CSC 2401 Introduction to Complexity — Fall 2001
Lecture #16, 6 November 2001
Lecturer: Allan Borodin
Scribe: Hyonho Lee

Lecture #16

Last Time: Generalization of NP characterization

Theorem 1 (General NP characterization) *$L \in N^P_{k+1}$ if and only if there exists a set $B \in \Pi^P_k$ and a polynomial fuction $q(n)$ such that,*
$L = \{\ x \mid \exists y,\ |y| \leq q(|x|)$ *and* $< x, y > \in B\}$

This is just a generalization of the alternative NP definition. The proof is also similar but a little bit harder.

Alternative Characterization

Theorem 2 (Corollary of the previous theorem) *$L \in \Sigma^P_{k+1}$ if and only if there exists a set $B \in P$ and a polynomial function $p(n)$ such that,*

$$
\begin{aligned}
L = \{\ x \mid\ & \exists y_1,\ |y_1| \leq p(n), \\
& \forall y_2,\ |y_2| \leq p(n), \\
& \exists y_3,\ |y_3| \leq p(n), \\
& \ldots, \\
& Q_{k+1} y_{k+1},\ |y_{k+1}| \leq p(n), \\
& \textit{and}\ < x, y_1, y_2, \ldots, y_{k+1} > \in B\}
\end{aligned}
$$

$$
\textit{where } Q_{k+1} = \begin{cases} \forall, & \textit{if } k \textit{ is odd.} \\ \exists, & \textit{if } k \textit{ is even.} \end{cases}
$$

Note that as a consequence, a language $L \in \Pi^P_{k+1}$ can also be expressed in terms of alternating quantifiers but now starting with a universal quantifier.

We can prove the $\Rightarrow$ direction of this theorem by induction. First, we know it is true for Σ^P_1. Now, assume that $L \in \Sigma^P_{k+1}$. Then, by the previous theorem, there exists $B \in \Pi^P_k$ such that $L = \{\ x \mid \exists y_1,\ |y_1| \leq q(|x|)$ and

$< x, y_1 > \in B\}$. Also, by the induction hypothesis, $< x, y_1 > \in B$ if and only if there exists a set $\widetilde{B} \in P$ such that

$$\begin{aligned} B = \{ < x, y_1 > \mid & \forall y_2,\ |y_2| \leq p(n), \\ & \exists y_3,\ |y_3| \leq p(n), \\ & \ldots, \\ & Q_{k+1} y_{k+1},\ |y_{k+1}| \leq p(n), \\ & and\ \widetilde{B}(< x, y_1, y_2, \ldots, y_{k+1} >)\} \end{aligned}$$

Here we assume an encoding such that
$< x, y_1, y_2, \ldots, y_{k+1} > = << x, y_1 >, y_2, \ldots, y_{k+1} >$ Then, our theorem follows.

Some Examples

1. $L = \{\ F \mid F$ is a satisfiable formula with a unique satisfiable assignment $\}$

   $F \in L$ if and only if $\exists \sigma, \forall \sigma'$, [σ satisfies F (which is polytime computable) and $\sigma' \neq \sigma \Rightarrow \sigma'$ does not satisfy F]

   $$\therefore\ L \in \Sigma_2^P$$

2. $L = \{\ C \mid$ a circuit C computes a boolean function f and $\forall C'$, if $|C'| < |C|$, then $C' \neq C\}$

   $C \in L$ if and only if $\forall C', \exists x$, [$|C'| < |C| \Rightarrow C'(x) \neq C(x)$ (which is polytime computable)]

   $$\therefore\ L \in \Pi_2^P$$

3. $L =$ minDNF $= \{(F, k) \mid F$ is a formula and $\exists$ equivalent DNF with less than k literal occurrences $\}$

   Then, L is Σ_2^p-complete (say with respect to $\leq_p$)

   $(F, k) \in L$ if and only if $\exists F', \forall \sigma, [F'$ is DNF with less than k literal occurrences and $F(\sigma) = F'(\sigma)]$

There are complete sets for each Σ class. In particular, there are quantified versions of the SAT problem that are complete for their respective class. However, minDNF seems perhaps more natural.

