15-855: Intensive Intro to Complexity Theory
Spring 2009

Lecture 7: **The Permanent, Toda's Theorem, XXX**

1 #P and Permanent

Recall the class of counting problems, $\#P$, introduced last lecture. It was introduced by Valiant in 1979; he was interested in the following intriguing question:

Question: We saw that the natural counting versions of NP-complete problems tend to be #P-complete. Could the counting version of a natural problem in P be #P-complete?

Well, the answer is yes, because of #3DNF: it's #P-complete, but as we saw, deciding 3DNF is trivial. On the other hand, this example seems unfair because it's really the #3CNF problem in disguise. Another example is on the homework, #MONOTONE-SAT; but again, this seems just like SAT in disguise. Valiant's Theorem is a famous "real" example of this phenomenon:

Theorem 1.1. *Given a bipartite graph, counting the number of perfect matchings is #P-complete.*

It's well known that deciding if a bipartite graph has a perfect matching is in P (using a Max-Flow algorithm). Valiant's Theorem is usually stated another way:

Valiant's Theorem: *Computing the* permanent *of matrix of 0's and 1's is #P-complete.*

Definition 1.2. *The* permanent *of the $n \times n$ matrix A is*

$$\mathrm{perm}(A) = \sum_{\substack{\text{permutations} \\ \text{on } [n]}} \pi \prod_{i=1}^{n} A_{i\pi(i)}.$$

Exercise 1.3. *Check that if G is a bipartite graph on $n+n$ nodes, the number of perfect matchings in G is precisely the permanent of the $n \times n$ "adjacency matrix" of G.*

Valiant's Theorem is interesting and important for several reasons. First, it's somewhat surprising that computing the permanent is as hard as solving #3SAT, since computing the *determinant* of a matrix is easy, and the permanent and determinant have very similar-looking formulas. Second, computing the permanent is a beautiful problem because it is both "downward self-reducible" and "randomly self-reducible". We will see the importance of these facts when we discuss the theories of "interactive proofs" and "derandomization".

Since it is such an important theorem, you would expect us to prove it. But we won't! It seems to be the result in complexity theory with the *least* enlightening proof. The outline of the proof is standard and obvious: make a "gadget-based" reduction from #3SAT to #PERFECT-MATCHING. However the gadgets used are very complicated and somewhat magical-seeming.

One reason for this is that one has to avoid reducing SAT to PERFECT-MATCHING! Ben-Dor and Halevi have slightly simplified Valiant's original proof. I don't recommend reading this paper though; instead, try reading the proof in Papadimitrious's book. If you can't obtain it, try either the Arora-Barak book (which has an elegant-looking but sketched proof) or the Goldreich book (which heroically does it in a long, detailed way).

2 Toda's Theorem

Toda's Theorem says that solving #3SAT exactly is extremely hard — much harder than approximating it. Specifically, it's at least as hard as any problem in the Polynomial-Time Hierarchy. We will see a proof of the following:

Theorem 2.1. *(Slightly weak version of Toda's Theorem.)* $\mathsf{PH} \subseteq \mathsf{BPP}^{\#\mathsf{P}}$.

I.e., given an oracle for #3SAT, you can solve any problem in the Hierarchy with an efficient, zero-error randomized algorithm. In particular, it follows that it's very unlikely that #3SAT can be solved in $\mathsf{BPP}^{\mathsf{NP}}$ like Approximate-#3SAT can. The reason is that this would imply (via Lautemann's Theorem, $\mathsf{BPP} \in \Sigma_2$) that PH is in some *fixed* level of the Hierarchy (specifically, the 3rd, if you're careful), contradicting the Infinite Hierarchy Assumption.

Although Theorem 2.1 already contains the main point — solving $\#\mathsf{P}$ problems is harder than anything in the Hierarchy — the full Toda's Theorem has some slight extra touches:

Theorem 2.2. $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}[1]}$.

