CS 710: Complexity Theory 12/6/2011

Lecture 25: Counting vs. Alternation

Instructor: Dieter van Melkebeek Scribe: Brian Nixon & Sachin Ravi

This lecture will continue on the topic of counting, #P, and how counting relates to the polynomial hierarchy in terms of power.

Recall by definition, #P consists of all $f : \{0,1\}^* \to \mathbf{N}$ where $\exists c \in \mathbf{N}$, $\exists V \in$ P such that $f(x) = |S_x|$ (the size of the solution set) $S_x = \{y \in \{0,1\}^{|x|^c} | \langle x, y\rangle \in V\}$.

Last time we introduced two theorems but did not prove either. We will complete a proof for both in this lecture. First, we will show that we can approximate #P using the second level of the polynomial time hierarchy. In the previous class we proved a this fact for the third level, Σ_3^p.

Theorem 1. $(\forall f \in \#\mathrm{P})(\forall a > 0)$ *there exists a function g computable in* poly *time with oracle access to Σ_2^p where* $|f(x) - g(x)| \leq \frac{f(x)}{|x|^a}$

Second we want to show that the power of exact counting yields all of PH. In fact, we can get the entire polynomial hierarchy using only one call to a #P oracle.

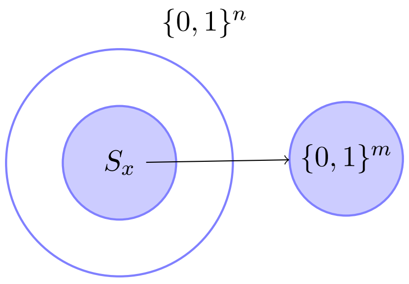
Theorem 2. $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}[1]}$

Crucial to both proofs is the existence of efficient universal families of hash functions that are small. These are small function families that behave in certain respects as if they were random, allowing efficient random sampling. By small we mean we can describe each function h with a small number of bits, $O(n^c + m)$. For completeness, we repeat the definition from last lecture.

Definition 1. *A family of hash functions $H = \{h : \{0,1\}^{n^c} \to \{0,1\}^m\}$ is* universal *if* $\forall y_1 \neq y_2 \in \{0,1\}^{n^c}$, $\forall z_1, z_2 \in \{0,1\}^m$, $\Pr_{h \in_R H}[h(y_1) = z_1 \wedge h(y_2) = z_2] = \frac{1}{2^{2m}}$.

Note that this is the probability in the definition is the same the probability that we would get if h picked output values for y_1 and y_2 uniformly at random.

1 Proof of Theorem 1



We want to prove $(\forall f \in \#\mathrm{P})(\forall a > 0)\exists g$ computable in poly time with access to Σ_2^p where $|f(x) - g(x)| \leq \frac{f(x)}{|x|^a}$.

Our method will proceed in a similar fashion to the proof showing BPP $\subseteq \Sigma_2^p$ back in lecture 15. It will consist of hashing the set of witness strings $S_x \subseteq \{0,1\}^{n^c}$ into a set of roughly the same size without collisions. As $f(x) = |S_x|$, getting a target set of roughly the same size will provide our approximation.

If 2^m is large relative to $|S_x|$, we expect there are no collisions in the image of S_x because our hash functions offer similar behavior to random assignment. When 2^m is small enough, the image of S_x should cover all of $\{0,1\}^m$. Our method will take a small number of hash functions and try to cover $\{0,1\}^m$. Finding the value where this breaks down gives us 2^m as an approximation of the size of S_x.

Recall from last lecture that $|S_x| \geq 2^{m+1} \Rightarrow \Pr_{h_1,...h_m \in_R H}[\forall z \in \{0,1\}^m, z \in \bigcup_{i=1}^m h_i(S_x)] \geq \frac{1}{2}$. Also, $m|S_x| \leq 2^{m-1} \Rightarrow \forall h_1, ...h_m \in H$, $\Pr_{z \in_R \{0,1\}^m}[z \in \bigcup_{i=1}^m h_i(S_x)] \leq \frac{1}{2}$. This second inequality is a simple application of the union bound.

In this way, $(\exists h_1, ...h_m \in H)(\forall z \in \{0,1\}^m)[z \in \bigcup_{i=1}^m h_i(S_x)]$ serves as a distinguisher between the two two states. While this may seem like a Σ_2^p predicate, the test $[z \in \bigcup_{i=1}^m h_i(S_x)]$ cannot be done simply in polynomial time. We need to transform it using an additional clause into $(\exists x, i)[z = h_i(x)]$ to verify, making it a Σ_3^p predicate. Note that if we could swap the leading $\exists$ and $\forall$ phrases we could reduce this to Π_2^p.

