CMPUT 675: Computational Complexity Theory **Winter 2019**

Lecture 11 (Feb 12th): **BPP** & **Interactive proofs**

Lecturer: Zachary Friggstad *Scribe: Gao Yue*

11.1 BPP

We have defined **BPP** using probabilistic TMs, but we can also define **BPP** using verifiers:

Definition 1 *An Alternative Definition of* **BPP** *:*
BPP *contains language L if there exists a polynomial time* **NP** *verifier M (time cost of M = p(n)) such that*

- $x \in L \rightarrow \underset{y \sim \{0,1\}^{p(|x|)}}{\mathbf{Pr}}[M(x,y) = \text{ACCEPT}] \geq \frac{2}{3}$
- $x \notin L \rightarrow \underset{y \sim \{0,1\}^{p(|x|)}}{\mathbf{Pr}}[M(x,y) = \text{ACCEPT}] \leq \frac{1}{3}$

11.1.1 BPP $\subseteq$ $\mathbf{P}_{/poly}$

Theorem 1 *(Adleman '78)* ***BPP*** $\subseteq$ $\boldsymbol{P}_{/poly}$

Proof. Suppose $L \in$ **BPP**, then by the error reduction procedure there exists a poly-time TM M and a polynomial p such that for $\forall x$:

$$\underset{r \sim \{0,1\}^{p(|x|)}}{\mathbf{Pr}}[\text{M(x,r) correctly determines if x is in L or not}] \geq 1 - 2^{-(|x|+1)}$$

For any fixed n, let's build a circuit C_n deciding all length-n inputs. For $x \in$ {0,1}^{n}, let S_x = {y: M(x,y) incorrectly determines if x is in L or not}.
Note :

$$\begin{aligned}
|\bigcup_{x \in \{0,1\}^n} S_x| &\leq \sum_x |S_x| && (11.1)\\
&\leq \sum_x \frac{2^{p(n)}}{2^{n+1}} && (11.2)\\
&= 2^n \cdot \frac{2^{p(n)}}{2^{n+1}} && (11.3)\\
&= 2^{p(n)-1} && (11.4)\\
&< |\{0,1\}^{p(n)}| && (11.5)
\end{aligned}$$

This implies that $\exists r' \in \{0,1\}^{p(n)}$ such that $\forall x \in \{0,1\}^n$, $M(x, r')$ correctly determines if x is in L or not. Since $M(\cdot, r')$ is a deterministic poly-time TM given such a string y that correctly determines all $x \in \{0,1\}^n$, then following the circuit construction procedure in **Theorem 1** of **Lec 8**, there exists a poly-size circuit C_n such that $C_n(x) = L(x)$ for every $x \in \{0,1\}^n$ and we can compute C_n in $poly(n)$ time. More explicitly, the language $L' = \{(x, r) : M(x, r) = \text{ACCEPT}\}$ is in **P** so it has a uniform family of polynomial-size circuits. Let $C'_{n+p(n)}$ be the family of circuits for inputs of size $n + p(n)$ ($|x| = n, |r| = p(n)$). Hard code the last $p(n)$ wires to r' to produce the circuit C_n with n inputs. Hence for any language L in **BPP**, L has polynomial size circuit family. ■

11.1.2 BPP is in PH

In this section, we'll describe a relation between **BPP** and the polynomial hierarchy.

Theorem 2 *(Sipser-Gacs '83)* $\mathbf{BPP} \subseteq \sum_2^p \cup \prod_2^p$

Proof. As $\sum_2^p$ is complement of $\prod_2^p$ and **BPP** is closed under complementation, it sufficies to show $\mathbf{BPP} \subseteq \sum_2^p$.

Let $L \in \mathbf{BPP}$, by error reduction (again) there is a poly-time TM M such that $\forall x \underset{r \sim \{0,1\}^{|p(x)|}}{\mathbf{Pr}}$ [M(x,r) correctly determines if x is in L or not] $\geq 1 - \frac{1}{2^{|x|}}$.

Fix n, for $x \in \{0,1\}^n$, let $A_x = \{y \in \{0,1\}^{p(|x|)}$: M(x,y) = ACCEPT$\}$. For a fixed x, if $x \in L$, $\underset{y \sim \{0,1\}^{|p(x)|}}{\mathbf{Pr}}[M(x, y) = \text{ACCEPT}] \geq 1 - \frac{1}{2^{|x|}}$.