There are two advantages here. First, the [1] indicates that you only need to make 1 call to the $\#\mathsf{P}$ oracle. Second, there is a "derandomization" from BPP to P.

The proof of Toda's Theorem passes through a rather goofy complexity class called $\oplus\mathsf{P}$:

Definition 2.3. $\oplus\mathsf{P}$ *is the class of all languages L such that there is a polynomial-time nondeterministic Turing Machine M such that for all x,*

$$x \in L \Leftrightarrow M(x) \text{ has an odd number of accepting computation paths.}$$

You can think of $\oplus\mathsf{P}$ as the "least significant bit" version of $\#\mathsf{P}$. By the parsimonious Cook reduction, we get:

Proposition 2.4. *$\oplus$SAT, the problem of determining if $\#\varphi$ is odd for a given formula φ, is $\oplus\mathsf{P}$-complete under parsimonious reductions.*

2.1 Outline of the proof

The most basic theorem about PH is the Collapse Theorem: If $\mathsf{NP} = \mathsf{P}$ then $\mathsf{PH} = \mathsf{P}$. In other words, if NP can be solved efficiently, the whole Hierarchy can be solved efficiently. Now actually, we usually identify "practical efficiency" with BPP, not P. Does the Collapse Theorem still hold? Although the proof is not completely straightforward, the answer is yes. The following is a good exercise:

Theorem 2.5. $\mathsf{NP} \subseteq \mathsf{BPP} \Rightarrow \mathsf{PH} \subseteq \mathsf{BPP}$.

The basic idea behind Toda's Theorem is that we have an interesting *unconditional* containment for NP, from the Valiant-Vazirani Theorem. This theorem roughly speaking tells us that there is a randomized reduction from SAT to Unique-SAT. But of course, if you have an oracle for $\oplus$SAT, you can immediately use it to distinguish whether a formula has 0 or 1 satisfying assignments. Hence:

Corollary 2.6. *(of Valiant-Vazirani)* $\mathsf{NP} \subseteq \mathsf{RP}^{\oplus\mathsf{P}} \subseteq \mathsf{BPP}^{\oplus\mathsf{P}}$.

Given this, one can imagine repeating the proof of Theorem 2.5, but with $\mathsf{BPP}^{\oplus\mathsf{P}}$ as the notion of "practical efficiency"! This is essentially what we'll do, although one has to be rather careful in the proof. The key aspect is that "$\oplus\mathsf{P}$ calculations" can be composed with one another.[1] The ingenuity of Toda's Theorem is in working with neither "Unique-SAT" (as one would guess from Valiant-Vazirani) nor $\#\mathsf{P}$ (the class of interest), but rather with this goofy but flexible intermediate class.

We will not fully prove Theorem 2.2 (sometimes called "Toda's Second Theorem"). Instead we will prove something halfway between Theorems 2.1 and 2.2 (sometimes called "Toda's First Theorem"):

Theorem 2.7. $\mathsf{PH} \subseteq \mathsf{BPP}^{\oplus\mathsf{P}[1]}$.

This comes pretty close to the full Toda's Theorem, Theorem 2.2 — in fact, we're using an even weaker oracle than it does. We will not actually show how to get the full theorem, "derandomizing" BPP to P (at the expense of calling a $\#\mathsf{P}$ oracle once rather than a $\oplus\mathsf{P}$ oracle once). It is a bit of a funny trick for reducing $\mathsf{BPP}^{\oplus\mathsf{P}}$ to $\mathsf{P}^{\#\mathsf{P}[1]}$. You can find the trick in many Complexity Theory lecture notes.

2.2 Proof of Theorem 2.2

Let $c \in \mathbb{N}$ and recall the problem Σ_cSAT: Given is a size-n "c-level quantified formula" of the form

$$\Phi = \exists x_1 \forall x_2 \exists x_3 \cdots Q x_c \varphi(x_1, \ldots, x_c);$$

The task is to decide if the formula is true or not. As we saw, this problem is complete for Σ_c. Similarly, Π_cSAT is complete for the class Π_c.

