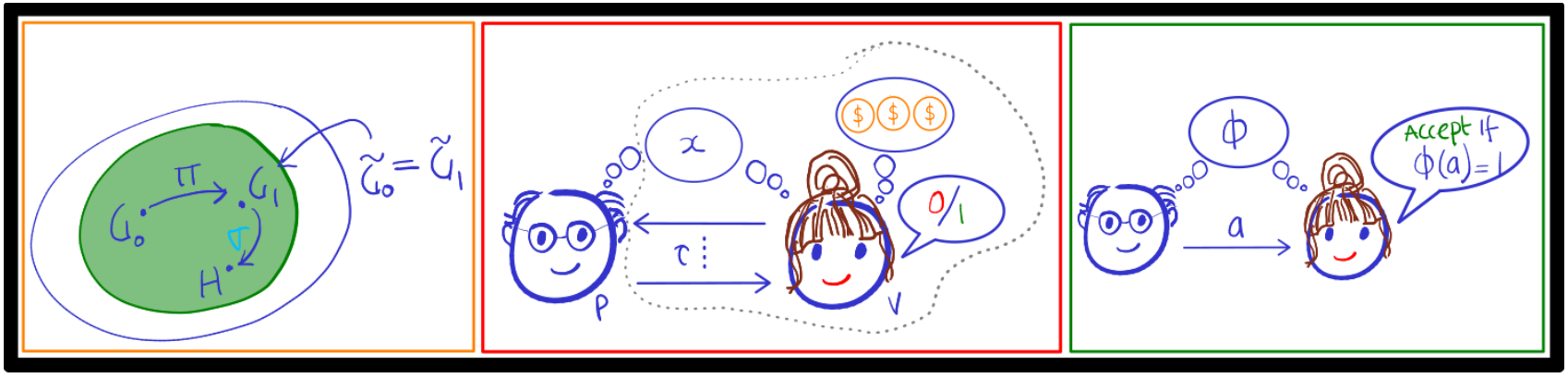


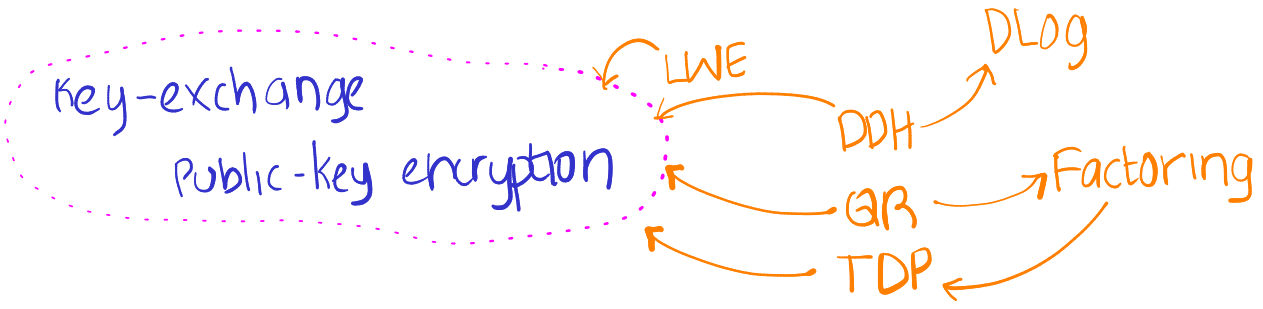
CS783: Theoretical Foundations of Cryptography

Lecture 14 (24/Sep/24)

Instructor: Chethan Kamath

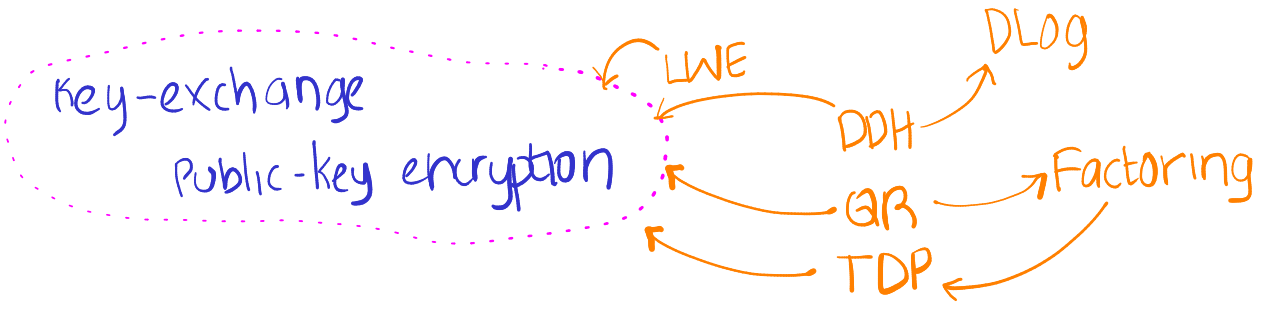
To Recap Previous Module

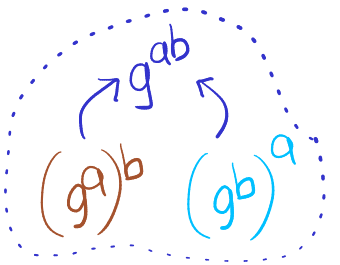
- We learnt: secure communication in the *public-key* setting
- Cryptographic primitives encountered: key-exchange, public-key encryption, signature, hash function, TDP
- Hardness assumptions: Factoring, DLog, QR, LWE, RSA



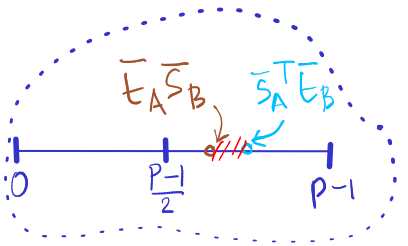
To Recap Previous Module

- We learnt: secure communication in the *public-key* setting
- Cryptographic primitives encountered: key-exchange, public-key encryption, signature, hash function, TDP
- Hardness assumptions: Factoring, DLog, QR, LWE, RSA



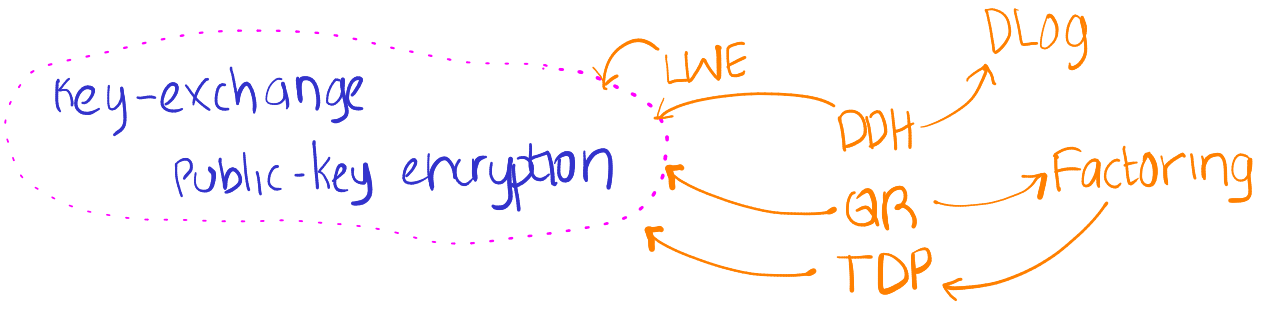


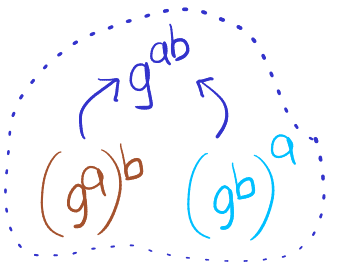
- Key conceptual takeaway: structure vs. hardness



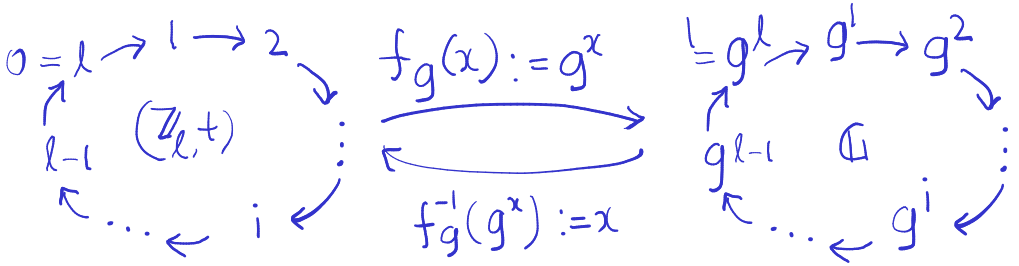
To Recap Previous Module

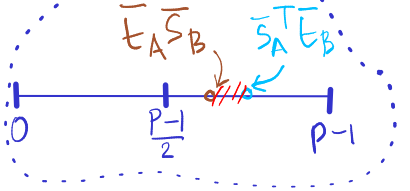
- We learnt: secure communication in the *public-key* setting
- Cryptographic primitives encountered: key-exchange, public-key encryption, signature, hash function, TDP
- Hardness assumptions: Factoring, DLog, QR, LWE, RSA





- Key conceptual takeaway: structure vs. hardness





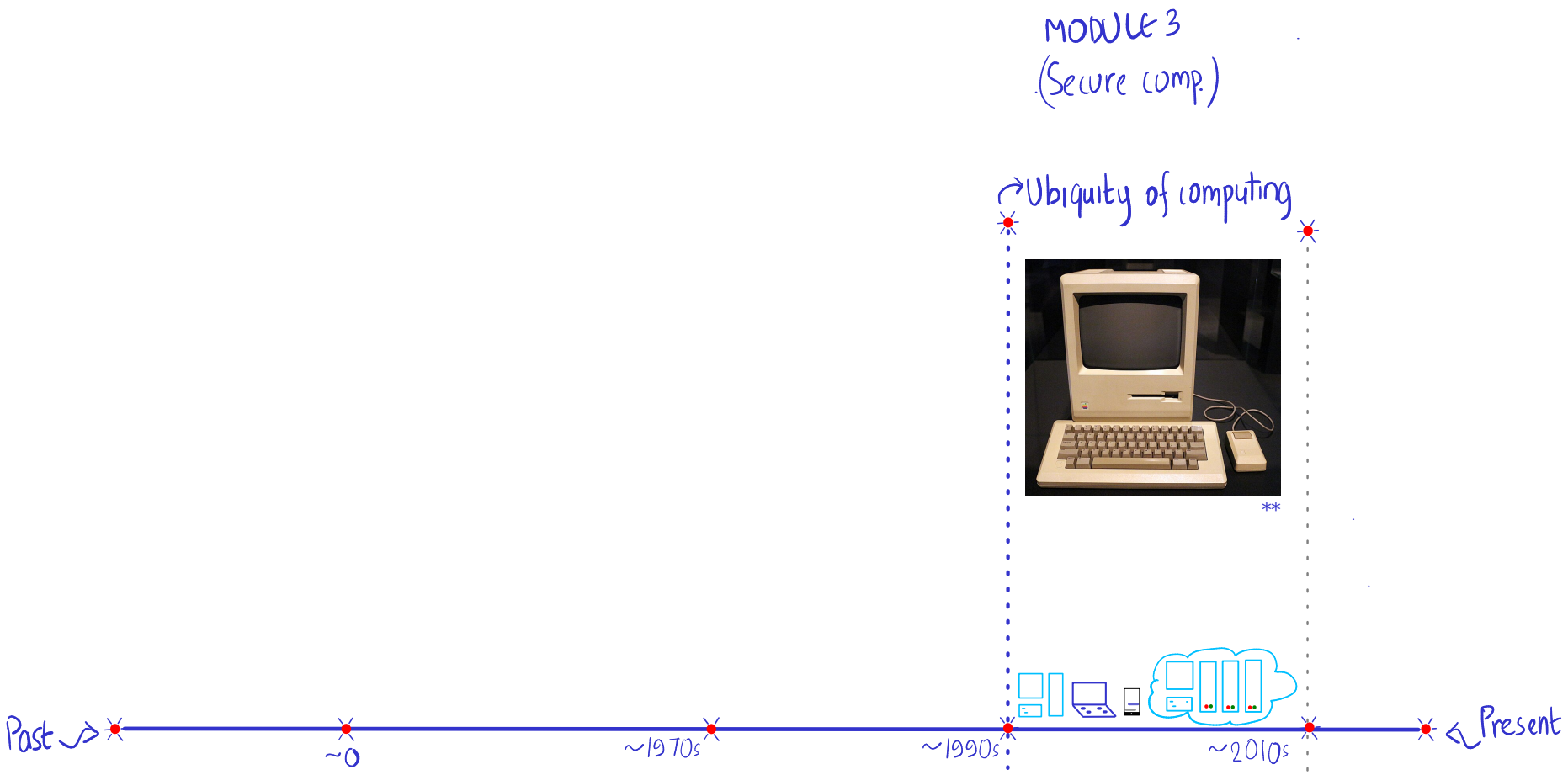
$PK^* =$

y_{00}	y_{10}	y_{20}	y_{30}
y_{01}	y^*	y_{21}	y_{31}

$i^* \leftarrow [\ell]$ $\quad b^* \leftarrow \{0,1\}$

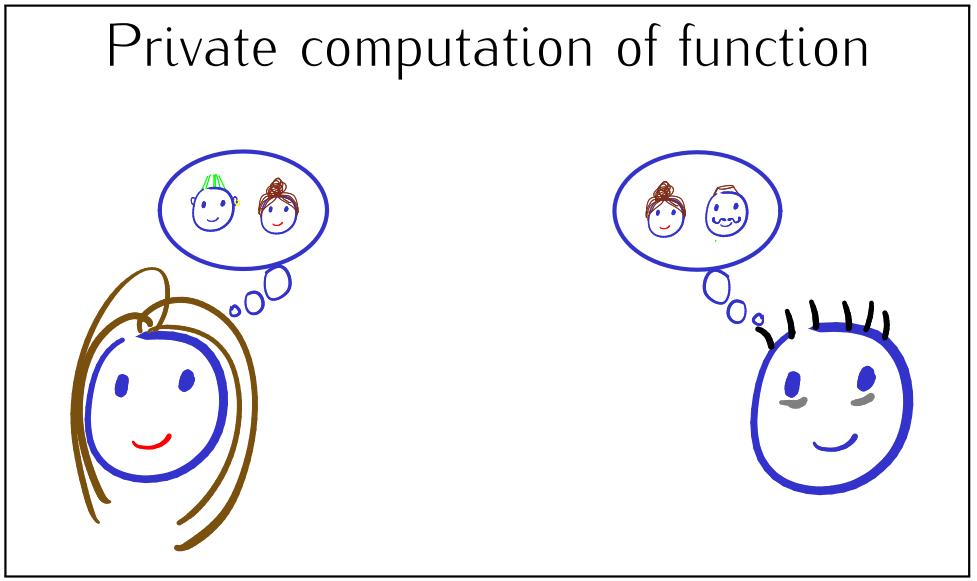
- Key tools: groups, random self-reducibility, plug and pray

This Module: Secure Computation...

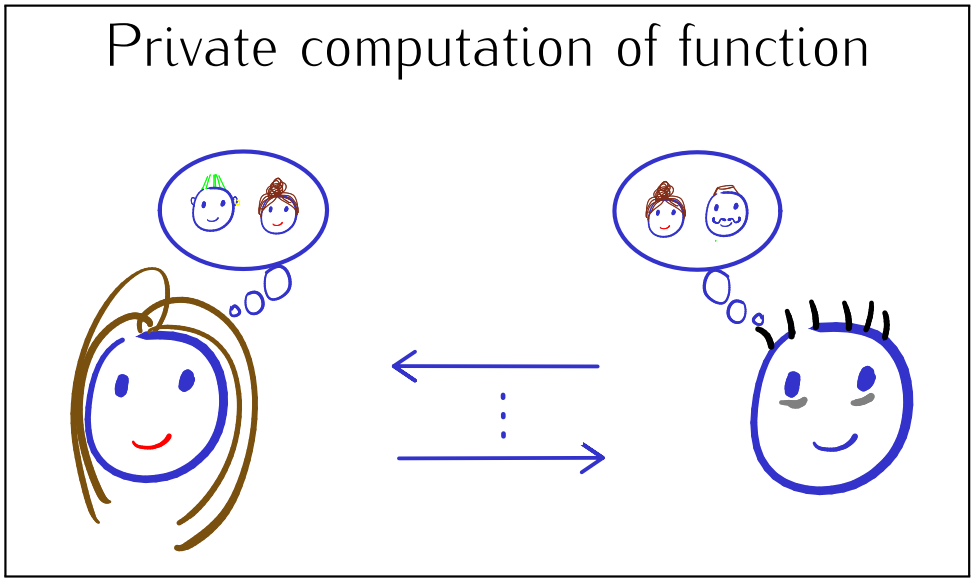


Credit for images: Wikipedia (**User:Saliko)

This Module: Secure Computation...

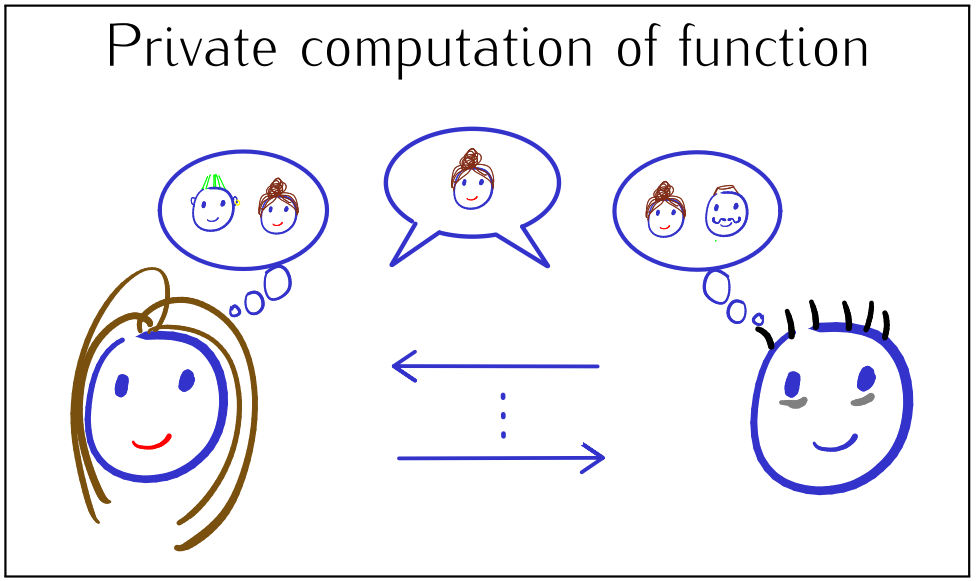


This Module: Secure Computation...



This Module: Secure Computation...

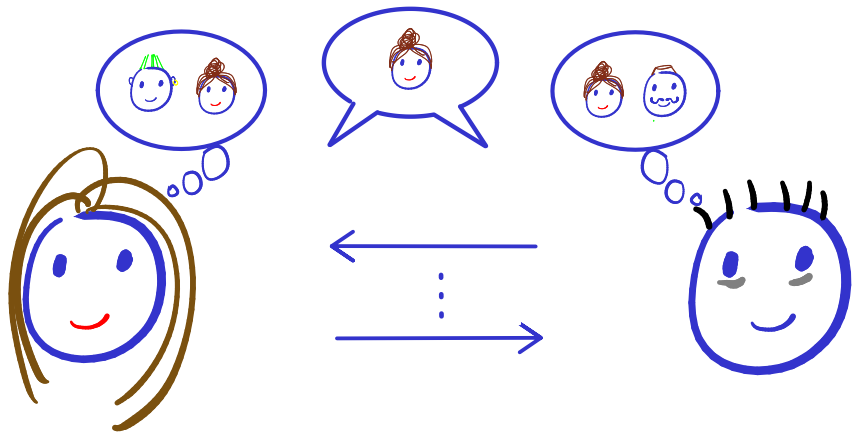
L17-18



This Module: Secure Computation...