Let us return to the beginning and modify the second inequality to $m|S_x| \Rightarrow \forall h_1, ...h_m \in H, \Pr_{z_1,...z_\ell \in_R \{0,1\}^m}[\forall j, z_j \in \bigcup_{i=1}^m h_i(S_x)] \leq \frac{1}{2^\ell}$ by increasing the number of points in the target we sample. Thus $\Pr_{z_1,...z_\ell \in_R \{0,1\}^m}[(\exists h_1, ...h_m \in H)|\forall j, z_j \in \bigcup_{i=1}^m h_i(S_x)] \leq \frac{\text{\# of choices of } h_1,...h_\ell}{2^\ell}$ by the union bound. We want this to be less than 1 to be useful. Note that the first inequality is unchanged by this modification as the property holding for all z lower bounds the probability of the property holding for any set of values $\{z_1, ...z_\ell\}$.

Consider the following predicate:

$$(\forall z_1, ...z_\ell \in \{0,1\}^m)(\exists h_1, ...h_m \in H)[\{z_j\}_{j=1}^m \subseteq \bigcup_{i=1}^m h_i(S_x)] \tag{1}$$

This is a Π_2^p predicate. If the property holds then we can pick $h_1, ...h_m$ to form a covering set. If not, then by a probablistic argument we can pick a choice that causes this to fail. Let m^* be the smallest m for which the above predicate fails. Here $(m^* - 1)2^{m^*-2} < |S_x| < 2^{m^*+1}$ and $|S_x| < 2^{m^*+1} < 8(m^* - 1)|S_x| < 8n^c|S_x|$. Let our estimate be 2^{m*+1}.

The relative error of the polynomial is non-trivial as $|S_x|$ can be exponential in n. We need to tighten this up. Consider changing the intial problem slightly.

Consider $f' = f^{n^d}$. $f' \in \#\text{P}$ as the class is closed under multiplication (using concatenation). Running the above algorithm for f' yields $|S'_x| < 2^{m^*+1} < 8n^{c+d}|S'_x|$. $|S'_x| = |S_x|^{n^d}$.

This lets us approximate $f(x)$ with $2^{\frac{m*+1}{n^d}}$. The error is $\sim 2^{\frac{(c+d)\log n}{n^d}} \sim 1 + \frac{(c+d)\log n}{n^d}$. A choice of d large enough shrinks this second error formula to cn^{-a}. This is the bound we wanted.

2 Exact Counting

We now prove the following result:

Theorem 3. *$PH \subseteq P^{\#P[1]}$.*

We will prove this theorem in the following three parts:

1. NP $\subseteq$ RP$^{\text{UNIQUE-SAT}}$. UNIQUE-SAT is the promise problem defined on SAT formulae that have exactly one or zero satisfying assignments, with the positive instance being the uniquely satisfiable formulas. THE UNIQUE-SAT oracle is guaranteed to give the correct answer on such specified formulae but can act arbitrarily for formulas with multiple satisfying assignments, with inconsistent answers on such queries that are outside of the promise being allowed.

2. Using the first part, we show that PH $\subseteq$ BPP$^{\oplus\text{P}}$.

3. And, we finish the proof by showing BPP$^{\oplus\text{P}}$ $\subseteq$ P$^{\#\text{P}[1]}$.

   In this lecture, we show the proof for parts 1 and 2 and leave part 3 for the next lecture.

2.1 Proof of NP $\subseteq$ RP$^{\text{UNIQUE-SAT}}$

Since SAT is NP-complete, we will show how to solve SAT in RP$^{\text{UNIQUE-SAT}}$ and then the result follows. We use hash functions again for this proof. Consider a NTM solving SAT that runs in time n^c. We look at the set of all possible assignments for the input forumla and consider applying a hash function on these. The idea is to try to choose the range of the hash function about the size of S_x, the set of all satisfying assignments for the formula. If we can do this, then a randomly chosen hash function with high probability will map one of these satisfying assignments to 0^m, giving us a potential UNIQUE-SAT oracle query.

We first bound the probability that a randomly chosen hash function maps a unique satisfying assignment to 0^m. As before, let $\mathcal{H}_m$ be a universal family of hash functions mapping $\{0,1\}^{n^c}$ to $\{0,1\}^m$. Then, the probablity that a randomly chosen h from $\mathcal{H}_m$ maps a unique satisfying assignment to 0^m is given by

$$\Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}=1]=\Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}\geq 1]-\Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}\geq 2]$$

