

# Chapter 8: Interactive proofs

## Interaction between prover & verifier

- **NP:** Prover sends proof to verifier's certificate tape, then verifier takes over.
- **Oracles:** Prover is an **oracle**; verifier asks questions about **instances of a single problem**, prover answers always **trusted**
- Can we have **more general interactions** between prover & verifier for **cryptography, program checking...**?
- **Deterministic** or **randomized** verifier? **Deterministic** or **randomized** prover?

Our provers will be:

- 1 All-powerful
- 2 Not trusted
- 3 Deterministic

# Chapter 8: Interactive proofs

## Deterministic prover & verifier

---

### Protocol 1 Deterministic 3SAT

---

**for each** clause  $C = (l_1 \vee l_2 \vee l_3)$  **do**

**Verifier:** Values  $l_1, l_2, l_3$ ?

**Prover:** Send  $l_1, l_2, l_3$  to Verifier

**Verifier:** If (all clauses satisfied)  $\wedge$  (all literals consistent) **then** ACCEPT **else** REJECT

---

- If  $m$  clauses, then we have  $2m$  rounds of (alternate) interaction
- We can have only 2 rounds, where verifier asks for all clauses simultaneously and prover replies
- The verifier speaks last to accept or reject

# Chapter 8: Interactive proofs

## Definition 1 (Interaction of deterministic functions)

Let  $V, P : \{0, 1\}^* \rightarrow \{0, 1\}^*$  functions. A  $k$ -round interaction of  $V, P$  on input  $x$  is string sequence  $a_1, a_2, \dots, a_k$  s.t.

$$a_1 = V(x)$$

$$a_2 = P(x, a_1)$$

...

$$a_{2i+1} = V(x, a_1, \dots, a_{2i})$$

$$a_{2i+2} = P(x, a_1, \dots, a_{2i+1})$$

...

$$a_k = P(x, a_1, \dots, a_{k-1})$$

The output of the interaction is  $out_{V,P}(x) = V(x, a_1, \dots, a_k)$  (0 or 1).

**Note:** Think of  $a_{2i+1}$  as Verifier questions, and  $a_{2i+2}$  as Prover replies.

# Chapter 8: Interactive proofs

## Definition 2 (Deterministic proof system)

$L$  has a  **$k$ -round deterministic interactive proof system** if there is TM  $V$  s.t.  $V(x, a_1, \dots, a_i)$  runs in time  $\text{poly}(|x|)$ , can have  $k$ -round interactions with provers  $P$ , and

$$x \in L \Rightarrow \exists P : \text{out}_{V,P}(x) = 1 \quad (\text{Completeness})$$

$$x \notin L \Rightarrow \forall P : \text{out}_{V,P}(x) = 0 \quad (\text{Soundness})$$

## Definition 3

$L \in \text{dIP}$  if  $L$  has  $k(n)$ -round deterministic interactive proof system where  $k(n) = \text{poly}(n)$ .

**Note 1:** Both the verifier *and* the number of rounds are **polynomial** on the size of input  $|x| = n$ .

**Note 2:**  $\exists P$  and  $\forall P$  above mean  $\exists(a_2, a_4, \dots, a_k)$  and  $\forall(a_2, a_4, \dots, a_k)$ .

# Chapter 8: Interactive proofs

## Lemma 4

$dIP = NP$ .

### Proof:

- $NP \subseteq dIP$ : If  $L \in NP$  then  $V$  is the certifier for  $L$  and just asks for a certificate (2 rounds).
- $dIP \subseteq NP$ : If  $L \in dIP$  then let  $a_2, a_4, \dots, a_{k(n)}$  be the prover answers (our certificate). Certifier gets certificate, runs  $V(x) \rightarrow a_1, V(x, a_1, a_2) \rightarrow a_3, \dots$ , and finally checks  $V(x, a_1, a_2, \dots, a_{k(n)}) = 1$ . Certifier is good because:
  - $x \in L$ :  $\exists$  Prover answers  $a_2, a_4, \dots, a_{k(n)}$  s.t. everything consistent and  $V(x, a_1, a_2, \dots, a_{k(n)}) = 1$  (i.e.,  $\exists$  certificate to make certifier accept)
  - $x \notin L$ :  $\forall$  Prover answers  $a_2, a_4, \dots, a_{k(n)}$ , either inconsistent or  $V(x, a_1, a_2, \dots, a_{k(n)}) = 0$  (i.e.,  $\forall$  certificates certifier rejects)

$\Rightarrow L \in NP$

□

# Chapter 8: Interactive proofs

What if  $V$  is a **probabilistic** TM?

**Example:** How can colour-blind **A**rthur ( $V$ ) figure out whether **M**erlin ( $P$ ) wears socks of different colours? If **A** is deterministic, then **M** easily tricks him. What if **A** is **probabilistic**?