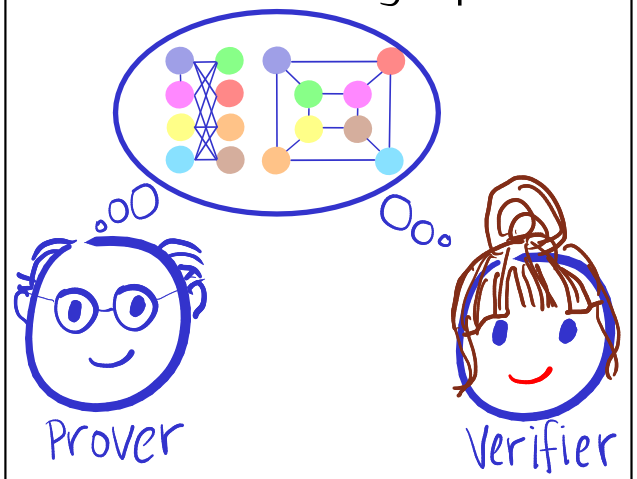
L17-18

Private computation of function



L14-16

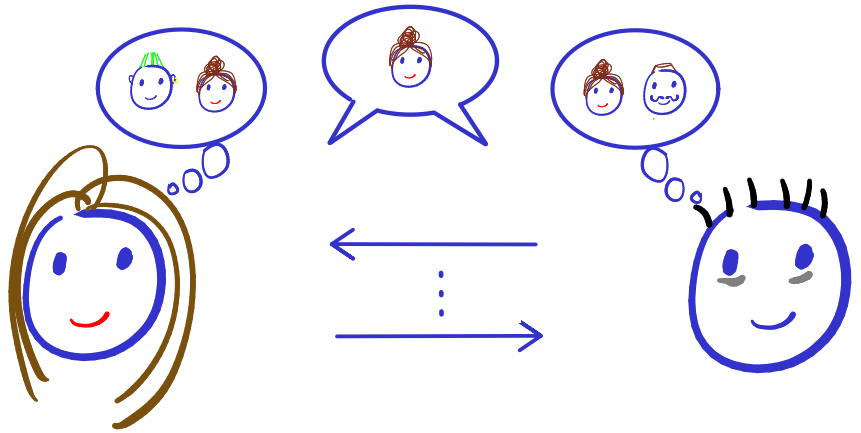
Zero-knowledge proofs



This Module: Secure Computation...

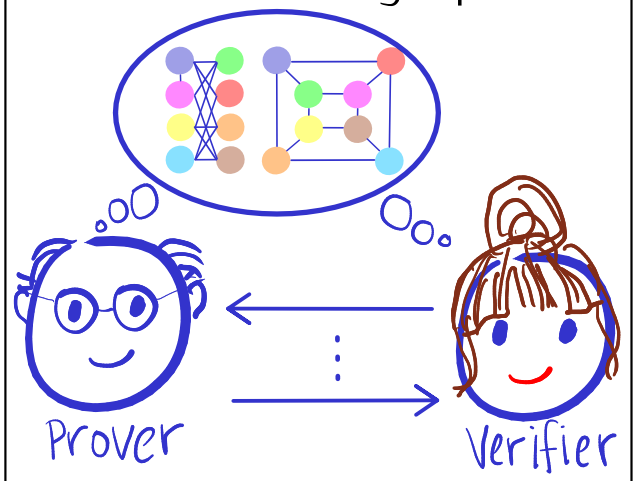
L17-18

Private computation of function



L14-16

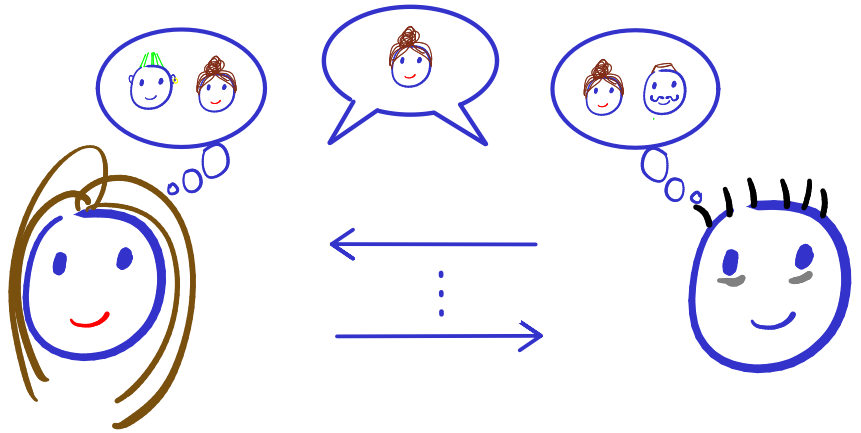
Zero-knowledge proofs



This Module: Secure Computation...

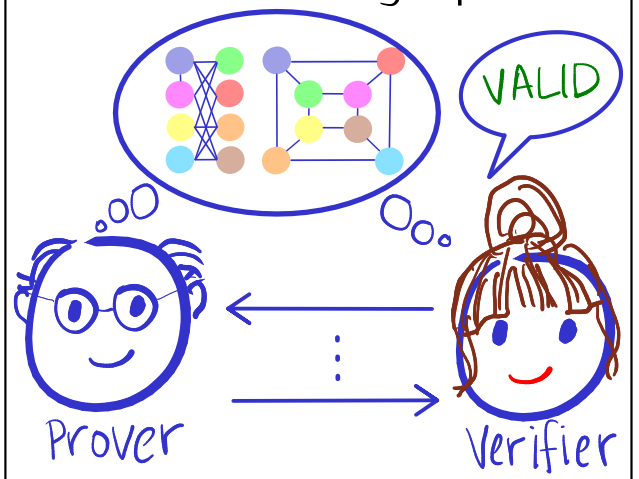
L17-18

Private computation of function



L14-16

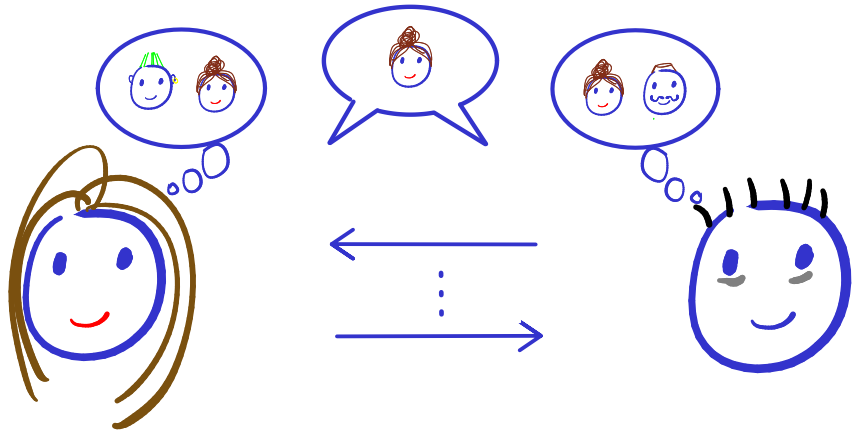
Zero-knowledge proofs



This Module: Secure Computation...

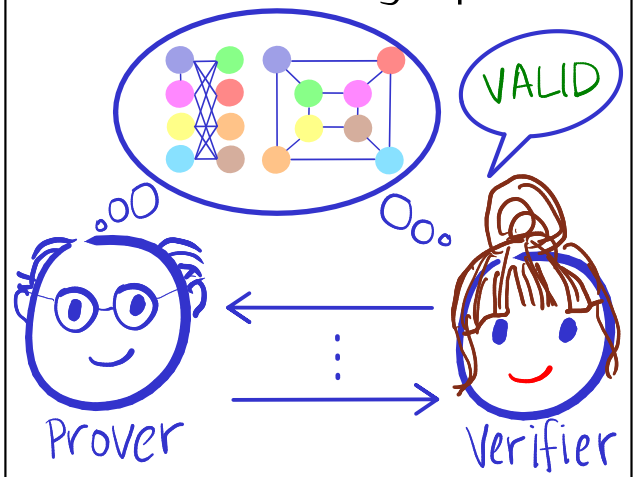
L17-18

Private computation of function

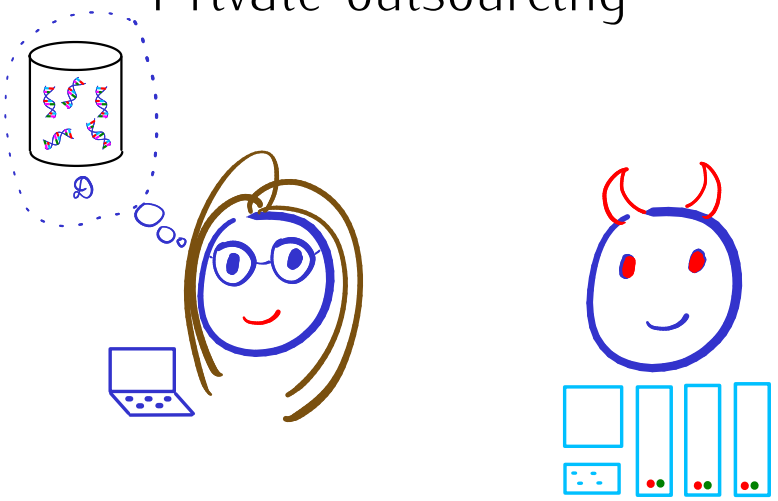


L14-16

Zero-knowledge proofs



Private outsourcing



L19

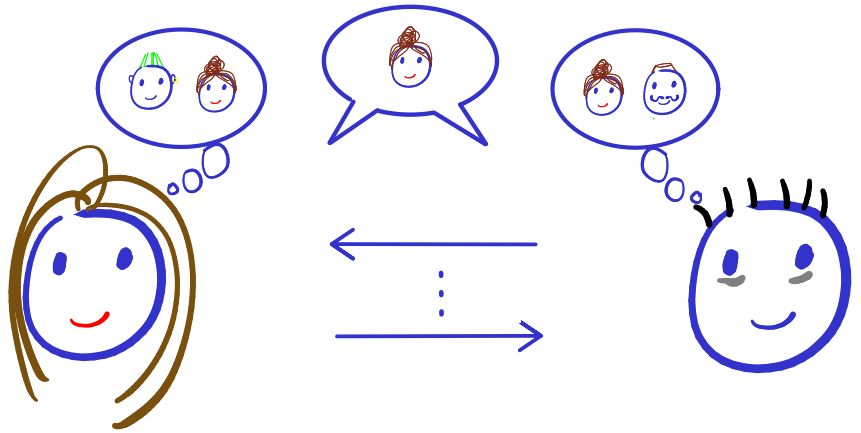
Verifiable outsourcing

L20

This Module: Secure Computation...

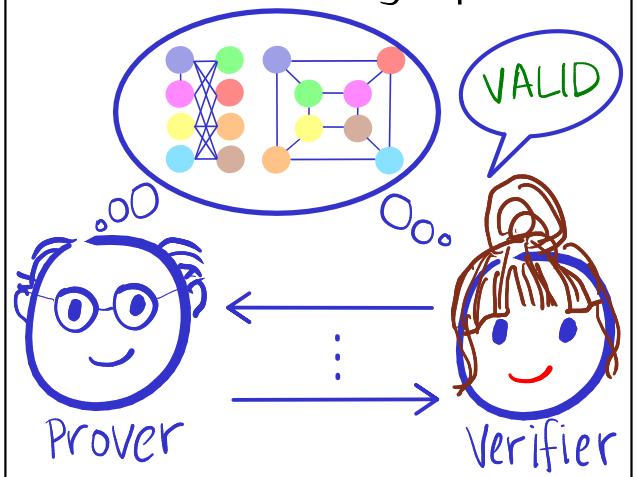
L17-18

Private computation of function

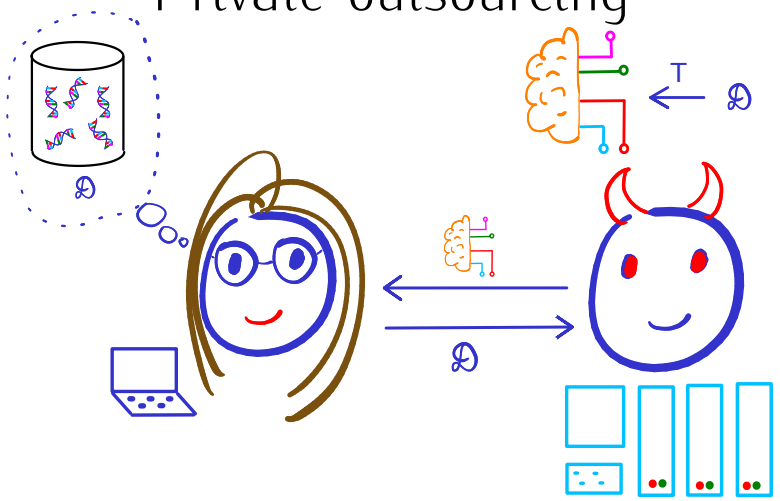


L14-16

Zero-knowledge proofs



Private outsourcing



L19

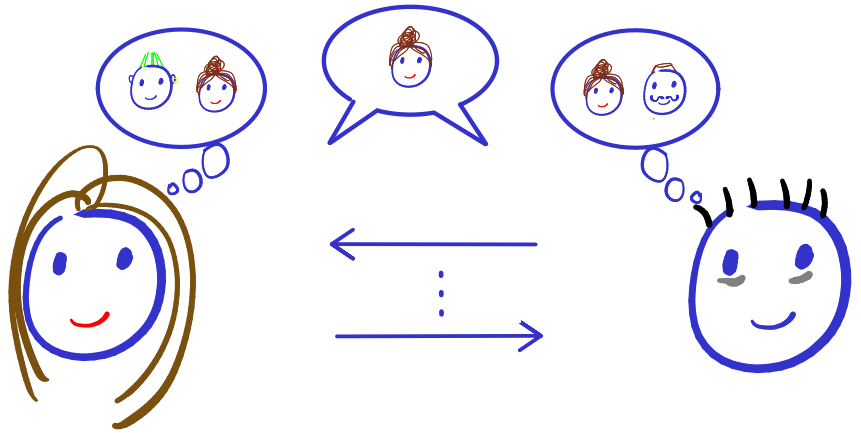
Verifiable outsourcing

L20

This Module: Secure Computation...

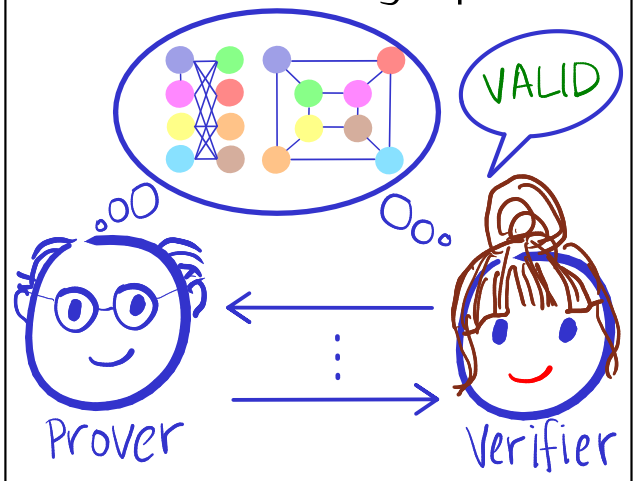
L17-18

Private computation of function

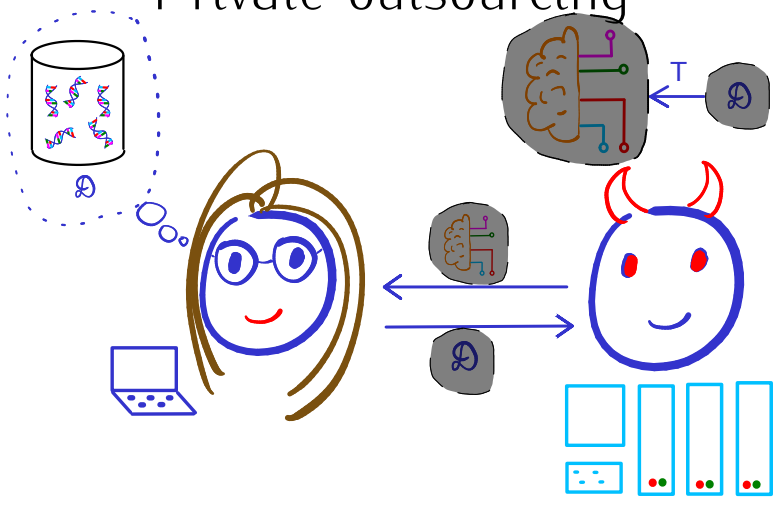


L14-16

Zero-knowledge proofs



Private outsourcing



L19

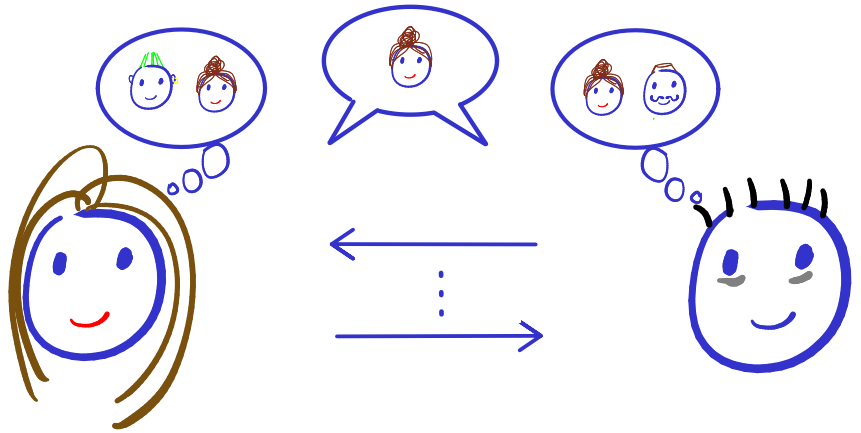
Verifiable outsourcing

L20

This Module: Secure Computation...

