

# Advanced Proof Systems - Course Material

Yosef Israel Goren

November 23, 2022

## Lecture 1

### Recap

#### **$P$ - Polynomial (Class)**

$L \in \{0, 1\}^*$  is also in  $P$  if there exists an efficient algorithm which decides it.

#### **$NP$ - Nondeterministic Polynomial**

$L \in \{0, 1\}^*$  is also in  $NP$  if there exists an efficient algorithm  $V$  and a polynomial  $p$  which follow:

1. Completeness:  $\forall x \in L, \exists y : V(x, y) = 1 \wedge |y| < p(|x|)$
2. Soundness:  $\forall x \notin L \forall y : V(x, y) \neq 1 \vee |y| \geq p(|x|)$

#### **$PPT$ - Probabalistic Polynomial Time**

This is a class of algorithms which must run in time polynomial to the size of their input, but also - must be capable of randomization, or 'flipping coins'.

#### **$IP$ - Interactive Proof**

A key difference between an Interactive Proof and a proof for an NP proof is that the latter necessarily requires the prover to provide the verifier with something he can use to prove the truth of the claim to others.

We denote  $(P, V)(x)$  to be the output of  $V$  (verifier) after the interaction between  $P$  and  $V$  on the input  $x$ . Both  $P$  and  $V$  can be thought of as  $PPT$  algorithms or programs which are capable of communicating with one another.

These interactions are often described with an interaction diagram:

- $P$  sends to  $V$  something

- $V$  sends to  $P$  something else
- ...
- $V$  accepts iff ...

Formal Definition: We say that  $L \in IP$  if there exists a polynomial algorithm  $V$ , an unbounded algorithm  $P$  and some constant  $c \in (0.5, 1]$  s.t.

1. Completeness:  $\forall x \in L, Pr[(P, V)(x) = 1] > c$
2. Soundness: if  $x \notin L, \forall P^* \in \mathbb{M}, Pr[(P^*, V)(x) = 1] < 1 - c$

Note:  $\mathbb{M}$  denotes the set of turing machines.

## Equivalence of $IP$ separation constants

### Iterative Runs

Given an  $IP$  protocol  $(P, V)$ , let  $(P^k, V^k)$  be the protocol obtained by running  $(P, V)$   $k$  times sequentially.  $V^k$  accepts iff in all iterations  $V$  accepted.

### Lemma

if  $(P, V)$  is  $IP$  with perfect completeness then for every polynomial  $k$ ,  $(P^k, V^k)$  is  $IP$  with perfect completeness and soundness error  $2^{-\Omega(k)}$

Proof:

1.  $V^k$  is efficient (composition of polynomials).
2. Perfect Completeness - due to Perfect Completeness of the original protocol - each iteration is guaranteed to succeed thus the protocol always does.
3. Soundness: Let  $x \in L, P^*$ . We will show that  $Pr[(P^*, V^k)(x) = 1] \leq 2^{-k}$ . Denote by  $E_i$  the event that  $V^k$  accepts in the  $i$ 'th iteration. Thus:

$$Pr[E_1 \wedge E_2 \wedge \dots E_k] = \prod_{i=1}^k Pr[E_i | E_1 \wedge E_2 \wedge \dots E_{i-1}]$$

Claim:  $Pr[E_i | E_1 \wedge E_2 \wedge \dots E_{i-1}] \leq 0.5$ .

Proof: Assume toward a contradiction that  $Pr[E_i | E_1 \wedge E_2 \wedge \dots E_{i-1}] > 0.5$

We design a prover  $P^{**}$  that convinces  $V$  with up to  $> 0.5$ :

$P^{**}$  emulates  $(P^*, V)$  for iterations  $1 \dots i - 1$  until the event  $E_1 \wedge E_2 \wedge \dots E_{i-1}$  happens and then runs  $(P^*, V)$  as the  $i$ 'th iteration. Since the run of  $(P^*, V)$  for the  $i$ 'th iteration only happens under the condition  $Pr[E_i | E_1 \wedge E_2 \wedge \dots E_{i-1}]$  - the probability for  $(P^*, V)$  to happen on the  $i$ 'th iteration is exactly  $Pr[E_i | E_1 \wedge E_2 \wedge \dots E_{i-1}] > 0.5$  but this is also the

probability for  $Pr[(P^{**}, V)(x) = 1]$ . Contradiction.

Thus:

$$Pr[E_1 \wedge E_2 \wedge \dots E_k] = \prod_{i=1}^k Pr[E_i | E_1 \wedge E_2 \wedge \dots E_k] \leq \prod_{i=1}^k 0.5 = 2^{-k}$$

## Graph Isomorphism & $IP$ Example

### Graph Isomorphism - Definition

The graphs  $G_1 = (V, E_1), G_2 = (V, E_2)$  are isomorphic or  $(G_1, G_2) \in GI$  if  $\exists \pi : V \rightarrow V$  s.t.  $(u, v) \in E_1 \iff (\pi(u), \pi(v)) \in E_2$ .

More simply - two graphs are isomorphic if they are identical up to a renaming of their vertices.

