits.

Without loss of generality we can always assume that the AND and OR gates of the AC^0 circuit alternates (If not combine the gates of same type appearing at consecutive levels. This is always possible since we allow unbounded fan-in).

At level i, we apply the lemma and get a decision tree of depth $\log S$. We replace the gates by equivalent DNFs. Say for the gates at level i, the parent of the gates are OR gates. Since DNF will have a OR at the top, these two can be merged resulting in decrease in depth by 1.

[3] We will be using level and height interchangeably.

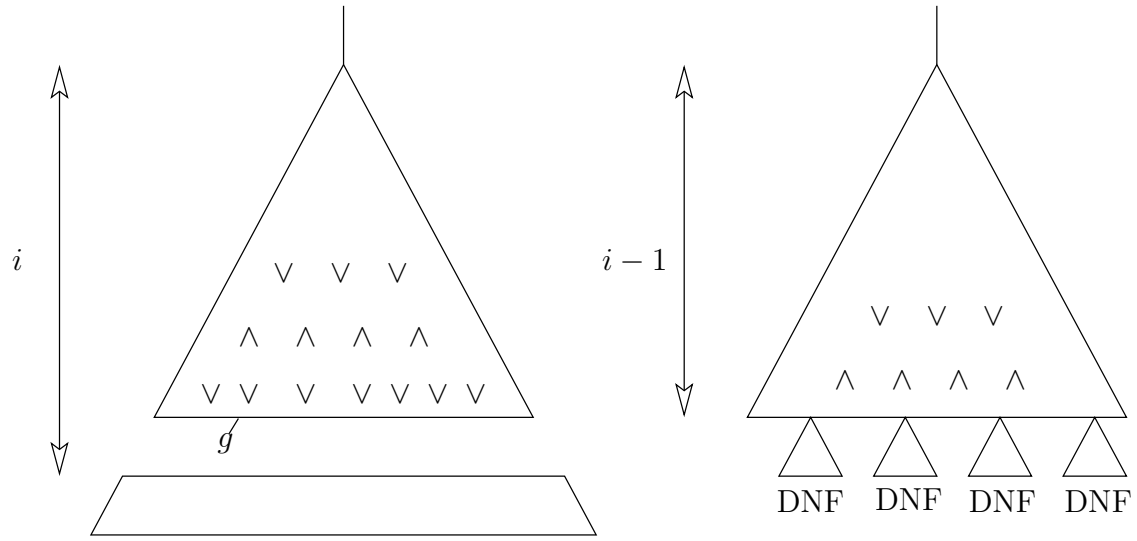

Figure 3: Application of Lemma 8

Suppose the parent gate is an AND what can be done? We replace it the CNF corresponding to the decision tree. Since, switching lemma holds for CNFs also, we can safely do this. Now again, there will be two AND gates appearing at consecutive levels which can be merged to get a decrease in height (Essentially we are "switching" between the two normal forms as we go up the levels). Applying the lemma $d-2$ times we end up in a decision tree of depth $O(\log S)$ that looks at n_{d-1} variables. Existence of such a restriction ρ achieving this is also guaranteed. Now, plug in the back the equivalent circuit. Since F computes parity, $F|_\rho$ that looks at n_{d-1} variables will still be computing the parity function (or its negation). But we know that parity function requires a decision tree of depth $\geq n_{d-1}$. Hence, we have

$$\log S \geq n_{d-1}$$
$$\log S \geq \frac{n}{14(14 \log S)^{d-2}}$$

giving us,

$$S \geq 2^{\frac{1}{14}\left(\frac{n}{14}\right)^{1/(d-1)}}$$

or $S = 2^{\Omega(n^{(1/(d-1))})}$ proving our main theorem 6.