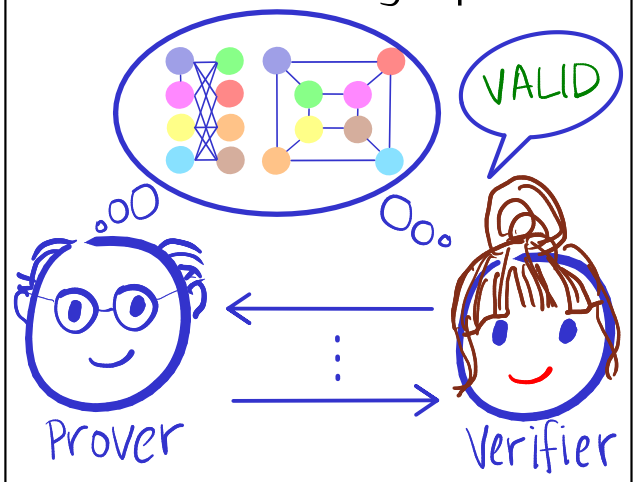
L17-18

Private computation of function

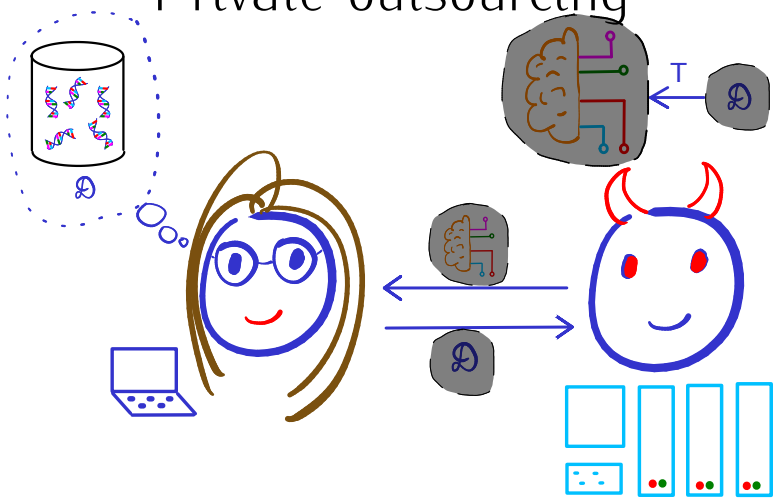


L14-16

Zero-knowledge proofs

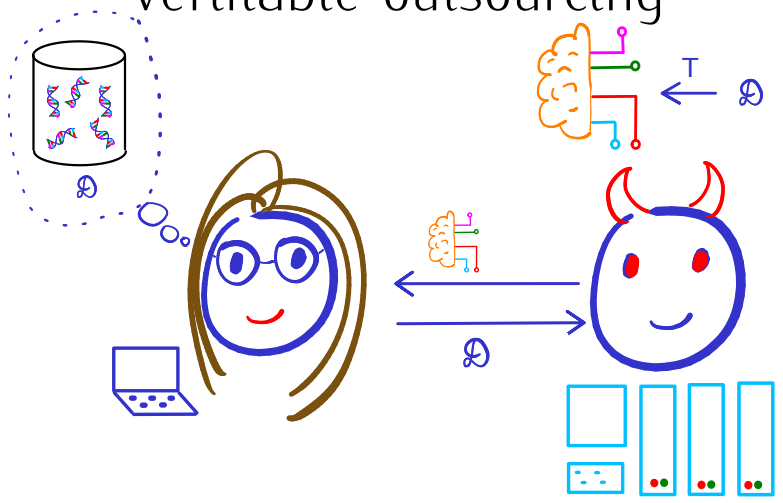


Private outsourcing



L19

Verifiable outsourcing

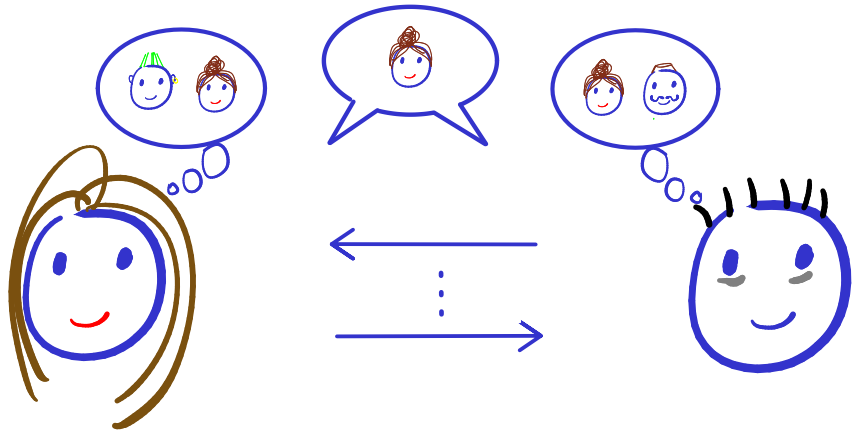


L20

This Module: Secure Computation...

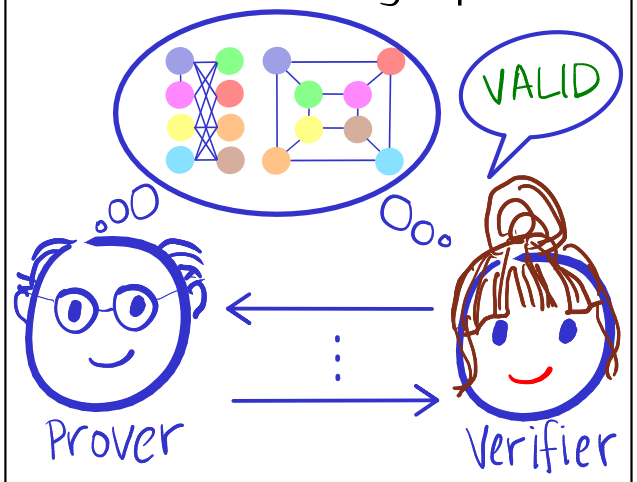
L17-18

Private computation of function

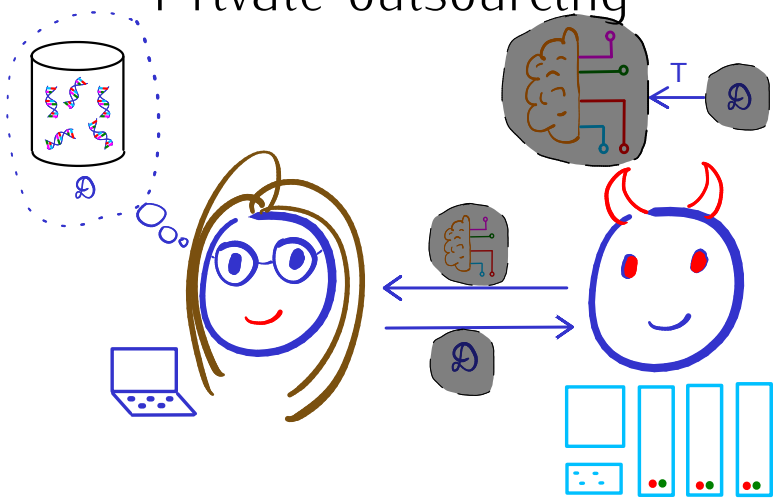


L14-16

Zero-knowledge proofs

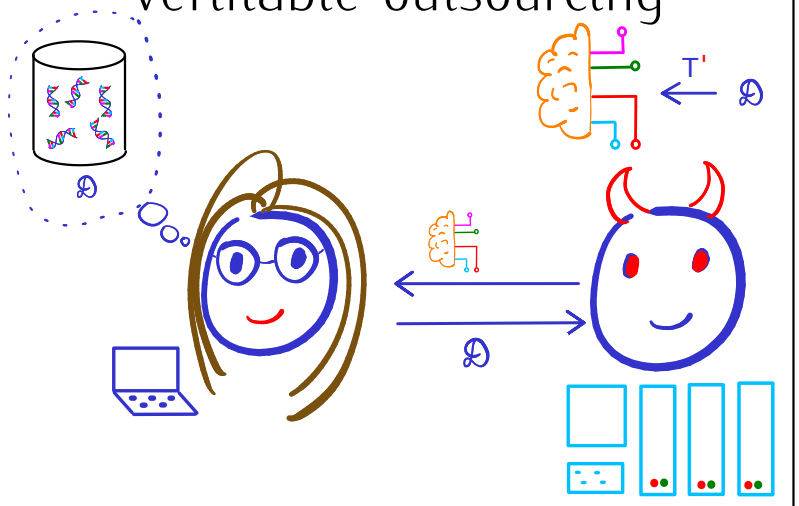


Private outsourcing



L19

Verifiable outsourcing

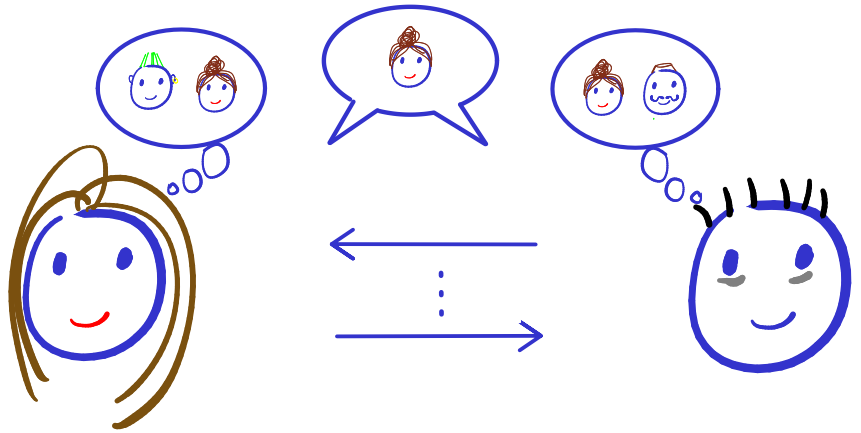


L20

This Module: Secure Computation...

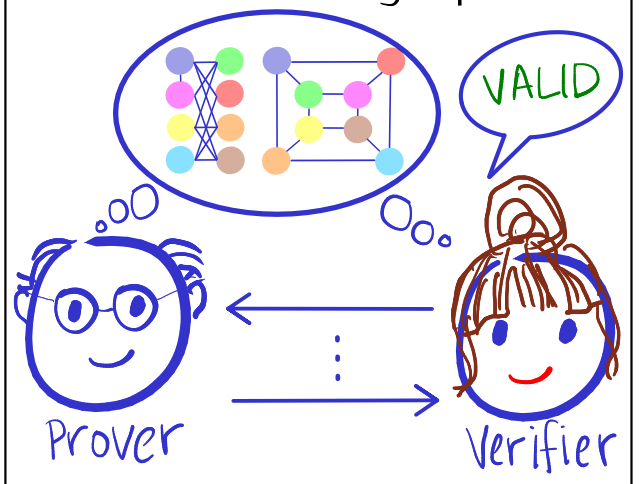
L17-18

Private computation of function

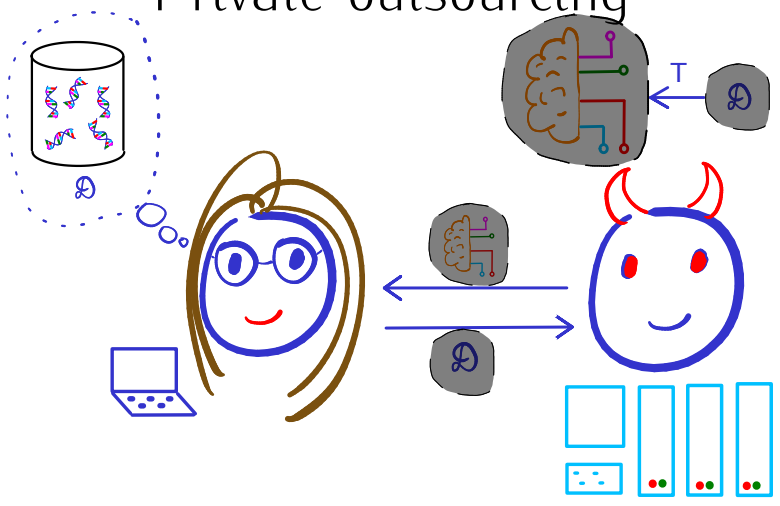


L14-16

Zero-knowledge proofs

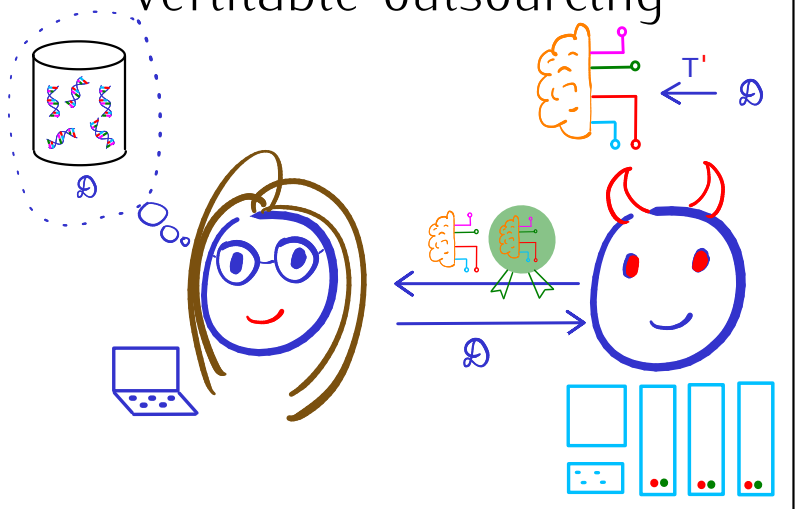


Private outsourcing



L19

Verifiable outsourcing

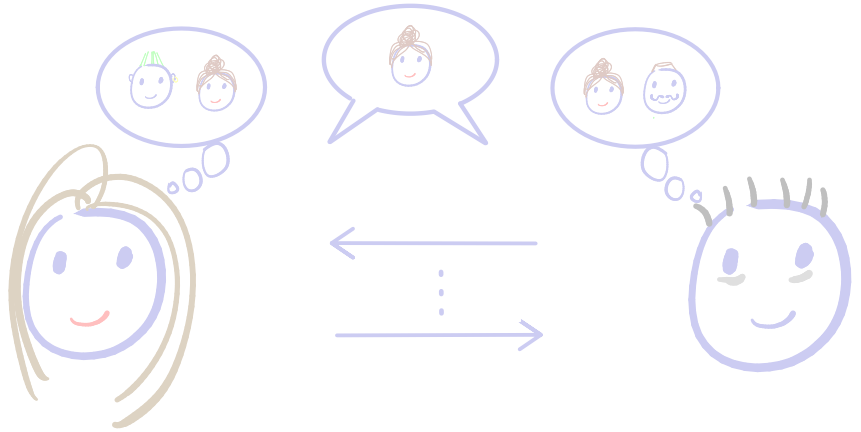


L20

This Module: Secure Computation...

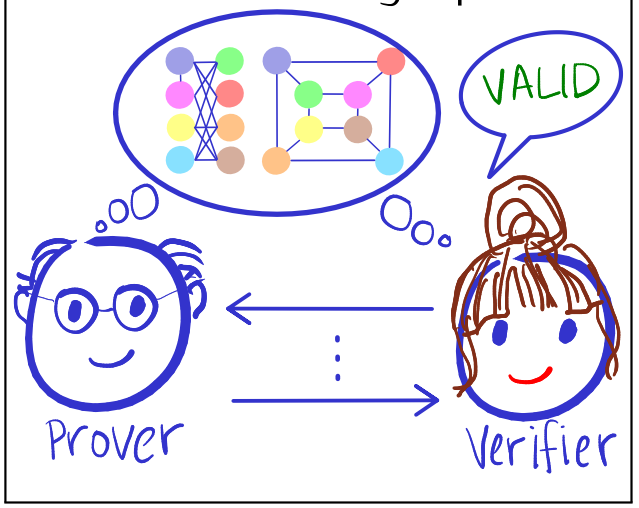
L17-18

Private computation of function

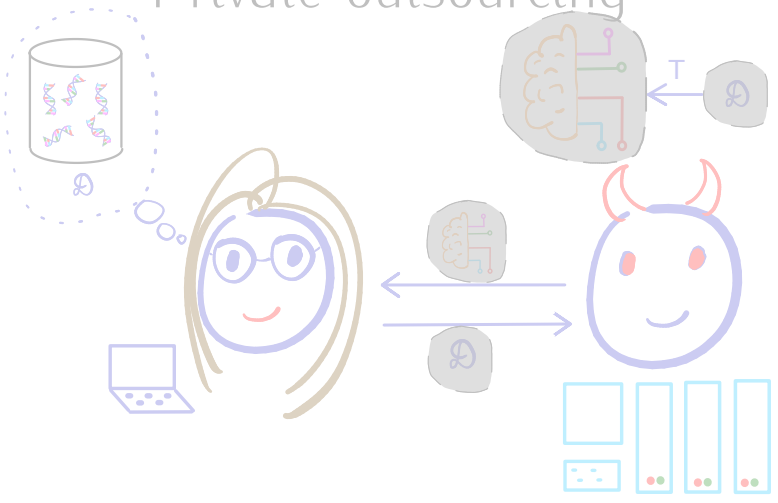


L14-16

Zero-knowledge proofs

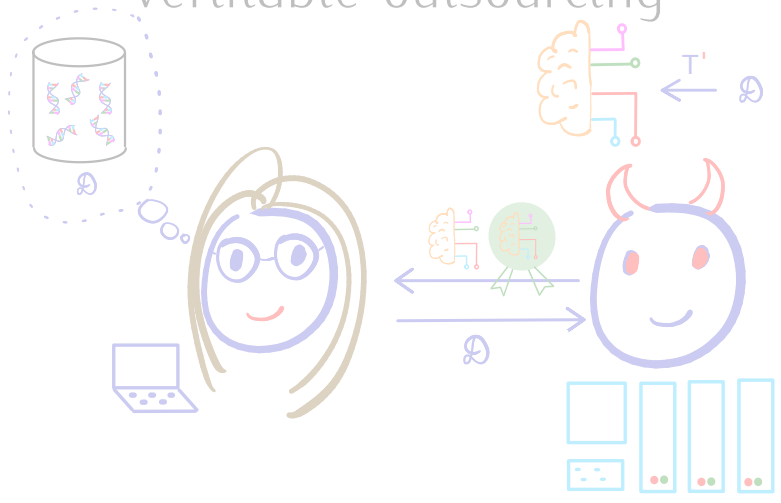


Private outsourcing



L19

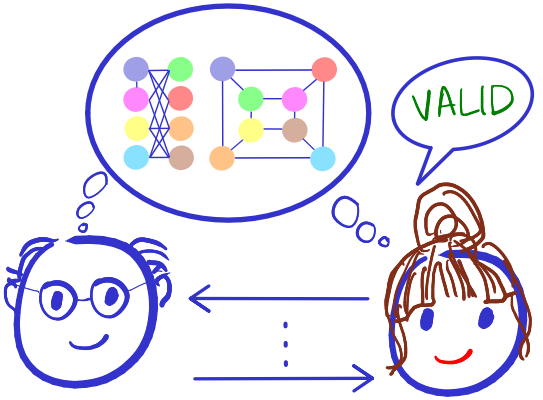
Verifiable outsourcing



L20

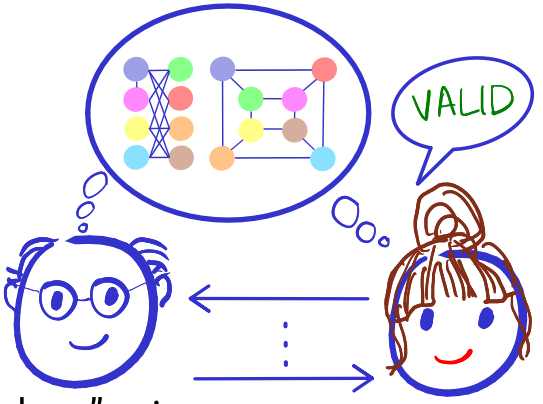
Plan for Today's Lecture

- What constitutes a proof?
  - Traditional "NP" proofs vs *interactive* proofs

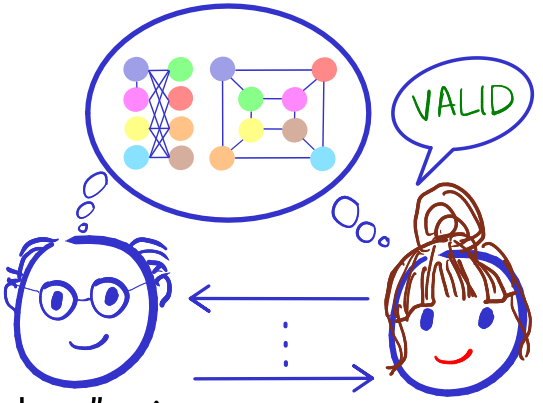


Plan for Today's Lecture

- What constitutes a proof?
  - Traditional "NP" proofs vs *interactive* proofs
- Zero-knowledge proofs: capture "zero knowledge" via simulation paradigm

