CMPT 407 - Complexity Theory
Lecture 7: Alternation

Valentine Kabanets

June 6, 2017

1 Collapsing the Polynomial-Time Hierarchy

Many results in complexity theory have the form "Statement X is true, unless the polynomial-time hierarchy collapses". Since it is generally believed that all the levels of the polytime hierarchy are distinct, i.e., that the polytime hierarchy *does not collapse*, such results give us some evidence that the statement X is probably true.

Let us see under what conditions the polytime hierarchy would collapse to some finite level, i.e., $\mathsf{PH} = \Sigma_i^p$, for some constant i.

Theorem 1. *If* $\mathsf{NP} = \mathsf{coNP}$*, then* $\mathsf{PH} = \Sigma_1^p = \mathsf{NP} = \mathsf{coNP}$.

Proof. We want to show that $\Sigma_i^p = \Sigma_1^p$ for every i. Our proof is by induction. The base case of $i = 1$ is obvious. Suppose the truth of the statement for $i \geq 1$, and let us prove it for $i+1$.

Take any $L \in \Sigma_{i+1}^p$. By definition, there is an $(i+2)$-ary polytime polybalanced relation R such that $x \in L$ iff $\exists y_1 \forall y_2 \ldots R(x, y_1, \ldots, y_{i+1})$. Consider the language $L' = \{(x, y) \mid \forall y_2 \exists y_3 \ldots R(x, y, y_2, \ldots, y_{i+1})$. It is clear that $L' \in \Pi_i^p$. By the inductive hypothesis, we have $\Pi_i^p = \mathsf{co}\Sigma_i^p = \mathsf{co}\Sigma_1^p = \Pi_1^p$. On the other hand, our assumption is that $\mathsf{NP} = \mathsf{coNP}$, i.e., that $\Sigma_1^p = \Pi_1^p$. Thus, we obtain that $L' \in \Sigma_1^p$.

Finally, observe that $x \in L$ iff $\exists y\ (x, y) \in L'$. Since $L' \in \Sigma_1^p$, it follows that there is a polytime polybalanced relation R' such that $w \in L'$ iff $\exists z R'(w, z)$. Thus, $x \in L$ iff $\exists (y, z) R'((x, y), z)$. Therefore, $L \in \Sigma_1^p$. □

Corollary 2. *If* $\mathsf{NP} = \mathsf{P}$*, then* $\mathsf{PH} = \mathsf{P}$.

Proof. If $\mathsf{NP} = \mathsf{P}$, then clearly $\mathsf{NP} = \mathsf{coNP} = \mathsf{P}$. So by Theorem 1, $\mathsf{PH} = \mathsf{NP}$, and hence, $\mathsf{PH} = \mathsf{P}$. □

Theorem 1 above can be easily generalized to show the following.

Theorem 3. *If* $\Sigma_i^p = \Pi_i^p$ *for some* $i \geq 1$*, then* $\mathsf{PH} = \Sigma_i^p$.

In other words, if the ith level of the polytime hierarchy is closed under complement, then the polytime hierarchy collapses to the ith level. It is easy to see that the converse is also true.

Theorem 4. *If* $\mathsf{PH} = \Sigma_i^p$ *for some* $i \geq 1$*, then* $\Sigma_i^p = \Pi_i^p$.

Proof. Note that $\Pi_i^p \subseteq \mathsf{PH} = \Sigma_i^p$. So $\Pi_i^p \subseteq \Sigma_i^p$. Taking the complements of both sides of the last inclusion, we get that $\Sigma_i^p \subseteq \Pi_i^p$ as well, and so indeed, $\Sigma_i^p = \Pi_i^p$. □

Putting the last two theorems together, we get the following

Corollary 5. *For every* $i \geq 1$, $\mathsf{PH} = \Sigma_i^p \Leftrightarrow \Sigma_i^p = \Pi_i^p$.

2 Time-Space tradeoff for SAT

Recall that the class $\mathsf{TISP}(t, s)$ is the set of languages that can be decided by a TM in time $t(n)$ and *simultaneously* in at most $s(n)$ space on inputs of length n; here t and s are proper complexity functions.