## Definition 5 (Probabilistic verifiers with private coins)

$L$  is in **IP** $[k]$  if there is **probabilistic** TM  $V$  with **private coins**  $r$  s.t.  $V(x, r, a_1, \dots, a_i)$  runs in time  $\text{poly}(|x|)$ , can have  $k$ -round interactions with provers  $P$ , and

$$x \in L \Rightarrow \exists P : \Pr_r[\text{out}_{V(r),P}(x) = 1] \geq 2/3 \quad (\text{Completeness})$$

$$x \notin L \Rightarrow \forall P : \Pr_r[\text{out}_{V(r),P}(x) = 1] \leq 1/3 \quad (\text{Soundness})$$

## Definition 6

$$\text{IP} = \cup_{c \geq 0} \text{IP}[n^c].$$

# Chapter 8: Interactive proofs

We have:

$$x \in L \Rightarrow \exists P : Pr_r[out_{V(r),P}(x) = 1] \geq 2/3 \quad (\text{Completeness})$$

$$x \notin L \Rightarrow \forall P : Pr_r[out_{V(r),P}(x) = 1] \leq 1/3 \quad (\text{Soundness})$$

## Lemma 7 (Probability boosting)

We can replace  $2/3$  by  $1 - 2^{-n^c}$ , and  $1/3$  by  $2^{-n^c}$  for any  $c > 0$  without changing IP.

**Proof:** Same as for BPP (repeat interaction protocol  $m$  times and  $V$  outputs majority of outputs), apply Chernoff...

**Objection:**  $P$  learns from previous interactions! Yes, but (Soundness) works  $\forall P$  (even for  $P$  that learns)!

□

$2/3$  can be even pushed to 1, i.e., **Perfect Completeness**! (Non-trivial proof...) Can we push  $1/3$  to 0, i.e., **Perfect Soundness** at the same time? If yes,  $IP = NP$ !

## Chapter 8: Interactive proofs

What about a **probabilistic** Prover?

$$x \in L \Rightarrow \exists P : Pr_{r,s}[out_{V(r),P(s)}(x) = 1] \geq 2/3 \quad (\text{Completeness})$$

$$x \notin L \Rightarrow \forall P : Pr_{r,s}[out_{V(r),P(s)}(x) = 1] \leq 1/3 \quad (\text{Soundness})$$

It doesn't make any difference (**averaging argument**)...

### Lemma 8

$$IP \subseteq PSPACE$$

#### Proof:

- Since  $V$  runs in  $O(n^c)$  time,  $a_1, a_2, \dots, a_{n^d}$  are of length  $O(n^c)$  each, for a total of  $O(n^{c+d})$  space.
- Enumerate all  $a_1, a_2, \dots, a_{n^d}$  to find (consistent) one that maximizes  $Pr_r[out_{V(r),P}(x) = 1]$  (how to compute this?). If  $\geq 2/3$  then ACCEPT, else REJECT.

## Chapter 8: Interactive proofs

### Graph Isomorphism (GI)

**Input:** Graphs  $G_1 = (V_1, E_1)$ ,  $G_2 = (V_2, E_2)$

**Output:** ACCEPT if  $\exists \pi$  permutation of  $V_1$  labels, so that  $\pi(G_1) = G_2$ .

### Graph Non-Isomorphism (GNI) = $\overline{GI}$

#### Lemma 9

$GI \in NP$  and  $GNI \in coNP$

Is  $GI \in P$ ? **OPEN**

Is  $GI$  NP-complete? **OPEN**

# Chapter 8: Interactive proofs

## Lemma 10

$GNI \in IP$

### Proof:

---

#### Protocol 2 Private coin GNI

---

**V:** Pick  $i \in_R \{1, 2\}$  and random  $\pi$ . Let  $H = \pi(G_i)$ . Send  $H$  to **P**.

**P:** Identify which of  $G_1, G_2$  generated  $H$ , say  $G_j$ . Send  $j$  to **V**.

**V:** If  $i = j$  then ACCEPT else REJECT

---

Graphs are socks! If different colour, then **P** always finds correct  $i$ , and

$$Pr_r[out_{V(r),P}(x) = 1] = 1$$

If same colour ( $G_1 \cong G_2$ ), then **P** can guess  $i$  with probability  $1/2$ , i.e.,

$$Pr_r[out_{V(r),P}(x) = 1] \leq 1/2.$$

Can reduce  $1/2$  to  $2/3$  by repetition (Lemma 7).

□

# Chapter 8: Interactive proofs

## Zero knowledge proofs (ZKP)

Can **P** persuade **V** about the truth of a statement, **without revealing any information to V**?

**Without revealing any information to V:** Whatever **V** learns from interaction with **P** to prove statement  $x$ , it could have **computed by itself**, without participating in any interaction.