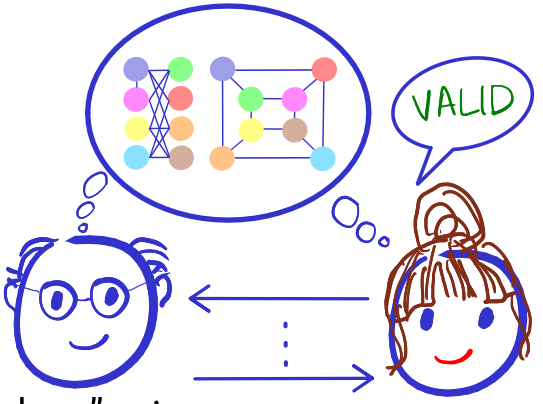


Plan for Today's Lecture

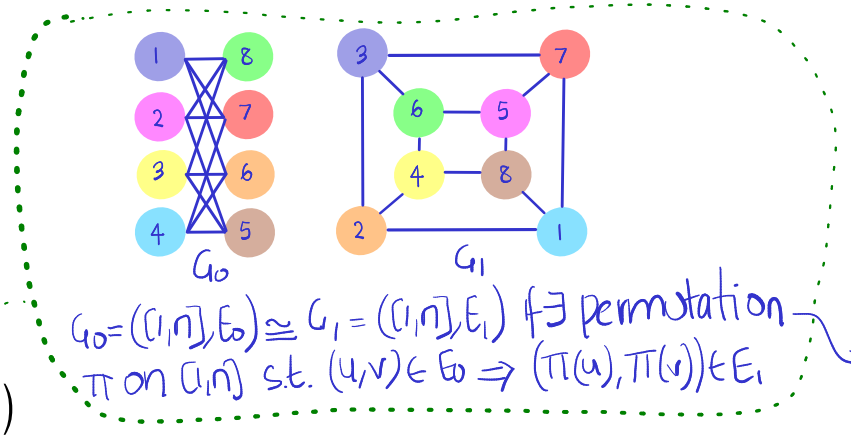


- What constitutes a proof?
  - Traditional "NP" proofs vs *interactive* proofs
- Zero-knowledge proofs: capture "zero knowledge" via simulation paradigm
- Examples. ZKP for:
  - Graph isomorphism (GI)
  - Quadratic residuosity (QR)
  - Graph non-isomorphism (GNI)
  - Quadratic non-residuosity (QNR)

Plan for Today's Lecture

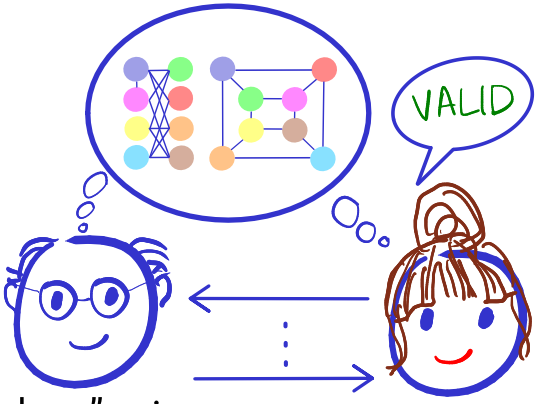


- What constitutes a proof?
  - Traditional "NP" proofs vs *interactive* proofs
- Zero-knowledge proofs: capture "zero knowledge" via simulation paradigm

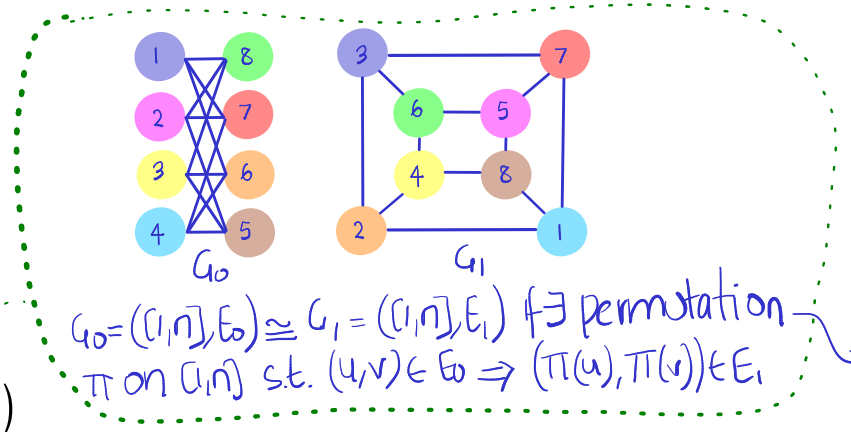


- Examples. ZKP for:
  - Graph isomorphism (GI)
  - Quadratic residuosity (QR)
  - Graph non-isomorphism (GNI)
  - Quadratic non-residuosity (QNR)

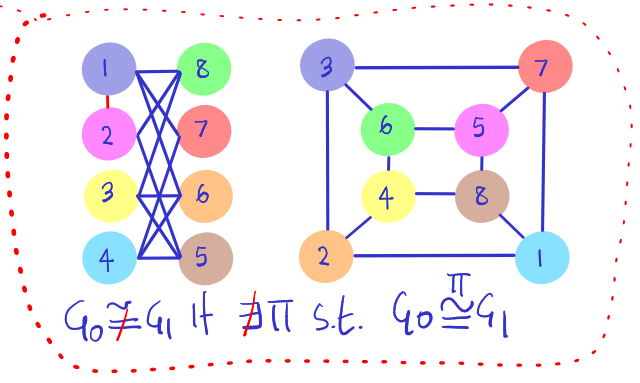
Plan for Today's Lecture



- What constitutes a proof?
  - Traditional "NP" proofs vs *interactive* proofs
- Zero-knowledge proofs: capture "zero knowledge" via simulation paradigm



- Examples. ZKP for:
  - Graph isomorphism (GI)
  - Quadratic residuosity (QR)
  - Graph non-isomorphism (GNI)
  - Quadratic non-residuosity (QNR)



Plan for Today's Lecture

1. Interactive Proof (IP)
2. Zero Knowledge (Interactive) Proof (ZKP)
3. Honest Verifier ZKP for Graph (Non-) Isomorphism

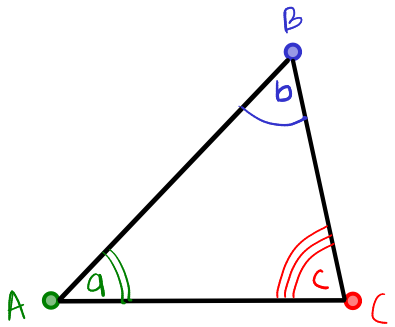
Plan for Today's Lecture

1. Interactive Proof (IP)
2. Zero Knowledge (Interactive) Proof (ZKP)
3. Honest Verifier ZKP for Graph (Non-) Isomorphism

Traditional "NP" Proof...

- Axioms $\xrightarrow{\text{derivation rules}}$ theorems=true statements
  - E.g.: Axioms of Euclidean geometry

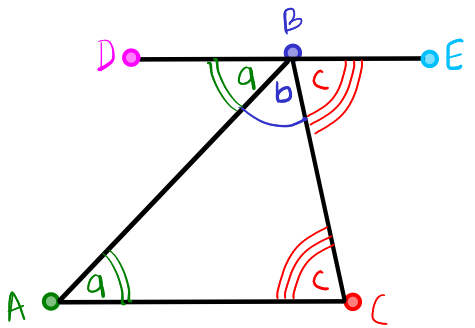
    → Theorem: "Sum of angles of a triangle equals 180°"



Traditional "NP" Proof...

- Axioms $\xrightarrow{\text{derivation rules}}$ theorems=true statements
  - E.g.: Axioms of Euclidean geometry

    ↳ Theorem: "Sum of angles of a triangle equals 180°"

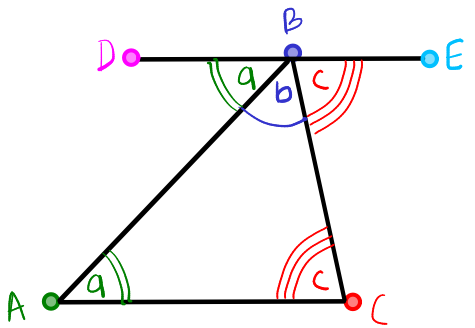


- Prover vs. verifier
  - Prover does the heavy lifting: derives the proof
    1. Construct a line through B parallel to $\overline{AC}$
    2. $\angle DBA = \angle a$ and $\angle EBC = \angle c$ (alternate interior angles)
    3. $2 \Rightarrow \angle a + \angle b + \angle c = \angle DBA + \angle b + \angle EBC = 180°$

Traditional "NP" Proof...

- Axioms $\xrightarrow{\text{derivation rules}}$ theorems=true statements
  - E.g.: Axioms of Euclidean geometry

    → Theorem: "Sum of angles of a triangle equals 180°"



- Prover vs. verifier
  - Prover does the heavy lifting: derives the proof
    1. Construct a line through B parallel to $\overline{AC}$
    2. $\angle DBA = \angle a$ and $\angle EBC = \angle c$ (alternate interior angles)
    3. $2 \Rightarrow \angle a + \angle b + \angle c = \angle DBA + \angle b + \angle EBC = 180°$
  - Verifier checks the proof, step by step

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

$$\forall x \in \mathcal{L}\ \exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$$

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

statement → $\forall x \in \mathcal{L}$ $\exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$ ← witness/proof

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

statement → $\forall x \in \mathcal{L}$ witness/proof ← $\exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$

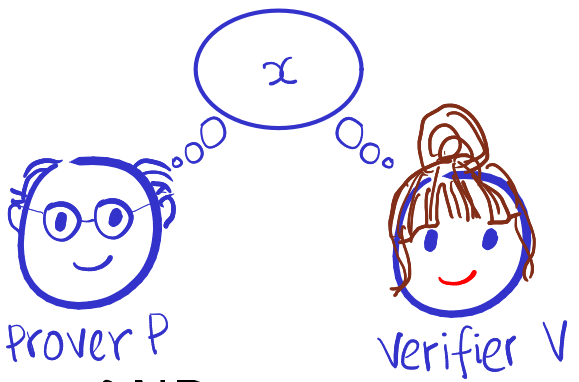
  - NP is the class of all such $\mathcal{L}$s

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

statement → $\forall x \in \mathcal{L}$ $\exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$ ← witness/proof

  - NP is the class of all such $\mathcal{L}$s



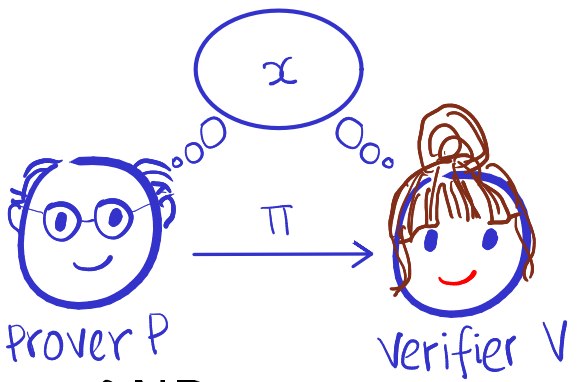
- "Proof system" view of NP
  - Prover P is *unbounded*: finds short proof π for x (if one exists)
  - Verifier V is *efficient*: checks proof π against the statement x

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

statement → $\forall x \in \mathcal{L}$ $\exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$ ← witness/proof

  - NP is the class of all such $\mathcal{L}$s



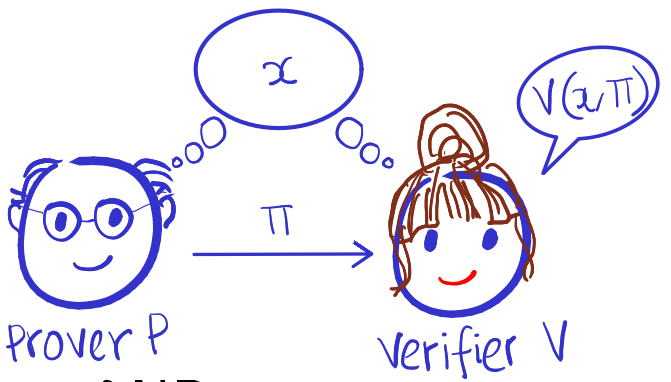
- "Proof system" view of NP
  - Prover P is *unbounded*: finds short proof π for x (if one exists)
  - Verifier V is *efficient*: checks proof π against the statement x

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

statement → $\forall x \in \mathcal{L}$ $\exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$ ← witness/proof

  - NP is the class of all such $\mathcal{L}$s



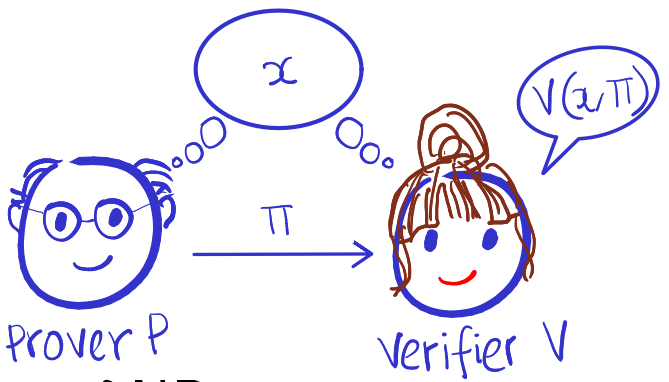
- "Proof system" view of NP
  - Prover P is *unbounded*: finds short proof π for x (if one exists)
  - Verifier V is *efficient*: checks proof π against the statement x

Traditional "NP" Proof...

- Corresponds to class NP
  - A language $\mathcal{L} \in \mathsf{NP}$ if there exists a polynomial-time *deterministic* machine V such that

statement → $\forall x \in \mathcal{L}\ \exists \pi \in \{0,1\}^{\mathsf{poly}(|x|)} : \mathsf{V}(x, \pi) = 1$ ← witness/proof

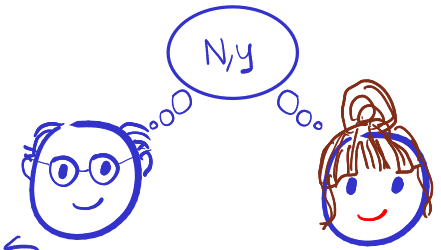
  - NP is the class of all such $\mathcal{L}$s



- "Proof system" view of NP
  - Prover P is *unbounded*: finds short proof π for x (if one exists)
  - Verifier V is *efficient*: checks proof π against the statement x
  - Completeness: $x \in \mathcal{L} \Rightarrow \mathsf{P}$ finds $\pi \Rightarrow \mathsf{V}(x, \pi) = 1$
  - Soundness: $x \notin \mathcal{L} \Rightarrow \nexists \pi \in \{0,1\}^{\mathsf{poly}(|x|)}$ s.t. $\mathsf{V}(x, \pi) = 1$

Which Languages have "NP" Proofs?

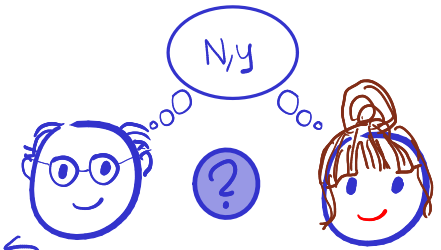
Quadratic residuosity (QR)



$$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$$

Which Languages have "NP" Proofs?

Quadratic residuosity (QR)

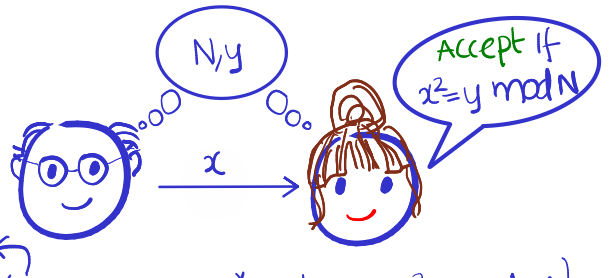


PQ

$$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Which Languages have "NP" Proofs?

Quadratic residuosity (QR)

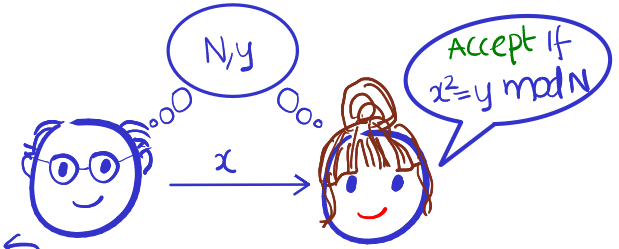


pq

$$\mathcal{L}_{QR} = \{(N, y) : \exists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$$

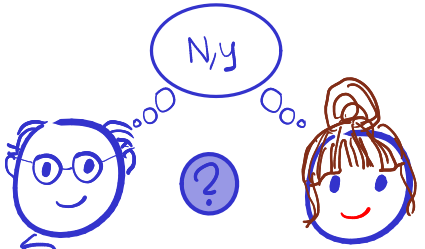
Which Languages have "NP" Proofs?

Quadratic residuosity (QR)



$$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

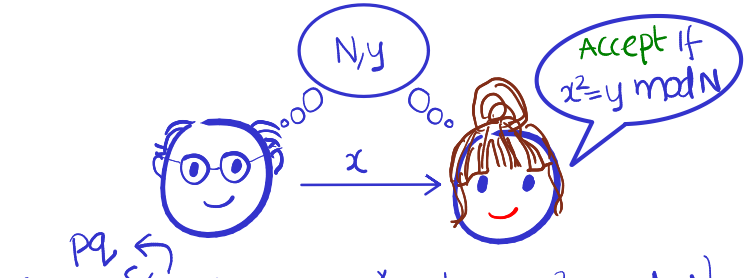
Quad. non-residuosity (QNR)



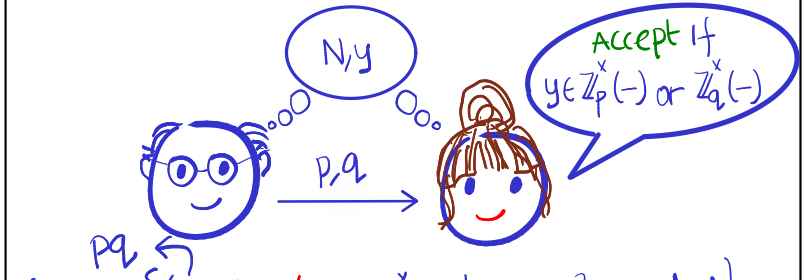
$$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Which Languages have "NP" Proofs?

Quadratic residuosity (QR)



Quad. non-residuosity (QNR)



Which Languages have "NP" Proofs?

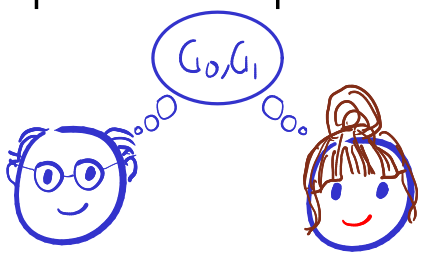
Quadratic residuosity (QR)

$$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Quad. non-residuosity (QNR)

$$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Graph isomorphism (GI)



$$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$$

Which Languages have "NP" Proofs?

