E0224 Computational Complexity Theory **Indian Institute of Science, Bangalore**
Fall 2014 **Department of CSA**

Lecture 13: September 17, 2014

Lecturer: Chandan Saha *Scribe: Sumant Hegde*

13.1 Introduction

Previously we were introduced to the Polynomial Hierarchy(PH). In this lecture we study properties of the PH in detail.

13.1.1 Recap: Classes Σ_i^p, Π_i^p and PH

Definition: A language L is in Σ_i^p if there is a polynomial q and a deterministic polynomial time machine M such that

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} ... Q_i u_i \in \{0,1\}^{q(|x|)} \; M(u_1, u_2, ..., u_i) = 1$$

where Q_i is $\exists$ if i is odd and $\forall$ if i is even. (This interpretation of Q_i is implicit in the rest of the notes. Also, for brevity we may write just $Q_i u_i$ instead of $Q_i u_i \in \{0,1\}^{q(|x|)}$.)
Definition: $\Pi_i^p = \{L : \overline{L} \in \Sigma_i^p\} \;\; \forall i \in \mathbb{N}$
Definition: $\mathsf{PH} = \cup_{i \geq 1} \Sigma_i^p \;\; \forall i \in \mathbb{N}$

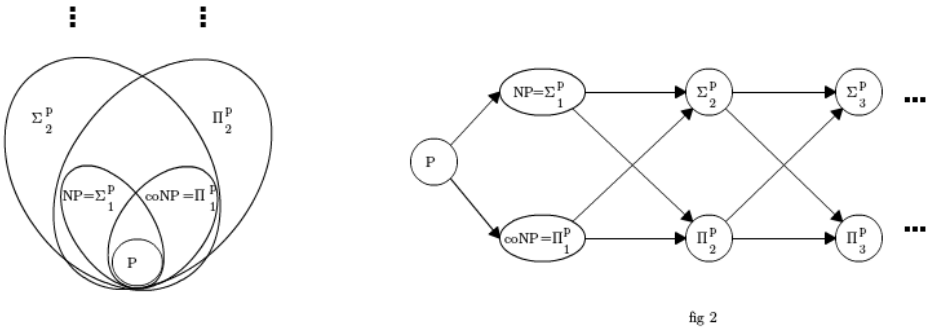
Observations:

- $\Pi_i^p = co\Sigma_i^p \;\; \forall i \in \mathbb{N}$
- $\mathsf{PH} = \cup_{i \geq 1} \Pi_i^p$
- $\Sigma_i^p \subseteq \Sigma_{i+1}^p \;\; \forall i \in \mathbb{N}$
- $\Pi_i^p \subseteq \Sigma_{i+1}^p \;\; \forall i \in \mathbb{N}$
- $\Sigma_i^p \subseteq \Pi_{i+1}^p \;\; \forall i \in \mathbb{N}$
- $\mathsf{PH} \subseteq \mathsf{PSPACE}$

13.2 Properties of the Polynomial Hierarchy

13.2.1 PH collapse

We say that the PH *collapses* to level i if $\mathsf{PH} = \Sigma_i^p$. In other words, if PH collapses to some level then a finite number of quantifiers (followed by a poly-time computable predicate) are sufficient to define all the languages in PH. It is conjectured that the PH does not collapse to any finite level i as shown in the figures below. More the number of (alternating) quantifiers, more the expressive power.



Next we prove the equivalence "PH collapses to level i" $\iff$ $\Sigma_i^p = \Sigma_{i+1}^p$ $\iff$ $\Sigma_i^p = \Pi_i^p$. Lemma 1.a proves that $\Sigma_i^p = \Sigma_{i+1}^p \implies \Sigma_i^p = \Pi_i^p$ while lemma 1.b proves that if $\Sigma_i^p = \Pi_i^p$ then PH collpses to level i, which by definition implies $\Sigma_i^p = \Sigma_{i+1}^p$. Lemma 1.c proves a corollary: if $\mathsf{P} = \mathsf{NP}$ then PH collapses to P.