- Restrict to  $NP$  statements, i.e., statements  $x \in L$  for  $L \in NP$ . Let poly-time TM  $M$  s.t.

$$x \in L \Leftrightarrow \exists u \in \{0, 1\}^{\text{poly}(|x|)} : M(x, u) = 1$$

- ZKP means: **P** tries to persuade **V** that it has a **certificate  $u$**  s.t.  $M(x, u) = 1$
- Can define ZKP for other classes, but  $NP$  is enough to demonstrate

# Chapter 8: Interactive proofs

## Definition 11 (Perfect zero knowledge proof)

Let pair of **poly-time probabilistic algorithms**  $P$ ,  $V$  have interaction  $\langle P(x, u), V(x) \rangle$  and  $\text{out}\langle P(x, u), V(x) \rangle \in \{0, 1\}$  be  $V$ 's output at the end.

- **Completeness:** If  $x \in L$  and  $u$  certificate for  $x$  (i.e.,  $M(x, u) = 1$ ), then

$$\Pr[\text{out}\langle P(x, u), V(x) \rangle = 1] \geq 2/3$$

- **Soundness:** If  $x \notin L$ , then

$$\forall P^*, u : \Pr[\text{out}\langle P^*(x, u), V(x) \rangle = 1] \leq 1/3$$

- **Perfect ZK:**  $\forall$  poly-time probabilistic  $V^*$ ,  $\exists$  expected poly-time **simulator**  $S^*$  s.t.

$$\forall x \in L, u : \Pr[\text{out}\langle P(x, u), V^*(x) \rangle = 1] = \Pr[S^*(x) = 1]$$

## Chapter 8: Interactive proofs

- Perfect ZK relaxed to small statistical distance  $\Rightarrow$  Statistical ZK (SZK)
- Perfect ZK relaxed to computationally indistinguishable  $\Rightarrow$  Computational ZK
- People believe  $P \subset \text{SZK} \subset NP$

# Chapter 8: Interactive proofs

---

## Protocol 3 PZK for $GI(G_0, G_1)$

---

**P:** Has node label permutation  $\pi(G_0) = G_1$  (GI certificate). Picks random permutation  $\pi_1$ . Sends  $\pi_1(G_1)$ .

**V:** Choose random  $b \in \{0, 1\}$ . Send  $b$ .

**P:** If  $b = 1$  then send  $\pi_1$  else send  $\pi_1 \circ \pi$ .

**V:**  $H :=$  first message (graph) received;  $\pi_2 :=$  second message (permutation) received.

If  $H = \pi_2(G_b)$  then return 1 else return 0

---

**Completeness:**  $G_0 \cong G_1 \Rightarrow \pi(G_0) = G_1$ .

- If  $b = 1$  then  $\pi_2(G_b) = \pi_1(G_1) = H$ .

- If  $b = 0$  then  $\pi_2(G_b) = \pi_1 \circ \pi(G_0) = \pi_1(G_1) = H$

$\Rightarrow \Pr[\text{out}\langle P(x, u), V(x) \rangle = 1] = 1$

**Soundness:**  $G_0 \not\cong G_1 \Rightarrow \pi(G_0) \neq G_1$ .

- If  $b = 1$  as before (**wrong**)

- If  $b = 0$  then  $\pi_2(G_b) = \pi_1 \circ \pi(G_0) \neq \pi_1(G_1) = H$  (**correct**)

$\Rightarrow \Pr[\text{out}\langle P(x, u), V(x) \rangle = 1] \leq 1/2$

# Chapter 8: Interactive proofs

---

## Protocol 4 PZK for $GI(G_0, G_1)$

---

**P:** Has node label permutation  $\pi(G_0) = G_1$  (GI certificate). Picks **random permutation**  $\pi_1$ . Sends  $\pi_1(G_1)$ .

**V:** Choose **random**  $b \in \{0, 1\}$ . Send  $b$ .

**P:** If  $b = 1$  then send  $\pi_1$  else send  $\pi_1 \circ \pi$ .

**V:**  $H :=$  first message (graph) received;  $\pi_2 =$  second message (permutation) received.

If  $H = \pi_2(G_b)$  then return 1 else return 0

---

**Perfect ZK:** What does **V** get from **P**?

- **Random permutation of  $G_1$**  ( $\pi_1(G_1)$ )
- Either **same random permutation  $\pi_1$**  (if  $b = 1$ ) or **another random permutation  $\pi_1 \circ \pi$**

**Crucial fact:** A permutation of a **random permutation** is itself a **random permutation**! Does this reminds us of something?

**Yes!** XOR  $x \oplus y$  of a **random**  $x$  with a number  $y$  is also **random**! (but  $x$ ,  $x \oplus y$  not independent)

**AHA!** **P** used **random  $\pi_1$**  to **mask** certificate  $\pi$ !

# Chapter 8: Interactive proofs

---

**Protocol 5** PZK for  $GI(G_0, G_1)$ 

---

**P:** Has node label permutation  $\pi(G_0) = G_1$  (GI certificate). Picks **random permutation**  $\pi_1$ . Sends  $\pi_1(G_1)$ .

**V:** Choose **random**  $b \in \{0, 1\}$ . Send  $b$ .

**P:** If  $b = 1$  then send  $\pi_1$  else send  $\pi_1 \circ \pi$ .

**V:**  $H :=$  first message (graph) received;  $\pi_2 =$  second message (permutation) received.

If  $H = \pi_2(G_b)$  then return 1 else return 0

---

## Simulator $S^*(G_0, G_1)$ :

① Pick random  $b' \in_R \{0, 1\}$  and random permutation  $\pi_2$ .