- $x \in L$: In this case, $|A_x| \geq (1-\frac{1}{2^n}) \cdot 2^{p(n)}$
- $x \notin L$: In this case, $|A_x| \leq \frac{2^{p(n)}}{2^n}$

But we need a statement like this that has perfect completeness in the "yes" case to show L resides in the polynomial hierarchy. To do this, we consider some translations about a small set of vectors so translating each A_x about these vectors will cover the entire space of random strings in the yes case, but still not cover the entire space in the no case.

For $u, v \in \{0,1\}^k$, let u $\oplus$ v be the bitwise XOR of u and v. (i.e, 101 $\oplus$ 011 = 110). For $S \subseteq \{0,1\}^n$, $u \in \{0,1\}^k$, define $S + u = \{v \oplus u : v \in S\}$. Let $t = \lceil \frac{p(n)}{n} \rceil + 1$.

Claim 1 *For* $u_1, ...u_t \in \{0,1\}^{p(n)}$, $\cup_{i=1}^t (A_x + u_i) \neq \{0,1\}^{p(n)}$ *if* $x \notin L$.

Proof. If $x \notin L$,

$$\begin{aligned}
|\cup_{i=1}^t (A_x + u_i)| &\leq \sum_{i=1}^t |A_x + u_i| && (11.6)\\
&= \sum_{i=1}^t |A_x| && (11.7)\\
&= t|A_x| && (11.8)\\
&\leq t\frac{2^{p(n)}}{2^n} && (11.9)\\
&< 2^{p(n)} && (11.10)
\end{aligned}$$

The last bound holds for large enough n. Of course, for bounded n we can just solve the problem in constant time. This shows $\cup_{i=1}^t (A_x + u_i) \neq \{0,1\}^{p(n)}$. ■

Claim 2 *If $x \in L$, $\exists u_1, u_2, ..., u_t$ such that $\cup_{i=1}^{t}(A_x + u_i) = \{0,1\}^{p(n)}$.*

Proof. Sample each u_i randomly and independently from $\{0,1\}^{p(n)}$. For any $y \in \{0,1\}^{p(n)}$,

$$\begin{aligned} \mathbf{Pr}[y \notin \cup_{i=1}^{t}(A_x + u_i)] &= \prod_{i=1}^{t} \mathbf{Pr}[y \notin (A_x + u_i)] \qquad \text{By independence} && (11.11)\\ &\leq \prod_{i=1}^{t} \frac{1}{2^n} && (11.12)\\ &= \frac{1}{2^{nt}}. && (11.13) \end{aligned}$$

Where (11.12) is because $y \notin (A_x + u_i)$ if and only if $y \oplus u_i \notin A_x$, and $y \oplus u_i$ is uniform random variable in $\{0,1\}^{p(n)}$, so $y \oplus u_i$ is in A_x with probability $\geq 1 - 2^{-n}$ since $x \in L$.

Hence,

$$\begin{aligned} \mathbf{Pr}[\exists y \in \{0,1\}^{p(n)} \text{such that } y \notin \bigcup_i (A_x + u_i)] &\leq \sum_y \mathbf{Pr}[y \notin \bigcup_i (A_x + u_i)] \qquad \text{By union bound} && (11.14)\\ &\leq \sum_y \frac{1}{2^{nt}} && (11.15)\\ &= \frac{2^{p(n)}}{2^{nt}} && (11.16)\\ &\leq \frac{2^{p(n)}}{2^{n(\frac{p(n)}{n}+1)}} \qquad \text{Since t=}\lceil \frac{p(n)}{n} \rceil\text{+1} && (11.17)\\ &< 1. && (11.18) \end{aligned}$$

So $\exists u_1, u_2, ..., u_t$ such that $\cup_{i=1}^{t}(A_x + u_i) = \{0,1\}^{p(n)}$. ■

Together **Claims 1** and **2** show $x \in L$ if and only if:

$$\exists u_1, ..., u_t \in \{0,1\}^{p(n)} \forall y \in \{0,1\}^{p(n)} \cup_{i=1}^{t} M(x, y \oplus u_i) = \text{ACCEPT}.$$

Hence $L \in \sum_2^p$. ■

By now we have learned complexity classes including $\mathbf{P}, \mathbf{NP}, \mathbf{BPP}, \mathbf{ZPP}, \mathbf{RP}, \mathbf{coRP}$ and $\mathbf{P}_{/poly}$. In summary, the hierarchy of complexity classes possibly looks like Figure 11.1 where the picture considers all classes that are not known to be equal as distinct (that is, some might actually be equal).