To prove Theorem 2.2, we will look at quantified formulas with a new type of quantifier, $\oplus$. A formula

$$\oplus x \in \{0,1\}^n \Psi(x)$$

is true iff the number of x's for which $\Psi(x)$ is true is *odd*. This new quantifier has similarities and differences when compared with $\exists$ and $\forall$:

Proposition 2.8.

1. *Like $\exists$ and $\forall$, the $\oplus$ quantifier "merges": the formula $\oplus x \in \{0,1\}^n \oplus y \in \{0,1\}^m \Phi(x,y)$ is equivalent to $\oplus(x,y) \in \{0,1\}^{n+m} \Phi(x,y)$.*
2. *Unlike $\exists$ and $\forall$, the $\oplus$ quantifier can't be tacked on with unused variables: assuming Ψ does not mention x, we don't in general have $\oplus x\ \Psi \equiv \Psi$.*
3. *Peculiarly, $\oplus x\ \neg\Phi(x)$ always has the same truth value as $\oplus x \Phi(x)$.*

[1] In fact, here is another good exercise: Show $\oplus\mathsf{P}^{\oplus\mathsf{P}} = \oplus\mathsf{P}$.

The proofs of these statements are exercises.

Finally, to show Theorem 2.2, it suffices to show the following:

Theorem 2.9. *For any $c \geq 1$, there is an efficient randomized reduction that takes a Σ_cSAT or $\oplus\Sigma_c$SAT formula Φ and outputs a $\oplus\Pi_{c-1}$SAT formula Φ' which has the same truth value as Φ with probability at least $1 - 1/(10c)$.*

This suffices to put $\Sigma_c \in \mathsf{BPP}^{\oplus\mathsf{P}[1]}$: First, we can go from a Σ_cSAT formula to a $\oplus\Pi_{c-1}$SAT formula which is equivalent except with probability $1/(10c)$. This formula has the form $\oplus x \forall y \Psi$. By De Morgan's laws, this is the same as $\oplus x \neg \exists y \neg \Psi$. By Proposition 2.8.3, this is equivalent to $\oplus x \exists y \neg \Psi$, which is a $\oplus\Sigma_{c-1}$SAT formula. Now we repeat, going to a $\oplus\Sigma_{c-2}$SAT formula, $\oplus\Sigma_{c-3}$SAT formula, etc., ending up with a $\oplus$SAT formula having the same truth value as the original Σ_cSAT formula, except with probability $1/10$. Finally, we determine the truth of the $\oplus$SAT formula with one call to the oracle.

Proof. We assume the initial formula Φ is a $\oplus\Sigma_c$SAT formula; the case where it is just Σ_cSAT is a simplification. So express Φ as

$$\Phi = \oplus x \exists y \Psi(x, y),$$

where $\Psi(x, y)$ is a Π_{c-1}SAT formula. The idea is to use the Valiant-Vazirani reduction on the "$\exists y \Psi(x, y)$" part. Say y ranges over $\{0,1\}^\ell$. Valiant-Vazirani would involve picking a random hash function h on $\{0,1\}^\ell$ and outputting a formula

$$\Theta = \Psi(x, y) \wedge \text{“}h(y) = 0\text{”}.$$

Here the "formula" for $h(y) = 0$ will also have some auxiliary variables z in it. Note that for each x, y, z, the formula Θ is equivalent to a Π_{c-1}SAT formula, since Ψ is Π_{c-1}SAT.

