18.405J/6.841J Advanced Complexity Theory	March 2, 2000
Lecture 9	
Lecturer: Dan Spielman	*Scribe: Abie Flaxman*

At the end of last lecture, we defined the complexity class #P. In today's lecture we will identify some #P-complete problems. We will also begin a proof of Toda's theorem, which shows that $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}}$.

1 #P-completeness

Definition 1 #P *is the set of functions f for which there exists a polynomial time, nondeterministic Turing Machine M such that $f(x)$ is the number of accepting paths of M on input x.*

Definition 2 FP *is the set of functions computable in polynomial time.*

Definition 3 *A function f is called* #P*-complete if*

1. $f \in \#\mathrm{P}$,
2. *For all* $g \in \#\mathrm{P}, g \in \mathrm{FP}^f$.

Let #CIRCUIT-SAT be the function that associates every circuit with the number of inputs that cause the circuit to output TRUE.

Claim 4 #CIRCUIT-SAT *is* #P*-complete.*

Proof #CIRCUIT-SAT is clearly in #P. The meat of the proof is verifying condition (2). Let M be a polynomial time, nondeterministic Turing Machine. Consider its computation tableau. We outlined in a previous lecture how to turn the tableau into a circuit. Do this. To count the number of accepting paths M has on input x, hardwire x as the input to the circuit for M, and make the actual input to the circuit the branch direction at each nondeterministic operation. This circuit will output TRUE for every accepting path of M on x. ■

Let #SAT be the function that associates every boolean function ϕ with the number of assignments that satisfy ϕ.

Claim 5 #SAT *is* #P*-complete.*

Proof Given a circuit C with input $\bar{x}$, construct a 3CNF ϕ with inputs $(\bar{x}, \bar{y})$ such that if $\bar{x}$ satisfies C, there exists a unique $\bar{y}$ such that $(\bar{x}, \bar{y})$ satisfies ϕ. This construction requires one variable for each gate and a small number of clauses allowing only legal configurations of each gate. ■

Remark These reductions have exactly as many accepting paths as the original problem. Why don't we strengthen the definition of #P-completeness to reflect this?

The answer is that #MATCHINGS (also known as PERMANENT) is #P-complete only under our current definition. Given an instance of #SAT with c clauses and S satisfying assignments, you get a graph with something like $4^c S$ matchings. (But definitely not exactly this...)

Finding a matching (or even a bunch of matchings) is a polynomial time operation. So this is pretty weird. Especially considering our next theorem, which shows that #P is at least as powerful as PH.

2 Machinary for Toda's Theorem

Theorem 6 *(Toda's Theorem)* $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}}$

Question:	Is $\mathrm{P}^{\#\mathrm{P}} \subseteq \mathrm{PSPACE}$?
Answer:	Yes.

To get an intuition for Toda's theorem, we need to define some new complexity classes. For class $\mathcal{C}$, we will define four class operators. We can use these to get all kinds of new classes.

Definition 7 $\Sigma \cdot \mathcal{C}$ *is the set of all L for which there exists a polynomial $p(n)$ and a language $A \in \mathcal{C}$ such that $x \in L$ iff there exists $y, |y| = p(|x|)$ such that $(x, y) \in A$.*

Definition 8 $\Pi \cdot \mathcal{C}$ *is the set of all L for which there exists a polynomial $p(n)$ and a language $A \in \mathcal{C}$ such that $x \in L$ iff for all $y, |y| = p(|x|)$, $(x, y) \in A$.*

Definition 9 $\mathrm{BP} \cdot \mathcal{C}$ *is the set of all L for which there exists a polynomial $p(n)$ and a language $A \in \mathcal{C}$ such that*

1. $x \in L \Rightarrow \Pr_{|y|=p(|x|)}[(x, y) \in A] > 2/3$
2. $x \notin L \Rightarrow \Pr_{|y|=p(|x|)}[(x, y) \in A] < 1/3$

Definition 10 (Parity operator) $\oplus \cdot \mathcal{C}$ *is the set of all L for which there exists a polynomial $p(n)$ and a language $A \in \mathcal{C}$ such that $x \in L$ iff $|\{y \text{ s.t. } |y| = p(|x|), (x, y) \in A\}|$ is odd.*

Using these class operators, we can state the key elements of Toda's theorem.

Claim 11 $\mathrm{NP} \subseteq \mathrm{BP} \cdot \oplus \cdot \mathrm{P}$, *and more generally,* $\Sigma_k^P \subseteq (\mathrm{BP} \cdot \oplus)^k \cdot \mathrm{P}$.

Claim 12 *For any class $\mathcal{C}$ formed by applying these operators to* P,

$$\begin{aligned}
\oplus \cdot \mathrm{BP} \cdot \mathcal{C} &\subseteq \mathrm{BP} \cdot \oplus \cdot \mathcal{C}, \\
\oplus \cdot \oplus \cdot \mathcal{C} &\subseteq \oplus \cdot \mathcal{C}, \\
\mathrm{BP} \cdot \mathrm{BP} \cdot \mathcal{C} &\subseteq \mathrm{BP} \cdot \mathcal{C}, \\
\mathrm{BP} \cdot \oplus \cdot \mathrm{P} &\subseteq \mathrm{P}^{\#\mathrm{P}}.
\end{aligned}$$

This may look like operator soup, but it's going to pay off.

We're not going to prove the boldest of these claims directly, but we will do some of the special cases.

