CS6840: Advanced Complexity Theory March 12 2012

Lecture 37 : Uniformity of Circuits

Lecturer: Jayalal Sarma M.N. *Scribe: Princy Lunawat*

LECTURE PLAN: In this lecture, we complete the proof of $\mathsf{P\backslash poly} = \mathsf{PSIZE}$. We talk about a special class of polynomial sized circuit families called P-Uniform circuits and characterize the languages associated with these circuit families. We finally prove that the class of such languages is actually P.

Proof of $\mathsf{PSIZE} = \mathsf{P\backslash poly}$

Recall the definition of PSIZE and $\mathsf{P\backslash poly}$ from the previous lectures.

Definition 1.

$$\mathsf{PSIZE} = \bigcup_{k \geq 0} \mathsf{SIZE}(n^k)$$

It is the class of all those languages that can be computed by a family of polynomial sized circuits.

Definition 2. A language L is in $\mathsf{P\backslash poly}$ if there exists a polynomial sized advice string $h : \mathbb{N} \to \{0,1\}^*$, there exists a language $B \in \mathsf{P}$ such that,

$$x \in L \iff (x, h(|x|)) \in B$$

In the last lecture we saw the forward containment, that is, $\mathsf{PSIZE} \subseteq \mathsf{P\backslash poly}$. The idea was to provide as an advice string, for a given in put x, the circuit $C_{|x|}$. The resultant machine takes the input x and the advice $C_{|x|}$ and checks whether the circuit evaluates to true on in put x. Observe that this computation is essentially a *Circuit Value Problem* instance which is in P. We now prove the reverse containment , that is $\mathsf{P\backslash poly} \subseteq \mathsf{PSIZE}$.
Given a language $L \in \mathsf{P\backslash poly}$, and an input string x, the computation is deterministic when we have the associated advice $h(|x|)$. Also, the no. of configurations in the computation is is polynomial in size since the length of h is polynomial in $|x|$. Each bit of a configuration in the deterministic computation is uniquely determined by exactly 3 bits of the previous configuration. Hence, for each bit i of a configuration j, we find a circuit that takes the three bits determining bit i from the configuration $j-1$. Of course, the size of this circuit is constant. We repeat this for each bit in each configuration of the run. Observe that this circuit depends only on the length of x and not the input x itself. We are essentially

hardwiring $h(|x|)$ into this circuit to obtain $C_{|x|}$. Hence, we get a family of polynomially sized circuits computing L.

One subtlety that needs to be taken care of is the input to the circuit. The input will be the start configuration of the machine B along with the advice h. Depending upon the advice for a particular string length, a different circuit is formed. This leads to what is referred to as *non-uniformity* in circuits. Circuits of a family deciding the same language may differ in size and depth depending on the advice. Since, the family in infinite, it is not possible to describe these non-uniform ciruits in polynomial time or space.

1 Uniform Circuits

To address this issue,let us find the source of the non-uniformity in our previous construction.Given an input x, we do the following:

1. Obtain the description of $C_{|x|}$ using the initial configuration, the input length *and the advice* $h(|x|)$.
2. Run the circuit on x.

Non- uniformity in the circuits is induces due to step 1, since the advice varies with the string length. We look at a specific class of circuits where this is eliminated.

1.1 P-Uniform Circuits

Consider the class Pof languages that can be decided in polynomial time. We know that $\mathsf{P} \subseteq \mathsf{P\backslash poly}$. The natural question is to ask for an advice string an input x for a language $L \in P$. by definition of $\mathsf{P\backslash poly}$, observe that once an advice is given, the remaining computation needs to be done in polynomial time. Since, L itself can be decided in polynomial time, we donot require an advice. In other words, any trivial advice works for deciding all inputs x. Hence, in the above step 1, the circuit $C_n$can be described depending only on the input length n. Such a family of circuits is said to be $P - Uniform$.

Definition 3. A circuit family $C_n, n \geq 0$ is said to be $P - Uniform$ if there exists a polynomial time determinitic turing machine which on in put 1^n outputs the description of C_n.

Claim 4. *The languages computed by* $P - Uniform$ *circuit families can be decided in* P.

Proof. Let us denote the class of languages decided by $P-Uniform$ circuits by UPSIZE. By our earlier discussion, we know that any language in P(without advice) can be computed by a $P-Uniform$ circuit family. Hence $P \subseteq \mathsf{UPSIZE}$. We now have to prove that $\mathsf{UPSIZE} \subseteq \mathsf{P}$. Consider a language $L \in \mathsf{UPSIZE}$. Let M be the polynomial time TM that outputs the description of C_n on input n. We construct a new machine N to decide L as follows:
N on input x,

1. Run M on $|x|$ to obtain $C_{|x|}$.
2. Check if $(C_{|x|}, x) \in \mathsf{CVP}$. If yes, accept x, else reject.

Machine M runs in polynomial time in step 1 and the checking in step 2 takes polynomial time since $\mathsf{CVP} \in \mathsf{P}$(Circuit Value Problem). Hence, $L(N) \in P$. Also, since N accepts only those strings for which the corresponding circuit of the circuit family of L outputs 1. Hence $L(N) = L$. Hence $\mathsf{UPSIZE} \subseteq \mathsf{P}$. □

Now that we know that $\mathsf{UPSIZE} = \mathsf{P}$, the natural question to ask if $\mathsf{NP} = \mathsf{UPSIZE}$. That is, if there exists a $P - Uniform$ circuit deciding SAT. Aiming to prove this is intuitively more tough that proving if there is any polynomial sized circuit at all deciding SAT. This goes well with our turing machine world where proving that $\mathsf{SAT} \in \mathsf{P\backslash poly}$ is intuitively an easier task than proving if $\mathsf{SAT} \in \mathsf{P}$, since we know that $\mathsf{P} \subseteq \mathsf{P\backslash poly}$.