$H := \pi_2(G_{b'})$

② Send  $H$  to  $V^*$  to get  $b$ .

③ If  $b = b'$  then send  $\pi_2$  to  $V^*$ ; return what  $V^*$  returns else rerun  $S^*$

$\Rightarrow \Pr[S^*(G_0, G_1) = \text{out}\langle P(x, u), V^*(x) \rangle \text{ in 1 iter}] = \Pr[b = b'] = 1/2$

$\Rightarrow E[T(n)] = \sum_{i=1}^{\infty} 2^{-i} V^*(n) = O(V^*(n))$

# Chapter 8: Interactive proofs

## Public coins vs. private coins

### Definition 12 ( $AM$ , $MA$ )

$AM[k] \subseteq IP[k]$  is class of interactive protocols, where  $V$  always reveals the random bits it used to  $P$ .

- Book says “ $V$ ’s messages to  $P$  contain only its random bits”. No need, since if  $P$  knows  $V$ ’s random bits up to now, then it can figure out the rest of the message  $V$  sends.
- Traditionally, computationally-restricted  $V$  called  $A$ thur, and all-powerful  $P$  called  $M$ erlin.  $AM[k]$  if  $A$  starts,  $MA[k]$  if  $M$  starts the interaction.
- $AM[2]$ ,  $MA[2]$  traditionally called  $AM$ ,  $MA$ .
- $AM = BP \cdot NP = \{L : L \leq_r 3SAT\}$  (why?)
- For any constant  $k \geq 2$ ,  $AM[k] = AM$  (proof omitted)

# Chapter 8: Interactive proofs

## Set Lower Bound

**Given:** Set  $S \subseteq \{0,1\}^n$  that  $x \in S$  can be **certified**, i.e., has poly-time TM  $M$  s.t.  $x \in S \Leftrightarrow \exists u : M(x, u) = 1$  (so  $x \in S$  is an **NP** question).  
Number  $K$  with  $2^{k-2} < K \leq 2^{k-1}$ .

**Prover:** Tries to persuade  $V$  that  $|S| \geq K$ .

**Verifier:** Rejects with “good” probability if  $|S| \leq \frac{K}{2}$ .

What about  $\frac{K}{2} < |S| < K$ ? We don't care what  $V$  answers! Our first example of a **gap** problem.

# Chapter 8: Interactive proofs

## Hashing detour

- Hash function  $h : \{0, 1\}^n \rightarrow \{0, 1\}^k$ , usually  $n \geq k$
- If  $x \neq x'$  and  $h(x) = h(x')$  then this is a **collision**
- What is a **good** hash function?
  - We want  $x$ 's to be **uniformly** spread (mapped) to  $y$ 's, i.e.,
  - We want every  $y \in \{0, 1\}^k$  to get the same number of pre-images, i.e.,  $|\{x : h(x) = y\}| = \frac{2^n}{2^k} = 2^{n-k}$ .
  - Equivalently,  $Pr_x[h(x) = y] = \frac{1}{2^k} = 2^{-k}$  (why?).
  - What if I keep  $x$  fixed (like  $y$ ) and I pick a **random**  $h$  from a **family**  $\mathcal{H}_{n,k}$  of hash functions? If  $Pr_{h \in \mathcal{H}_{n,k}}[h(x) = y] = 2^{-k}$  for every  $x, y$  then again I have a **uniform** mapping of  $x$ 's to  $y$ 's, and the family  $\mathcal{H}_{n,k}$  is **good**.

### Definition 13 (Pairwise independent hash family)

Hash family  $\mathcal{H}_{n,k}$  is **pairwise independent** if

$$\forall x \neq x', \forall y, y' : Pr_{h \in \mathcal{H}_{n,k}}[h(x) = y \wedge h(x') = y'] = 2^{-2k}$$

# Chapter 8: Interactive proofs

## Definition 14 (Pairwise independent hash family)

Hash family  $\mathcal{H}_{n,k}$  is **pairwise independent** if

$$\forall x \neq x', \forall y, y' : \Pr_{h \in \mathcal{H}_{n,k}} [h(x) = y \wedge h(x') = y'] = 2^{-2k}$$

## Corollary 1

If family  $\mathcal{H}_{n,k}$  is pairwise independent, then it is also **good**, i.e.,  $\Pr_{h \in \mathcal{H}_{n,k}} [h(x) = y] = 2^{-k}$ .