We now show the following:

Theorem 6 (Fortnow)**.** $\mathsf{SAT} \notin \mathsf{TISP}(n^{1.1}, n^{0.1})$.

By the careful analysis of the Cook-Levin reduction, one can show that there is a reduction from any language in $\mathsf{NTIME}(t(n))$ to a SAT-instance of size $O(t \log t)$, where every output bit of the reduction is computable in time and space $\mathsf{poly}(\log n)$. (Exercise![1])

Hence, if $SAT \in \mathsf{TISP}(n^{1.1}, n^{0.1})$, then $\mathsf{NTIME}(n) \subseteq \mathsf{TISP}(n^{1.1}\mathsf{poly}(\log n), n^{0.1}\mathsf{poly}(\log n))$. Thus, Theorem 6 will follow from the next lemma.

Lemma 7. $\mathsf{NTIME}(n) \not\subseteq \mathsf{TISP}(n^{1.2}, n^{0.2})$.

Before we start proving this lemma, we state the following definition of a Σ_2-computation with runtime t: By $\Sigma_2\mathsf{Time}(t)$ we mean the complexity class of languages such that the membership of input x in the language is determined by a formula of the following type:

$$\phi(x) = \exists y \in \{0,1\}^{O(t(|x|))}\ \forall z \in \{0,1\}^{O(t(|x|))}\ \psi(x, y, z),$$

where $\psi(x, y, z)$ is a predicate computable in deterministic time $O(t(|x|))$.

Proof of Lemma 7. Our proof is by contradiction. We outline the structure of our argument, with some pieces being proved as separate claims below.

Suppose $\mathsf{NTIME}(n) \subseteq \mathsf{TISP}(n^{1.2}, n^{0.2})$.

[1] Use the fact that for every deterministic TM with runtime t, there exists an oblivious deterministic TM with runtime $O(t \log t)$ (cf. Homework 1, Question 5.b). Then, argue that an oblivious TM with runtime T can be simulated by a Boolean circuit of size $O(T)$ (in contrast to a non-oblivious time-T TM which may require circuit size $O(T^2)$).

1. By padding, $\mathsf{NTIME}(n^{10}) \subseteq \mathsf{TISP}(n^{12}, n^2)$.
2. By Claim 8 below, $\mathsf{TISP}(n^{12}, n^2) \subseteq \Sigma_2\mathsf{Time}(n^8)$.
3. By Claim 9 below, $\Sigma_2\mathsf{Time}(n^8) \subseteq \mathsf{NTIME}(n^{9.6})$.

Chaining these three inclusions together, we get

$$\mathsf{NTIME}(n^{10}) \subseteq \mathsf{NTIME}(n^{9.6}),$$

which contradicts the Nondeterministic Time Hierarchy theorem. □

Claim 8. $\mathsf{TISP}(n^{12}, n^2) \subseteq \Sigma_2\mathsf{Time}(n^8)$.

Proof. Let $L \in \mathsf{TISP}(n^{12}, n^2)$ be an arbitrary language, decided by some TM M. Break up the computation of M on input x, $|x| = n$, into n^6 blocks of length n^6 each. Let $c_0, c_1, \ldots, c_{n^6}$ be the delimiting configurations for the partition. Each such configuration can be described by $O(n^2)$-bit string by the assumption on the space usage. Hence, we can write all n^6 of them using only $O(n^8)$ bits total.

Finally, we need to check for every $0 \leq i < n^6$ that there is valid computation of length at most n^6 leading from c_i to c_{i+1}.

Overall, we get that $x \in L$ iff

$$\exists c_0, c_1, \ldots, c_{n^6}\ \forall 0 \leq i < n^6 \quad \text{“}c_{i+1} \text{ follows from } c_i \text{ in } O(n^6) \text{ steps”}.$$

The latter is easily seen to be a $\Sigma_2\mathsf{Time}(n^8)$-formula. □

Claim 9. *If* $\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(n^{1.2})$, *then* $\Sigma_2\mathsf{Time}(n^8) \subseteq \mathsf{NTIME}(n^{9.6})$.