$GNI$  is the set of pairs of graphs which are isomorphic.

- Claim (no proof):  $NP \subseteq IP$ .
- Claim (proof sketch):  $GI \in IP$ .  
By finding the permutation  $\pi$  it is easy to check the  $GI$  condition over a given  $(G_1, G_2)$  thus we have an  $NP$  relation, meaning  $GI \in NP$ .
- Claim (no proof):  $GNI \in IP$ .

## Tutorial 1

### $IP$ and $NP$

Claim:  $NP \subseteq IP$ .

Proof: let  $L \in NP$ . There exists some  $NP$  relation  $R$  for  $L$ , with an efficient algorithm  $M_R$  which decides it.

Now define an  $IP$  protocol:

- Both  $P$  and  $V$  get  $x$ .
- If  $x \in L$   $P$  find  $y$  s.t.  $(x, y) \in R$ , and send it to  $V$ . Otherwise send  $\epsilon$ .
- $V$  checks if  $(x, y) \in R$  by running  $M_R$  (known to be efficient) and accepts iff  $M(x, y)$  accepts.

1. Completeness: If  $x \in L$ , such  $y$  must exist ( $NP$  definition) thus  $P$  will find it, and  $V$  will have  $(x, y) \in R$  so  $M_R$  and  $V$  will accept.
2. Soundness: If  $x \notin L$ , there is no  $y$  which such that  $(x, y) \in R$  so no matter what any  $P^*$  sends -  $M_R(x, y)$  rejects and so  $V$  rejects too.

## Similar Proof Systems

### Arthur-Merlin - $AM$

- Both parties get some input  $x$ .
- Arthur sends Merlin some randomized  $\alpha$ .
- Merlin sends back some  $\beta$ .
- Arthur accepts according to some  $PPT$  algorithm which is a function of  $x, \alpha$  and  $\beta$  (usually denoted  $A(x, \alpha, \beta)$ ).

### Merlin-Arthur - $MA$

- Both parties get input  $x$ .
- Merlin sends  $\beta$  to Arthur.
- Arthur generates some random value  $\alpha$ .
- Arthur accepts according to some  $PPT$  algorithm which is a function of  $x, \alpha, \beta$ .

Theorem:  $MA \subseteq AM$ .

Proof: Let  $L \in MA$ . WLOG (without loss of generality),  $L$  has a  $MA$  protocol with perfect completeness (we will come back to this assumption later in the course).

Denote by  $p(n)$  the length  $\beta$  ( $|\beta| \leq p(n)$ ).

Using repetition we can get to any protocol with perfect completeness and a soundness error of  $2^{-p(n)-1}$  (as seen in lecture).

Sketching the repeated protocol would look like:

- $M'$  sends  $\beta$  to  $A'$
- $A'$  sends back a list of  $\alpha$  values (as many as there are repetitions).
- $A'$  decides whether to accept.

This is because there is no reason for the prover's proof ( $\beta$ ) to change due to different sampling of  $\alpha$ , so it is always the same and can be sent once. So the length of Merlin's message does not change in the repeating protocol.

Now consider the same  $M', A'$  protocol but where  $A'$  sends the aggregated  $\alpha$  before  $M'$  sends  $\beta$ .

Claim: This new protocol is  $AM$ . Proof:

1. Completeness:  $\forall x \in L, M'$  sends the same  $\beta$  without looking at  $\alpha$ :

$$Pr[(M', A)(x) = 1] = Pr[A(x, \alpha, \beta) = 1] = 1$$

2. Soundness: Let  $x \notin L$ , fix  $M^*$ . Consider:

$$\begin{aligned} Pr[(M^*, A')(x) = 1] &= Pr[\exists \beta \in \{0, 1\}^p : A'(x, \alpha, \beta) = 1] \\ &= \cup_{\beta \in \{0, 1\}^p} Pr[A'(x, \alpha, \beta) = 1] \leq_{UB} 2^p 2^{-(p+1)} = \frac{1}{2} \end{aligned}$$

Note:  $UB$  denotes Union Bound.

## Lecture 2

### Recap

#### Remarks on $GI$

- For any graph  $G$ , the set of all  $(G, G')$ , where  $G'$  is isomorphic to  $G$  is an equivalence relation.
- For any graph  $G$ :  $(\pi(G))_{\pi \leftarrow \mathbb{U}} \approx \mathbb{I} \sim \times (\mathbb{G})$  - meaning uniformly sampling a permutation (over the set of nodes) of  $G$  and applying it to  $G$  will yield a uniformly sampled graph from the same equivalence partition.

#### $GNI \subseteq IP$

To prove this we will present an interactive protocol that runs on input  $G_0, G_1$ :

- $V$  samples  $b \leftarrow \{0, 1\}, \pi \leftarrow \{\text{permutations}\}$ .
- $V$  sends  $G = \pi(G_b)$ .
- $P$  checks: if  $G \approx G_0, b' = 0$  else  $b' = 1$ .
- $P$  sends  $b'$  to  $V$ .
- $V$  accepts iff  $b = b'$ .