**Proof:** We use the following simple fact:

If event space  $\Omega = \{B_1, B_2, \dots, B_m\}$ , then:

$$\begin{aligned} \Pr[A] &= \Pr[A \wedge \Omega] = \Pr[A \wedge (B_1 \vee B_2 \vee \dots \vee B_m)] \\ &= \Pr[(A \wedge B_1) \vee (A \wedge B_2) \vee \dots \vee (A \wedge B_m)] \\ &= \sum_{i=1}^m \Pr[A \wedge B_i] \quad (\text{events } A \wedge B_i \text{ are mutually independent}) \end{aligned}$$

Pick any  $x'$  and apply with  $A \leftarrow h(x) = y$  and  $B_i \leftarrow h(x') = y_i$  for all  $m = 2^k$   $k$ -strings  $y_i$ . □

# Chapter 8: Interactive proofs

## Definition 15 (Pairwise independent hash family)

Hash family  $\mathcal{H}_{n,k}$  is **pairwise independent** if

$$\forall x \neq x', \forall y, y' : Pr_{h \in \mathcal{H}_{n,k}}[h(x) = y \wedge h(x') = y'] = 2^{-2k}$$

## Corollary 2

If family  $\mathcal{H}_{n,k}$  is pairwise independent, then it is also **good**, i.e.,  
 $Pr_{h \in \mathcal{H}_{n,k}}[h(x) = y] = 2^{-k}$ .

## Theorem 16

*There is a (easily computable) pairwise independent hash family  $\mathcal{H}_{n,k}$ .*

**Proof:** See Theorem 8.15 in book. □

# Chapter 8: Interactive proofs

## Set Lower Bound (SLB)

**Given:** Set  $S \subseteq \{0, 1\}^n$  that  $x \in S$  can be **certified**, i.e., has poly-time TM  $M$  s.t.  $x \in S \Leftrightarrow \exists u : M(x, u) = 1$  (so  $x \in S$  is an **NP** question). Number  $K$  with  $2^{k-2} < K \leq 2^{k-1}$ .

**Prover:** Tries to persuade  $V$  that  $|S| \geq K$ .

**Verifier:** Rejects with “good” probability if  $|S| \leq \frac{K}{2}$ .

---

**Protocol 6** Goldwasser-Sipser public-coin protocol for SLB

---

**V:** Pick random hash function  $h \in_R \mathcal{H}_{n,k}$ , pick random  $y \in_R \{0, 1\}^k$ . Send  $h, y$ .

**P:** Try find  $x \in S$  s.t.  $h(x) = y$ . Send  $x$  and **certificate**  $u$  of  $x \in S$ .

**V:** If  $(h(x) = y \wedge M(x, u) = 1)$  then return 1 else return 0

---

**Proof intuition:** Let  $p^* = K/2^k$ . If  $S$  is **big** ( $|S| \geq K$ ) then **P** has very good chance ( $\geq \frac{3}{4}p^*$ ) to find  $x : h(x) = y$ , but if  $S$  is **small** ( $|S| \leq K/2$ ) its chances fall a lot ( $\leq \frac{1}{2}p^*$ )

## Chapter 8: Interactive proofs

### Lemma 17

$SLB \in AM$ .

**Proof:** Let  $p^* = K/2^k$ . If  $S$  is **big** ( $|S| \geq K$ ) then  $P$  has very good chance ( $\geq \frac{3}{4}p^*$ ) to find  $x : h(x) = y$ , but if  $S$  is **small** ( $|S| \leq K/2$ ) its chances fall a lot ( $\leq \frac{1}{2}p^*$ )

### Claim 1

If  $|S| \leq \frac{2^k}{2} < K$  and  $p = |S|/2^k$ , then

$$p \geq \Pr_{h,y}[\exists x \in S : h(x) = y] \geq \frac{3p}{4}.$$

- If  $|S| \leq \frac{K}{2} \leq 2^{k-2}$  then  $p \leq 1/4$  and  $3p/4 \geq \frac{3}{8} \frac{|S|}{K}$
- We can boost (Chernoff bound) to  $\geq 2/3, \leq 1/3$ : Run protocol constant  $M$  times,  $V$  accepts if accepting iterations  $\geq 5p^*M/8$ .
- Can run in parallel in 2 rounds. □

## Chapter 8: Interactive proofs

### Lemma 18

$SLB \in AM$ .

We have  $AM$  protocol to decide whether a set is big or small. We can apply it to show

### Theorem 19

$GNI \in AM$ .

**Proof:**

$$S = \{H : H \cong G_0 \text{ or } H \cong G_1\}$$

- If  $G_0 \not\cong G_1$  then  $|S| = 2n!$  ( $S$  big)
- If  $G_0 \cong G_1$  then  $|S| = n!$  ( $S$  small)

(not exactly, but this is the main idea)

□

# Chapter 8: Interactive proofs

## Lemma 20

$SLB \in AM$ .

We have  $AM$  protocol to decide whether a set is big or small. We can apply it to show

## Theorem 21

$GNI \in AM$ .

## Theorem 22 (Goldwasser-Sipser)

For every  $k \geq 2$ ,  $IP[k] = AM[k + 2]$ .

**Proof idea:** There is a gap in the number of private random strings making  $IP[k]$   $\forall$  accept in YES and NO instances.  $P$  of  $AM[k + 2]$  uses  $SLB$  to persuade the  $AM[k + 2]$   $\forall$  that the number is large.

