CS538, Spring 1998

Scribe for Lectures on 4/20,22/98

by Takahiro Murata

In these notes, we prove Toda's Theorem, which says the polynomial hierarchy is contained in the class of languages recognized by polynomial time deterministic machines with an oracle in PP, the class of languages recognized by probabilistic machines with unbounded two-sided error. The development is divided into three major parts; (1) NP $\subseteq$ BPP$^{\oplus \mathrm{P}}$ is shown, which is also known as the theorem of Valiant and Vazrani; then (2) PH $\subseteq$ BPP$^{\oplus \mathrm{P}}$ follows; and finally (3) BPP$^{\oplus \mathrm{P}} \subseteq$ P$^{\mathrm{PP}}$ concludes.

Lemma 1 (Isolation Lemma) *Let $G = \langle V, E \rangle$ be a graph.* [1] *Randomly pick a weight $w_i \in \{1, 2, \ldots, 4n^4\}$ for each edge $e \in E$ where $n = |V|$. Then, with probability $\geq 3/4$, $\forall x, y \in V$ there are not two minimum-weight paths from x to y where the weight of path $p = \langle v_0, \ldots v_i \rangle$ is:*

$$\Sigma_{j=0}^{i-1} weight(\langle v_j, v_{j+1} \rangle).$$

Proof: Let us call the following situation $BAD(x, y, e)$: when there are two minimum-weight paths from x to y, an edge e is on one of them, but not on the other.

$$\begin{aligned}
&\text{Prob(there exist two minimum-weight } x \rightsquigarrow y \text{ paths)} \\
&= \text{Prob}(\exists x \exists y \exists e\, BAD(x, y, e)) \\
&= \Sigma_{(x,y,e)} \text{Prob}(BAD(x, y, e)) \\
&= \Sigma_{(x,y,e)} \Sigma_{w' : \text{wt assignment to all edges except } e} \text{Prob}(BAD(x, y, e) | w') \cdot \text{Prob}(w') \\
&\leq \Sigma_{(x,y,e)} \Sigma_{w'} \frac{1}{4n^4} \cdot \text{Prob}(w') \\
&= \Sigma_{(x,y,e)} \frac{1}{4n^4} \Sigma_{w'} \text{Prob}(w') \\
&= \Sigma_{(x,y,e)} \frac{1}{4n^4} \quad (\text{since } \Sigma_{w'} \text{Prob}(w') = 1) \\
&\leq \frac{1}{4}
\end{aligned}$$

Note in the above, we need the following claim:

Claim 1 *Prob$(BAD(x, y, e)|w') \leq \frac{1}{4n^4}$.*

This is argued as follows: Assume that picking $c \in \{1, \ldots, 4n^4\}$ as the weight for e causes $BAD(x, y, e)$ to happen. Consider what happens if we choose some weight other than c to be the weight for e. Recall when $BAD(x, y, e)$ happens, e is on one of such paths but not on the other. Then, if the value of c were greater, the weight of the path on which e is would become greater than the other, causing it not to be the minimum one. If the value of c were smaller, the weight of the path on which e is would be less than the other, contradicting that the other one is one of the minimum ones. Therefore, once all the edges are given their weight by w', there is only one weight to cause $BAD(x, y, e)$ if it is possible at all. $\square$

Corollary 1 (Valiant & Vazrani) NP $\subseteq$ BPP$^{\oplus \mathrm{P}}$.

[1] This lemma comes with various settings. Here we talk of it in terms of paths in a given graph.

Proof: Let $A \in \mathrm{NP}, A = L(M)$ and M runs in time n^l. Let

$$B = \{(x, w, k) : M \text{ has an odd } \# \text{ of accepting paths on input } x \text{ having weight } k\}$$

where $w : \{1, \ldots, n^l\} \to \{1, \ldots, 4n^l\}$; the weight of a path $p \in \{L, R\}^{n^l} = \Sigma_{i, i-\text{th symbol of } p = L} w(i)$. Observe that $B \in \oplus\mathrm{P}$. For a $\mathrm{BPP}^{\oplus\mathrm{P}}$, or in fact $\mathrm{RP}^{\oplus\mathrm{P}}$, algorithm, consider the following:

On input x
  randomly pick w;
  for $k := 0$ to $4n^{2l}$
    ask if $(x, w, k) \in B$;
    if so, then halt and accept
  end-for;
  if the control reaches here, then halt and reject

Observe that the above program runs in polynomial time while the for-loop making sure to pick up the minimum weight given the weight assignment w. Since a single minimum weight path is separated with the probability $\geq 3/4$ by Isolation Lemma, if input x is in A, the above program accepts x with the probability $\geq 3/4$, and if x is not in A, (x, w, k) would never be in B for any w and k, and thus, the program rejects x. $\square$

Also, note that the above corollary relativizes with respect to any oracle D.