Now we prove that this interactive protocol is indeed an interactive proof for  $GNI$ :

1. Soundness: Let  $G_0 \approx G_1$  and let  $P^*$  be a (possibly) cheating prover.  
 If  $b = 0$  then  $G$  is uniformly distributed over all graphs isomorphic to  $G_0$ .  
 If  $b = 1$  then  $G$  is uniformly distributed over all graphs isomorphic to  $G_1$ ,  
 but due to transitivity, this is the same as the prior case, meaning in any  
 case,  $G$  is distributed uniformly over the equivalence to  $G_0$ .  
 Since this is the case, no information has been passed from  $V$  to  $P^*$  by  
 sending  $G$  ( $P^*$  could have just sampled a graph from  $Iso(G_0)$  itself!),  $P^*$   
 could not do more than guess  $b$ , and it's probability for success is no more  
 than  $\frac{1}{2}$ .
2. Completeness: If the graphs are not isomorphic,  $G \approx G_0$  iff  $b = 0$ , which  
 will mean  $b = b'$  is always the case and  $V$  always accepts.
3. Runtime:  $V$  does nothing of high complexity,  $P$  is unbounded.

## Equivalence between $IP$ and $AM[Poly]$

### Lemma: Set Lower Bound Protocol

#### Theorem

- $IP[k]$ :  $k$ -round interactive proof.
- $AM[k]$ :  $k$ -round public coin interactive proof.

Claim (no proof):  $AM[k] \subseteq IP[k]$ .

Theorem:  $IP[k] \subseteq AM[k + 2]$ .

Corollary:  $IP = AM[Poly]$ .

Theorem Proof:

Let  $L$  be a language with an  $IP$  which satisfies the following assumptions:

1.  $k = 2$ .
2. Perfect Completeness.
3. There is a set of  $N$  of possible verifier messages which is constant for any  
 specific protocol input  $x$ .
4. The message  $m$  sent by  $V$  is uniformly distributed over the  $N$  possible  
 messages that could be sent.
5. The soundness error is  $\frac{1}{100}$ .

We want to show that  $L$  is in  $AM[k + 2]$ .

First, note how our  $IP$  can be described as follows when running on input  $x$ :

- $V$  samples some  $r \leftarrow \{0, 1\}$ .
- $V$  calculates some  $m = V_1(x, r)$  for some efficient function  $V_1$ .

- $V$  sends  $m$  to  $P$ .
- $P$  sends back some  $\pi$ .
- $V$  accepts iff  $V_2(x, \pi, r) = 1$  for some efficient function  $V_2$ .

Define:

$$R_m = \{r' \mid m = V_1(x, r')\}, S_m = \{r' \in R_m \mid V_2(x, \pi, r') = 1\}$$

Now consider an alternative protocol, which is the same as the one above, only  $V$  also samples  $r' \leftarrow R_m$  and accepts iff  $V_2(x, \pi, r')$ .

Note how  $|R_m| = \frac{2^l}{N}$ , and if  $x \in L$  then  $S_m = R_m$ . Claim (missing proof): if  $x \in L$  then  $\mathbb{E}(|S_m|) = \frac{2^l}{N}$  and otherwise  $\mathbb{E}(|S_m|) \leq \frac{1}{100} \frac{2^l}{N}$ .

## Pairwise Independent Hash Functions

Definition: A set of hash functions  $\{h : X \leftarrow Y\}$  is pairwise independent if for all  $x_1, x_2 \in X$  where  $x_1 \neq x_2$  and  $y_1, y_2 \in Y$ , they satisfy:

$$\Pr_{h \leftarrow H}[h(x_1) = y_1 \wedge h(x_2) = y_2] = \frac{1}{|Y|^2}$$

Construction:

Let  $\mathbb{F}$  be a finite field. Given  $a, b \in \mathbb{F} : h_{a,b}(x) = ax + b$ .

Let  $H = \{h_{a,b} \mid a, b \in \mathbb{F}\}$ .

Construction proof:

Let  $x_1, x_2, y_1, y_2$  as described in definition.

$$\begin{aligned} \Pr_{a,b}[ax_1 + b = y_1 \wedge ax_2 + b = y_2] &= \Pr_{a,b}\left[\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\right] \\ &= \Pr_{a,b}\left[\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \end{pmatrix}^{-1} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\right] = \frac{1}{|\mathbb{F}|^2} \end{aligned}$$

For the last transition: Note how the right value of the equality (in the probability) is just a constant value, so we just need to randomly sample  $a, b$  to be some specific values within a finite field.

## Tutorial 2

### Perfect Completeness?

We have seen in the lecture how an  $IP$  can be reduced to an  $AM$  proof, or in other words; a public coin  $IP$ . Here, we would like to show how any  $AM$  can be reduced to an  $AM$  with perfect completeness (which is also an  $IP$  with perfect

completeness).

This means that  $\forall L \in IP$ , it must have a public coin, perfectly complete interactive proof.