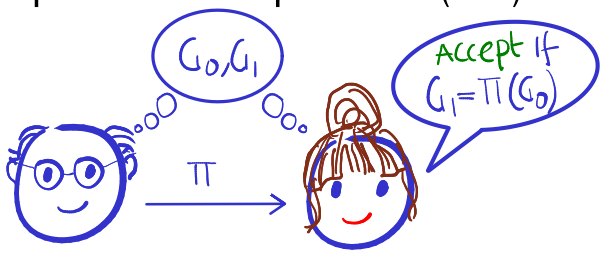
Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

Quad. non-residuosity (QNR)

$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

Graph isomorphism (GI)



$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

Which Languages have "NP" Proofs?

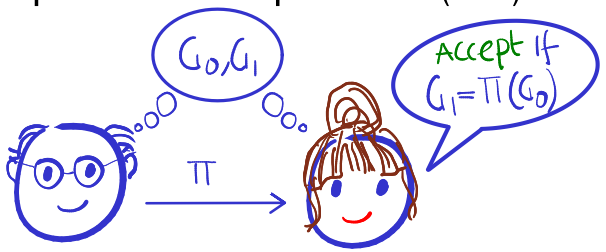
Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

Quad. non-residuosity (QNR)

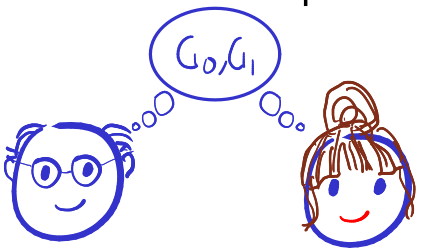
$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

Graph isomorphism (GI)



$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

Graph non-isomorphism (GNI)



$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

Which Languages have "NP" Proofs?

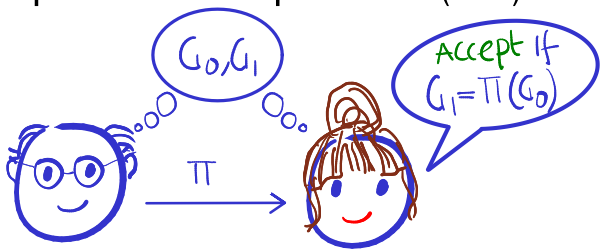
Quadratic residuosity (QR)

$$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Quad. non-residuosity (QNR)

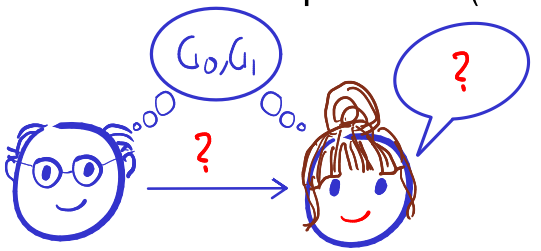
$$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Graph isomorphism (GI)



$$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$$

Graph non-isomorphism (GNI)



$$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$$

Which Languages have "NP" Proofs?

Quadratic residuosity (QR)

$$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Quad. non-residuosity (QNR)

$$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

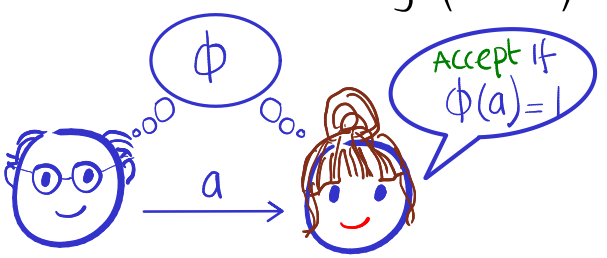
Graph isomorphism (GI)

$$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$$

Graph non-isomorphism (GNI)

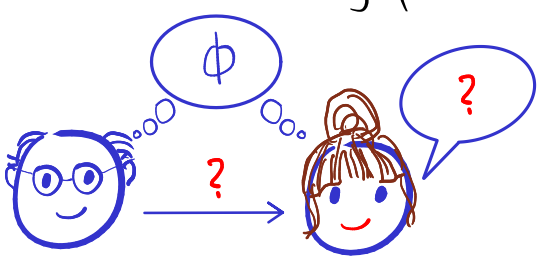
$$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$$

Boolean satisfiability (SAT)



$$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$$

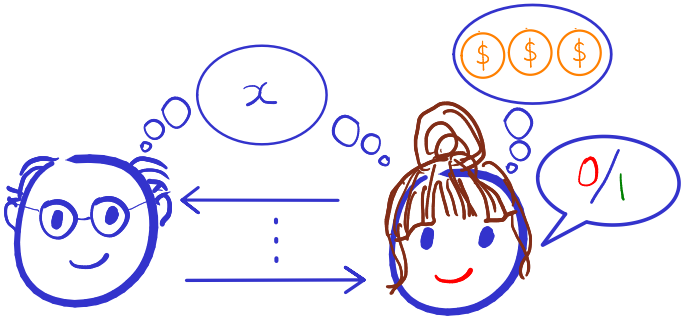
Bool. unsatisfiability (UNSAT)



$$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$$

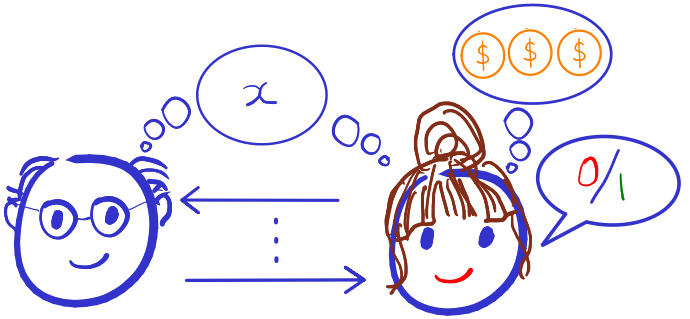
Interactive Proof (IP)

- Difference from NP proofs:
  1. Verifier V is *randomised*
  2. Prover P and V *interact* and V accepts/rejects in the end



Interactive Proof (IP)

- Difference from NP proofs:
  - 1 Verifier V is *randomised*
  - 2 Prover P and V *interact* and V accepts/rejects in the end

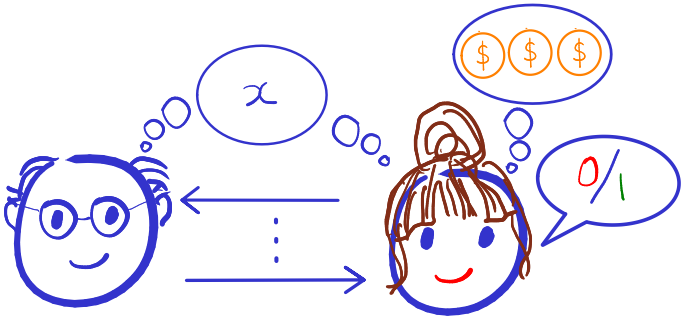


Defintion 1

An interactive protocol (P, V) *for a language* $\mathcal{L}$ *is an interactive proof (IP) system if the following holds:*

Interactive Proof (IP)

- Difference from NP proofs:
  - 1 Verifier V is *randomised*
  - 2 Prover P and V *interact* and V accepts/rejects in the end



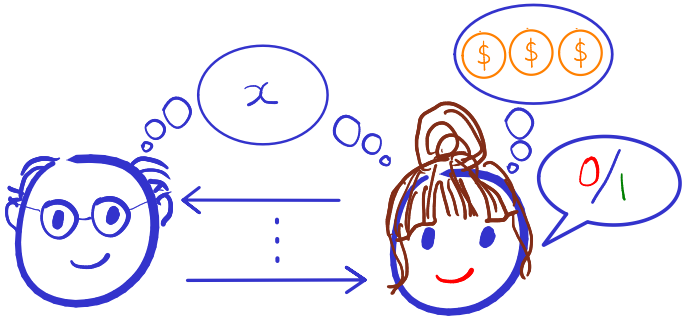
Defintion 1

An interactive protocol (P, V) *for a language* $\mathcal{L}$ *is an interactive proof (IP) system if the following holds:*

- *Completeness: for every* $x \in \mathcal{L}$, $\Pr[1 \leftarrow \langle \mathsf{P}, \mathsf{V} \rangle(x)] \geq 1 - 1/3$

Interactive Proof (IP)

- Difference from NP proofs:
  1. Verifier V is *randomised*
  2. Prover P and V *interact* and V accepts/rejects in the end



Defintion 1

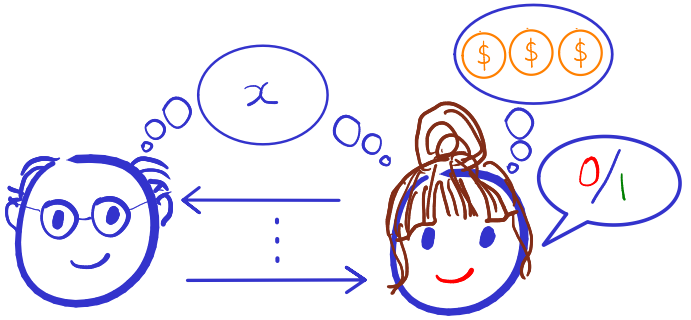
An interactive protocol (P, V) *for a language* $\mathcal{L}$ *is an interactive proof (IP) system if the following holds:*

- *Completeness: for every* $x \in \mathcal{L}$, $\Pr[1 \leftarrow \langle \mathsf{P}, \mathsf{V} \rangle(x)] \geq 1 - 1/3$
- *Soundness: for every* $x \notin \mathcal{L}$ *and malicious prover* P^*,

$$\Pr[1 \leftarrow \langle \mathsf{P}^*, \mathsf{V} \rangle(x)] \leq 1/3$$

Interactive Proof (IP)

- Difference from NP proofs:
  - 1 Verifier V is *randomised*
  - 2 Prover P and V *interact* and V accepts/rejects in the end



Defintion 1

An interactive protocol (P, V) *for a language* $\mathcal{L}$ *is an interactive proof (IP) system if the following holds:* completeness error $\varepsilon_c(n)$

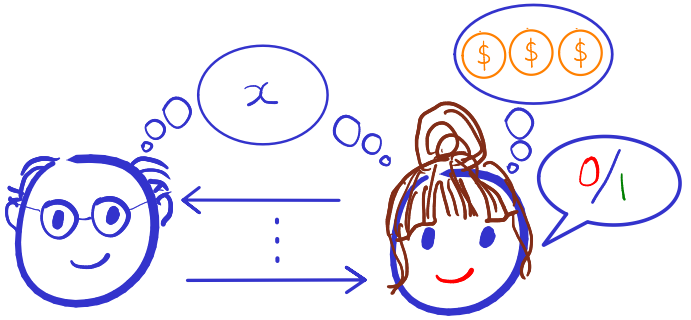
- *Completeness: for every* $x \in \mathcal{L}$, $\Pr[1 \leftarrow \langle \mathsf{P}, \mathsf{V} \rangle(x)] \geq 1 - 1/3$
- *Soundness: for every* $x \notin \mathcal{L}$ *and malicious prover* P^*,

$$\Pr[1 \leftarrow \langle \mathsf{P}^*, \mathsf{V} \rangle(x)] \leq 1/3$$

soundness error $\varepsilon_s(n)$

Interactive Proof (IP)

- Difference from NP proofs:
  1. Verifier V is *randomised*
  2. Prover P and V *interact* and V accepts/rejects in the end



Defintion 1

An interactive protocol (P, V) *for a language* $\mathcal{L}$ *is an interactive proof (IP) system if the following holds:* completeness error $\varepsilon_c(n)$

- *Completeness: for every* $x \in \mathcal{L}$, $\Pr[1 \leftarrow \langle \mathsf{P}, \mathsf{V} \rangle(x)] \geq 1 - 1/3$
- *Soundness: for every* $x \notin \mathcal{L}$ *and malicious prover* P^*,

$$\Pr[1 \leftarrow \langle \mathsf{P}^*, \mathsf{V} \rangle(x)] \leq 1/3$$

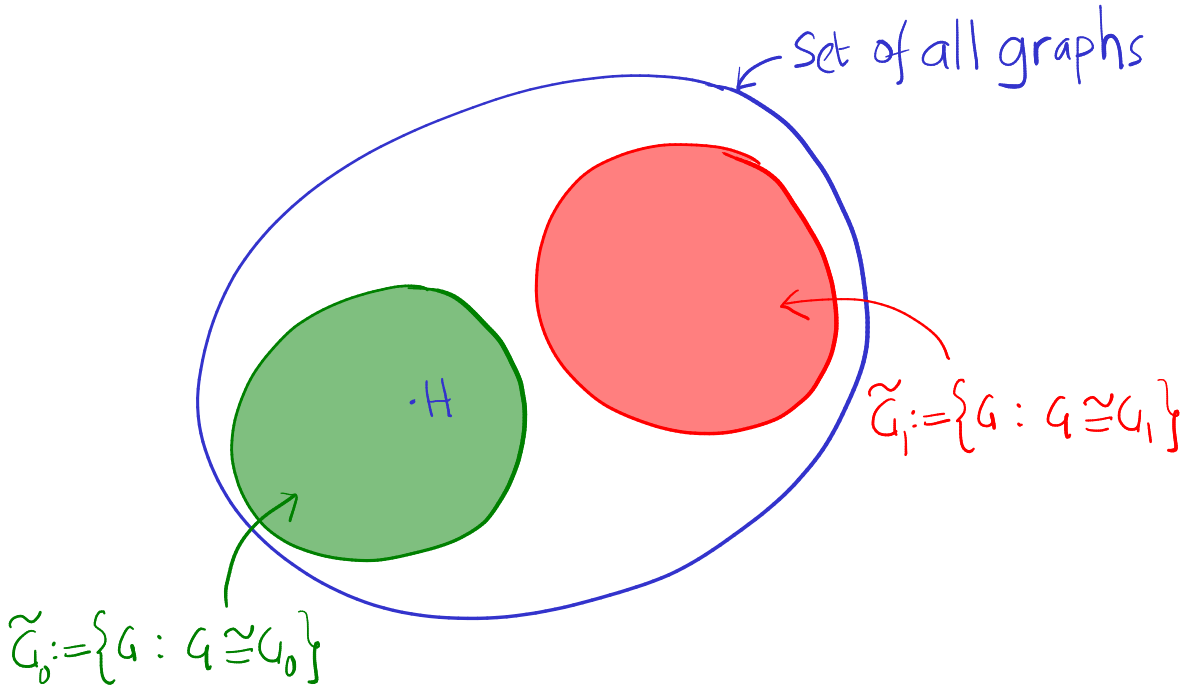
soundness error $\varepsilon_s(n)$

Exercise 1 (Robustness of Defintion 1)

Show that languages captured by Defintion 1 doesn't change when 1) $\epsilon_c \leq 1/2^{|x|}, \epsilon_s \leq 1/2^{|x|}$*; 2)* $\epsilon_c \leq 1/2 - 1/|x|, \epsilon_s \leq 1/2 - 1/|x|$

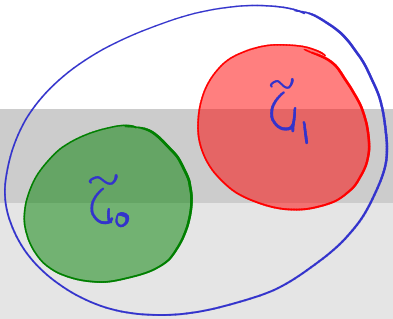
Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

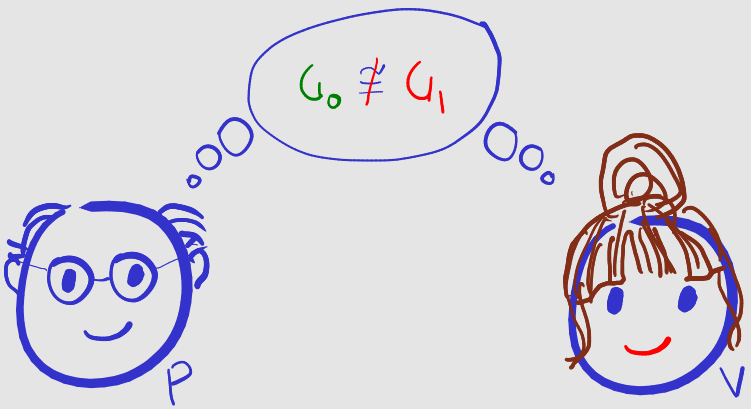


Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

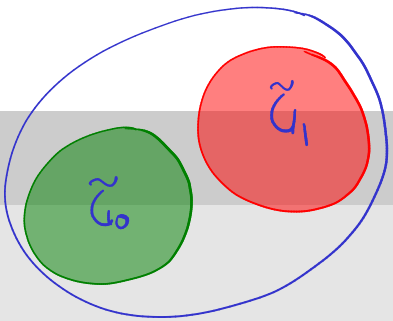


Protocol 1 (Π_{GNI}: IP for GNI)

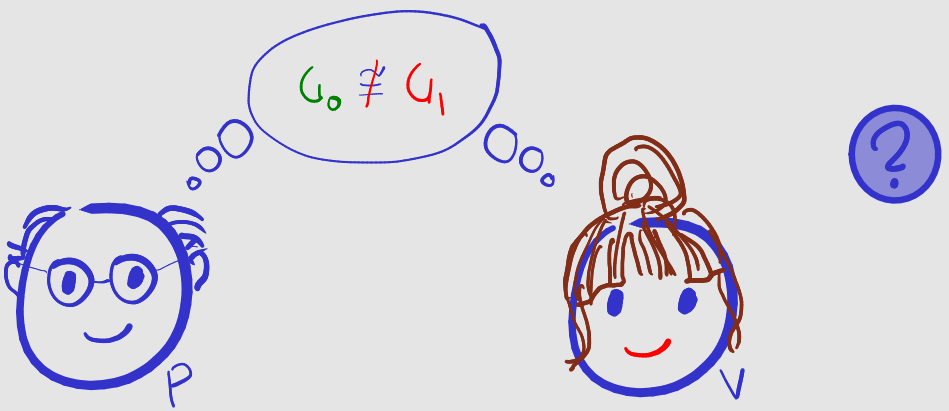


Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \ncong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*