The following is the next lemma to show.

Lemma 2 $\mathrm{NP}^{\mathrm{NP}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}$.

Proof: Due to Corollary 1 and its relativized version,

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

Problem 5 in our assignment 6 shows $\oplus\mathrm{P}^{\mathrm{BPP}^D} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}^D}$ for any oracle D. Therefore, the last set in the above inclusion sequence is reduced to $\mathrm{BPP}^{\mathrm{BPP}^{\oplus\mathrm{P}^{\oplus\mathrm{P}}}}$, which is further reduced to $\mathrm{BPP}^{\oplus\mathrm{P}}$ thanks to the following two claims. Note again, however, that this lemma as well as each intermediate step in the proof relativizes with respect to any oracle D.

Claim 2 $\oplus\mathrm{P}^{\oplus\mathrm{P}} = \oplus\mathrm{P}$.

Proof: We need some preparation for the claim as follows:

Subclaim 1 *Let $f(1^i, x)$ be a $\#\mathrm{P}$ function. Then, for all k,*

$$g(x) = \Pi_{i=1}^{n^k} f(1^i, x) \in \#\mathrm{P}.$$

Proof: Let M be the NP machine such that $\#\mathrm{acc}_M(1^i, x) = f(1^i, x)$. Then, consider the following machine M':

On input x $(n = |x|)$
  for $i := 1$ to n^k
    run M on input $(1^i, x)$;
    if this rejects, then halt and reject
    else continue

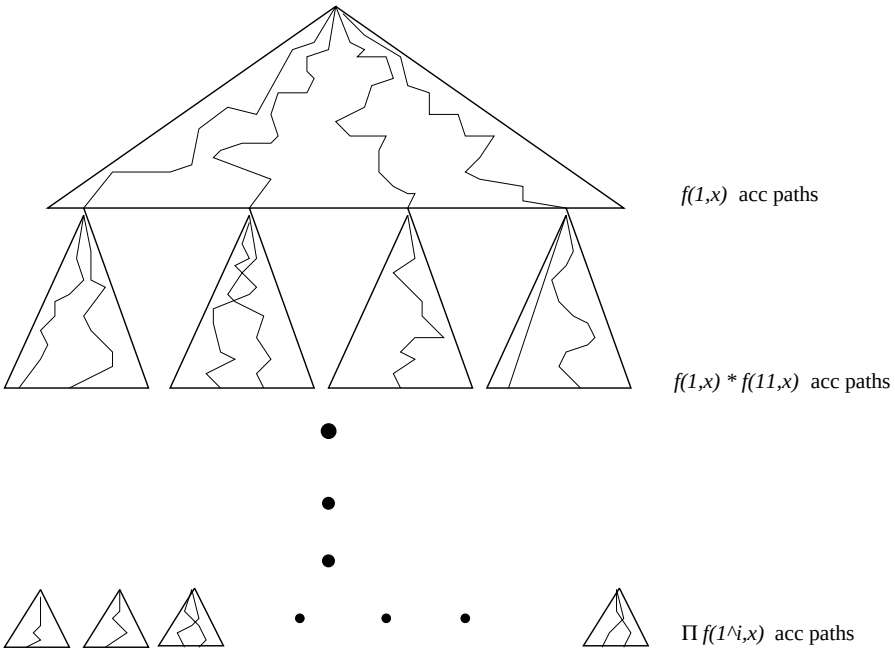

Figure 1: picture for Subclaim 1

Observe with the help of Figure 1 that $g(x) = \#\mathrm{acc}_{M'}(x)$.