## Chapter 8: Interactive proofs

It can be shown that class  $AM[k]$  doesn't change if we require **perfect completeness** (probability of success for YES instance is 1). Like  $GM$ , every private coin protocol can be transformed to public coin protocol with perfect completeness. We can use this to prove

### Theorem 23

*If  $GI$  is NP-complete, then  $\Sigma_2 = \Pi_2$ .*

**Proof:** Omitted

# Chapter 8: Interactive proofs

## Polynomials

- Polynomials defined over fields, e.g.,  $GF(p) = \{0, 1, 2, \dots, p-1\}$  for a prime  $p$  (same as *modulo  $p$*  arithmetic).  $GF(2)$  = *binary*.
- Univariate polynomial  $p(x) = 3x^5 + x^3 - 0.5x - 1$  with  $\deg(p) = 5$
- Multivariate polynomial  
 $p(X_1, X_2, X_3, X_4) = 5X_1^3X_3X_4 - X_2^2X_3^2X_4^2 + X_2^5 - 3$  with  $\deg(p) = 6$ .
- If  $\deg(p_1) = d_1$  and  $\deg(p_2) = d_2$  then  $\deg(p_1p_2) = d_1 + d_2$
- Univariate polynomial of  $\deg(p) = d$  has *at most  $d$  roots*, i.e. solutions of  $p(x) = 0 \Rightarrow$  at most  $d$  solutions of  $p(x) = K$ .
- If  $p \gg d$  for a univariate polynomial of  $\deg(p) = d$  over  $GF(p)$ , then *very difficult to guess a root*, i.e.,

$$\Pr_{s \in GF(p)}[p(s) = K] \leq \frac{d}{p}$$

**Idea:** Use this to catch *lying provers*! If  $V$  has  $p_1(x)$  and  $P$  sends  $p_2(x)$ , then  $p_1(x) = p_2(x)$  *only on  $d$  points*.

# Chapter 8: Interactive proofs

## Sumcheck

**Given:** Polynomial  $g(X_1, X_2, \dots, X_n)$  over  $GF(p)$  for prime  $p$  and  $\deg(g) = d$ , integer  $K$ .

**Prover:** Tries to persuade  $V$  that

$$\sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(b_1, b_2, \dots, b_n) = K. \quad (1)$$

**Verifier:** Rejects with “good” probability if (1) not true.

**Assumption:** Polynomial  $g(\cdot)$  has a  $\text{poly}(n)$  representation, and  $V$  can evaluate  $g(x_1, x_2, \dots, x_n)$  in  $\text{poly}(n)$  time.

- Fully expanded  $g$  can have  $\text{exp}(n)$  number of terms!
- If we set  $X_i := b_i$ ,  $i = 2, 3, \dots, n$  then we get univariate polynomial  $g(X_1, b_2, \dots, b_n)$  with  $\deg(p) = d$ . Define

$$h(X_1) := \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(X_1, b_2, \dots, b_n)$$

Then  $(1) \Leftrightarrow h(0) + h(1) = K$ .

# Chapter 8: Interactive proofs

## Sumcheck

**Given:** Polynomial  $g(X_1, X_2, \dots, X_n)$  over  $GF(p)$  for prime  $p$  and  $\deg(g) = d$ , integer  $K$ .

**Prover:** Tries to persuade  $V$  that

$$h(0) + h(1) = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(b_1, b_2, \dots, b_n) = K.$$

---

### Protocol 7 Sumcheck IP

---

**V:** If  $n = 1$  then accept only if  $g(0) + g(1) = K$ . Else ( $n \geq 2$ ) ask **P** to send  $h(X_1)$ .

**P:** Send polynomial  $s(X_1)$ . (**P** can “cheat” by sending  $s(X_1) \neq h(X_1)$ )

**V:** If  $(s(0) + s(1) \neq K)$  then return 0 else pick random  $a \in_R GF(p)$ . Recursively check that

$$s(a) \stackrel{?}{=} h(a) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(a, b_2, \dots, b_n)$$

---

- **Completeness:** If  $h(0) + h(1) = K$  then  $s(X_1) = h(X_1)$  and  $Pr[V \text{ accepts}] = 1$
- **Soundness:** If  $h(0) + h(1) \neq K$  then  $Pr[V \text{ accepts}] \leq 1 - (1 - \frac{d}{p})^n$ .

# Chapter 8: Interactive proofs

---

## Protocol 8 Sumcheck IP

---

**V:** If  $n = 1$  then accept only if  $g(0) + g(1) = K$ . Else ( $n \geq 2$ ) ask **P** to send  $h(X_1)$ .

**P:** Send polynomial  $s(X_1)$ . (**P** can “cheat” by sending  $s(X_1) \neq h(x_1)$ )

**V:** If  $(s(0) + s(1) \neq K)$  then return 0 else pick random  $a \in_R GF(p)$ . Recursively check that

$$s(a) \stackrel{?}{=} h(a) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(a, b_2, \dots, b_n)$$

- 
- **Soundness:** If  $h(0) + h(1) \neq K$  then  $Pr[\text{V accepts}] \leq 1 - (1 - \frac{d}{p})^n$ .