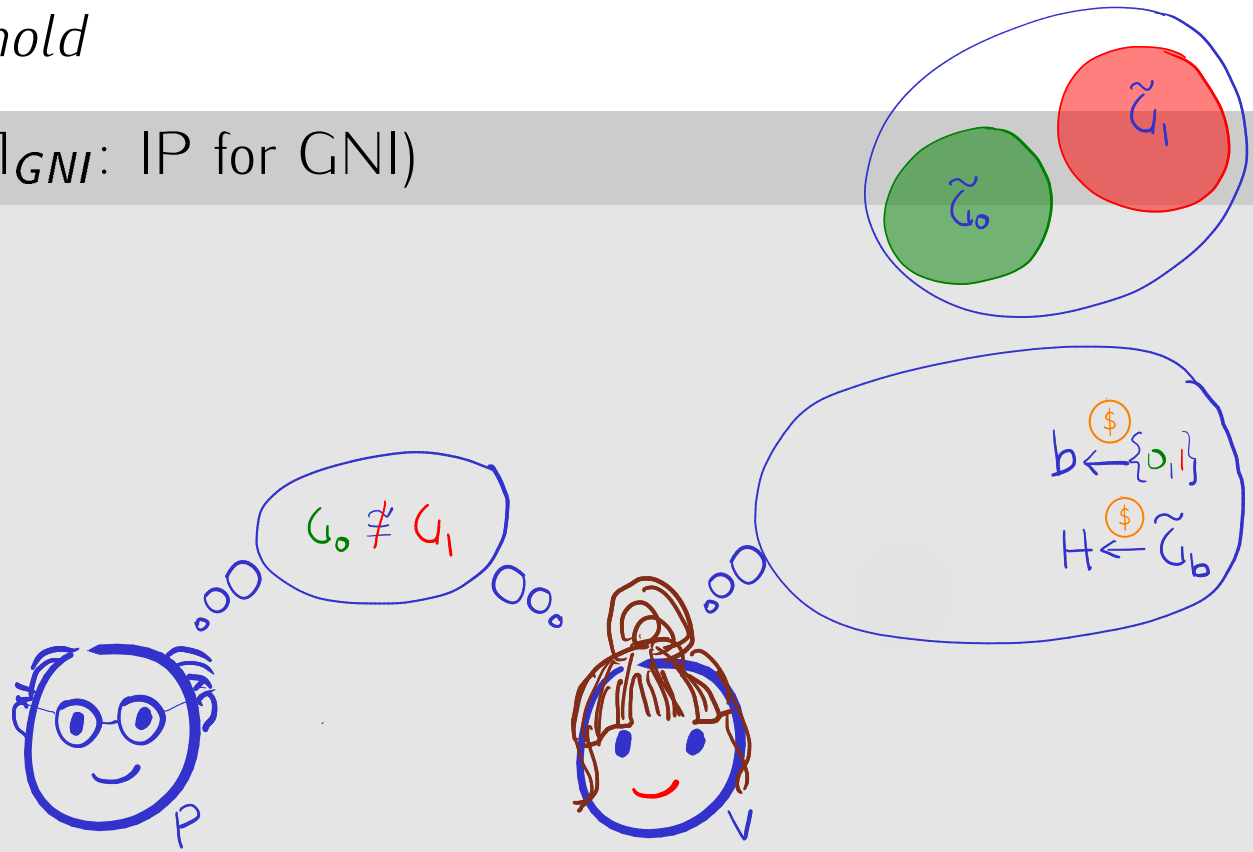
Protocol 1 (Π_{GNI}: IP for GNI)



Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

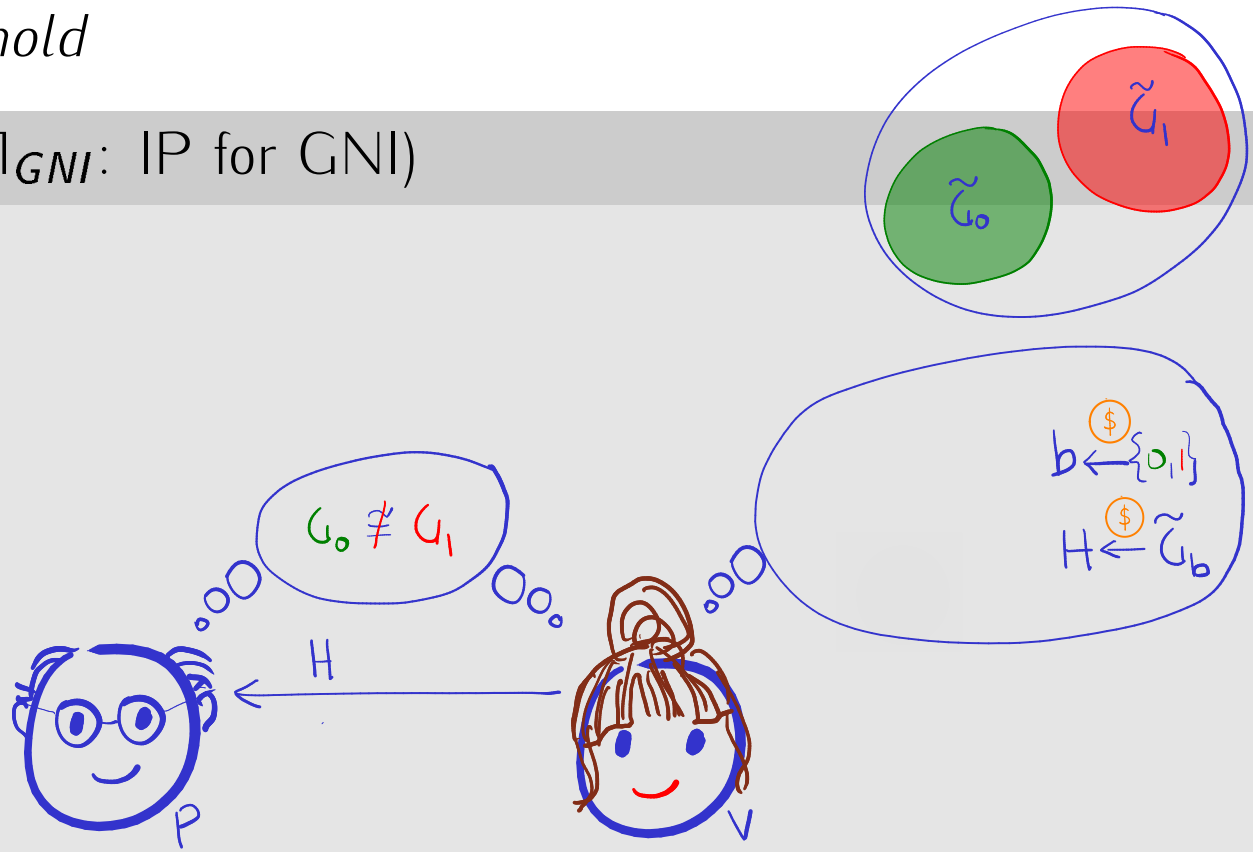
Protocol 1 (Π_{GNI}: IP for GNI)



Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

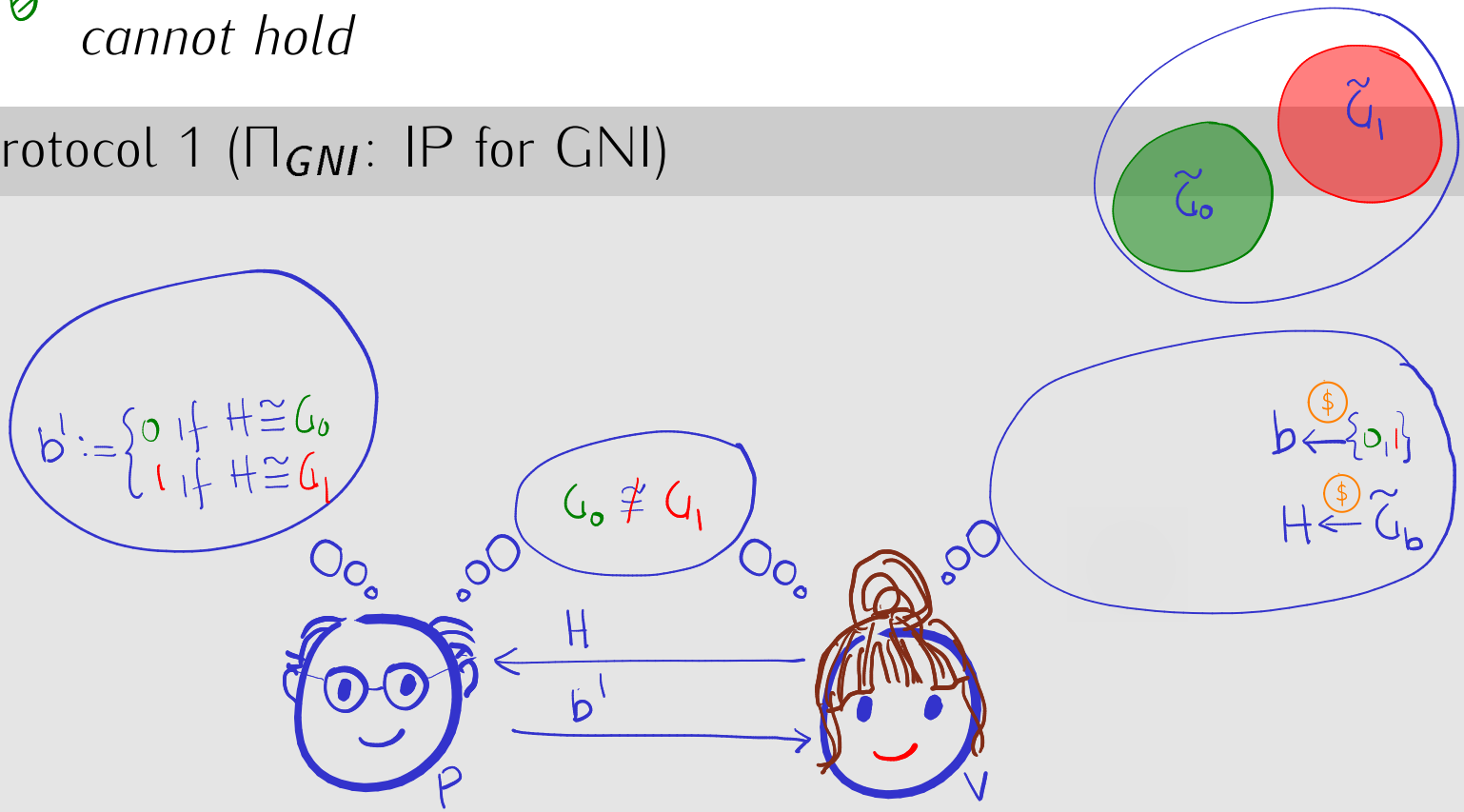
Protocol 1 (Π_{GNI}: IP for GNI)



Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

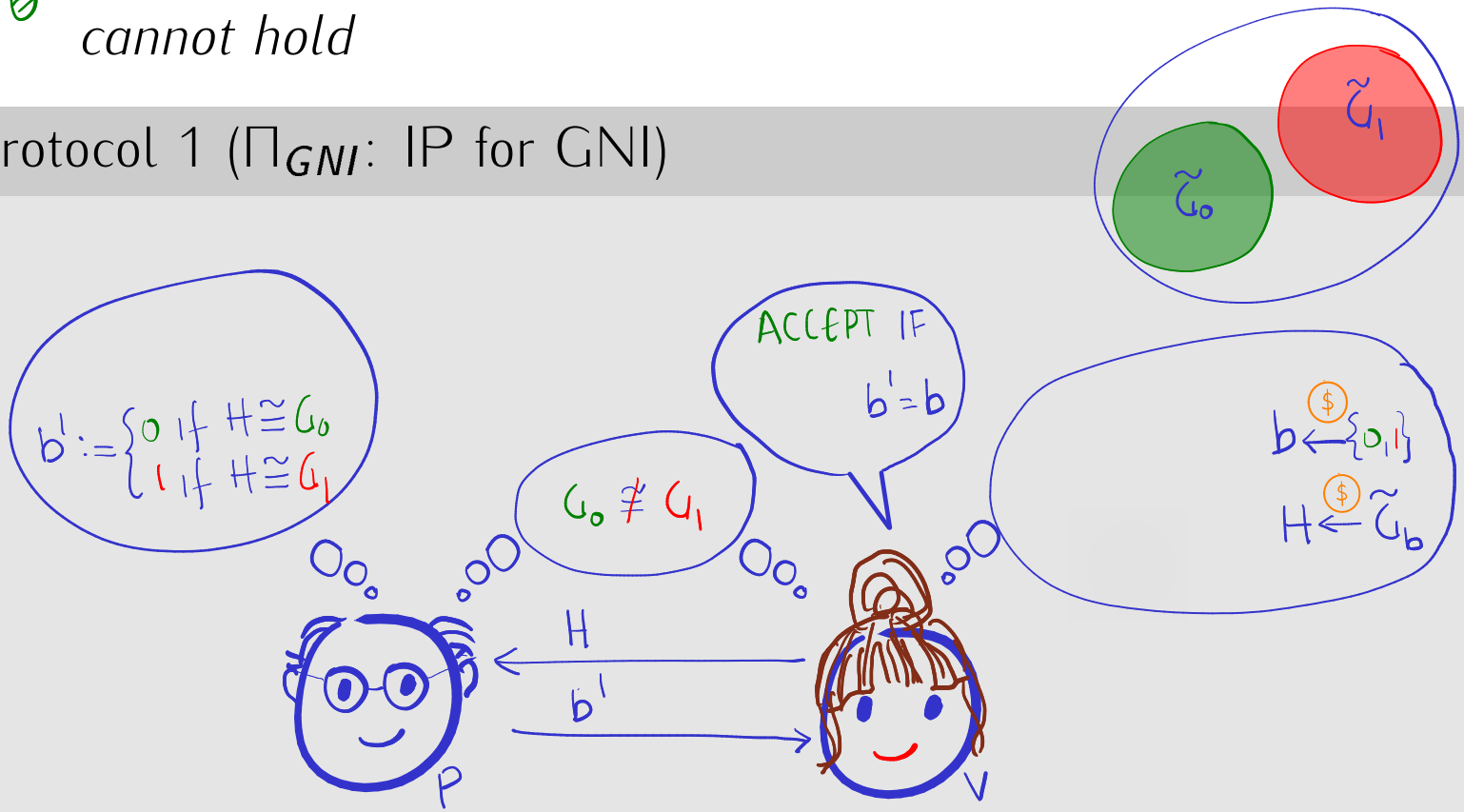
Protocol 1 (Π_{GNI}: IP for GNI)



Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

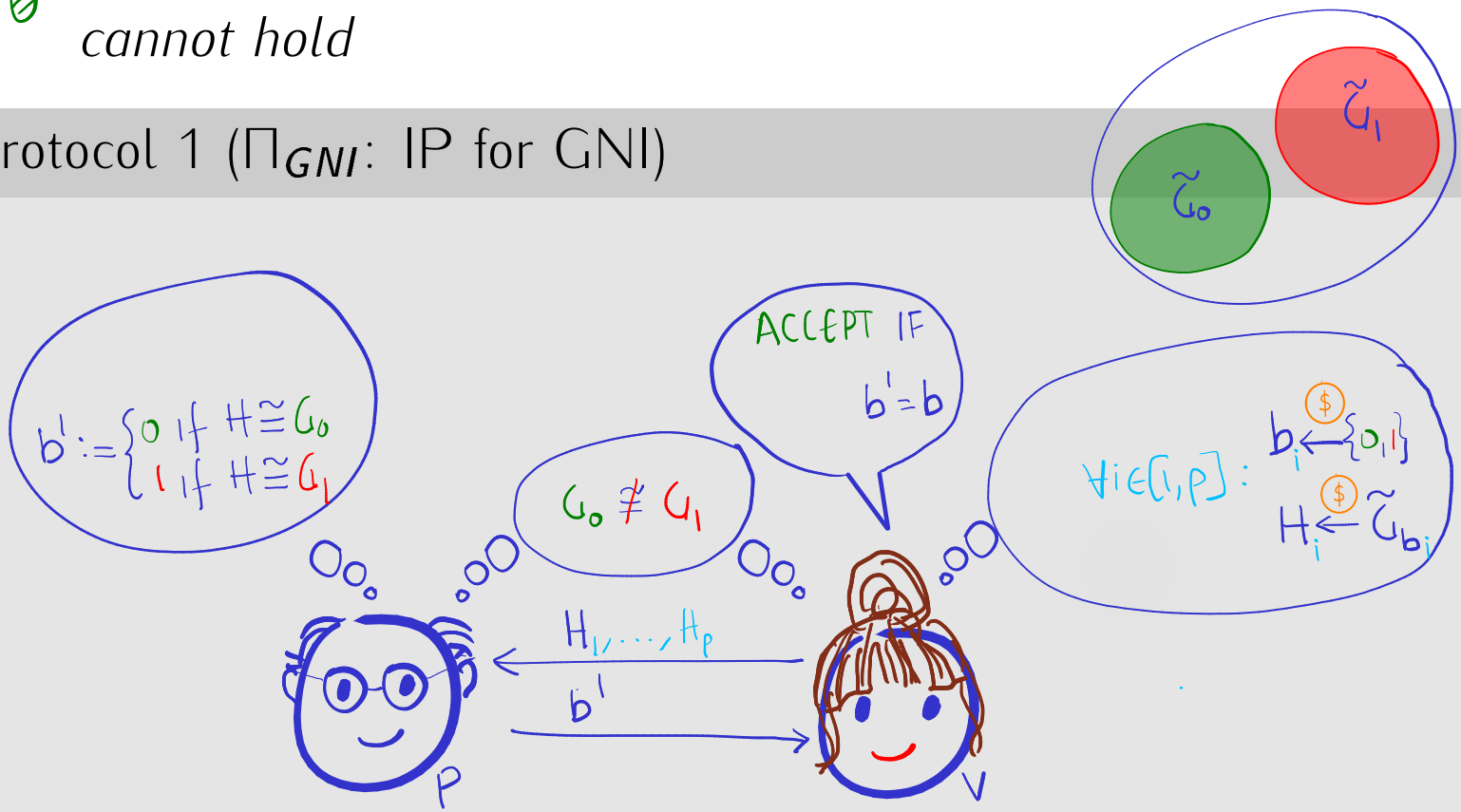
Protocol 1 (Π_{GNI}: IP for GNI)



Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

Protocol 1 (Π_{GNI}: IP for GNI)

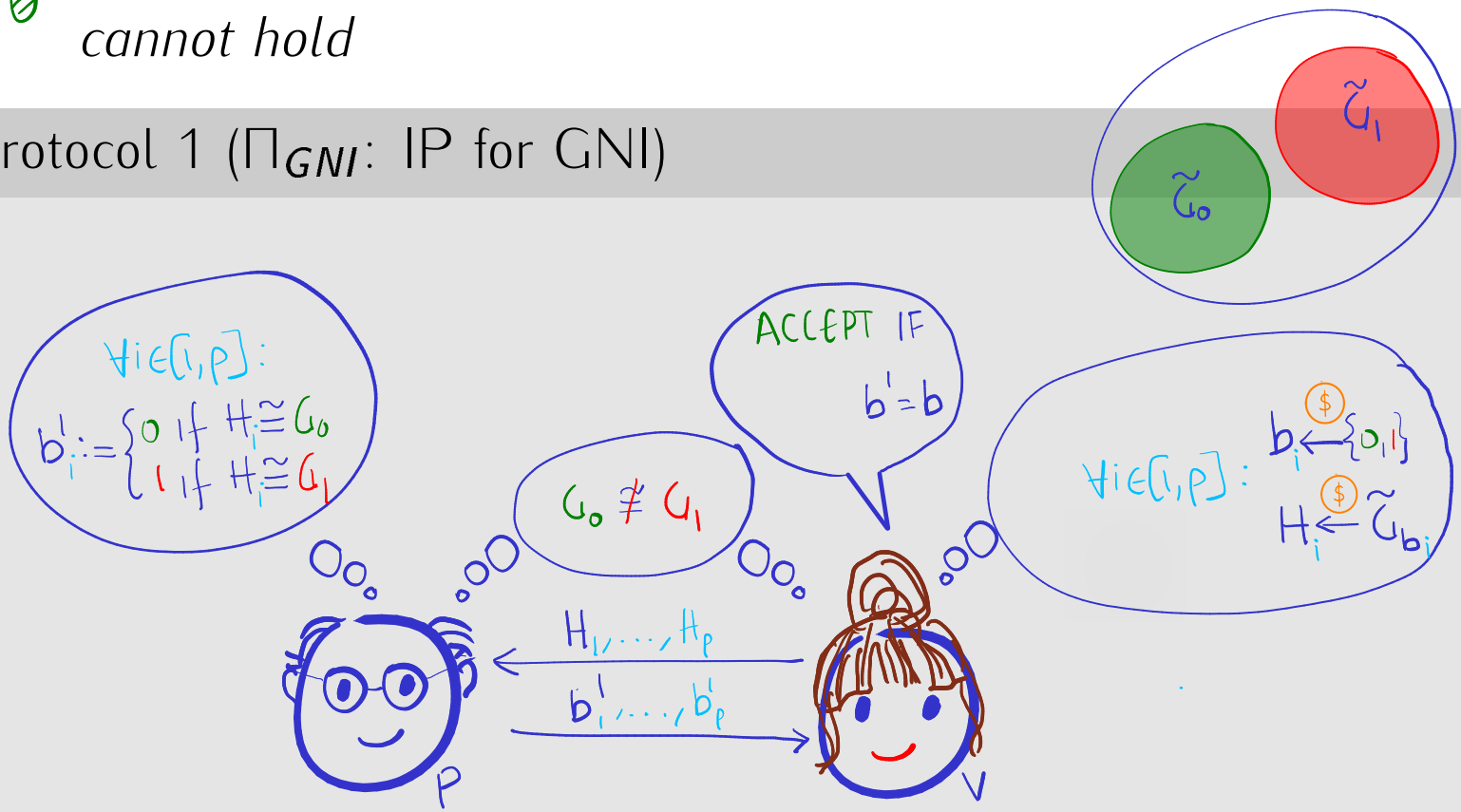


Parallel/sequentially repeat to boost soundness

Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

Protocol 1 (Π_{GNI}: IP for GNI)