Subclaim 2 *Let $f(x,y)$ be a $\#\mathrm{P}$ function. Then,*

$$g(x) = \Sigma_{y,|y|=|x|^k} f(x,y) \in \#\mathrm{P}.$$

Proof: Let M be the machine such that $\#\mathrm{acc}_M(x,y) = f(x,y)$. Consider the following machine M':

On input x
compute $|x|^k$;
guess y of length $|x|^k$;
run $M(x,y)$

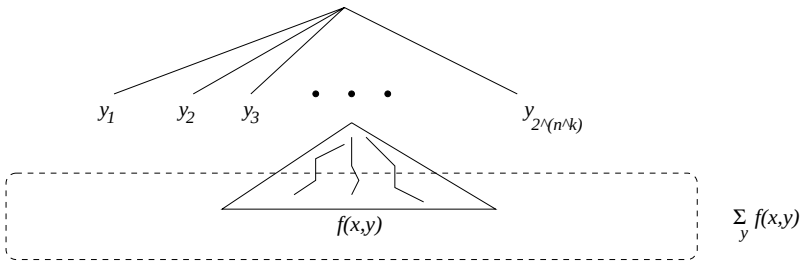

Figure 2: picture for Subclaim 2

Observe with the help of Figure 2 that $g(x) = \#\mathrm{acc}_{M'}(x)$.

For the claim, let $A \in \oplus\mathrm{P}^{\oplus\mathrm{P}}$, i.e., $A \in \oplus\mathrm{P}^B$ for some $B \in \oplus\mathrm{P}$. Let A be accepted by M^B, i.e., M is an NP machine such that:

$$x \in A \iff \#\mathrm{acc}_{M^B}(x) \equiv 1 \pmod 2.$$

We also assume with no loss of generality M makes exactly $|x|^k$ queries along any path. Also, let B be accepted by an NP machine N. Let $f(1^i, x, y, z)$ be the #P function given by the following NP machine:

On input $(1^i, x, y, z)$
  simulate $M(x)$ along the path given by y using the sequence
    of bits in z as the oracle answers until the i-th query
    is asked (call this query input w);
  if the i-th bit of z is 1 then
    run $N(w)$;
  if the i-th bit of z is 0 then
    flip a coin and
      accept on one path & run $N(w)$ on the other;
  if $i = |x|^k + 1$ then // i-th query not asked
    accept $\iff$ M accepts along this path y

Note that for $i \leq |x|^k$:

$$\#\mathrm{acc}_N(w) + (1 - z_i) \equiv f(1^i, x, y, z) \pmod 2$$

where z_i is the i-th bit of z, and w is the i-th query asked, or equivalently:

$$f(1^i, x, y, z) \equiv 1 \pmod 2 \iff \chi_B(w) = z_i.$$

Also, observe that for each x and y, there is exactly one z ($|z| = |x|^k$, henceforth called z_{xy}) such that the bits of z correctly give the answers that oracle B would give. Therefore, together with what we noted above:

$$\text{If } z \neq z_{xy} \text{ then } \Pi_{i=1}^{n^k+1} f(1^i, x, y, z) \equiv 0 \pmod 2.$$

Therefore:

$$\Sigma_{z,|z|=n^k} \Pi_{i=1}^{n^k+1} f(1^i, x, y, z) \equiv 1 \pmod 2$$

$$\iff$$

$$M^B(x) \text{ accepts along path } y.$$

Now define:

$$g(x) = \Sigma_{y,|y|=n^k} \Sigma_{z,|z|=n^k} \Pi_{i=1}^{n^k+1} f(1^i, x, y, z).$$

Due to the two subclaims we have shown above, g is a #P function. Then, observing that $x \in A \iff g(x) \equiv 1 \pmod 2$, we conclude. □

Claim 3 $\mathrm{BPP}^{\mathrm{BPP}} = \mathrm{BPP}$.

Proof: Let $A \in \mathrm{BPP}^B$ where $B \in \mathrm{BPP}$. Let B be accepted by a machine M that runs in time n^k for some constant k with error probability $\leq 1/2^{n^4}$. Note that most sequences z of length n^k has the property that:

$$M(y, z) = \chi_B(y).$$

To simulate our BPP^B algorithm for A, first guess such a z, and then use $M(y, z)$ instead of an oracle for $y \in B$. This also concludes the proof of Lemma 2. □

Corollary 2 $\mathrm{PH} \subseteq \mathrm{BPP}^{\oplus \mathrm{P}}$.

