Notes on Computer Theory Last updated: April, 2018

The Polynomial Heirarchy.

Notes by Jonathan Katz, lightly edited by Dov Gordon

1 The Polynomial Hierarchy

We have seen the classes $\mathcal{NP}$ and $\mathsf{co}\mathcal{NP}$, which are defined as follows:

$L \in \mathcal{NP}$ if there is a (deterministic) Turing machine M running in time polynomial in its first input, such that

$$x \in L \Leftrightarrow \exists w\, M(x, w) = 1.$$

$L \in \mathsf{co}\mathcal{NP}$ if there is a (deterministic) Turing machine M running in time polynomial in its first input, such that

$$x \in L \Leftrightarrow \forall w\, M(x, w) = 1.$$

It is natural to generalize the above; doing so allows us to capture problems that are "more difficult" than $\mathcal{NP} \cup \mathsf{co}\mathcal{NP}$. As an example, consider again the language CLIQUE :

$$\text{CLIQUE} \stackrel{\text{def}}{=} \{(G, k) \mid G \text{ has a clique of size } k\}.$$

We know that CLIQUE $\in \mathcal{NP}$; a certificate is just a set of vertices of size at least k (that the vertices are fully-connected can easily be verified). How about the following language?

$$\text{MAX-CLIQUE} \stackrel{\text{def}}{=} \{(G, k) : \text{the largest clique in } G \text{ has size exactly } k\}.$$

This language does not appear to be in $\mathcal{NP}$: we can certify that some graph G has a clique of size k, but how do we certify that this is the *largest* clique in G? The language does not appear to be in $\mathsf{co}\mathcal{NP}$, either: although we could prove that $(G, k) \notin$ MAX-CLIQUE if G happened to have a clique of size *larger* than k, there is no easy way to prove that $(G, k) \notin$ MAX-CLIQUE when the largest clique has size *smaller* than k.

As another example, consider the problem of CNF-formula minimization. A CNF formula ϕ on n variables naturally defines a function $f_\phi : \{0,1\}^n \to \{0,1\}$, where $f_\phi(x) = 1$ iff the given assignment x satisfies ϕ. Can we tell when a given formula is minimal? Consider the language

$$\text{MIN-CNF} \stackrel{\text{def}}{=} \{\phi : \text{no formula with fewer clauses than } \phi \text{ computes } f_\phi\}.$$

This language does not appear to be in $\mathcal{NP}$ or $\mathsf{co}\mathcal{NP}$, either. (Even if I give you a smaller formula ϕ' that computes f_ϕ, there is no obvious way for you to verify that fact efficiently.)

The above examples motivate the following definition:

Definition 1 *Let i be a positive integer. $L \in \Sigma_i$ if there is a (deterministic) Turing machine M running in time polynomial in its first input, such that*

$$x \in L \Leftrightarrow \underbrace{\exists w_1 \forall w_2 \cdots Q_i w_i}_{i \text{ times}}\, M(x, w_1, \ldots, w_i) = 1.$$

where $Q_i = \forall$ if i is even, and $Q_i = \exists$ if i is odd.

$L \in \Pi_i$ if there is a (deterministic) Turing machine M running in time polynomial in its first input, such that

$$x \in L \Leftrightarrow \underbrace{\forall w_1 \exists w_2 \cdots Q_i w_i}_{i \text{ times}} \; M(x, w_1, \ldots, w_i) = 1.$$

where $Q_i = \forall$ if i is odd, and $Q_i = \exists$ if i is even.

As in the case of $\mathcal{NP}$, we may assume without loss of generality that the w_i each have length polynomial in x.

Returning to our examples, note that MAX-CLIQUE $\in \Sigma_2$ since $(G, k) \in$ MAX-CLIQUE iff there exists a set of vertices S such that for all sets of vertices S' the following (efficiently verifiable) predicate is true:

> $|S| = k$ and S forms a clique in G; moreover, either $|S'| \leq k$ or S' does not form a clique in G.

(It is easier to express the above in English as: "there exists a set S of k vertices, such that for all sets S' containing more than k vertices, S is a clique and S' is not clique". But one has to be careful to check that anytime the quantification is limited [e.g., by quantifying over all sets of size greater than k, rather than all sets], the limitation is efficient to verify.) We also have MAX-CLIQUE $\in \Pi_2$ since we can swap the order of quantifiers in this case (since S' does not depend on S, above). Turning to the second example, note MIN-CNF $\in \Pi_2$ since $\phi \in$ MIN-CNF iff for all formulas ϕ' that are smaller than ϕ there exists an input x for which $\phi(x) \neq \phi(x')$. Here, however, we cannot just swap the order of quantifiers (do you see why?), and so we do not know, or believe, that MIN-CNF $\in \Sigma_2$.