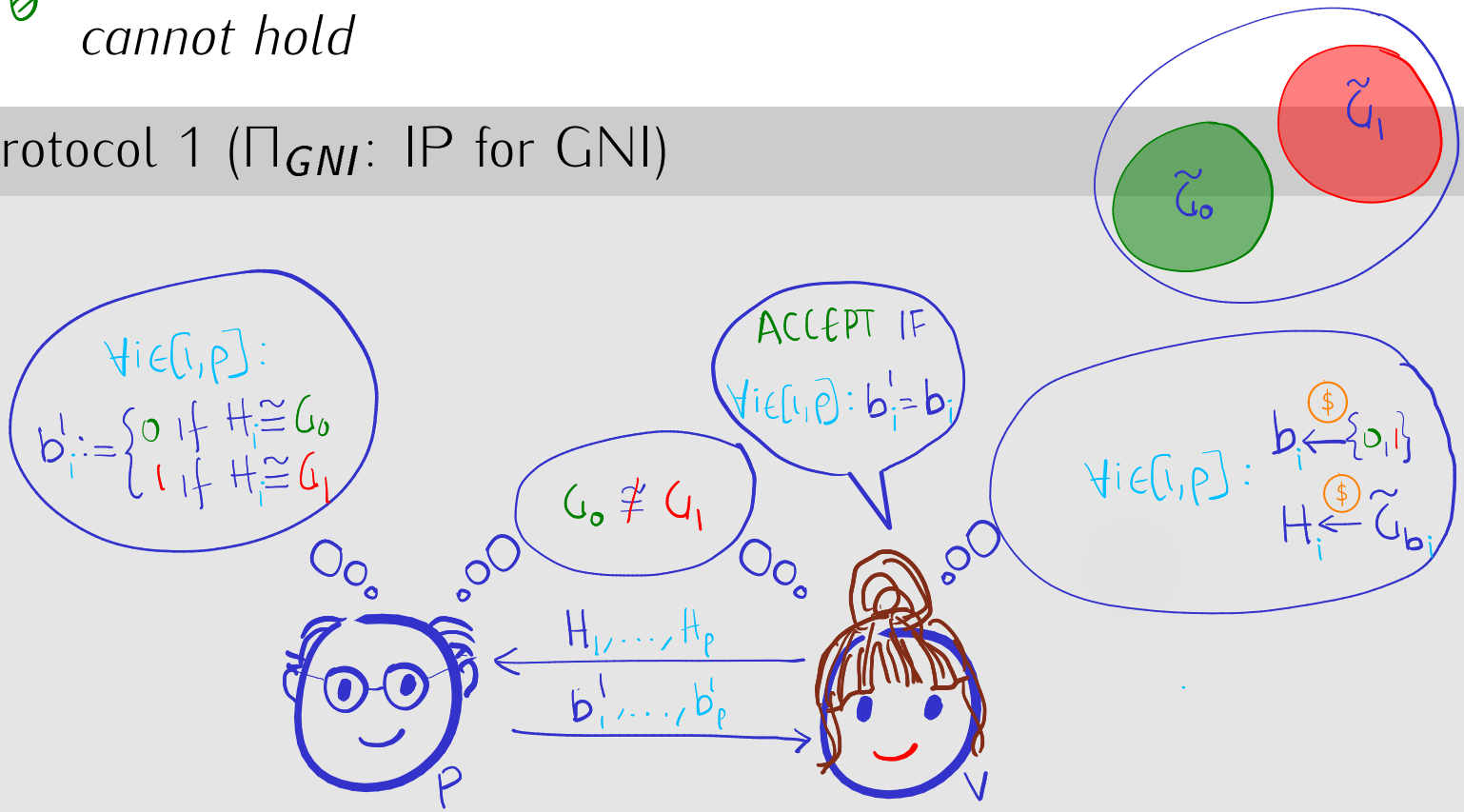


Parallel/sequentially repeat to boost soundness

Power of Randomness+Interaction: IP for GNI...

Idea: $G_0 \not\cong G_1 \Rightarrow$ for any graph H, $G_0 \cong H$ *and* $G_1 \cong H$ *both cannot hold*

Protocol 1 (Π_{GNI}: IP for GNI)



Parallel/sequentially repeat to boost soundness

Power of Randomness+Interaction: IP for GNI...

Theorem 1

Π_{GNI} *is an IP for* $\mathcal{L}_{GNI}$

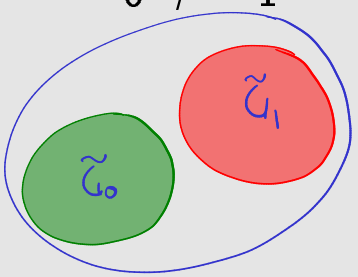
Power of Randomness+Interaction: IP for GNI...

Theorem 1

Π_{GNI} *is an IP for* $\mathcal{L}_{GNI}$

Proof.

- Completeness:
  - $G_0 \not\cong G_1 \Rightarrow \mathsf{P}$ can recover b_i from H_i with certainty



$$\Pr[1 \leftarrow \langle \mathsf{P}, \mathsf{V} \rangle (G_0, G_1)] = 1 \geq 2/3$$

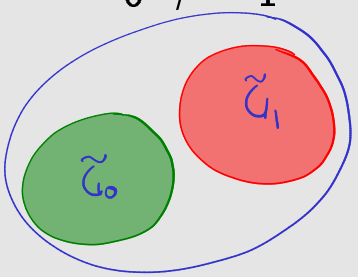
Power of Randomness+Interaction: IP for GNI...

Theorem 1

Π_{GNI} *is an IP for* $\mathcal{L}_{GNI}$

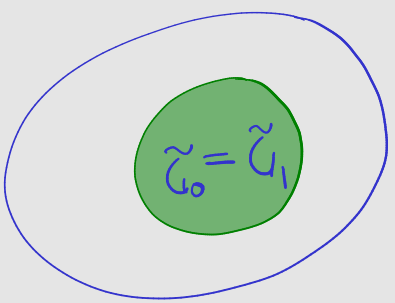
Proof.

- Completeness:
  - $G_0 \not\cong G_1 \Rightarrow \mathsf{P}$ can recover b_i from H_i with certainty



$$\Pr[1 \leftarrow \langle \mathsf{P}, \mathsf{V} \rangle (G_0, G_1)] = 1 \geq 2/3$$

- Soundness:
  - $G_0 \cong G_1 \Rightarrow H_i$ loses information about bits b_i
  - Hence best P^* can do is guess b_is

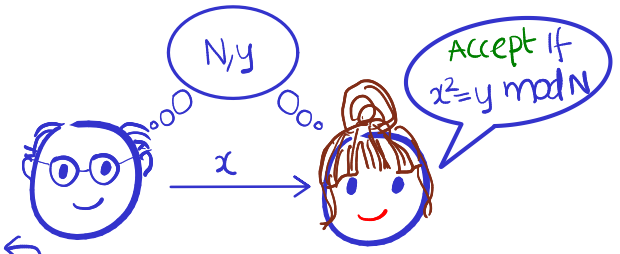


$$\Pr[1 \leftarrow \langle \mathsf{P}^*, \mathsf{V} \rangle (G_0, G_1)] = 1/2^{\rho} < 1/3$$

□

Which Languages have IPs?

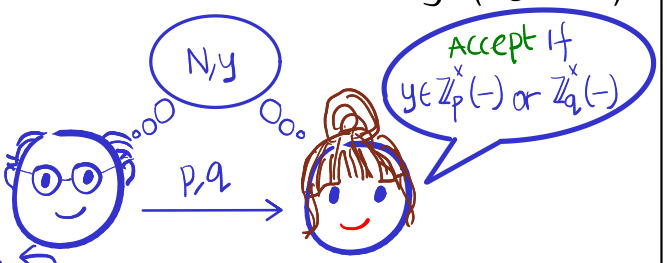
Quadratic residuosity (QR)



$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

✓

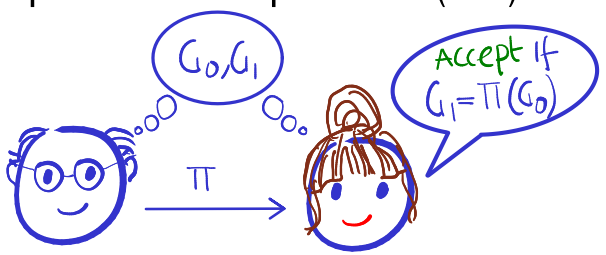
Quad. non-residuosity (QNR)



$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

✓

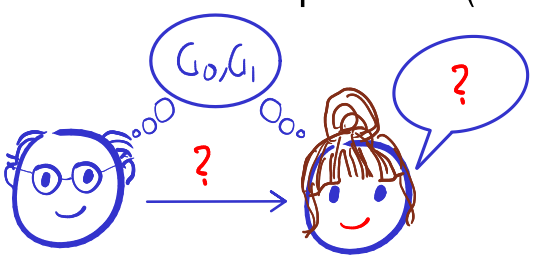
Graph isomorphism (GI)



$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

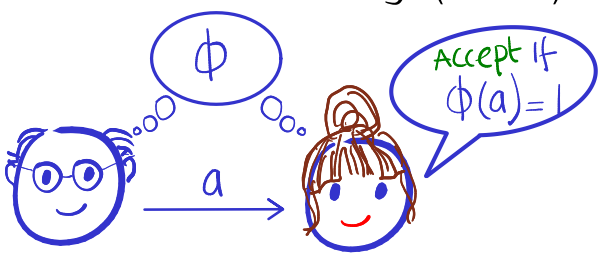
✓

Graph non-isomorphism (GNI)



$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

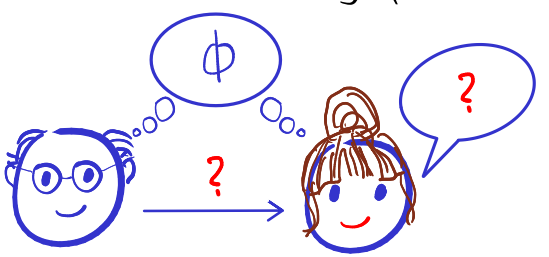
Boolean satisfiability (SAT)



$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

✓

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

Which Languages have IPs?

Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

✓

Quad. non-residuosity (QNR)

$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

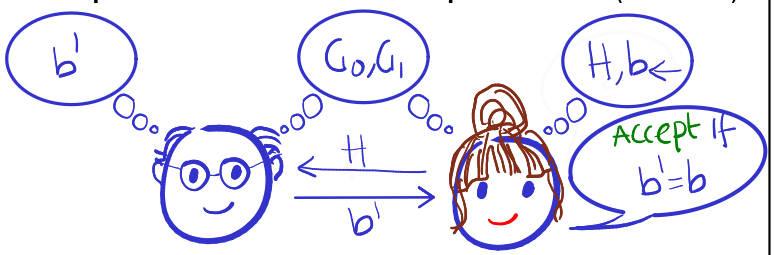
✓

Graph isomorphism (GI)

$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

✓

Graph non-isomorphism (GNI)



$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

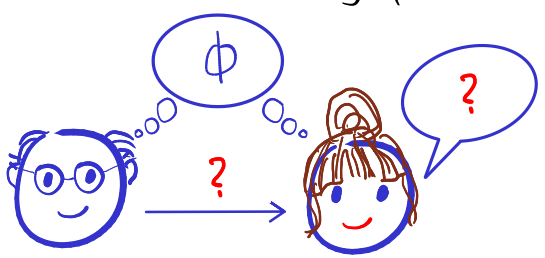
✓

Boolean satisfiability (SAT)

$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

✓

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

Which Languages have IPs?

Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

✓

Quad. non-residuosity (QNR)

$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

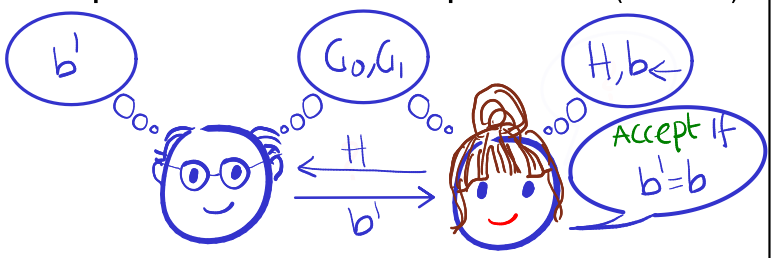
✓

Graph isomorphism (GI)

$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

✓

Graph non-isomorphism (GNI)



$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

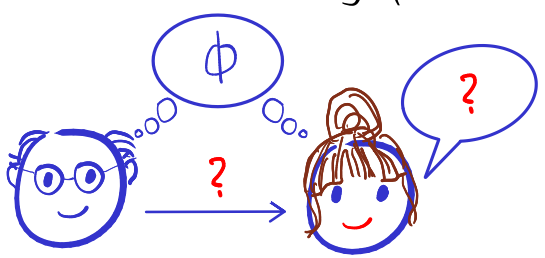
✓

Boolean satisfiability (SAT)

$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

✓

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

"Sumcheck"

✓

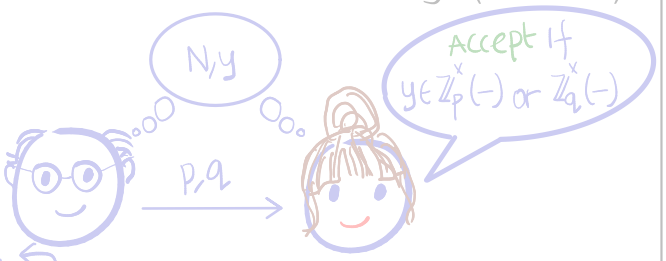
Which Languages have IPs? PSPACE Languages

Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

✓

Quad. non-residuosity (QNR)



$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$

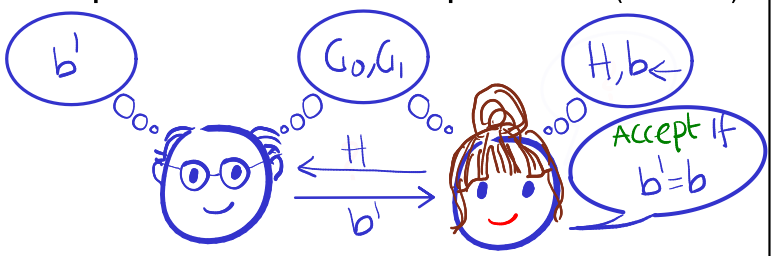
✓

Graph isomorphism (GI)

$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

✓

Graph non-isomorphism (GNI)



$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

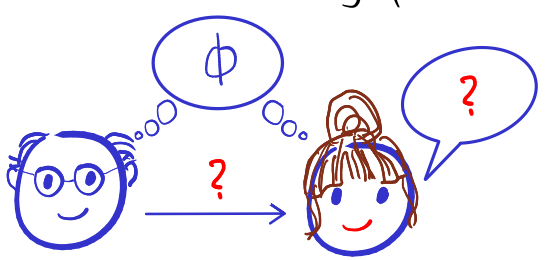
✓

Boolean satisfiability (SAT)

$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

✓

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

"Sumcheck"

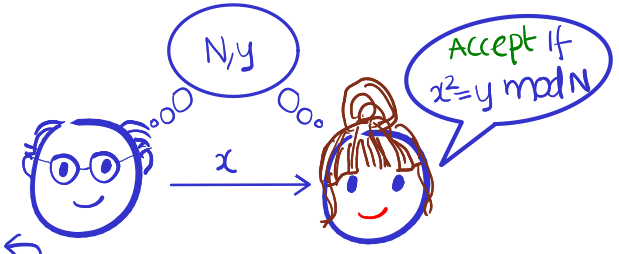
✓

Plan for Today's Lecture

1. Interactive Proof (IP)
2. Zero Knowledge (Interactive) Proof (ZKP)
3. Honest Verifier ZKP for Graph (Non-) Isomorphism

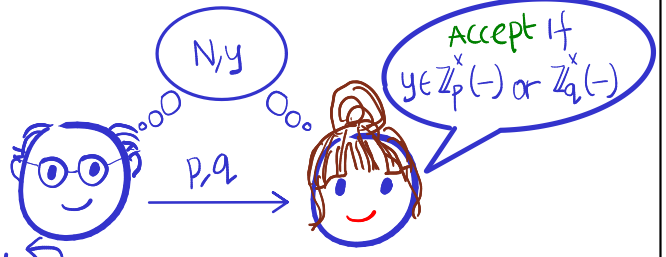
Any Issues with the NP Proofs We Saw?