The Valiant-Vazirani guarantee is that:

For each x,

$$\exists y \Psi(x, y) \text{ false} \Rightarrow \Theta(x, y, z) \text{ unsatisfiable w.p. } 1 \Rightarrow \oplus(y, z)\Theta(x, y, z) \text{ false w.p. } 1,$$

$$\exists y \Psi(x, y) \text{ true} \Rightarrow \Theta(x, y, z) \text{ uniquely satisfiable w.p. } \geq \frac{1}{8\ell} \Rightarrow \oplus(y, z)\Theta(x, y, z) \text{ true w.p. } \geq \frac{1}{8\ell}.$$

We now need two main ideas. First, we need to boost this probability of $1/8\ell$ to something extremely close to 1, because we want to take a *union bound* over all x's. So we run the Valiant-Vazirani reduction $t = O(m\ell \log c)$ times independently, where m is the number of x_i variables, producing $\Theta_1, \ldots, \Theta_t$. Then:

For each x,

$$\exists y \Psi(x, y) \text{ false} \Rightarrow \oplus(y, z)\Theta_1(x, y, z), \ldots, \oplus(y, z)\Theta_t(x, y, z) \text{ all false w.p. } 1,$$

$$\exists y \Psi(x, y) \text{ true} \Rightarrow \text{ at least one of } \oplus(y, z)\Theta_1(x, y, z), \ldots, \oplus(y, z)\Theta_t(x, y, z) \text{ true w.p. } \geq 1 - \frac{2^{-m}}{10c}.$$

Note here that the random Valiant-Vazirani reduction *does not depend on* x; i.e., the t hash functions chosen are independent x. So we can union-bound over all $x \in \{0,1\}^m$ and conclude that, with probability at least $1 - 1/(10c)$ over the choice of the hash functions (and hence $\Theta_1, \ldots, \Theta_t$),

$$\Phi = \oplus x \exists y \Psi(x,y) \text{ has the same truth value as } \oplus x \left(\bigvee_{i=1}^{t} \oplus(y,z) \Theta_t(x,y,z) \right). \tag{1}$$

The second and last main idea in the proof is to convert the parenthesized expression in (1), viz.,

$$(\oplus(y,z)\Theta_1(x,y,z)) \vee \cdots \vee (\oplus(y,z)\Theta_t(x,y,z)), \tag{2}$$

to some $\oplus\Pi_{c-1}$SAT formula $\Xi(x)$ with the same truth value. This will complete the proof because $\oplus x \Xi(x)$ is also a $\oplus\Pi_{c-1}$SAT formula (using Proposition 2.8.1).

For each i, let c_i denote the number of (y,z)'s for which $\Theta_i(x,y,z)$ is true. We would like a single $\oplus$-formula which is true if and only if *at least one of the c_i's is odd*. Observe that this holds if and only if $(c_1+1)(c_2+1)\cdots(c_t+1)+1$ is odd. Now we do some tricks. Let b be a single-bit variable. Then the number of (y,z,b)'s for which

$$\widetilde{\Theta}_i := (b = 0 \wedge \Theta_i(x,y,z)) \vee (b = 1 \wedge y = \overline{0} \wedge z = \overline{0})$$

is true is precisely $c_i + 1$. Next, introduce t copies (y^i, z^i, b^i) of the variable set (y,z,b). Then the number of $(y^1, z^1, b^1, \ldots, y^t, z^t, b^t)$ for which

$$\Xi' = \widetilde{\Theta}_1(y^1, z^1, b^1) \wedge \cdots \wedge \widetilde{\Theta}_t(y^t, z^t, b^t)$$

is true is exactly $(c_1+1)\cdot\cdots\cdot(c_t+1)$. Finally, we can use the same trick we used in going from Θ_i to $\widetilde{\Theta}_i$ to get a formula Ξ'' for which the number of satisfying assignments is $(c_1+1)\cdot\cdots\cdot(c_t+1)+1$. It is easy to check that this Ξ'' is equivalent to a Π_{c-1}SAT formula, because the Θ_i's and hence $\widetilde{\Theta}_i$'s are. Hence $\Xi = \oplus(y^1, \ldots, b^t, b')\Xi''$ is the desired $\oplus\Pi_{c-1}$SAT formula equivalent to (2), completing the proof. □