Using Lemma 2, this follows by induction on the level of the hierarchy.

> **Remark:** AC^0 can be simulated by probabilistic depth 2 circuit of size $2^{\log^{O(1)} n}$, and this is useful in showing that certain functions require very large AC^0 circuits.

Before showing the final lemma, let us acknowledge the following fact for a moment (this will be shown in the next lecture):

Fact 1 *Let $k \in \mathbf{N}$, and let $f \in \#\mathrm{P}$. Then, there exists $g \in \#\mathrm{P}$ such that*

$f(x)$ *is odd* $\implies g(x) \equiv 1 \pmod{2^{n^k}}$,
$f(x)$ *is even* $\implies g(x) \equiv 0 \pmod{2^{n^k}}$.

Now, let us see the final lemma:

Lemma 3 $\mathrm{BPP}^{\oplus \mathrm{P}} \subseteq \mathrm{P}^{\mathrm{PP}}$.

Proof: Recall that $\mathrm{P}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}}$ (by our assignment #6, problem 1). Let $A \in \mathrm{BPP}^{\oplus \mathrm{P}}$, where A is accepted by M^B and let f be the #P function for B. Let n^k be the running time of M. Assume first that M makes only one query along any path. Then, let $g(x, y)$ be a #P function that is defined to be the number of accepting paths of the following machine:

On input x, y
  run $M(x)$ along path y;
    when a query "$w \in B$?" is made,
    then flip a coin $c \in \{0, 1\}$ and
      use this as the oracle answer and
      continue simulating $M(x)$;
  if this simulation accepts, then generate $f(w) + (1 - c)$ paths and accept

Observe that:

$$g(x, y) \text{ is odd} \iff M^B(x) \text{ accepts with probabilistic sequence } y.$$

For $g(x, y)$ above, consider a #P function $g'(x, y)$ which has the warranted property by Fact 1, i.e.:

$g(x, y)$ is odd $\implies g'(x, y) \equiv 1 \pmod{2^{n^k}}$,
$g(x, y)$ is even $\implies g'(x, y) \equiv 0 \pmod{2^{n^k}}$.

Then, define:

$$h(x) = \Sigma_y g'(x, y).$$

First, observe that $h(x)$ is a #P function due to Subclaim 2. Also, observe that the value $h(x) \pmod{2^{n^k}}$ represents the number of y's such that $M^B(x)$ accepts along path y. Therefore, our $\mathrm{P}^{\#\mathrm{P}}$ algorithm, on input x, using the oracle $h(x)$, decides if the following holds:

$$h(x) \geq \frac{1}{2} 2^{n^k} \pmod{2^{n^k}}.$$

If so, x is accepted, and if not x is rejected. Observing this algorithm accepts A, we conclude under the simplified assumption that the number of oracle queries made by M is one on any path.

If M makes more than one query, modify $g(x, y)$ as follows:

On input x, y
repeat
  run $M(x)$ along path y;
    when a query "$w \in B$?" is made,
    then flip a coin $c \in \{0,1\}$ and
      generate $f(w) + (1 - c)$ paths,
      use c as the oracle answer, and continue simulating $M(x)$
until no more queries are asked;
if the simulation of $M(x)$ along path y accepts with this sequence of guessed oracle queries
  then accept
  else reject

Let us call this machine N.

Claim 4 *M^B accepts x along y if and only if $\#acc_N(x, y)\,(= g(x, y)) \equiv 1 \pmod 2$.*

This is observed by the following:

$$\begin{aligned} &\#\mathrm{acc}_N(x, y) \\ &= \Sigma_{\text{coin-flip seq } s \in \{0,1\}^*} \#\text{acc paths generated by } N(x, y) \text{ with "guess sequence" } s \\ &\equiv \#\text{acc paths generated by } N(x, y) \text{ with the correct guess } s \pmod 2. \end{aligned}$$

Noting that the property in the above claim is the only one we relied on regarding $g(x, y)$ in the proof for the simplified case, we conclude. □

Theorem 1 (Toda) $\mathrm{PH} \subseteq \mathrm{P}^{\mathrm{PP}}$.

Immediate by combining Corollary 2 and Lemma 3.