11.2 Interactive Proofs

The mechanism of interactive proof system is like a multi-round interaction between the prover (P) and the verifier (V). In each round, the verifier asks a question according to all messages obtained so far and the prover respond to that question, in the last round the verifier decides whether to accept. The formal definitions are as follows :

Definition 2 (Interaction of Deterministic Functions) *Let f, g : $\{0,1\}^* \to \{0,1\}^*$ be functions, k be an integer ≥ 0. A k-round interaction of f and g on given input $x \in \{0,1\}^*$ is defined as :*

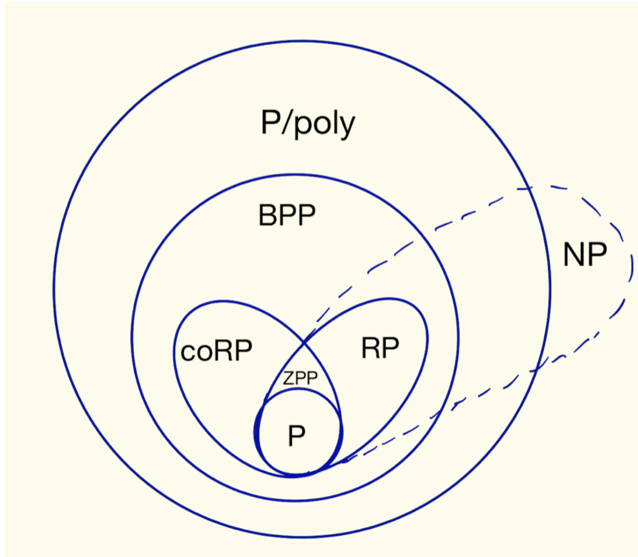

Figure 11.1: Relation between complexity classes

$$
\begin{gathered}
a_1 = f(x) \\
a_2 = g(x, a_1) \\
\dots \\
a_{2i+1} = f(x, a_1, \dots, a_{2i}) \\
a_{2i+2} = g(x, a_1, \dots, a_{2i+1}) \\
\dots \\
output = f(x, a_1, \dots, a_k)
\end{gathered}
$$

The output of the interaction is assumed to be 0 (REJECT), or 1(ACCEPT) in deterministic proof systems.

Definition 3 *Deterministic Proof Systems*
A language L has a k-round deterministic interactive proof if there exists a poly-time TM V that on input x, a_1, ..., a_i runs in time poly($|x|$) such that:

- $x \in L \rightarrow \exists P : \{0,1\}^* \rightarrow \{0,1\}^*$, *the output of V and P interacting on x in k rounds is 1(ACCEPT).*
- $x \notin L \rightarrow \forall P : \{0,1\}^* \rightarrow \{0,1\}^*$, *the output of V and P interacting on x in k rounds is 0(REJECT).*

Here, are treating the output of V as a string rather than just ACCEPT or REJECT. Also, k may be a function of $|x|$: the number of rounds of interaction may not necessarily be bounded by a constant.

This yields another complexity class!

Definition 4 ***dIP***

$\boldsymbol{dIP} = \bigcup_{c\geq 0}$ *(Languages with n^c-round deterministic interactive proof systems)*

The following theorem shows that a language can be determined by a poly-time verifier if and only if it has a poly-round deterministic interactive proof system.

Theorem 3 **dIP = NP**

Proof.

- **NP** $\subseteq$ **dIP** : Recall that definition of **NP** is all languages decided by a poly-time verifier, so any **NP** language L has a 1-round proof system as follows:

P : The prover just sends the certificate.
V : Verifies the certificate just like the NP verifier would.

- **dIP** $\subseteq$ **NP** : Let L be a language L that has a deterministic interactive proof system with a verifier V and a prover P. In input x, consider the interaction:

$$\boxed{\begin{array}{l} V(x) = a_1 \\ P(x, a_1) = a_2 \\ ... \\ V(x, a_1, ..., a_k) = 1 \end{array}}$$

  We build a PTV that can decide L according to this proof system. The certificate is just $(a_1, a_2, ..., a_k)$, satisfying V(x) = a_1, V(x,a_1, a_2) = a_3, ..., and V(x,$a_1, ..., a_k$) = 1. Define a poly-time verifier M that verifies V(x, $a_1, a_2, a_3, \ldots, a_i$) = a_{i+1} for all odd i. It is easy to see that if $x \in L$, there exists such certificate and the output of the verifier M is 1(ACCEPT); if $x \notin L$, there does not exist such certificate since the output of the proof system is always 0(REJECT). Thus, $L \in$ **NP**.

■

11.2.1 Randomized Interactive Proof for Graph Non-Isomorphism

In this section, all graphs have their nodes labelled 1 through $n = \#$ nodes.

