1 Administrative

- PS2 is graded and available from Swastik.
- PS3 is out and due Friday 4/6.

2 Overview

In this lecture we prove the two containments of Toda’s Theorem:

1. \( \text{PH} \subseteq \text{BP} \cdot \oplus \cdot \text{P} \)
2. \( \text{BP} \cdot \oplus \cdot \text{P} \subseteq \text{P}^\# \text{P} \)

3 Operator Definitions

The alternation operators acting on a language \( L \) or complexity class \( C \) are defined as follows:

\[
\exists \cdot L = \{ x | \exists y \text{ s.t. } (x, y) \in L \} \quad \exists \cdot C = \{ \exists \cdot L | L \in C \}
\]

\[
\forall \cdot L = \{ x | \forall y \text{ s.t. } (x, y) \in L \} \quad \forall \cdot C = \{ \forall \cdot L | L \in C \}
\]

\[
\oplus \cdot L = \{ x | \#y \text{ s.t. } (x, y) \in L \text{ is even} \} \quad \oplus \cdot C = \{ \oplus \cdot L | L \in C \}
\]

\[
\text{BP} \cdot L = (\Pi_{Y \in S}, \Pi_{N O}) \quad \text{BP} \cdot C = \{ \text{BP} \cdot L | L \in C \}
\]

\[
\Pi_{Y \in S} = \{ x | \text{Pr}_{y}(x, y) \in L \geq 1 - \frac{1}{2^{q(n)}} \} \\
\Pi_{N O} = \{ x | \text{Pr}_{y}(x, y) \in L \leq \frac{1}{2^{q(n)}} \}
\]

The classes in \( \text{PH} \) can be defined using these operators: \( \Sigma^P_k \exists \cdot \forall \cdot \exists \cdot \ldots \cdot \text{P} \) and \( \Pi^P_k = \forall \cdot \exists \cdot \forall \cdot \ldots \cdot \text{P} \).

4 Toda’s Theorem 1: \( \text{PH} \subseteq \text{BP} \cdot \oplus \cdot \text{P} \)

Lemma 1 \( \text{BP} \cdot \oplus \cdot \text{P} = \text{BP} \cdot \oplus \cdot \text{BP} \cdot \oplus \cdot \text{P} \)

First we note that for a complexity class \( C \) closed under complement, \( \text{BP} \cdot C \) is closed under complement since the error bounds for \( \text{BP} \) are symmetric. \( \oplus \cdot C \) is also closed under complement since by a method from Lecture 12 we can add exactly one satisfying assignment to a formula, flipping the parity. Therefore all the classes we’re dealing with are closed under complement. The method from Lecture 12 also allows us to exchange ‘odd’ for ‘even’ in the definition of \( \oplus \).

We see that \( \text{BP} \cdot \oplus \cdot \text{P} \subseteq \text{BP} \cdot \oplus \cdot \text{BP} \cdot \oplus \cdot \text{P} \) trivially. The opposite containment can be shown based on some general properties:

Property 1.1 \( \oplus \cdot \oplus \cdot C = \oplus \cdot C \).
Proof For $L \in C$, let $L' = \oplus \cdot L$ and $L'' = \oplus \oplus \cdot L$. Now,

$$x \in L'' \iff \# y_1 \text{ s.t. } (x, y_1) \in L' \text{ is odd}$$

$$\iff \sum_{y_1} L'(x, y_1) \equiv 1 \pmod{2}$$

$$\iff \sum_{y_1} \sum_{y_2} L(x, y_1, y_2) \equiv 1 \pmod{2}$$

$$\iff \sum_{(y_1, y_2)} L(x, y_1, y_2) \equiv 1 \pmod{2}$$

$$\iff \# (y_1, y_2) \text{ s.t. } (x, y_1, y_2) \in L \text{ is odd}$$

$$\iff x \in L'$$

So, $L'' = L'$. ■

Property 1.2 $BP \cdot BP \cdot C \subseteq BP \cdot C$

Proof Again, for $L \in C$ let $L'' = BP \cdot BP \cdot L$ and $L' = BP \cdot L$. Now,

$$x \in L'' \Rightarrow Pr_{y_1}[(x, y_1) \in L] \geq 1 - 2^{-q_1(n)}$$

$$\Rightarrow Pr_{y_1} [Pr_{y_2}[(x, y_1, y_2) \in L] \geq 1 - 2^{-q_2(n)}] \geq 1 - 2^{-q_1(n)}$$

$$\Rightarrow Pr_{y_1, y_2}[(x, y_1, y_2) \in L] \geq 1 - 2^{-q_1(n)} - 2^{-q_2(n)}$$

$$\Rightarrow x \in L'$$

Since we can similarly show that $x \notin L'' \Rightarrow x \notin L'$, $L'' = L'$. ■

Property 1.3 $\oplus \cdot BP \cdot C \subseteq BP \oplus \cdot C$

Proof Fix $L \in C$. We will show that $\oplus \cdot BP \cdot L \subseteq BP \cdot \oplus \cdot C$.