To construct the reduction, we start with the following  $z$ -round public coin protocol ( $AM[z]$ ), which runs on input  $X$ :

- $A$  samples  $\alpha \leftarrow \{0, 1\}^{r_c}$ , and sends it to  $M$ .
- $M$  calculates  $\beta = M(X, \alpha)$  and sends it to  $m$ .
- $A$  accepts iff  $A(X, \alpha, \beta) = 1$ .

Where we assume completeness error  $\epsilon > 0$ :

$$\begin{aligned} \forall x \in L : Pr[(M, A)(X) = 1] &\geq 1 - \epsilon \\ \Rightarrow \forall x \in L : Pr[\exists \beta : A(X, \alpha, \beta) = 1] &\geq 1 - \epsilon \end{aligned}$$

Now, we want to use it to construct an equivalent with perfect completeness; for that end, consider the alternative protocol:

- $M'$  samples  $s_1, s_2, \dots, s_k \leftarrow \{0, 1\}^{r_c}$ . s.t. (\*)

$$\forall \alpha \{0, 1\}^{r_c}, \exists i \in [k] : s_i \oplus \alpha \notin REJ$$

.

- $M'$  sends  $s_1, s_2, \dots, s_k$  to  $A'$ .
- $A'$  samples  $\alpha \leftarrow \{0, 1\}^{r_c}$  and sends it to  $M'$ .
- $M'$  calculates  $\forall i : \beta_i = M(X, s_i \oplus \alpha)$  and sends it.
- $A'$  accepts iff  $(\exists i : A(X, s_i \oplus \alpha, \beta_i)) = 1$

Lemma 1: if  $x \in L$  then pre-processing succeeds.

We denote  $\bar{s} = (s_1, s_2, \dots, s_k)$ . We say that  $\bar{s}$  is 'good' if it satisfies (\*).

To prove the lemma, we can show that:

$$\exists \bar{s} : \bar{s} \text{ is good}$$

We will show this by first showing:

$$Pr[\bar{s} \text{ is good}] > 0$$

To start off:

$$\begin{aligned} Pr_{\bar{s}}[\bar{s} \text{ is not good}] &= Pr_{\bar{s}}[\exists \alpha, \forall i : s_i \oplus \alpha \in REJ] \\ &= Pr_{\bar{s}}\left[\bigcup_{\alpha} (\forall i : s_i \oplus \alpha \in REJ)\right] \leq 2^{r_c} \cdot \epsilon^k \end{aligned}$$



From here we can see that for  $k$  of at-least  $rc$ , the probability is less then 1, meaning that the complementary probability is more than 0, meaning :

$$\exists \bar{s} : \bar{s} \text{ is good}$$

Completeness is trivial given Lemma 1.  
Soundness of the protocol can be found in notes of Tutorial 2 in the website.

## Lecture 3

### Public Coin Protocol For Set Size

#### Lemma (Set LB)

Exists a protocol  $(P, V)$  such that: Given membership access to  $s \subseteq U, t \subseteq \mathbb{N}$ :

1. Completeness:

$$|s| \geq t \Rightarrow \Pr[V \text{ accepts}] \geq \frac{2}{3}$$

2. Soundness:

$$\forall P^*, |s| \leq \frac{t}{100} \Rightarrow \Pr[V \text{ accepts}] \leq \frac{1}{3}$$

#### The Protocol

1.  $V$  samples  $h \leftarrow H$  where  $H = \{h : U \rightarrow [t]\}$
2.  $P$  finds  $x \in S$  s.t.  $h(x) = 17$  and sends it to  $V$  (if none exist,  $P$  fails).
3. accepts iff  $x \in S$  and  $h(x) = 17$

#### Proof

For  $x \in U, h \in H$ , denote  $E_x$  if  $h(x) = 17$ .

Soundness:  $|S| \leq \frac{t}{100}$ .

$$\Pr_{h \leftarrow H}[\exists x \in S, h(x) = 17] = \Pr_{h \leftarrow H}[\bigcup_{x \in S} E_x] \leq \sum_{x \in S} \Pr[E_x] = \frac{|S|}{t} \leq \frac{1}{100}$$

Completeness:  $|S| = t$ .

$$\begin{aligned} \Pr_{h \leftarrow H}[\exists x \in S, h(x) = 17] &= \Pr_{h \leftarrow H}[\bigcup_{x \in S} E_x] \geq \sum_{x \in S} \Pr[E_x] - \Pr_{x, x' \in S, x < x'}[E_x \cap E_{x'}] \\ &= \frac{|S|}{t} - \binom{|S|}{2} \frac{1}{t^2} \geq \frac{|s|}{t} - \frac{|s|^2}{2t^2} = \frac{1}{2} \end{aligned}$$

## Zero Knowledge Proofs (ZKP)

### ZKP protocol for $GI$

Given input  $G_0, G_1$ :

- $P$  finds a permutation  $\psi \in S_n$  s.t.  $G_0 = \psi(G_1)$ .
- $P$  uniformly samples a random permutation  $\pi \leftarrow S_n$ .
- $P$  sends  $G = \pi(G_0)$  to  $V$ .
- $V$  samples  $b \leftarrow \{0, 1\}$  and sends it to  $P$ .
- $P$  defines  $\sigma = \pi$  if  $b = 0$  else  $\sigma = \pi \circ \psi$  and sends it to  $V$ .
- $V$  accepts iff  $\sigma(G_b) = G$ .

