COM S 6810 Theory of Computing	February 26, 2009

Lecture 12: Complexity of Counting II

Instructor: Rafael Pass — *Scribe: Navin Sivakumar*

Recall that we would like to prove that given access to an NP oracle, we can probabilistically approximate the #P-complete problem #SAT. Formally, we are proving the following theorem:

Theorem 1 *Given any polynomial p, there exists a p.p.t. A such that for any formula ϕ of length n,*

$$\boldsymbol{Pr}\left[\#\mathsf{SAT}(\phi)\left(1-\frac{1}{p(n)}\right) \leq A^{\mathsf{NP}}(\phi) \leq \#\mathsf{SAT}(\phi)\left(1+\frac{1}{p(n)}\right)\right] \geq 1-2^{-n}.$$

In the last lecture, we showed that given a coarse approximation for #SAT, we can amplify the accuracy of the estimate to satisfy the bounds required by theorem **??**. More precisely, we proved the following proposition:

Proposition 1 *Assume that there exists a p.p.t. B which on input ϕ approximates #SAT(ϕ) within a factor of c for some constant c. Then theorem* **??** *holds.*

We also explicitly constructed an oracle machine which approximates #SAT within a factor of 64, given an oracle O which solves the promise problem GAP-SAT defined below:

Definition 1 *The promise problem GAP-SAT is given by the pair of languages (Π_Y, Π_N) defined as follows:*

$$\begin{aligned}\Pi_Y &= \{(\phi,k) \mid \#\mathsf{SAT}(\phi) > 8k\}\\ \Pi_N &= \left\{(\phi,k) \mid \#\mathsf{SAT}(\phi) < \frac{k}{8}\right\}\end{aligned}$$

Thus, if we can implement the GAP-SAT oracle as a p.p.t., then we will complete the proof of theorem **??**. At first glance, this seems to be a major obstacle because deciding GAP-SAT is considered a hard problem; it would be surprising if we could solve it in polynomial time. However, recall that we are only trying to solve GAP-SAT when we are given access to an NP oracle, which turns out to be a feasible task. Intuitively, we will use the following approach. Given a pair ϕ, k we will modify ϕ to a formula ϕ' in such a way that the number of solutions to ϕ' is a $\frac{1}{k}$ fraction of the number of solutions to ϕ; we will then query the NP-oracle on ϕ'. The idea is that if $(\phi, k) \in \Pi_Y$, then ϕ has at least $8k$ solutions, so ϕ' should still have some remaining solutions. On the other hand, if $(\phi, k) \in \Pi_N$, then ϕ has at most $\frac{k}{8}$ solutions and so ϕ' should not have any solutions.

The approach to reducing the number of solutions to ϕ is to insert additional constraints so that some fraction of the solutions to ϕ fail to satisfy the additional constraints; for example, we might require that the variables $x_1, \ldots, x_k$ satisfy $x_i = 0$. However, if we apply this procedure deterministically, it is possible that, for example, all solutions (or no solutions) to a given formula ϕ satisfy the additional constraints we impose. In order for our construction to work, we must reduce the number of solutions by a factor that is very close to k. Fortunately, it is sufficient for our needs to solve $\mathsf{GAP\text{-}SAT}$ probabilistically. To see how this can help, suppose we have an oracle RO computing a random function from $\{0,1\}^n$ to $\{0,1\}^m$. Then we can generate ϕ' from ϕ by setting

$$\phi'(x) = \phi(x) \wedge (\mathsf{RO}(x) = 0^m).$$

Because RO computes a random function, we expect approximately a fraction of 2^{-m} inputs to satisfy $\mathsf{RO}(x) = 0^m$; moreover, because it is a completely independent random function, we expect approximately a fraction of 2^{-m} of the solutions to $\phi(x)$ to satisfy $\mathsf{RO}(x) = 0^m$. Hence if we set m to be roughly $\log k$, then this construction of ϕ' should satisfy our needs with sufficiently high probability.