Lemma 1.a: $\Sigma_i^p = \Sigma_{i+1}^p$ implies that $\Sigma_i^p = \Pi_i^p$
Proof: Since $\Sigma_i^p = \Sigma_{i+1}^p$, the observation $\Pi_i^p \subseteq \Sigma_{i+1}^p$ (see (13.1.1)) can be rewritten as

$$\Pi_i^p \subseteq \Sigma_i^p \tag{13.1}$$

Since there is a one-to-one correspondence between languages in a class and languages in its co-class, (13.1) implies

$$\begin{aligned} co\Pi_i^p &\subseteq co\Sigma_i^p \\ \Sigma_i^p &\subseteq \Pi_i^p \end{aligned} \tag{13.2}$$

From (13.1) and (13.2) we have $\Sigma_i^p = \Pi_i^p$. □

Lemma 1.b: If $\Sigma_i^p = \Pi_i^p$ then $\mathsf{PH} = \Sigma_i^p$
Proof: We prove that $\Sigma_j^p = \Sigma_i^p \ \ \forall j \geq i$, by doing induction on j.
Hypothesis: $\Sigma_j^p = \Sigma_i^p$
Base case: $j = i$: obviously $\Sigma_i^p = \Sigma_i^p$
Goal: We would like to show that $\Sigma_{j+1}^p = \Sigma_i^p$.
Let L be a language in Σ_{j+1}^p. Then there exists a polynomial-time function q and a polynomial-time machine M such that

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} \cdots Q_{j+1} u_{j+1} \in \{0,1\}^{q(|x|)} \ M(x, u_1, u_2, \cdots, u_{j+1}) = 1 \tag{13.3}$$

Define a new language L' as follows.

$$\langle x, u_1 \rangle \in L' \iff \forall u_2 \in \{0,1\}^{q(|x|)} \exists u_3 \in \{0,1\}^{q(|x|)} \cdots Q_{j+1} u_{j+1} \in \{0,1\}^{q(|x|)} \ M(x, u_1, u_2, \cdots, u_{j+1}) = 1$$

(Here we are treating $\langle x, u_1 \rangle$ as one string belonging to L'.)
Clearly, L' is defined by a boolean formula of j alternating quantifiers, starting with $\forall$. Therefore $L' \in \Pi_j^p$.

Now,

$$\begin{aligned} & L' \in \Pi_j^p \\ \implies & L' \in co\Sigma_j^p \\ \implies & L' \in co\Sigma_i^p \text{ (from induction hypothesis)} \\ \implies & L' \in \Pi_i^p \\ \implies & L' \in \Sigma_i^p \text{ (given } \Pi_i^p = \Sigma_i^p) \end{aligned}$$

which means there exists a polynomial-time machine M' such that

$$\langle x, u_1 \rangle \in L' \iff \exists v_1 \in \{0,1\}^{q(|x|)} \forall v_2 \in \{0,1\}^{q(|x|)} \cdots Q_i v_{i+1} \in \{0,1\}^{q(|x|)} \; M'(x, u_1, v_1, v_2, \cdots, v_i) = 1$$

Plugging the above expression in (13.3),

$$\begin{aligned} x \in L & \iff \exists u_1 \exists v_1 \forall v_2 \exists v_3 \cdots Q_i v_i M'(x, u_1, v_1, v_2, \cdots, v_i) = 1 \\ & \iff \exists u_1 v_1 \forall v_2 \exists v_3 \cdots Q_i v_i M'(x, u_1 v_1, v_2, \cdots, v_i) = 1 \end{aligned}$$

From the last step $L \in \Sigma_i^p$. Thus $\Sigma_{j+1}^p = \Sigma_i^p$. □

Lemma 1.c: If $\mathsf{P} = \mathsf{NP}$ then $\mathsf{PH} = \mathsf{P}$
Proof: We prove that $\Sigma_i^p = \mathsf{P} \;\; \forall i \in \mathbb{N}$, by doing induction on i.
Hypothesis: $\Sigma_i^p = \mathsf{P}$
Base case: $\Sigma_1^p = \mathsf{NP} = \mathsf{P}$ (given)
Goal: We would like to show that $\Sigma_{i+1}^p = \mathsf{P}$.
Let L be a language in Σ_{i+1}^p. Then there exists a polynomial-time function q and a polynomial-time machine M such that