We make some relatively straightforward observations, leaving their proofs as exercises.

- $\Sigma_1 = \mathcal{NP}$ and $\Pi_1 = \mathsf{co}\mathcal{NP}$.
- For all i, we have $\mathsf{co}\Sigma_i = \Pi_i$ (and $\mathsf{co}\Pi_i = \Sigma_i$).
- For all i, we have $\Sigma_i, \Pi_i \subseteq \Pi_{i+1}$ and $\Sigma_i, \Pi_i \subseteq \Sigma_{i+1}$.

The *polynomial hierarchy* PH consists of all those languages of the form defined above; that is, $\mathsf{PH} \stackrel{\text{def}}{=} \bigcup_i \Sigma_i = \bigcup_i \Pi_i$. Since $\Sigma_i \subseteq \mathsf{PSPACE}$ for all i, we have $\mathsf{PH} \subseteq \mathsf{PSPACE}$. Each Σ_i (resp., Π_i) is called a *level* in the hierarchy.

As in the case of $\mathcal{NP}$ and $\mathsf{co}\mathcal{NP}$, we believe

Conjecture 1 *$\Sigma_i \neq \Pi_i$ for all i.*

If the above conjecture is not true, then the polynomial hierarchy *collapses* in the following sense:

Theorem 2 *If $\Sigma_i = \Pi_i$ for some i, then $\mathsf{PH} = \Sigma_i$.*

Proof We show that for all $j > i$, we have $\Sigma_j = \Sigma_{j-1}$. Let $L \in \Sigma_j$. So there is a polynomial-time Turing machine M such that

$$x \in L \Leftrightarrow \exists w_j \forall w_{j-1} \cdots Q w_1 \; M(x, w_j, \ldots, w_1) = 1,$$

where we have numbered the variables in descending order for clarity in what follows. Assume $j - i$ is even. (A symmetric argument works if $j - i$ is odd.) Define language L' as

$$(x, w_j, \ldots, w_{i+1}) \in L' \Leftrightarrow \exists w_i \forall w_{i-1} \cdots Q w_1 \; M(x, w_j, \ldots, w_1) = 1,$$

and note that $L' \in \Sigma_i$. By the assumption of the theorem, $L' \in \Pi_i$ and so there is some machine M' running in time polynomial in $|x| + |w_j| + \cdots + |w_{i+1}|$ (and hence polynomial in $|x|$) such that

$$(x, w_j, \ldots, w_{i+1}) \in L' \Leftrightarrow \forall w'_i \exists w'_{i-1} \cdots Q' w'_1 \; M'(x, w_j, \ldots, w_{i+1}, w'_i, \ldots, w'_1) = 1.$$

(Note that the $\{w'_i, \ldots, w'_1\}$ need not have any relation with the $\{w_i, \ldots, w_1\}$.) But then

$$\begin{aligned} x \in L &\Leftrightarrow \exists w_j \forall w_{j-1} \cdots \forall w_{i+1} \;\; (x, w_j, \ldots, w_{i+1}) \in L' \\ &\Leftrightarrow \exists w_j \forall w_{j-1} \cdots \forall w_{i+1} \left[\forall w'_i \exists w'_{i-1} \cdots Q' w'_1 \;\; M'(x, w_j, \ldots, w'_1) = 1 \right] \\ &\Leftrightarrow \exists w_j \forall w_{j-1} \cdots \forall w_{i+1} w'_i \exists w'_{i-1} \cdots Q' w'_1 \;\; M'(x, w_j, \ldots, w'_1) = 1, \end{aligned}$$

and there are only $j - 1$ quantifiers in the final expression. Thus, $L \in \Sigma_{j-1}$. ■

A similar argument gives:

Theorem 3 *If $\mathcal{P} = \mathcal{NP}$ then $\mathsf{PH} = \mathcal{P}$.*

Each Σ_i has the complete problem Σ_i-SAT, where this language contains all true expressions of the form $\exists w_1 \forall w_2 \cdots Q w_i \;\; \phi(w_1, \ldots, w_i) = 1$ for ϕ a boolean formula in CNF form, and where each w_i is a vector of variables. However, if PH has a complete problem, the polynomial hierarchy collapses: If L were PH-complete then $L \in \Sigma_i$ for some i; but then every language $L' \in \Sigma_{i+1} \subseteq \mathsf{PH}$ would be reducible to L, implying $\Sigma_{i+1} = \Sigma_i$.