### HV – ZKP definition

An interactive proof  $(P, V)$  is a honest-verifier zero knowledge proof of  $L \subseteq \{0, 1\}^*$  if there exists a *PPT* algorithm  $S$  called 'the simulator' for which:  $\forall x \in L$  : The following distributions are 'similar':

- $View^{x,P}(x) := (\text{the input } x, \text{ randomized coins } r, \text{ communications transcript})$
- $S(x)$  (the output of the simulator)

Here 'similar' can mean two different things:

- If it means identical, meaning these two distributions are the same distribution, we call the protocol a **perfect ZKP**.
- If it means that the distributions are computationally indistinguishable, we call the protocol a **statistical ZKP**, the definition for 'computationally indistinguishable' is in the next tutorial.

The idea here is that since the verifier  $V$  could have seen everything which is seen during the interactive protocol  $(P, V)$  by running the simulator  $S$ , the verifier did not actually learn anything from it.

The reason this is 'honest-verifier' is because this definition does not catch the case where the verifier 'cheats' and does not run the protocol  $V$ , and might be able to extract information  $P$  by doing it.

### ZKP definition

An interactive proof  $(P, V)$  is a zero knowledge proof for  $L$  if:

$$\forall x, V^*, \exists S : View^{P, V^*}(x) \approx S(x)$$

Like to 'similar' from the honest verifier definition, ' $\approx$ ' can mean either 'identical' or 'computational indistinguishable'; with the former being a perfect ZKP

and the latter being a statistical *ZKP*. We sometimes denote computational indistinguishability by  $X \approx_C Y$  and perfect indistinguishability with  $X \approx_P Y$ .

In words, the difference from the honest-verifier zero ZKP is that now the simulator should be able to simulate anything that ANY verifier protocol manages to extract, meaning that regardless of the verifier protocol - no information is extracted.

### ***HZ – ZKP proof for GI protocol above***

$$View := (G_0, G_1, G, b, \sigma), G = \sigma(G_b)$$

Define the following protocol (simulator); on input  $G_0, G_1$  do:

- Sample  $\sigma \leftarrow S_n$
- Sample  $b \leftarrow \{0, 1\}$
- Define  $G = \sigma(G_b)$
- return  $(G_0, G_1, G, b, \sigma)$

Now consider each cell in the output tuple of the two distributions (*View* and  $S(G_0, G_1)$ ),  $G_0, G_1$  are constant and always the same.  $b, \sigma$  are sampled uniformly from their ranges in both cases. So far there is no correlation between the cells.  $G$  is defined exactly given the rest of the variables meaning that since the rest of the variables are the same for both distributions  $G$  is also the same. Furthermore, since the correlations were identical so far, the new correlations are too.

These proofs often (such as in this case) follow the pattern of first showing that given a subset of the cells of the distribution tuples are the same due to being uniformly sampled respectively ( $(G_0, G_1, b, \sigma)$  in this case), then showing how the rest of the cells ( $G$  in this case) are a deterministic function of the other cells, and so the addition of these new cells does not impair with the distributions being the same.

Formally, if we have distributions  $D_1, D_2$  and a (deterministic) function  $f$ :

$$D_1 \approx D_2 \Rightarrow (D_1, f_{d \leftarrow D_1}(d)) \approx (D_2, f_{d \leftarrow D_2}(d))$$

### ***NP $\subseteq$ ZKP (proof concept)***

In the lecture, proof by picture is shown for  $3COL \in NPC$  has a zero knowledge proof. The main course of the proof consists of the prover committing to a specific permutation of the colors on a graph (without showing the verifier what these are), and allowing the verifier to choose a specific edge and see that indeed the coloring on both sides of that edge differ. For each iteration a cheating prover has some chance of failing due to the verifier having a chance to guess an edge where the two sides have the same color. Furthermore, the protocol is

zero knowledge since the actual colors seen in each iteration are just two random colors.

## Tutorial 3

### Computational & Statistical Zero Knowledge

#### Distinguisher and Advantage - Definition

A distinguisher  $D$  is a probabilistic polynomial time algorithm; it receives an input  $w$  and tries to decide if  $w \in X$  or  $w \in Y$ .