Now, $\oplus \cdot BP \cdot L = \{x | \# y \text{ s.t. } Pr_z[(x, y, z) \in L] \geq 1 - 2^{-q_1(n)} \text{ is even}\}$.

We define $\tilde{L} = \{x | Pr_z[\# y \text{ s.t. } (x, y, z) \in L \text{ is even}] \geq 1 - 2^{-q_2(n)}\}$. $\tilde{L}$ is clearly $\in BP \cdot \oplus \cdot C$. We will show that for proper $q_1(n)$ and $q_2(n)$, $\oplus \cdot BP \cdot L = \tilde{L}$:

Define $L' = BP \cdot L = \{(x, y) | Pr_z[(x, y, z) \in L] \geq 1 - 2^{-q(n)}\}$.

Fix $x$. We say that $z$ is bad for $y$ if $L(x, y, z) \neq L'(x, y)$.

When we fix $y$, we see that:

$$Pr_z[z \text{ is bad for } y] \leq 2^{-q(n)}$$

$$Pr_z[\exists y \text{ s.t. } z \text{ is bad for } y] \leq 2^l 2^{-2q(n)} \text{ (where } l = |y|)$$

$$Pr_z[z \text{ is good for all } y] \geq 1 - 2^l 2^{-q(n)}$$

$$Pr_z[\forall y L(x, y, z) = L'(x, y)] \geq 1 - 2^l 2^{-q(n)}$$

By choosing $q(n)$ sufficiently large, we can get that for most $z$'s, $\forall y L(x, y, z) = L'(x, y)$. So, for most $z$'s, $\oplus \cdot L(x, z) = \oplus \cdot L'(x)$. Now, this is equivalent to saying that $\tilde{L} = \oplus \cdot L'(x) = \oplus \cdot BP \cdot L(x)$. ■

Lemma 2 $\exists \cdot C \subseteq BP \cdot \oplus \cdot C$

11-2
Proof Let’s look at some language \( L \in C \). By the same argument given by Valiant-Varizani (an RP reduction), there is a language \( L' \in C \) such that

\[
\exists y : (x, y) \in L \iff \Pr_z[\#y : (x, y, z) \in L' \text{ is even}] \geq 1/p(n) \tag{1}
\]

and

\[
\forall y : (x, y) \notin L \iff \Pr_z[\#y : (x, y, z) \in L' \text{ is even}] = 0 \tag{2}
\]

Now, the only problem is that RP only needs a probability of \( 1/p(n) \) when \( x \in L \), but strong-BP needs a probability \( 1 - \frac{1}{2^{-\frac{1}{n}}} \). But it is easy to get the stronger probability.

\[
L^k = \{(x, y|\forall i : (x, y_i, z_i) \in L')\}
\]

Clearly, \( L^k \) will still be in \( C \). So

\[
\exists y : (x, y) \in L \iff \Pr_z[\#y_1..y_k : (x, y, z) \in L' \text{ is even}] \geq 1 - (1 - 1/p(n))^k. \tag{4}
\]

Therefore,

\[
\exists y : (x, y) \in L \iff \Pr_z[\#y : (x, y, z) \in L' \text{ is even}] \geq 1/p(n) \tag{5}
\]

So \( L \in BP \cdot \oplus \cdot C \).

C is closed under complement, so we can easily make the same argument for \( \forall \cdot C \).

Theorem 3 (Toda’s Theorem 1) \( \forall k : \Sigma_k^P, \Pi_k^P \subseteq BP \cdot \oplus \cdot P \)

Proof We will prove by induction on \( k \), where the base case of \( k = 0 \) is trivial since \( \Sigma_0^P = \Pi_0^P = P \). We assume the induction hypothesis that \( \Sigma_{k-1}^P, \Pi_{k-1}^P \subseteq BP \cdot \oplus \cdot P \). We will now show the inductive case using a series of containments:

\[
\Sigma_k^P = \exists \cdot \Pi_{k-1}^P \subseteq \exists \cdot BP \cdot \oplus \cdot P \subseteq BP \cdot \oplus \cdot BP \cdot \oplus \cdot P \subseteq BP \cdot \oplus \cdot P
\]

The first containment, \( \Sigma_k^P = \exists \cdot \Pi_{k-1}^P \) is true by definition. The second containment is proven in Lemma 2. The last containment is true by the following:

\[
BP \cdot \oplus \cdot BP \cdot \oplus \cdot P \subseteq \oplus \cdot BP \cdot \oplus \cdot BP \cdot P \quad \text{(by Property 1.3)}
\]

\[
\subseteq \oplus \cdot BP \cdot P \quad \text{(by Property 1.1)}
\]

\[
\subseteq \oplus \cdot BP \cdot P \quad \text{(by Property 1.2)}
\]
5 Theorem 2: \( BP \cdot \oplus \cdot P \in P^\#P \)

Toda’s theorem states that we can reduce any language in \( BP \cdot \oplus \cdot P \) to a language of the form

\[
L_\# = \{ x \mid 0 \leq \#y \text{ s.t.} (x, y) \in L \leq a(mod b) \}
\]

for some language \( L \) and integers \( a \) and \( b \).