Proof. By definition, $\Sigma_2\mathsf{Time}(n^8)$ formula $\phi(x)$ has the form:

$$\exists y \in \{0,1\}^{O(|x|^8)}\ \forall z \in \{0,1\}^{O(|x|^8)}\ \psi(x, y, z),$$

where $\psi(x, y, z)$ is computable in deterministic time $O(n^8)$. The formula $\eta(x, y) = \forall z\ \psi(x, y, z)$ corresponds to a coNP computation in time $O(n^8)$. If $\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(n^{1.2})$, then clearly $\mathsf{coNTIME}(n) \subseteq \mathsf{DTIME}(n^{1.2})$, and hence, by padding, $\mathsf{coNTIME}(n^8) \subseteq \mathsf{DTIME}(n^{9.6})$.

Denoting the latter deterministic computation by $\xi(x, y)$, we get $x \in L$ iff $\exists y \xi(x, y)$, where y is of length $|x|^8$ and $\xi(x, y)$ is computable in deterministic time $n^{9.6}$. Hence, we get $L \in \mathsf{NTIME}(n^{9.6})$. □

One can improve the exponents of the above time-space tradeoff for SAT somewhat (getting $SAT \notin \mathsf{TISP}(n^c, n^{o(1)})$, for $c < 2\cos(\pi/7) \approx 1.8$, as shown by R. Williams (IEEE Conference on Computational Complexity, 2007), but still we can't rule out n^2-time SAT algorithms that use polylogarithmic amount of space. The techniques used in the proof presented above seem to get stuck just below the exponent 2 for the time bound. Moreover, there is some evidence that the current proof techniques (as the ones we used in the above proof) can't beat the time-bound exponent $2\cos(\pi/7) \approx 1.8$ (cf. S. Buss and R. Williams, Limits on Alternation-Trading Proofs for Time-Space Lower Bounds, *IEEE Conference on Computational Complexity*, 2012.)

Thus, we conclude with the following

Open Question: Prove (or disprove) that $SAT \notin \mathsf{TISP}(n^2, poly(\log n))$.

3 NP $\stackrel{?}{\subset}$ P/poly

There is an interesting connection between NP having polysize circuits and the collapse of a polytime hierarchy.

Theorem 10 (Karp-Lipton)**.** *If* NP $\subset$ P/poly*, then* PH $= \Sigma_2^p$.

Proof Sketch. As we argued earlier, to show that PH $= \Sigma_2^p$, it suffices to prove that $\Sigma_2^p = \Pi_2^p$. To prove the latter, it actually suffices to argue that $\Pi_2^p \subseteq \Sigma_2^p$. (Show that this is indeed sufficient!)

Since NP $\subset$ P/poly, there is a polysize family of circuits computing SAT. Moreover, since there is a polytime algorithm for finding a satisfying assignment, given access to an algorithm for SAT, we conclude that there is a polysize family of circuits C_n with the following property: Given a propositional formula ϕ of size n, $C_n(\phi)$ outputs a satisfying assignment for ϕ if one exists, or outputs the string of 0s if ϕ is unsatisfiable. (Check this!)

Now, consider any language $L \in \Pi_2^p$. By definition, there is a polytime polybalanced relation R such that $x \in L$ iff $\forall y \exists z \;\; R(x, y, z)$. Consider the language

$$L' = \{(x, y) \mid \exists z \;\; R(x, y, z)\}.$$

Obviously, $L' \in$ NP. By our assumption, there is a polysize circuit family such that $C_m(x, y)$ outputs a satisfying assignment z for $R(x, y, z)$ if one exists; here, $m = |x| + |y|$.

We can test if $x \in L$ by the following polytime polybalanced formula:

$$\exists C_m \forall y \;\; R(x, y, C_m(x, y)).$$

Indeed, if $x \in L$, then there will be a polysize circuit C_m that would produce a satisfying z-assignment for $R(x, y, z)$ for every y. Conversely, if there is some small circuit C_m that produces a satisfying z-assignment for $R(x, y, z)$ for every y, then the formula $\forall y \exists z \;\; R(x, y, z)$ must be true, and hence, $x \in L$. Thus, we have shown that $L \in \Sigma_2^p$, as required. □

4 EXP $\stackrel{?}{\subset}$ P/poly

Last time we saw that assuming NP $\subseteq$ P/poly implies PH $= \Sigma_2^p$. Next we consider what happens if EXP $\subseteq$ P/poly.