The advantage of  $D$  over  $X, Y$  is defined as:

$$adv_D(X, Y) := |Pr_{w \leftarrow X}[D(w) = 1] - Pr_{w \leftarrow Y}[D(w) = 1]|$$

#### Negligible Functions

A negligible function is a function  $\epsilon : \mathbb{N} \rightarrow \mathbb{R}$  s.t.

$$\forall c \in \mathbb{N}, \epsilon(n) = o\left(\frac{1}{n^c}\right)$$

This is equivalent to saying that:

$$\forall \text{polynomial } p(n), \epsilon(n) \leq \frac{1}{p(n)}$$

#### Computational Indistinguishability

Let  $X, Y$  be two ensembles of distributions, meaning that each of them consists of a series of distributions:

$$X = \{X_1, X_2, \dots\}, Y = \{Y_1, Y_2, \dots\}$$

We say that  $X$  is computationally indistinguishable from  $Y$  if for every distinguisher  $D$  there exists a negligible function  $\epsilon$  such that:

$$\forall n \in \mathbb{N}, adv_D(X_n, Y_n) \leq \epsilon(n)$$

#### ZKP for 3COL (more formally)

Given an input graph  $G = ([n], E)$ :

- $P$  finds a 3-coloring  $\phi$  of  $G$ .
- $P$  samples a permutation  $\xi$  over  $[3]$  (the colors).

- $\forall v \in [n]$ , 'put  $\xi(v)$  in a box'  $\beta_v$  and send it to  $V$ .
- $V$  samples  $(u, v) = e \leftarrow E$  and sends it to  $P$ .
- $P$  sends the keys for  $\beta_v$  and  $\beta_u$  to  $V$ .
- $V$  accepts iff colors 'inside'  $\beta_v, \beta_u$  are different.

To formalize the idea of these 'boxes' we define the notion of a commitment scheme:

A **commitment scheme** is a pair of PPT algorithms:  $commit, check$  with the following syntax:

- $commit(b; r) \rightarrow c$
- $check(c, b, r) \rightarrow \{0, 1\}$

And which satisfy the following conditions:

1. **Computationally Hiding:**  $commit(0) \approx_C commit(1)$ .
2. **Perfectly Binding:** There is no  $n_0, r_0, r_1$  and  $C^*$  s.t.

$$check(C^*, 0, r_0) = check(C^*, 1, r_1) = 1 \quad (1 \text{ means accept})$$

## Lecture 4

$$coNP \subseteq IP$$

### Arithmetization

We interested in a reduction from a  $coNP$  problem to an arithmetic problem.

#### The Sumcheck Problem:

Parameters: a finite field  $\mathbb{F}$ , and  $n, d \in \mathbb{N}$ .

Input:  $Q : \mathbb{F}^n \leftarrow \mathbb{F}$ ,  $\alpha \in \mathbb{F}$ . Problem: does  $\sum_{x \in \{0,1\}^n} Q(x) = \alpha$  ?

The reduction We will be reducing from  $coNP$  to the sumcheck problem by reducing  $3 - CNF$  to it (since  $3 - CNF$  is  $coNP$ -complete).

Let  $\phi \in 3 - CNF$  with  $n$  variables and  $m$  clauses.

We will start the construction by translating each building block of  $3 - CNF$  formulas and expressing it in terms of polynomials:

1.  $\phi(x_1, x_2, \dots, x_n) = x_1 \longrightarrow p(x_1, x_2, \dots, x_n) = x_1$
2.  $x_1 \wedge x_2 \longrightarrow x_1 \cdot x_2$
3.  $x_1 \wedge \neg x_3 \longrightarrow x_1 \cdot (1 - x_3)$

$$4. (x_1) \vee (x_2) = \neg((\neg x_1) \wedge (\neg x_2)) \longrightarrow 1 - (1 - x_1) \cdot (1 - x_2)$$

Lemma: for every 3-CNF formula  $\phi$  on  $m$  clauses and a finite field  $\mathbb{F}$ , there exists a polynomial  $p : \mathbb{F}^n \rightarrow \mathbb{F}$  with degree  $O(m)$  s.t.  $\phi = p$ .  
Furthermore, given  $\phi$  and  $z \in \mathbb{F}^n$ ,  $p(z)$  can be evaluated in  $\text{poly}(n, m, \log(|\mathbb{F}|))$  time.

Given the lemma, we can easily solve 3-CNF, by constructing the appropriate polynomial  $p$ , and evaluating the sum of the polynomial: and checking if the sum is 0.

### IP for the sumcheck problem

We will show that the sumcheck problem is in IP by the following protocol on the inputs  $Q : \mathbb{F}^n \rightarrow \mathbb{F}$ ,  $\alpha \in \mathbb{F}$ :

- $P$  defines  $Q_1 : \mathbb{F} \rightarrow \mathbb{F}$  as

$$Q_1(\lambda) = \sum_{x_2, x_3, \dots, x_n \in \{0,1\}} Q(\lambda, x_2, x_3, \dots, x_n)$$

And sends it as  $\hat{Q}_0$ .

- $V$  rejects if  $\hat{Q}_0(0) + \hat{Q}_0(1) \neq \alpha$ .
- for  $i \in [n]$ :
  - $V$  samples  $r_i \leftarrow \mathbb{F}$  and sends it.
  - $P$  defines:

$$Q_i(\lambda) = \sum_{x_{i+2}, x_{i+3}, \dots, x_n \in \{0,1\}} (r_1, \dots, r_i, \lambda, x_{i+1}, \dots, x_n)$$

And sends it as  $\hat{Q}_i$ .

- $V$  rejects if  $\hat{Q}_i(0) + Q(1) \neq \hat{Q}_{i-1}(r_i)$

- $V$  accepts.