\( L_\# \in P^\#P \), because it’s just counting the satisfying \( y \) values modulo some constant. In fact, if we prove this reduction, we will show that any language in \( BP \cdot \oplus \cdot P \) can be decided with just one query to a \( \#P \) oracle.

Let \( L \) be some language in \( P \) and let \( L' \in BP \cdot \oplus \cdot P \) be defined as:

\[
x \in L' \Rightarrow Pr_x[\#z\{(x, y, z) \in L\} = 1(mod 2)] \geq \frac{2}{3}
\]

\[
x \notin L' \Rightarrow Pr_x[\#z\{(x, y, z) \in L\} = 0(mod 2)] \leq \frac{1}{3}
\]

We are using a weak version of \( BP \) because it is all we’ll need for this proof.

Let’s suppose that the number of \( y \) values that we’re enumerating over is \( \leq 2^k \).

Now, what if we were somehow able to “pump up” the modulus to be \( 2^k \)? In other words, what if we could convert \( L' \) into a language defined as:

\[
x \in L' \Rightarrow Pr_x[\#z\{(x, y, z) \in L\} = 1(mod 2^k)] \geq \frac{2}{3}
\]

\[
x \notin L' \Rightarrow Pr_x[\#z\{(x, y, z) \in L\} = 0(mod 2^k)] \leq \frac{1}{3}
\]

Then, we could remove the probability notation and write the definition as

\[
x \in L' \Rightarrow \#(y, z)\{(x, y, z) \in L\} \geq \frac{2}{3} \cdot 2^k(mod 2^k)
\]

\[
x \notin L' \Rightarrow \#(y, z)\{(x, y, z) \in L\} \leq \frac{1}{3} \cdot 2^k(mod 2^k)
\]

Now this can be calculated by a language of the form described in (6)!

So how are we going to boost this modulus up to \( 2^k \)?

Given that \( \#y : \{(x, y) \in L_1\} = N_1 \) and \( \#y : \{(x, y) \in L_2\} = N_2 \) for some \( x \), we can construct languages \( L_+ \) and \( L_- \):

\[
L_+ = \{(x, by)\{(b = 0 \& (x, y) \in L_1) \text{ or } (b = 1 \& (x, y) \in L_2)\}
\]

\[
\#y : \{(x, y) \in L_+\} = N_1 + N_2
\]

11-4
\[ L_x = \{ (x, (y_1, y_2)) | ((x, y_1) \in L_1) \text{ and } ((x, y_2) \in L_2) \} \]  
\[ \#y : \{(x, y) \in L_x\} = N_1 \times N_2 \]  

(15)

(16)

Combining these constructions, it is clear that if \( N(x) = \#y : \{(x, y) \in L\} \), then for any \( f \), we can construct a language \( L_f \) such that \( \#y : \{(x, y) \in L_f\} = f(N(x)). \) So, we just need a function \( f \) such that:

\[
\begin{align*}
   x = 0 \pmod{2^z} &\Rightarrow f(x) = 0 \pmod{2^2^z} \quad (17) \\
   x = 1 \pmod{2^z} &\Rightarrow f(x) = 1 \pmod{2^2^z} \quad (18)
\end{align*}
\]

With such an \( f \), we can iteratively apply \( f \) to \( x \) \( k \) times, which will pump \( (\mod 2) \) up to \( (\mod 2^k) \). Well, it turns out there’s no function with these properties. However, if we replace 1 with \(-1\), then there is such a function, namely \( f(x) = 4x^3 + 3x^4 \).

\[
\begin{align*}
   x = 0 \pmod{2^z} &\Rightarrow 4x^3 + 3x^4 = 0 \pmod{2^2^z} \quad (19) \\
   x = -1 \pmod{2^z} &\Rightarrow 4x^3 + 3x^4 = -1 \pmod{2^2^z} \quad (20)
\end{align*}
\]

So now we can construct a language from \( L’ \) that we can use in (6) with \( a = 2^{k-1} \) and \( b = 2^k \), thus solving \( L’ \) in \( P^{\#P} \).