For the first term, we have

$$\Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}\geq 1]\geq \Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}-\sum_{\substack{y_1\neq y_2\\ y_1,y_2\in S_x}}\mathcal{X}_{h(y_1)=0^m}\wedge\mathcal{X}_{h(y_2)=0^m}] \tag{2}$$

$$=|S_x|\cdot\frac{1}{2^m}-\binom{|S_x|}{2}\cdot\frac{1}{2^{2m}}. \tag{3}$$

by considering the first two terms of the inclusion-exclusion principle of the probability and using the pariwise independence property of the hash functions. For the second term, we have

$$\Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}\geq 2]\leq\binom{|S_x|}{2}\cdot\frac{1}{2^{2m}}.$$

Thus,

$$\Pr_{h\in\mathcal{H}_m}[\sum_{y\in S_x}\mathcal{X}_{h(y)=0^m}=1]\geq|S_x|\cdot\frac{1}{2^m}-2\cdot\binom{|S_x|}{2}\cdot\frac{1}{2^{2m}} \tag{4}$$

$$\geq\frac{S_x}{2^m}\cdot\left(1-\frac{|S_x|}{2^m}\right). \tag{5}$$

Now, using the fact that for a function $f(t) = t(1-t)$, the max occurs at $t = \frac{1}{2}$, we get

$$\Pr_{h \in \mathcal{H}_m}[\sum_{y \in S_x} \mathcal{X}_{h(y)=0^m} = 1] \geq \frac{1}{3} \cdot \frac{2}{3} = \frac{2}{9},$$

where m can be picked such that $t \in [\frac{1}{3}, \frac{2}{3}]$ since $\frac{|S_x|}{2^m}$ increases by 2 for each value of m.

Using the above analysis, we have a $\mathrm{RP}^{\text{UNIQUE-SAT}}$ algorithm to solve SAT.

INPUT: formula ϕ

```
(1)    foreach $m = 0, 1, \cdots, n^c$
(2)        Pick $h \in \mathcal{H}_m$ uniformly at random.
(3)        Let $\zeta = (\exists y \in S_x) h(y) = 0^m$
(4)        $\zeta$ is NP-query so convert it into equivalent SAT formula.
(5)        Query UNIQUE-SAT oracle with $\zeta$.
(6)        if Oracle returns yes then Use self-reducibility to find $y$ that satisfies $\zeta$
(7)        if $\phi(y) = 1$ then Output "YES"
(8)    Output "NO"
```

2.2 Proof of PH $\subseteq$ BPP$^{\oplus\mathrm{P}}$.

From the previous proof, we have that NP $\subseteq$ BPP$^{\oplus\mathrm{P}}$ and the proof relativizes so we have that

$$\mathrm{NP}^{\mathrm{A}} \subseteq (\mathrm{BPP}^{\oplus\mathrm{P}})^{\mathrm{A}}$$

for some oracle A. From Homework 1, we know NP $\subseteq$ BPP $\Longrightarrow$ PH $\subseteq$ BPP and the proof of this fact relativizes so

$$\mathrm{NP}^{\mathrm{B}} \subseteq \mathrm{BPP}^{\mathrm{B}} \Longrightarrow \mathrm{PH}^{\mathrm{B}} \subseteq \mathrm{BPP}^{\mathrm{B}}$$

for some oracle B.

Letting the oracle A = $\oplus$P, we get

$$\mathrm{NP}^{\oplus\mathrm{P}} \subseteq (\mathrm{BPP}^{\oplus\mathrm{P}})^{\oplus\mathrm{P}}.$$

In general, giving an additional oracle O_2 to an oracle machine which already has access to O_1 can be solved by the base machine by giving it access to O_1 and $O_1^{O_2}$.

Thus,

$$\begin{aligned} \mathrm{NP}^{\oplus\mathrm{P}} \subseteq (\mathrm{BPP}^{\oplus\mathrm{P}})^{\oplus\mathrm{P}} &= \mathrm{BPP}^{\oplus\mathrm{P}, \oplus\mathrm{P}^{\oplus\mathrm{P}}} && (6) \\ &= \mathrm{BPP}^{\oplus\mathrm{P}}, && (7) \end{aligned}$$

where we leave it as an exercise to prove $\oplus\mathrm{p}^{\oplus\mathrm{P}} = \oplus\mathrm{P}$.

Letting the oracle B = $\oplus$P, because $\mathrm{NP}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}} \Longrightarrow \mathrm{PH}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}$, this means

$$\mathrm{PH} \subseteq \mathrm{PH}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}.$$

3 Next Lecture

We finish the proof of PH $\subseteq \mathrm{P}^{\#\mathrm{P}[1]}$ and begin talking about Interactive Proof Systems.

Acknowledgements

In writing the notes for this lecture, we perused the notes by Mushfeq Khan for lecture 23 from the Spring 2010 offering of CS 710, and the notes by Ashutosh Kumar for lecture 24 from the Spring 2010 offering of CS 710.