Definition 5 (Isomorphic) *Two graphs G_1 and G_2 are isomorphic if there is a permutation π of the labels of the nodes of G_1 such that $\pi(G_1) = G_2$. If G_1 and G_2 are isomorphic, write $G_1 \simeq G_2$.*

Graph Isomorphism: determine if $G_1 \simeq G_2$. This is in **NP**: a certificate is simply the description of the permutation π.

Graph Non-Isomorphism is the opposite of **Graph Isomorphism**: it is the problem deciding whether two given graphs are not isomorphic. It is not known if the problem is in **NP**. Here is a randomized interactive proof for Graph Non-Isomorphism:

V : Pick $i \in \{1, 2\}$ and randomly permute the labels of G_i, send this graph H to the prover.
P : Sends $j \in \{1, 2\}$, with the idea that $G_j \simeq$ H if $G_i \not\simeq G_j$.
V : Accept if and only if $i = j$.

Note that if $G_1 \not\simeq G_2$, then there exists a prover such that $\mathbf{Pr}[V \text{ACCEPTS}] = 1$.
If $G_1 \simeq G_2$, the best any prover can do is to randomly guess, $\mathbf{Pr}[V samples(G_1, H)] = \mathbf{Pr}[V samples(G_2, H)]$ for all $H \simeq G_1$. This is because the distribution over random graphs H is identical whether V sampled $i = 1$ or $i = 2$ so P has no way of "guessing" the value of i given H. That is, suppose the prover P answers with

$P(H) \in \{1,2\}$. As $\mathbf{Pr}[\text{V samples } H|i=1] = \mathbf{Pr}[\text{V samples } H|i=2]$, we have $\mathbf{Pr}_{i,H}[P(H)=i] = 1/2$. That is, if $G_1 \simeq G_2$ then for every prover, $\mathbf{Pr}[V accepts] = 1/2$. This can be reduced to 2^{-k} by repeating the protocol k times and having the verifier accept if and only if all answers from the prover were correct.

So we do not know whether **Graph Non-Isomorphism** is in **dIP**, but later we'll introduce another complexity class related to randomized interactive proofs which **Graph Non-Isomorphism** belongs to.

11.2.2 The class IP

We have shown that deterministic proof system does not change the class of language we can determine (**dIP** = **NP**), and there exists some problem (e.g. **Graph Non-isomorphism**) with an interactive proof that does not necessarily lie in **dIP**. So in order to realize the full potential of interaction, we try to replace the *deterministic* verifier by *probabilistic* verifier (a P.T.M.) that may generate its queries and its final choice using random bits that are not revealed to the prover.

Definition 6 ***IP***
*For $k \geq 0$, a language L is in **IP**[k] if there is a k-round interactive proof where the verifier is a P.T.M. such that:*

- *(Completeness)* $x \in L \rightarrow \exists P : \mathbf{Pr}[\text{The k-round verification outputs ACCEPT}] \geq \frac{2}{3}$
- *(Soundness)* $x \notin L \rightarrow \forall P : \mathbf{Pr}[\text{The k-round verification outputs ACCEPT}] \leq \frac{1}{3}$

$\mathbf{IP} = \bigcup_{c \geq 0} \mathbf{IP}[n^c]$

Remark : The following observations on the class **IP** are left as exercise :

- What if the prover can be randomized? Does it change the class **IP**?
  Allowing the prover to be randomized does not change the class **IP**. The reason is that for any language L, if a randomized prover P results in making verifier V accept with some probability, then in each step the prover could instead choose its answer deterministically to maximize the resulting probability of having the verifier accept (as the best deterministic answer is at least as good as the "average" answer if they are chosen randomly).
- Does the prover need arbitrary power?
  No. Even if the prover is restricted to computing answers in polynomial space, **IP** does not change. Given any verifier V, trying all answers, we can compute the optimum prover (which, given x, maximizes the verifier's acceptance probability) using poly($|x|$) space and $2^{poly(|x|)}$ time. Hence **IP** $\subseteq$ **PSPACE**. The assignment asks you to carefully describe this argument.
- The probabilities of correctly classifying an input can be made arbitrarily close to 1 by using the same error reduction procedure for **BPP** in **Lec10**:
  To replace $\frac{2}{3}$ by 1-exp(-m), sequentially repeat the protocol m times and take the majority answer.

References

AB09 S.Arora and B.Barak, Computational Complexity: A Modern Approach, *Cambridge University Press,New York, NY, USA*, 2009, pp. 126–151.