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} \cdots Q_{i+1} u_{i+1} \in \{0,1\}^{q(|x|)} \; M(x, u_1, u_2, \cdots, u_{i+1}) = 1 \quad (13.4)$$

Define a new language L' as follows.

$$\langle x, u_1 \rangle \in L' \iff \forall u_2 \in \{0,1\}^{q(|x|)} \exists u_3 \in \{0,1\}^{q(|x|)} \cdots Q_{i+1} u_{i+1} \in \{0,1\}^{q(|x|)} \; M(x, u_1, u_2, \cdots, u_{i+1}) = 1$$

Clearly, $L' \in \Pi_i^p$. Now,

$$\begin{aligned} & L' \in \Pi_i^p \\ \implies & L' \in co\Sigma_i^p \\ \implies & L' \in co\mathsf{P} \text{ (from induction hypothesis)} \\ \implies & L' \in \mathsf{P} \end{aligned}$$

Thus, there exists a polynomial-time machine M' that decides L'. Substitute M' in (13.4):

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} M'(x, u_1) = 1$$

But this implies $L \in \mathsf{NP}$, from the very definition of NP. Further, as $\mathsf{P} = \mathsf{NP}$ (given), we have $L \in \mathsf{P}$. Thus, $\Sigma_{i+1}^p = \mathsf{P}$. □

13.2.2 Σ_i^p completeness

As always, the definitions of completeness and reduction are guided by the complex-theoretical question we are interested in. When defining PSPACE-completeness, we asked "Is PSPACE = P?". Knowing that $\Sigma_i^p \in \mathsf{PSPACE}$ for all i, the natural question we now ask is: "Is $\Sigma_i^p = \mathsf{P}$?". Accordingly, we are interested in polynomial-time reduction.

Definition. A language L is Σ_i^p-hard if for every $L' \in \Sigma_i^p$, $L' \leq_p L$. Further, L is Σ_i^p-complete if $L \in \Sigma_i^p$.

Now we need to give an example for a Σ_i^p-complete problem in general (i.e., for any i). For this let us examine some special cases that we are already familiar with.
We have often said, "CNFSAT = { An unquantified formula φ such that $\exists x \in \{0,1\}^* \varphi(x) = 1\}$ is NP-complete". We could convey the same fact by saying "CNFSAT = {A *true* quantified boolean formula of the form $\exists x \in \{0,1\}^* \varphi(x)$, where φ is an unquantified boolean formula} is Σ_1^p-complete". Similarly, the statement "TAUTOLOGY = { An unquantified formula φ such that $\forall x \in \{0,1\}^* \varphi(x) = 1\}$ is coNP-complete" can by all means be rephrased as "TAUTOLOGY = {A *true* quantified boolean formula of the form $\forall x \in \{0,1\}^* \varphi(x)$, where φ is an unquantified boolean formula} is Π_1^p-complete". Notice that, in the rephrased version of the statement of a complete problem (of some class in the polynomial hierarchy), we are always defining a set of "all possible true quantified boolean formulas," of some form. The form is determined essentially by the definition of the class. We capture the idea formally, by first defining $\Sigma_i\mathsf{QBF}$ and $\Sigma_i\mathsf{SAT}$ as follows.

$$\begin{aligned}\Sigma_i\mathsf{QBF} =& \{\exists x_1 \forall x_2 \exists x_3 \cdots Q_i x_i \varphi(x_1, \cdots, x_i) \text{ where } \varphi(x_1, \cdots, x_i) \text{ is an unquantified boolean formula}\}\\ \Sigma_i\mathsf{SAT} =& \{\text{true } \Sigma_i\mathsf{QBF}\}\end{aligned}$$

$\Sigma_i\mathsf{SAT}$ is in Σ_i^p. To elaborate, let $y = \exists x_1 \forall x_2 \cdots Q_i x_i \varphi(x_1, \cdots x_i)$ be a QBF. Also, think of a DTM M such that $M(y, x_1, \cdots, x_i) = 1$ if and only if $\varphi(x_1, \cdots, x_i) = 1$. (That is, M first parses y and extracts the boolean formula φ. M is also supplied the assignments $x_1, \cdots, x_i$, using which it evaluates φ, in poly-time, and simply outputs the answer.) Now $y \in \Sigma_i\mathsf{SAT} \iff \exists x_1 \forall x_2 \cdots Q_i x_i \varphi(x_1, \cdots, x_i) = 1 \iff \exists x_1 \forall x_2 \cdots Q_i x_i M(y, x_1, \cdots, x_i) = 1$. From the last step, $\Sigma_i\mathsf{SAT}$ is in Σ_i^p.