$$\Sigma_k^P \cup \Pi_k^P \subseteq \Delta_{k+1}^P \subseteq \Sigma_{k+1}^P \cap \Pi_{k+1}^P$$

$$\begin{tikzcd}
 & \Sigma_1^P \arrow[dr, dash] & & \Sigma_2^P \arrow[dr, dash] & & \Sigma_k^P \arrow[dr, dash] & & \Sigma_{k+1}^P \arrow[dr, dash] & \\
P \arrow[ur, dash] \arrow[dr, dash] & & \Delta_2^P \arrow[ur, dash] \arrow[dr, dash] & & \cdots \arrow[ur, dash] \arrow[dr, dash] & & \Delta_{k+1}^P \arrow[ur, dash] \arrow[dr, dash] & & \cdots \\
 & \Pi_1^P \arrow[ur, dash] & & \Pi_2^P \arrow[ur, dash] & & \Pi_k^P \arrow[ur, dash] & & \Pi_{k+1}^P \arrow[ur, dash] &
\end{tikzcd}$$

Collapsing the PH?

For every $k \geq 1$, $\Sigma_k^P = \Pi_k^P \Longrightarrow PH = \Sigma_k^P$. In particular,
$NP = \Sigma_1^P = \Pi_1^P = \text{co-}NP \Longrightarrow PH = \Sigma_1^P = NP$.
Also, $\Sigma_k^P = \Delta_k^P \Longrightarrow PH = \Delta_k^P$, and in particular, $P = NP \Longrightarrow PH = P$.
Proof: Assume $\Sigma_k^P = \Pi_k^P$. Show $\Sigma_k^P = \Sigma_{k+1}^P$. (Then, This
$\Longrightarrow \Sigma_{k+1}^P = \Pi_{k+1}^P \Longrightarrow \Sigma_{k+2}^P = \Pi_{k+2}^P = \Sigma_k^P \cdots$)
Let $L \in \Sigma_{k+1}^P$.
Then, there exists $B \in \Pi_k^P = \Sigma_k^P$ such that $L = \{\ x \mid \exists y,\ |y| \leq q(|x|)$ and
$< x, y > \in B\}$
That is, $x \in L$ if and only if $\exists y_1, < x, y_1 > \in B$.
So, $< x, y_1 > \in B$ if and only if
$\exists y_2, \ldots, Q_k y_k, \widehat{B}(x, y_1, \ldots, y_k), where \widehat{B} \in P$.
That is, $x \in L$ if and only if $\exists y_1, \exists y_2, \forall y_3, \ldots, \widehat{B} \in P$. (
$= \exists < y_1, y_2 >, \forall y_3, \ldots, \widehat{B} \in P$.
So, $\exists y_1 \forall y_2$ collapses to $\exists < y_1, y_2 >$.

Chapter 6: Circuit Complexity

$\forall n$ every boolean function f on n variables $f : \{0,1\}^n \to \{0,1\}$ is computed by a boolean circuit C with gates $\wedge, \vee, \neg$, all negations at inputs, and $|C|$ (the number of gates) $\leq 3 \cdot 2^n$.

It is easy to see because $F(x_1, \ldots, x_n) = (x_1 \wedge F(1, x_2, \ldots, x_n)) \vee (\bar{x}_1 \wedge F(0, x_2, \ldots, x_n))$. Let $CMAX(n) =$ the maximun number of gates needed to compute any boolean function of n variables. Then, $CMAX(n) \leq 2 \cdot CMAX(n-1) + 3$ from which the bound follows.

In fact, $\forall n, CMAX(n) \leq (1 + O(\frac{1}{\sqrt{n}})\frac{2^n}{n}$ (this is not so easy to prove) and $CMAX(n) \geq \frac{2^n}{n}$.

Idea for a lower bound is counting argument. (Counting arguments are the analogue of diagonalization.) There are $2^{(2^n)}$ distinct boolean fuctions on n variables. (2^n possible input variable settings for which we assign one of two values.) A boolean function on n variables can be thought as a string on 2^n bits.

How many different circuits with t gates?

$$[\underbrace{2}_{\text{gate types}} \cdot (t + \underbrace{2n}_{\text{number of positive and negated input vars}} + 2)^2]^t.$$

That is, there are approximately t^t circuits and $2^{(2^n)}$ functions. Taking logs we get an $\Omega(2^n/n)$ lower bound on t.

In fact "most" boolean function require circuit size $(1-\epsilon)\frac{2^n}{n}$.