Unfortunately, describing a truly random function of this sort requires writing 2^n random m-bit strings, so we cannot compute ϕ' in polynomial time. Therefore, we need to use in place of RO a function which has a short description but still maintains sufficient randomness for our argument to hold. Fortunately, there is a class of functions called *pairwise independent hash functions* which meets these requirements.

Definition 2 *Let $H_{n,m}$ be a family of functions from $\{0,1\}^n$ to $\{0,1\}^m$. Then $H_{n,m}$ is said to be* pairwise independent *if for all $x, y \in \{0,1\}^n$ such that $x \neq y$ and for all $a, b \in \{0,1\}^m$,*

$$\boldsymbol{Pr}_{h \in H_{n,m}}[h(x) = a \wedge h(y) = b] = \frac{1}{2^{2m}}.$$

It is straightforward to verify that $\mathbf{Pr}[h(x) = a] = \frac{1}{2^m}$ for any x, a, from which it is clear that the random variables $h(x)$ for $x \in \{0,1\}^n$ are pairwise independent. Notice that this is essentially the only "randomness" we require from the function.

The following example shows that there are pairwise independent hash functions with short descriptions:

Example 1 *The family of functions $\{h_{a,b}\}_{a,b \in GF(2^m)}$ mapping $GF(2^n)$ to $GF(2^m)$ defined by $h_{a,b}(x) = ax + b$ is pairwise independent (homework exercise).*

We must now show that the pairwise independence property gives sufficient randomness for our purposes. The following theorem is key to establishing this result:

Theorem 2 (Pairwise mixing lemma) *Let $H_{n,m}$ be a family of pairwise independent hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$. Fix $\epsilon > 0$ and let S be a subset of $\{0,1\}^n$ such that $|S| geq \epsilon^{-3} 2^m$. Then*

$$\boldsymbol{Pr}_{h \in H_{n,m}} \left[(1-\epsilon)\frac{|S|}{2^m} \le |\{x \in S \mid h(x) = 0^m\}| \le (1+\epsilon)\frac{|S|}{2^m} \right] > 1 - \epsilon.$$

Proof. Note that it is equivalent to show that

$$\mathbf{Pr}\left[\left| |\{x \in S \mid h(x) = 0^m\}| - \frac{|S|}{2^m} \right| \le \frac{\epsilon |S|}{2^m} \right] > 1 - \epsilon.$$

For ease of notation, we define $Z = |\{x \in S \mid h(x) = 0^m\}|$. The first observation is that $\frac{|S|}{2^m} = \mathbf{E}[Z]$ (i.e. $\frac{|S|}{2^m}$ is the expected size of the set $\{x \in S \mid h(x) = 0^m\}$); we can then apply Chebyshev's inequality to obtain a bound on the probability. To see this, define for each $x \in S$ the indicator function δ_x, where $\delta_x = 1$ if and only if $h(x) = 0^m$. Then $\mathbf{E}[\delta_x] = 2^{-m}$ for each x and

$$\begin{aligned} \mathbf{Var}(\delta_x) &= \mathbf{E}[\delta_x^2] - \mathbf{E}[\delta_x]^2 = \frac{1}{2^m} - \frac{1}{2^{2m}} \\ &= \frac{1}{2^m}\left(1 - \frac{1}{2^m}\right) \end{aligned}$$

Observe also that the random variables δ_x are pairwise independent because $H_{n,m}$ is pairwise independent. Now note that $Z = \sum_{x \in S} \delta_x$. We can now calculate the expected value and variance of Z:

$$\begin{aligned} \mathbf{E}[Z] &= \mathbf{E}[\sum_{x \in S} \delta_x] = \sum_{x \in S} \mathbf{E}[\delta_x] = \frac{|S|}{2^m} \\ \mathbf{Var}[Z] &= \mathbf{Var}[\sum_{x \in S} \delta_x] = \sum_{x \in S} \mathbf{Var}[\delta_x] = \frac{|S|}{2^m}\left(1 - \frac{1}{2^m}\right) \end{aligned}$$