Claim: $\Sigma_i\mathsf{SAT}$ is Σ_i^p-complete.
Proof: The language is in Σ_i^p as we have just seen. Hence it suffices to show that for any language $L \in \Sigma_i^p$, $L \leq_p \Sigma_i\mathsf{SAT}$ is true. We know the following about L: There exist a poly-time TM M and a polynomial q such that

$$\forall x \in \{0,1\}^* \;\; x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} \cdots Q_i u_i \in \{0,1\}^{q(|x|)} \; M(x, u_1, u_2, \cdots, u_i) = 1 \tag{13.5}$$

Our goal is to define poly-time reduction function f such that $x \in L \iff f(x) \in \Sigma_i\mathsf{SAT}$. Noticing the alternating sequence of quantifiers in variables $u_1, \cdots, u_i$ in (13.5), we realize that if we can somehow convert the execution $M(x, u_1, \cdots, u_i)$ into a boolean formula φ_x in variables $(u_1, \cdots, u_i)$, then we are done, as we can then argue as follows.

$$\begin{aligned}x \in L &\iff \exists u_1 \forall u_2 \cdots Q_i u_i \varphi_x(u_1, \cdots, u_i) = 1\\ &\iff \exists u_1 \forall u_2 \cdots Q_i u_i \varphi_x(u_1, \cdots, u_i) \text{ is a true } \Sigma_i\mathsf{QBF}\\ &\iff f(x) = \exists u_1 \forall u_2 \cdots Q_i u_i \varphi_x(u_1, \cdots, u_i),\ f(x) \text{ is poly-size, } f \text{ is poly-time w.r.t } x.\end{aligned}$$

The size of $\varphi_x(u_1, \cdots, u_i)$ is indeed polynomial in x, and the conversion from M to φ_x is poly-time in x: this directly follows from the Cook-Levin theorem (Lecture 4).

13.2.3 PH-completeness

While classes Σ_i^p and Π_i^p (for all i) have complete problems, PH does not have one, under the assumption that PH does not collapse.
Claim: If there exists a PH-complete problem, then there exists an i such that $\mathsf{PH} = \Sigma_i^p$ (i.e., PH collapses to level i).
Proof: Suppose there exists a PH-complete language L. Then there exists some i such that $L \in \Sigma_i^p$, as $\mathsf{PH} = \cup_{i \geq 1} \Sigma_i^p \quad \forall i \in \mathbb{N}$. This implies that $L \leq_p \Sigma_i\mathsf{SAT}$ since $\Sigma_i\mathsf{SAT}$ is Σ_i^p-complete. But then, since all the languages in PH can be reduced to L, from the transitivity property of reduction we have $L' \leq_p \Sigma_i\mathsf{SAT} \quad \forall L' \in \mathsf{PH}$. In other words, $L' \in \Sigma_i^p \quad \forall L' \in \mathsf{PH}$. (This is because any language that reduces in poly-time to a language in Σ_i^p is in turn in Σ_i^p.) This, together with the fact that $\Sigma_i^p \subseteq \mathsf{PH}$, shows $\mathsf{PH} = \Sigma_i^p$. □