**Proof:** Induction on  $n$ . For  $n = 1$   $Pr[\text{V accepts}] = 0$ . True for  $n = k$ . For  $n = k + 1$ :

$$\begin{aligned} Pr[\text{V accepts}] &= Pr[(\text{V accepts round 3}) \wedge (\text{V accepts recursively})] \\ &= Pr[\text{V accepts round 3}] \cdot Pr[\text{V accepts recursively}] \\ &\leq Pr[s(a) = h(a)] \cdot (1 - (1 - \frac{d}{p}))^k \\ &\leq \frac{d}{p} \cdot (1 - (1 - \frac{d}{p}))^k = (1 - (1 - \frac{d}{p}))^{k+1} \end{aligned}$$

□

# Chapter 8: Interactive proofs

## Arithmetization

Why bother with **polynomials**? We have seen protocols for problems on graphs, sets, algebra... what about **logic**? **Idea:** Transform **logic** to **algebra** via **arithmetization**.

Given 3CNF formula  $\phi(x_1, x_2, \dots, x_n)$ :

- Binary var  $x_1 \rightarrow X_1$  variable in  $GF(p)$
- Literal  $x_i \rightarrow X_i$  and  $\bar{x}_i \rightarrow 1 - X_i$
- $x_i \wedge x_j \rightarrow X_i \cdot X_j$
- $x_i \vee x_j \rightarrow 1 - (1 - X_i)(1 - X_j)$

**Example:**

$C_l = (x_i \vee \bar{x}_j \vee x_k) \rightarrow p_l(X_1, X_2, \dots, X_n) = 1 - X_j(1 - X_i)(1 - X_k)$   
with  $\deg(p_l) = 3$  (book is **wrong** on this example!)

$\Rightarrow P_\phi(X_1, X_2, \dots, X_n) = \prod_{i=1}^m p_i(X_1, X_2, \dots, X_n),$   
with  $\deg(P_\phi) \leq 3m$  and representation size  $O(m)$  (don't expand!)

# Chapter 8: Interactive proofs

## 3SAT

$$\phi(x_1, x_2, \dots, x_n) \in 3SAT \Leftrightarrow \exists b_i \in \{0, 1\}, i = 1, 2, \dots, n : P_\phi(b_1, b_2, \dots, b_n) = 1$$

## #SAT

$$\#SAT = \{ \langle \phi, K \rangle : 3CNF \text{ formula } \phi \text{ has exactly } K \text{ satisfying assignments} \}$$

**Note**  $\overline{3SAT} \leq_P \#SAT$  (What about 3SAT?)

$$\langle \phi, K \rangle \in \#SAT \Leftrightarrow \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} P_\phi(b_1, b_2, \dots, b_n) = K$$

...but this is the **Sumcheck** problem! Apply Protocol 8 to show

Theorem 24

$$\#SAT \in IP$$

# Chapter 8: Interactive proofs

## Theorem 25

$$NP, coNP \subseteq IP \subseteq PSPACE$$

### Proof:

- $\overline{3SAT} \leq_P \#SAT \Rightarrow \overline{3SAT} \in IP$ . (We already know  $NP \subseteq IP$ .)
- All  $IP$  interaction and  $V$  computations are polynomial time. Go over **all** possible  $P$  answers, to discover optimal  $P$ , i.e., maximizes  $V$  acceptance probability (once a set of  $P$  replies is fixed,  $V$  acceptance probability calculated going over all its possible random bits). If best acceptance probability achieved  $\geq 2/3$  then ACCEPT, else REJECT.

□

## Theorem 26

$$IP = PSPACE$$

**Proof:** We show that  $\overline{TQBF} \in IP$  (we have  $PSPACE \subseteq IP$ )

Proof uses exactly the same ideas that show  $\#SAT \in IP$ .

## Chapter 8: Interactive proofs

**Arithmetization** for formula  $\Psi = \forall x_1 \exists x_2 \forall x_3 \dots \exists x_n \phi(x_1, \dots, x_n)$  implies

$$\Psi \in \overline{TQBF} \Leftrightarrow \prod_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \prod_{b_n \in \{0,1\}} P_\phi(b_1, b_2, \dots, b_n) = 0$$

This will produce  $h(\cdot)$  polynomials of degree  $2^n$  in Protocol 8!