The task therefore reduces to obtaining a lower bound on the probability

$$\mathbf{Pr}\left[|Z - \mathbf{E}[Z]| \le \epsilon \mathbf{E}[Z]\right].$$

We can apply Chebyshev's inequality to show

$$\mathbf{Pr}\left[|Z - \mathbf{E}[Z]| \le \epsilon \mathbf{E}[Z]\right] \ge 1 - \frac{\mathbf{Var}[Z]}{\epsilon^2 \mathbf{E}[Z]^2}.$$

Now we can use the assumption on the size of $|S|$ to obtain the following bound:

$$\begin{aligned} \frac{\mathbf{Var}[Z]}{\epsilon^2 \mathbf{E}[Z]^2} &= \frac{\left(1 - \frac{1}{2^m}\right)}{\epsilon^2 \frac{|S|}{2^m}} \\ &\le \frac{2^m - 1}{\epsilon^2(\epsilon^{-3} 2^m)} < \epsilon \end{aligned}$$

Therefore,

$$\mathbf{Pr}\left[|Z - \mathbf{E}[Z]| \le \epsilon \mathbf{E}[Z]\right] > 1 - \epsilon.$$

Now consider the following machine A operating on input (ϕ, k) with access to an NP oracle:

- Set $m = \lfloor \log k \rfloor$.
- Choose a random $h \in H_{n,m}$ (we are abusing notation by using n to denote the number of variables in ϕ rather than the length of ϕ, k)
- Set $\phi'(x) = \phi(x) \wedge (h(x) = 0^m)$.
- Query oracle on ϕ'; accept if and only if oracle accepts.

The following proposition shows that A solves $\mathsf{GAP\text{-}SAT}$ probabilistically:

Proposition 2 *Define A as above. Then*

$$\begin{aligned}(\phi, k) \in \Pi_Y \quad &\Rightarrow \quad \boldsymbol{Pr}\left[A^{\mathsf{NP}}(\phi, k) = 1\right] > \frac{1}{2} \\ (\phi, k) \in \Pi_N \quad &\Rightarrow \quad \boldsymbol{Pr}\left[A^{\mathsf{NP}}(\phi, k) = 1\right] < \frac{1}{4}\end{aligned}$$

Proof. Suppose $\#\mathsf{SAT}(\phi) > 8k$. Then it is sufficient to show that ϕ' constructed by A is satisfiable with probability at least $\frac{1}{2}$. Let $S_\phi = \{x \mid \phi(x) = 1\}$. Taking $\epsilon = \frac{1}{2}$, we have $|S_\phi| > 8k \ge 8 \cdot 2^m = \epsilon^{-3} 2^m$, where m is computed by A. Therefore, we can apply theorem **??** to get

$$\mathbf{Pr}\left[|\{x \in S_\phi \mid h(x) = 0^m\}| \ge \frac{|S_\phi|}{2^m}(1 - \frac{1}{2})\right] > 1 - \frac{1}{2} = \frac{1}{2}.$$

Observe that $\frac{|S_\phi|}{2^m} \cdot \frac{1}{2} > \frac{8}{2} = 4$, so ϕ' is satisfiable whenever $|\{x \in S_\phi \mid h(x) = 0^m\}| \ge \frac{|S_\phi|}{2^m}(1 - \frac{1}{2})$. Thus A accepts with probability at least $\frac{1}{2}$.

Now suppose $\#\mathsf{SAT}(\phi) < \frac{k}{8}$. Recall that for each x, $\mathbf{Pr}[h(x) = 0^m] = 2^{-m}$. Define S_ϕ as above. Observe that

$$|S_\phi| < \frac{k}{8} \le \frac{2^{m+1}}{8} = \frac{2^m}{4}.$$

Then applying the union bound, the probability that $h(x) = 0^m$ for some x in S_ϕ is at most

$$\frac{2^m}{4} \cdot 2^{-m} = \frac{1}{4}.$$

We now have a p.p.t. which solves GAP-SAT with sufficiently high probability, given access to an NP-oracle; therefore we can apply the results from the previous lecture to complete the proof of theorem **??**.