The assumption that EXP $\subseteq$ P/poly seems even crazier than NP $\subseteq$ P/poly, but it is still not disproved. However, we have the following.

Theorem 11 (A. Meyer)**.** *If* EXP $\subset$ P/poly*, then* EXP $= \Sigma_2^p$.

Proof. Given an arbitrary language $L \in$ EXP decided by a deterministic TM M in time 2^{n^c}. Let us denote by $T(x)$ the tableau of the computation of M on input x; the tableau is an array of size 2^{n^c} by 2^{n^c}, where the first row is the (encoding) of the initial configuration, and each row $i+1$ is the configuration that follows from the configuration in row i in one step of

the computation of M on x. With appropriate encoding, we can think of $T(x)$ as an array of constant-length binary strings (with each such string encoding a symbol in a corresponding cell position at a given time step), and will denote by $T(x)_{i,j}$ the (i,j)th entry of the array $T(x)$. Without loss of generality, we will use the first bit of the last configuration to signal if the computation is accepting or not, i.e., $T(x)_{2^{|x|^c},1} = 1$ iff $T(x)$ is the tableau of an accepting computation of M on x.

Define a new language

$$L' = \{(x,i,j,k) \mid 1 \leq i,j \leq 2^{|x|^c},\ 1 \leq k \leq O(1),\ T(x)_{i,j,k} = 1\},$$

where $T(x)_{i,j,k}$ is the kth bit of the constant-size binary string $T(x)_{i,j}$. It is easy to see that $L' \in \mathsf{EXP}$ (simply simulate M on x until you see the ith configuration, and then look up the jth tape cell in that configuration, and check its kth bit). By our assumption, $L' \in \mathsf{P/poly}$. Note that given a polysize circuit C for L', we can check if C is correct by testing that every pair of successive configurations given by C are indeed valid configurations such that one follows form the other. As in the proof of the Cook-Levin Theorem, it suffices to check that locally every 2×3 "window" in the tableau is correct, where the window is the sub-array $T(x)_{i-1,j-1}, T(x)_{i-1,j}, T(x)_{i-1,j+1}, T(x)_{i,j}$.

Consider the following Σ_2^p formula: $\phi(x) =$

$$\exists C \forall i,j\ [C(x,i,j) \text{ follows from } C(x,i-1,j-1), C(x,i-1,j), C(x,i-1,j+1)] \\ \& \left[C(x,2^{|x|^c},1) = 1\right]$$

where C is an encoding of a circuit of size $\mathsf{poly}(|x|)$, and i,j are numbers between 1 and $2|x|^c$; hence, C, i, and j can all be encoded with $\mathsf{poly}(|x|)$-size bit strings.

We claim that $x \in L$ iff $\phi(x)$ is true. As we've argued above, if $x \in L$ then there is a small circuit encoding the tableau of an accepting computation of M on x, and hence $\phi(x)$ will be true. On the other hand, if there exists some circuit C for which $\phi(x)$ is true, then this C must indeed encode an accepting computation of M on x, and hence $x \in L$. □

The above proof argument crucially relied on the fact that the tableau of a *deterministic* exponential-time TM is unique, and hence, each bit of the tableau can be computed also in deterministic exponential time. The assumption of $\mathsf{NEXP} \subseteq \mathsf{P/poly}$ is about nondeterministic TMs, and we don't know how to use the same argument as above in this case. Still, with a different proof argument (using, in particular, the connection between circuit complexity and derandomization of randomized algorithms), one can show the following:

Theorem 12 (Impagliazzo, Kabanets, Wigderson)**.** *If* $\mathsf{NEXP} \subseteq \mathsf{P/poly}$*, then* $\mathsf{NEXP} = \Sigma_2^p$.