Completeness: Here we know that:  $\sum_x Q(x) = \alpha$ . So we can write:

$$\begin{aligned} \hat{Q}_1(0) + \hat{Q}_1(1) &= \hat{Q}_1(0) + \hat{Q}_1(1) = \sum_{x_2, x_3, \dots, x_n} Q(0, x_2, x_3, \dots, x_n) + \sum_{x_2, x_3, \dots, x_n} Q(1, x_2, x_3, \dots, x_n) \\ &= \sum_x Q(x) = \alpha \end{aligned}$$

Thus first check passes. Additionally:

$$\hat{Q}_1(r_1) + \hat{Q}_1(r_1) = \sum_{x_2, x_3, \dots, x_n} Q(r_1, x_2, x_3, \dots, x_n)$$

By induction aspect w.p. 1.

Soundness: Assume  $\sum_x \neq \alpha$  and let  $P^*$  be a cheating prover.  
WLOG  $P^*$  always sends  $\hat{Q}_1 \neq Q_1$ .  
Observe that  $Q_1$  and  $\hat{Q}_1$  are polynomials of degree  $d$  so with all but a probability of  $\frac{d}{|\mathbb{F}|}$  - we have  $Q_1(r_1) \neq \hat{Q}_1(r_1)$ .  
Assume  $Q_1(r_1) \neq \hat{Q}_1(r_1)$ , thus:

$$\hat{Q}_1(r_1) \neq \sum_{x_2, x_3, \dots, x_n} Q(r_1, x_2, x_3, \dots, x_n)$$

$$\begin{aligned} Pr[V \text{ accepts}] &\leq Pr[Q_1(r_1) = \hat{Q}_1(r_1)] + Pr[V_{n-1} \text{ accepts} \mid Q_1(r_1) = \hat{Q}_1(r_1)] \\ &\leq \frac{d}{|\mathbb{F}|} + \frac{d \cdot (n-1)}{|\mathbb{F}|} = \frac{d \cdot n}{|\mathbb{F}|} \end{aligned}$$

## Tutorial 4

## Lecture 5

### Reminder

Lemma (sumcheck):

There exists an IP between  $P$  and  $V$  where  $P$  gets as input a polynomial  $Q : \mathbb{F}^n \rightarrow \mathbb{F}$ , of individual degree  $d$  and a value  $\alpha \in \mathbb{F}$  and  $V$  gets 'oracle access' to  $Q$  and  $\alpha$  explicitly.

- Completeness:

$$\left[ \sum_{x \in \{0,1\}^n} Q(x) = \alpha \right] \Rightarrow V \text{ accepts w.p. } 1$$

- Soundness:

$$\left[ \sum_{x \in \{0,1\}^n} Q(x) \neq \alpha \right] \Rightarrow \left[ \forall P^*, Pr[V \text{ accepts with } P^*] \leq \frac{nd}{|\mathbb{F}|} \right]$$

Lemma: For every 3CNF;  $\phi : \{0,1\}^n \rightarrow \{0,1\}^n$  on  $m$  clauses and field  $\mathbb{F}$

## Probobalistically Checkable Proofs - *PCP*

### Definition

We say that a *PPT* machine  $V$  is a *PCP* verifier for  $L$  if:

1. Completeness:

$$[x \in L] \Rightarrow \exists \pi : Pr[V^\pi(x) = 1] = 1$$

2. Soundness:

$$[x \notin L] \Rightarrow \forall \pi^* : Pr[V^{\pi^*}(x) = 1] \leq \frac{1}{2}$$

3. Parameters:

- Query complexity - number of queries to  $\pi$ , denoted with  $q$ .
- Proof length -  $|\pi|$ , denoted with  $l$ .
- Randomness length - number of random bits sampled by the verifier, denoted with  $r$ .

We denote  $PCP(q, r)$  the class of languages with *PCP* with query complexity  $q$  and randomness length  $r$ .

### Initial Properties & Conclusions

1. Claim:  $l \leq q \cdot 2^r$
2. Claim:  $NP \subseteq PCP(poly, 0)$ , more formally:

$$NP \subseteq \bigcup_{p \in poly} PCP(p, 0)$$

3. Claim: if  $L$  has an interactive proof in which each prover/verifier message has length  $a/b$  respectively. and with  $k$  rounds, then  $L$  has a *PCP* with length  $a \cdot 2^{kb}$  and query complexity  $a \cdot k$ .
4. Claim:  $PCP(q, r)$  has *NP* proof of length  $q \cdot 2^r$ .

3. results with that any  $L \in PSPACE$  has a proof of exponential length that can be checked by a polynomial verifier, with a polynomial number of queries.

### PCP theorem: $NP = PCP(O(1), O(\log(n)))$

We will see the proof of this claim in a few steps:

- $NP \subseteq PCP(O(1), poly(n))$
- $NP \subseteq PCP(O(\log(n)^2), O(\log(n)))$
- Combining the two claims above to get the theorem.