Lemma 13 $\mathrm{NP} \subseteq \mathrm{BP} \cdot \oplus \cdot \mathrm{P}$

Proof Recall the theorem of Valiant and Vazirani from last lecture: there exists a random polynomial time procedure that takes a circuit C and returns $n + 1$ circuits $C_1, ..., C_{n+1}$ such that if C is satistfiable,

$$\Pr[\text{there exists } i \text{ s.t. } C_i \text{ has exactly one satisfying assignment}] > 1/8,$$

and otherwise no C_i has a satisfying assignment.

Two things are not so good about this. First, the $1/8$ should be a $2/3$. Second, we have $n + 1$ circuits and we want to have 1.

To circumvent objection 1, we amplify the probability by repetition. Say we output $C_1^i, ..., C_{n+1}^i$ for $i = 1, 2, ..., 20$. Then

$$\Pr[\text{exists } i, j \text{ s.t. } C_j^i \text{ has exactly one satisfying assignment}] > 1 - (7/8)^{20} \approx 0.930.$$

How can we beat objection 2? We do something like "AND them all togther" (but not quite, because that doesn't work.)

Given $x_1, ..., x_n$, if there exists i such that x_i is even, then $\prod x_i$ is even. If there exists i such that x_i is odd, then $1 + \prod(1 + x_i)$ is odd. This is our inspiration.

Given $C_1(\bar{x}_1)$ has m_1 satisfying assignments and $C_2(\bar{x}_2)$ has m_2 satisfying assignements, $C_1 \cap C_2(\bar{x}_1, \bar{x}_2)$ has $m_1 m_2$ satisfying assignments.

It is easy to add one: given a circuit C with m satisfying assignments, we form a new circuit C' with $m + 1$ satisfying assignments by adding an additional input, z, with the rule that if $z = 0$, C' does what C would do, and if $z = 1$, C' accepts only if $x_1, ..., x_n$ are all 1.

So we have a randomized polynomial time procedure which takes a circuit C as input and outputs a circuit C' such that:

1. if $\#\mathrm{C} > 0, \Pr[\#\mathrm{C}' \text{ is odd}] > 2/3$
2. if $\#\mathrm{C} = 0, \#\mathrm{C}'$ is even.

This gives us a $\mathrm{BP} \cdot \oplus \cdot \mathrm{P}$ algorithm for recognizing the NP-complete language CIRCUIT-SAT. C is in CIRCUIT-SAT iff for most choices of random bits C' has an odd number of satisfying assignments. ■

Question:	Why did we use BP and not RP?
Answer:	Here RP would have worked also, but perhaps not for Σ_k^P.

Observation 14 $\mathrm{BP} \cdot \oplus \cdot \mathrm{P}$ *is closed under compliment because* $\oplus \cdot \mathrm{P}$ *is closed under compliment.*

Corollary 15 $\mathrm{coNP} \subseteq \mathrm{BP} \cdot \oplus \cdot \mathrm{P}$

A similar technique would prove $(\Pi \cdot \Sigma)^k \cdot \mathrm{P} \subseteq (\mathrm{BP} \cdot \oplus)^k \cdot \mathrm{P}$, but the bookkeeping is nasty, so we will develop more machinary and come back to this.

Theorem 16 $\oplus \cdot \mathrm{BP} \cdot \mathrm{P} \subseteq \mathrm{BP} \cdot \oplus \cdot \mathrm{P}$

Proof Let L be a language in $\oplus \cdot \mathrm{BP} \cdot \mathrm{P}$. Then there is a polynomial $p_1(n)$ and some language $A' \in \mathrm{BP} \cdot \mathrm{P}$ such that $x \in L$ iff $(x, y) \in A'$ for an odd number of y's with length $p_1(|x|)$.

We will amplify the error probabilities of the BP operator so the error is less than $2^{-2p_1(|x|)}$. Then there is a polynomial $p_2(n)$ and some language $A \in \mathrm{P}$ such that

1. $(x, y) \in A' \Rightarrow \Pr_{|z|=p_2(|(x,y)|)}[(x, y, z) \in A] > 1 - 2^{-2p_1(|x|)}$
2. $(x, y) \notin A' \Rightarrow \Pr_{|z|=p_2(|(x,y|)}[(x, y, z) \in A] < 2^{-2p_1(|x|)}$

Putting these definitions togther, we see that

$$x \in L \Rightarrow \Pr_z[(x, y, z) \in A] > 1 - 2^{2p_1(|x|)} \text{ for an odd number of } y.$$

Also,

$$x \notin L \Rightarrow \Pr_z[(x, y, z) \in A] > 1 - 2^{2p_1(|x|)} \text{ for an even number of } y.$$

And for any y,

$$\Pr_z[(x,y,z) \in A \text{ disagrees with } (x,y) \in A'] < 2^{2p_1(|x|)}.$$

So,

$$\Pr_z[(x,y,z) \in A \text{ agrees with } (x,y) \in A' \text{ for all } y] < 2^{2p_1(|x|)} \cdot 2^{p_1(|x|)} = 2^{p_1(|x|)}.$$

Therefore,

$$x \in L \Rightarrow \Pr_z[(x,y,z) \in A \text{ for an odd number of } y] > 1 - 2^{-p_1(|x|)}$$

and

$$x \notin L \Rightarrow \Pr_z[(x,y,z) \in A \text{ for an odd number of } y] < 2^{-p_1(|x|)}.$$

■