13.3 Defining polynomial hierarchy using oracle machines

A NTM with access to oracle-$\Sigma_i\mathsf{SAT}$ is able to decide exactly all languages in Σ_{i+1}^p.
Claim: $\Sigma_i^p = \mathsf{NP}^{\Sigma_{i-1}\mathsf{SAT}}$
Let us prove a particular case of the above claim, with $i = 2$.
Corollary: $\Sigma_2^p = \mathsf{NP}^{\mathsf{SAT}}$
Proof: The proof has two parts. Part 1 shows $\Sigma_2^p \subseteq \mathsf{NP}^{\mathsf{SAT}}$ and part 2 shows $\mathsf{NP}^{\mathsf{SAT}} \subseteq \Sigma_2^p$. Before starting part 1, let L be a language in Σ_2^p. Then there is a DTM M and a polynomial q such that

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} M(x, u_1, u_2) = 1$$

In part 1, we need to show that there is a NTM M' with access to oracle-SAT that can decide L. That is, we need to show that:

$$\exists u_1 \forall u_2 M(x, u_1, u_2) = 1 \iff M'^{\mathsf{SAT}}(x) = 1 \tag{13.6}$$

Being a NTM, M' can guess u_1 (which takes care of "$\exists u_1$" in (13.6)). Once u_1 is fixed this way, M' needs to find whether $\forall u_2 M(x, u_1, u_2) = 1$. This question itself makes a language L: $\langle x, u_1 \rangle \in L \iff \forall u_2 M(x, u_1, u_2) = 1$. Further $L \in \mathsf{coNP}$ from the alternative definition of coNP and hence $\langle x, u_1 \rangle$ can be Karp-reduced to a TAUTOLOGY instance φ_{x,u_1}. Indeed, M' reduces $\langle x, u_1 \rangle$ to φ_{x,u_1} in poly-time and checks if $\varphi_{x,u_1} \in \mathsf{TAUTOLOGY}$. The checking step is poly-time as M' has access to oracle-SAT and $\neg\varphi(\cdot) \notin \mathsf{SAT} \iff \varphi(\cdot) \in \mathsf{TAUTOLOGY}$. Thus

$$\exists u_1 \forall u_2 M(x, u_1, u_2) = 1 \iff \exists u_1 \varphi_{x,u_1} \in \mathsf{TAUTOLOGY} \iff M'^{\mathsf{SAT}}(x) = 1$$

That completes part 1 of the proof.
For part 2, our first attempt would be to mimick part 1: map the nondeterministic choice u_1 to $\exists u_1$, and map the rest of the activity of M' (including querying oracle) to some formula of the form $\forall \cdot M(\cdots) = 1$. That is,

$$M'^{\mathsf{SAT}}(x) = 1 \implies \exists u_1 \forall u_2 (\text{oracle-SAT's answer for question on } \varphi_{x,u_1} \text{ comes here})$$

The issue with the above mapping is, it assumes that M' makes only one query to the oracle, whereas in principle M' can make polynomially many queries to the oracle and every next move of M' can depend on answers given by the oracle in the past queries.
The main idea is to nondeterministically guess all the future queries as well as the SAT oracle's answers and then make a single coNP query whose answer verifies that all this guessing was correct[1].

Specifically, there exists a sequence of nondeterministic choices with which M' makes m queries to oracle and accepts x, as follows. Let c be the sequence of choices. Let $a_1, \cdots, a_m$ be the answers by the oracle for queries $\varphi_1, \cdots, \varphi_m$ respectively. Interpret a_i as follows. If $a_i = 0$ then φ_i is not satisfiable (i.e., $\forall v_i\ \varphi_i(v_i) = 0$), and if $a_i = 1$ then φ_i is satisfiable (i.e., $\exists u_i\ \varphi_i(u_i) = 1$). Thus we have the following description:

$$\begin{aligned} x \in L \iff & \exists c, u_1 \cdots, u_m \forall v_1, \cdots, v_m \text{ such that } M' \text{ accepts } x \text{ using choice sequence } c, \text{ answers } a_1, \cdots, a_m \text{ and} \\ & \forall i \in [m] a_i = 1 \implies \varphi_i(u_i) = 1 \text{ and} \\ & \forall i \in [m] a_i = 0 \implies \varphi_i(v_i) = 0 \end{aligned}$$

The expression shows that $L \in \Sigma_2^p$.□

13.4 References

[1] Sanjeev Arora and Boaz Barak, Computational Complexity: A Modern Approach, *Cambridge University Press*, 2009 .