## Hardness of Approximation (Application)

$GapSAT_\epsilon$ : Accept satisfiable *CNF* formulas, and reject formulas s.t. every assignment satisfies at most  $\epsilon$  clauses. (here  $\epsilon \in [0, 1]$  is the proportion of clauses satisfied).

Theorem:

$$\exists \epsilon > 0 : GapSAT_\epsilon \in NPC$$

Proof:

Let  $L \in NP$ .

Given  $x \in \{0, 1\}^n$ , if  $x \in L$ , there exists a *PCP* proof  $\pi$  for  $x$  (The PCP theorem).

For a given  $x, p$  and  $V_{PCP}$ , define  $V_{x,p} : \{0, 1\}^q \rightarrow \{0, 1\}$  as follows:  
There exists a CNF computing  $V_{x,p}$  with  $2^q$  clauses.  
Consider the following CNF:  $\bigwedge_p V_{x,p}$

- if  $x \in L$ , it is easy to see that  $V_{x,p}$  is satisfied.
- if  $x \notin L$  for any assignment  $\pi$  for half of  $p$ 's  $V_{x,p}(\pi) = 0$  so overall, at-least  $\frac{1}{2} \frac{1}{2^q}$  of the clauses are not satisfied.

With  $\epsilon = 1 - \frac{1}{2^{q+1}}$ , we have that  $L \in GapSAT_\epsilon$ .

## Tutorial 5

### Error-Correcting Codes (ECC)

#### Motivation

The purpose of error correcting codes is to handle systems where information is passed through a noisy channel, some information needs to be written and read from a medium, but there is no guarantee that the information will be read exactly as written - rather, there are some weaker guarantees about the amount of errors or changes that can occur between the information that is written and the information that is read.

#### Introduction

A simple solution to enable correcting errors would be to repeat the written message multiple times - for example, write the input 3 times, and if there is a conflict on some bit - choose the majority.

It is easy to see how any single bit error can be corrected by this method.

## Parameters for ECC

For an error correction code  $C : \mathbb{F}^k \rightarrow \mathbb{F}^n$

1. Rate:  $\frac{n}{k}$  (redundancy  $n - k$ )
2. Minimal Distance:

$$\min_{m_1 \neq m_2} \Delta(C(m_1), C(m_2))$$

$$hw(x) = |\{i | x_i \neq 0\}|, \Delta(x, y) = hw(x - y)$$

## Linear Codes

If  $C : \mathbb{F}^k \rightarrow \mathbb{F}^n$  is a linear function, meaning:

$$\forall m_1, m_2 \in \mathbb{F}^k, \forall \alpha, \beta : C(\alpha \cdot m_1 + \beta \cdot m_2) = \alpha \cdot C(m_1) + \beta \cdot C(m_2)$$

Then we get the following conclusions:

- $C(0) = 0$ .  
Easy to prove by adding 0 to the argument of  $C$ .
- $d = \min_{m \in \mathbb{F}^k \setminus \{0\}} C(m)$ .  
Proof:

1.

$$\begin{aligned} d &= \min_{m_1 \neq m_2} \Delta(C(m_1), C(m_2)) = \min_{m_1 \neq m_2} hw(C(m_1) - C(m_2)) \\ &= \min_{m_1 \neq m_2} hw(C(m_1 - m_2)) \leq \min_{m \neq 0} hw(C(m)) \\ d &\leq \min_{m \neq 0} \Delta(C(m), C(0)) = \min_{m \neq 0} hw(C(m)) \end{aligned}$$

- Singleton Bound:  $d \leq n - k + 1$ .

## ECC important examples

### Hadamard Code

Let  $Had : \{0, 1\}^k \rightarrow \{0, 1\}^{2^k}$  where:

$$\forall m \in \{0, 1\}^k : Had(m)_i = \langle m, i \rangle$$

Then get get:

1. Rate:  $\frac{n}{k} = \frac{2^k}{k}$

2. Absolute distance:  $\frac{2^k}{2}$ .  
 Proof:  $\frac{d}{n}$  of  $Had$  is  $\frac{1}{2}$ .  
 Claim: if  $m \in \{0, 1\}^k \setminus \{\bar{0}\}$  then

$$Pr_{x \in \{0,1\}^k}[\langle m, x \rangle = 0] = Pr_{x \in \{0,1\}^k}[\langle m, x \rangle = 1] = \frac{1}{2}$$

Let  $m \in \{0, 1\}^k \setminus \{\bar{0}\}$ , by claim from HW1, exactly half of the bits of  $Had(m)$  are 1, and thus  $hw(Had(m)) = \frac{1}{2} \cdot 2^k$

## Read-Solomon Codes

Presume we map each input to a polynomial:  $m \rightarrow \mathbb{F}^k[x]$ .

And let  $m_1 \neq m_2 \rightarrow p_1 \neq p_2$ .

Thus  $p_1, p_2$  agree on at most  $k - 1$  points in  $\mathbb{F}^k$ .

$$RS(m) = p(\alpha_1), p(\alpha_2), \dots, p(\alpha_n) \Rightarrow d = n - k + 1$$