We expand **arithmetization**:

$$\begin{aligned} \forall X_i \quad p(X_1, \dots, X_n) &= p(X_1, \dots, X_{i-1}, 0, X_{i+1}, \dots, X_n) \\ &\quad + p(X_1, \dots, X_{i-1}, 1, X_{i+1}, \dots, X_n) \end{aligned}$$

$$\begin{aligned} \exists X_i \quad p(X_1, X_2, \dots, X_n) &= p(X_1, \dots, X_{i-1}, 0, X_{i+1}, \dots, X_n) \\ &\quad + p(X_1, \dots, X_{i-1}, 1, X_{i+1}, \dots, X_n) \end{aligned}$$

$$\begin{aligned} \mathcal{L}X_i \quad p(X_1, X_2, \dots, X_n) &= X_i \cdot p(X_1, \dots, X_{i-1}, 1, X_{i+1}, \dots, X_n) \\ &\quad + (1 - X_i) \cdot p(X_1, \dots, X_{i-1}, 0, X_{i+1}, \dots, X_n) \end{aligned}$$

$\mathcal{L}X_i$  is a **linearization operator** that replaces  $X_i^k \rightarrow X_i$ , since  $X_i^k = X_i$  if  $X_i \in \{0, 1\}$ .

## Chapter 8: Interactive proofs

Use expanded arithmetization to compute polynomials for the following formula:

$$\forall x_1 \mathcal{L}_1 \exists x_2 \mathcal{L}_1 \mathcal{L}_2 \forall x_3 \mathcal{L}_1 \mathcal{L}_2 \mathcal{L}_3 \dots \exists x_n \mathcal{L}_1 \mathcal{L}_2 \dots \mathcal{L}_n \phi(x_1, \dots, x_n)$$

Note that now

$$h(X_1) = \sum_{b_2 \in \{0,1\}} \dots \prod_{b_n \in \{0,1\}} P_\phi(X_1, b_2, \dots, b_n)$$

has degree  $k > 1$  because of the  $\prod$ 's. That's why we linearize  $X_1$  next! (using  $\mathcal{L}_1$ ). As we go down in the recursion, we will need to also linearize  $X_2, X_3, \dots$

Read 8.3.3 in text

# Chapter 8: Interactive proofs

What is the power of using multiple Provers (*MIP*)? Play one prover against the other to force **non-adaptability**...

## Theorem 27

$$MIP = NEXP$$

If we define a **proof**=a table with all Prover answers to Verifier questions, then

## Definition 28

$PCP[r, q]$  is set of languages that are accepted by a Verifier that makes  $q$  queries to a table of size  $2^r$ .

## Theorem 29 (Theorem 27)

$$NEXP = PCP[poly(n), poly(n)]$$

## Theorem 30 (The *PCP* theorem)

$$NP = PCP[O(\log n), O(1)]$$

# Chapter 8: Interactive proofs

## Program checkers

**Program verification problem**, i.e. design algorithm  $C$  s.t.  $C(P) = 1$  iff program  $P$  for computation task  $T$  is **always** correct, is **undecidable**.

**Program checking on an input problem**, i.e. design algorithm  $C^P$  with access to code  $P$  s.t.  $C^P(x) = 1$  iff  $P(x)$  is correct ( $P(x) = T(x)$ ) is **surprisingly easy** if we also allow randomness!

### Definition 31 (Blum-Khanna)

A **checker for computational task  $T$**  is a **probabilistic poly-time** TM  $C$  that, given any **program  $P$**  for  $T$  and any **input  $x$** :

- $P(x) = T(x) \Rightarrow \Pr[C^P \text{ accepts } P(x)] \geq 2/3$  (boosting  $1 - n^{-k}$ )
- $P(x) \neq T(x) \Rightarrow \Pr[C^P \text{ accepts } P(x)] \leq 1/3$  (boosting  $n^{-k}$ )

...Do such animals even exist?

# Chapter 8: Interactive proofs

## Program checker for $GNI$ (or $GI$ )

- If  $P(G_1, G_2) = 1$  (i.e.,  $P$  says  $G_1 \not\cong G_2$ ) then  $C$  runs:

---

### Protocol 9 Private coin GNI

---

- V:** Pick  $i \in_R \{1, 2\}$  and random  $\pi$ . Let  $H = \pi(G_i)$ . Send  $H$  to **P**.  
**P:** Identify which of  $G_1, G_2$  generated  $H$ , say  $G_j$ . Send  $j$  to **V**.  
**V:** If  $i = j$  then ACCEPT else REJECT
- 

...but using code  $P$  instead of **P**.

- If  $P(G_1, G_2) = 0$  (i.e.,  $P$  says  $G_1 \cong G_2$ ) then  $C$  runs:
  - for each  $i \in V_1$  do
    - for each  $j \in V_2$  do
      - Delete  $i, j$  to get  $G'_1, G'_2$
      - if  $P(G'_1, G'_2) = 0$  then
        - $j := \pi(i)$
        - Move to next  $i$
      - else
        - Check correctness of  $P(G'_1, G'_2) = 1$
        - Move to next  $j$
  - if  $G_1 = \pi(G_2)$  then
    - Return ACCEPT
  - else
    - Return REJECT

# Chapter 8: Interactive proofs

## Program checkers

We can use IPs to design program checkers in general:

### Theorem 32

*GI, #SAT, TQBF have checkers.*

### Theorem 33

*P-complete problems have “easy” checkers.*