Quadratic residuosity (QR)



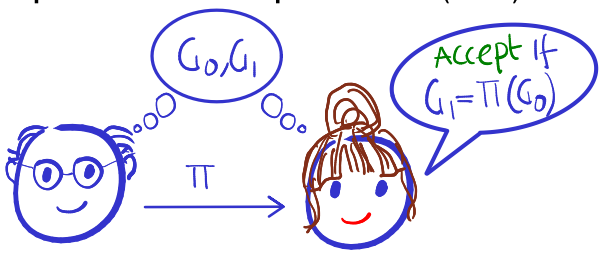
$$\alpha_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Quad. non-residuosity (QNR)



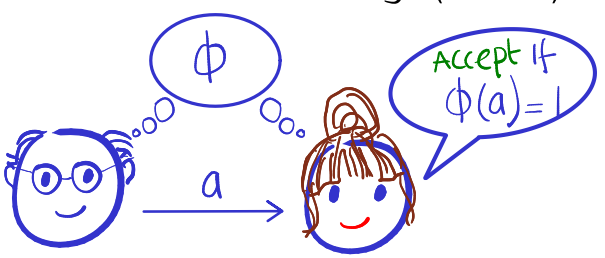
$$\alpha_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^\times \text{ s.t. } y = x^2 \bmod N\}$$

Graph isomorphism (GI)



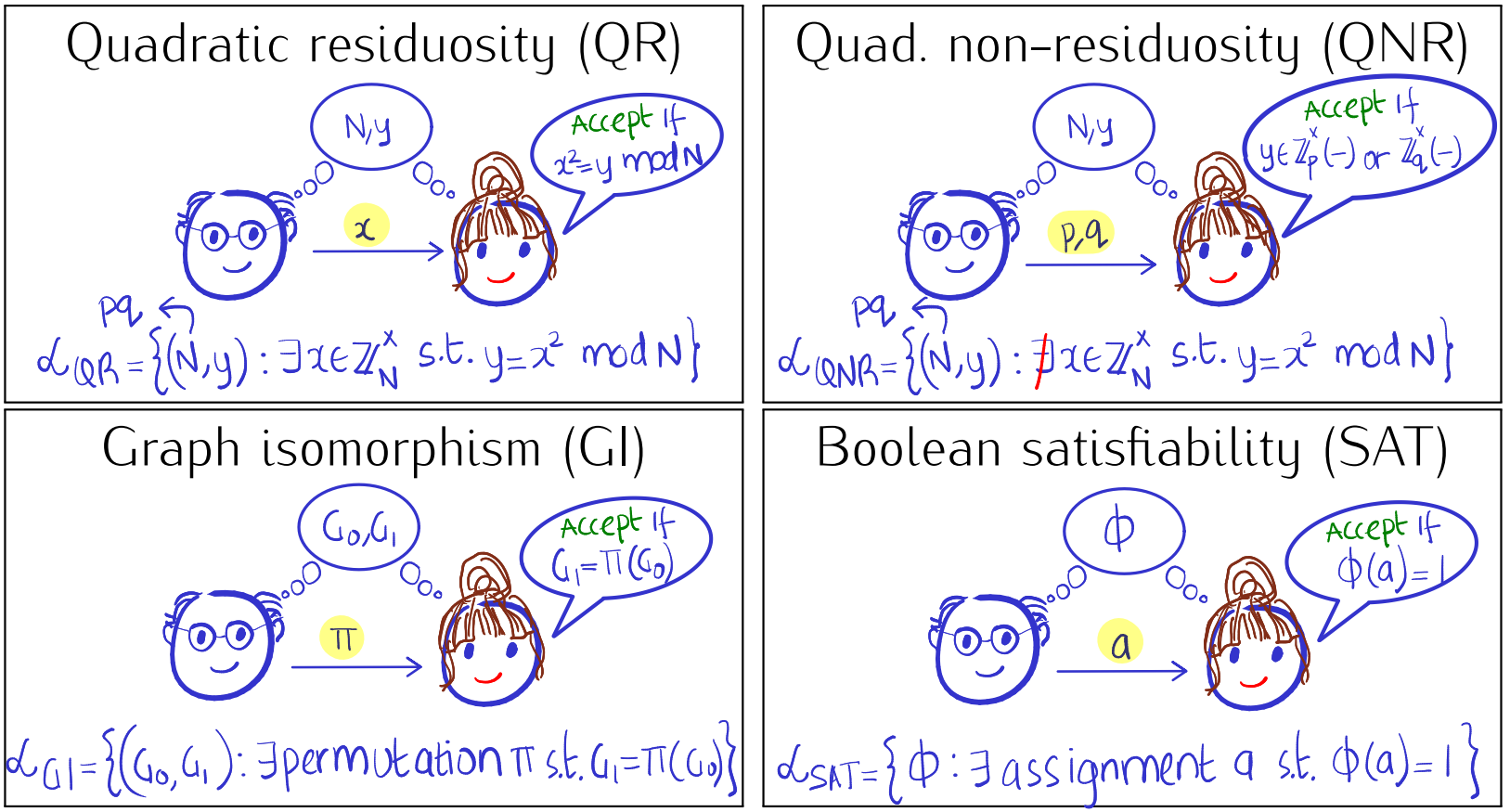
$$\alpha_{GI} = \{(G_0, G_1) : \exists \text{permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$$

Boolean satisfiability (SAT)



$$\alpha_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$$

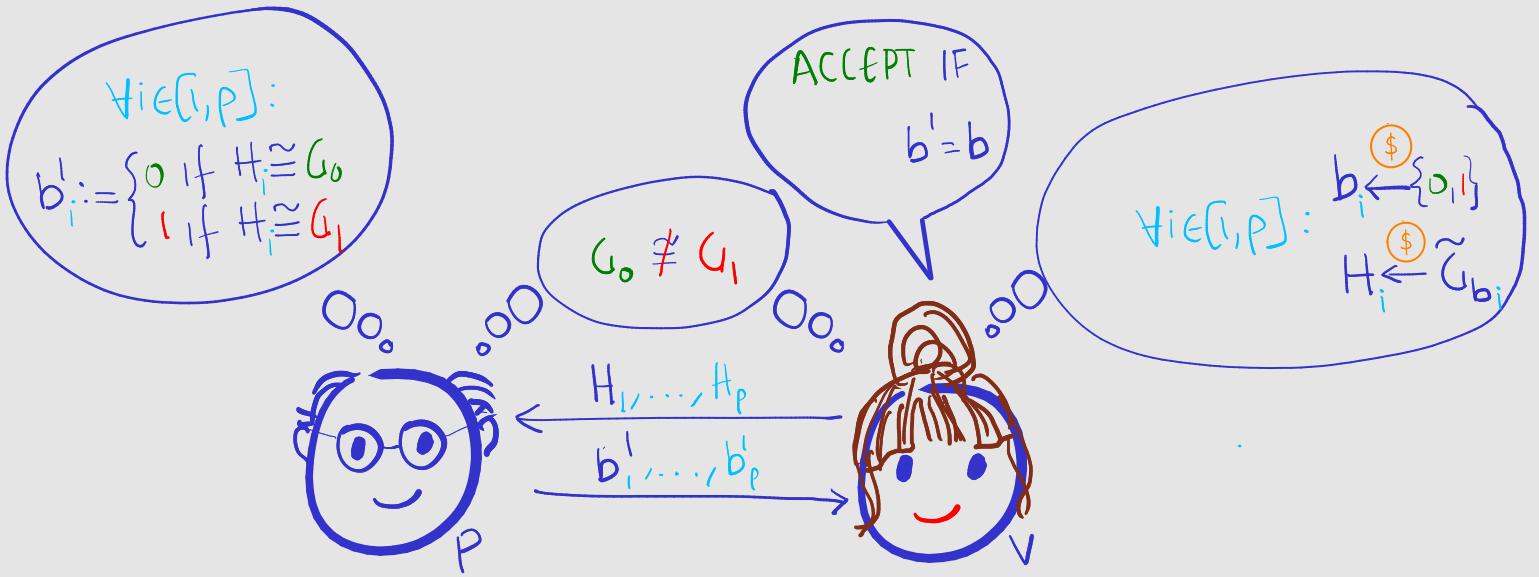
Any Issues with the NP Proofs We Saw?



- Verifier gains "non-trivial knowledge" about witness w
  - Not desirable, e.g., when $x = pk$ and $w = sk$ (identification)

What About the IP We Saw?

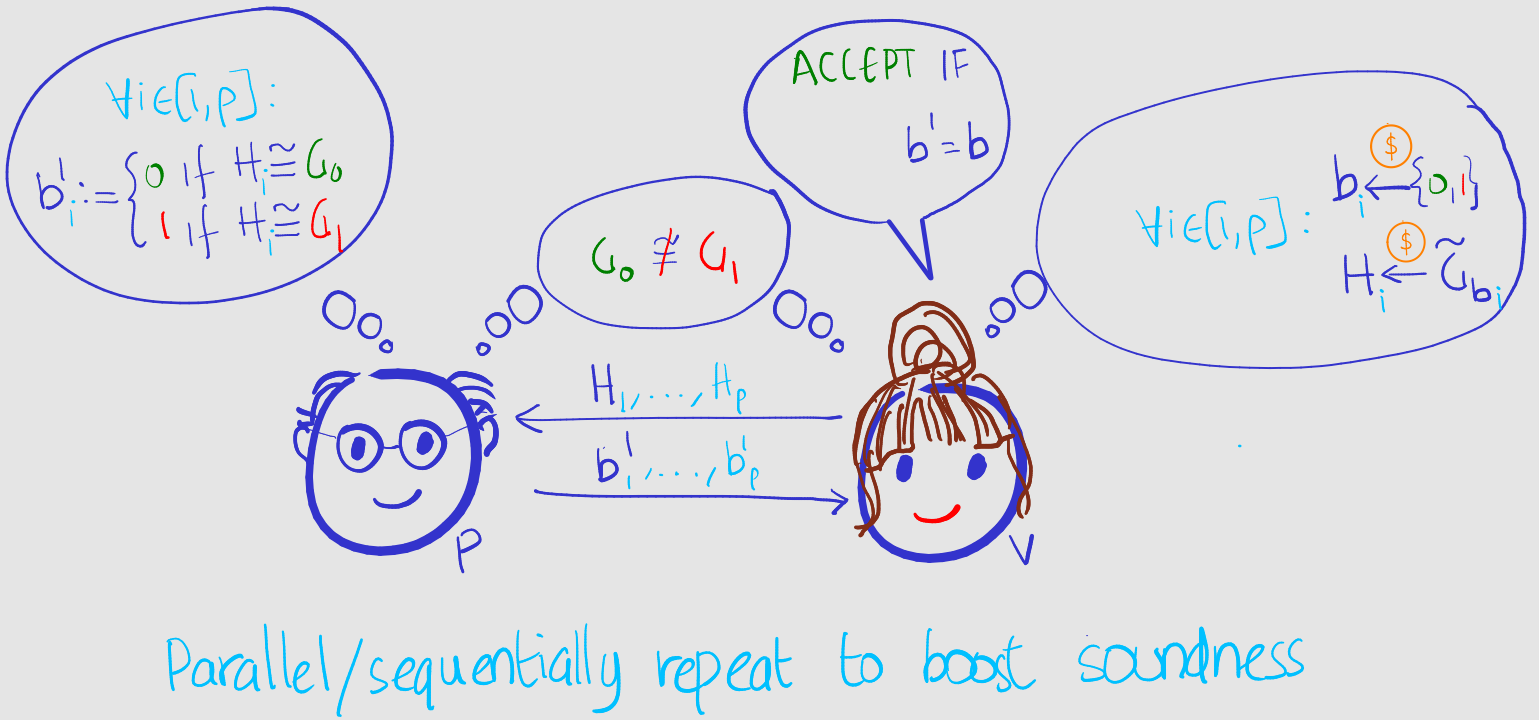
Protocol 1 (Π_{GNI}: IP for GNI)



Parallel/sequentially repeat to boost soundness

What About the IP We Saw?

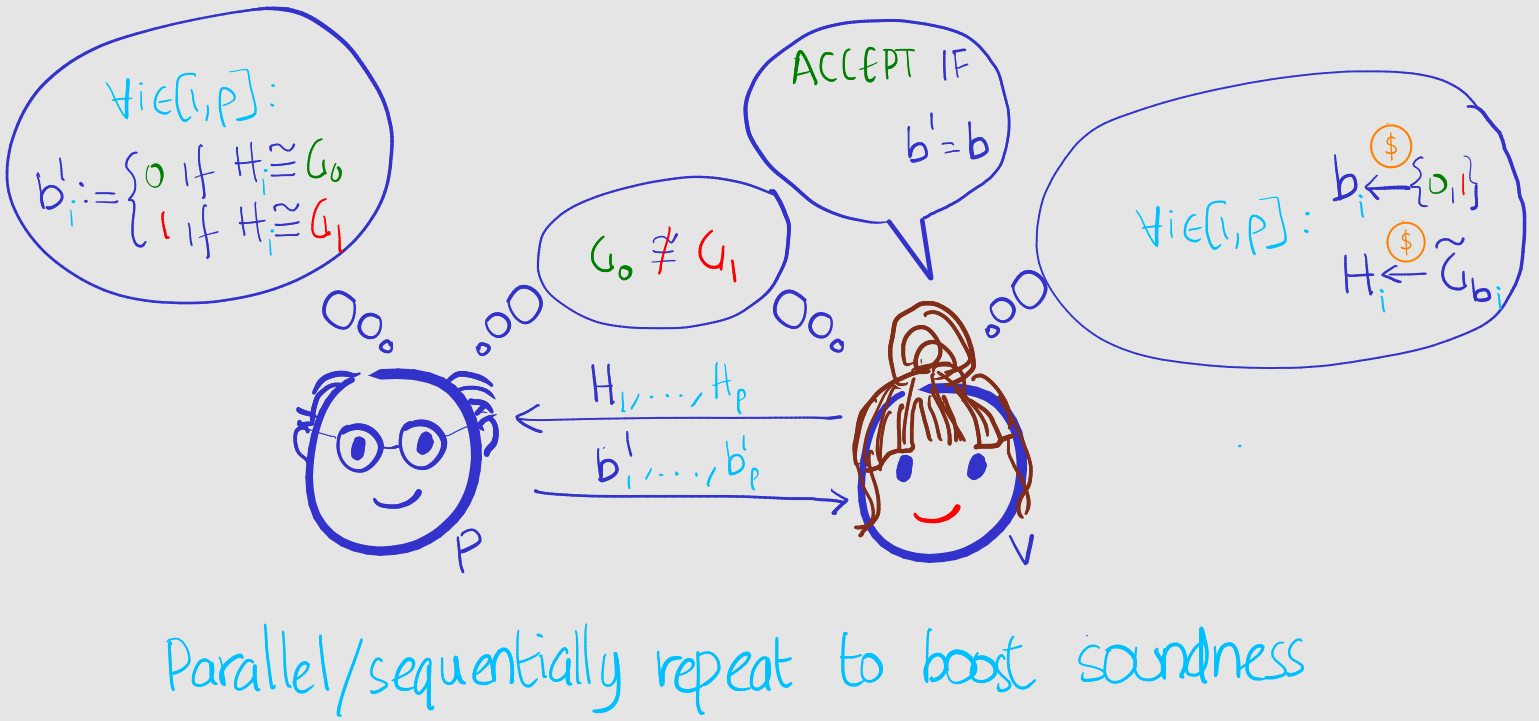
Protocol 1 (Π_{GNI}: IP for GNI)



- Seems V *gains no knowledge* beyond validity of the statement

What About the IP We Saw?

Protocol 1 (Π_{GNI}: IP for GNI)



Parallel/sequentially repeat to boost soundness

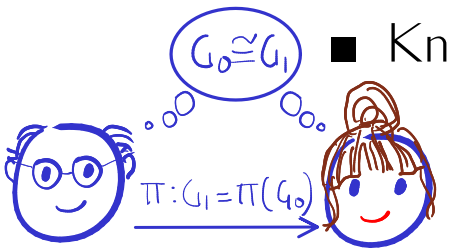
- Seems V *gains no knowledge* beyond validity of the statement
- We will see that Π_{GNI} is (honest-verifier) zero-knowledge!

How to Capture "V Gains No Knowledge"?

- Knowledge vs. information ← in the information-theoretic sense
  - Knowledge is computational

How to Capture "V Gains No Knowledge"?

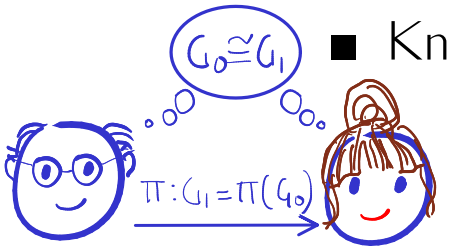
- Knowledge vs. information ← in the information-theoretic sense
  - Knowledge is computational: e.g., consider NP proof for GI



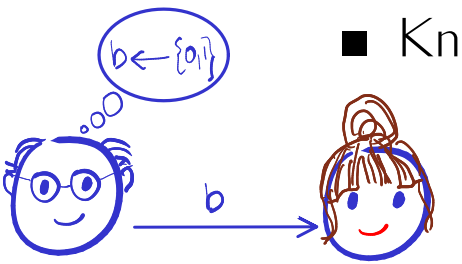
  - Given (G_0, G_1), the isomorphism π contains no *information*
  - But when given π, V "gains knowledge" since she couldn't have computed π herself

How to Capture "V Gains No Knowledge"?

- Knowledge vs. information ← in the information-theoretic sense



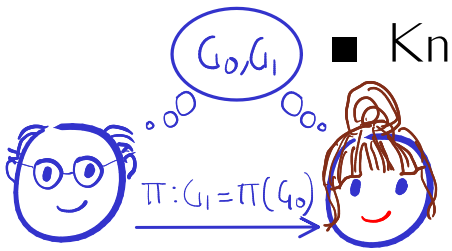
  - Knowledge is computational: e.g., consider NP proof for GI
    - Given (G_0, G_1), the isomorphism π contains no *information*
    - But when given π, V "gains knowledge" since she couldn't have computed π herself



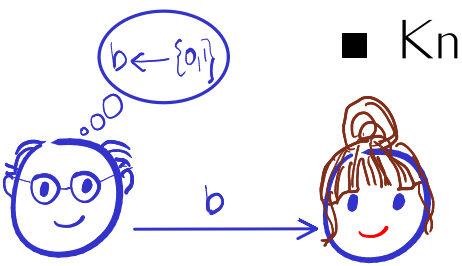
  - Knowledge pertains to public objects:
    - Flipping a *private* fair coin b and (later) revealing its outcome leads to V gaining *information*
    - But V *does not gain knowledge*: she could herself have tossed the private coin and revealed it

How to Capture "V Gains No Knowledge"?

- Knowledge vs. information ← in the information-theoretic sense
  - Knowledge is computational: e.g., consider NP proof for GI



    - Given (G_0, G_1), the isomorphism π contains no *information*
    - But when given π, V "gains knowledge" since she couldn't have computed π herself
  - Knowledge pertains to public objects:

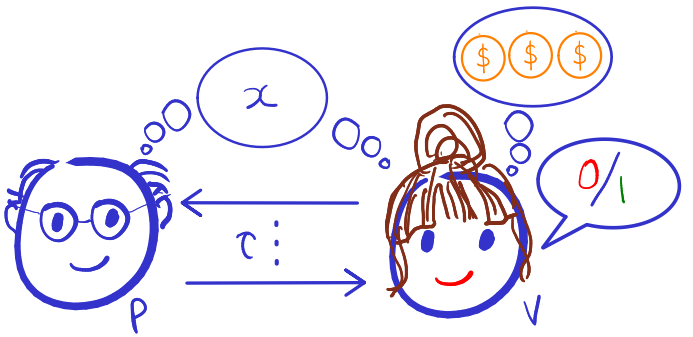


    - Flipping a *private* fair coin b and (later) revealing its outcome leads to V gaining *information*
    - But V *does not gain knowledge*: she could herself have tossed the private coin and revealed it

Intuitively, "V gains no knowledge" if anything V can *compute* (other than the validity of x) after the interaction, V could have computed *without it*

Defining Zero Knowledge via Simulators...

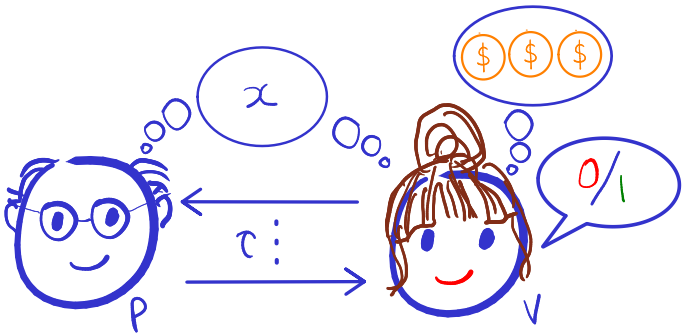
- Formalised via "simulation paradigm": $\mathsf{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V} \rangle(x))$ can be *efficiently* simulated given only the instance



Defining Zero Knowledge via Simulators...

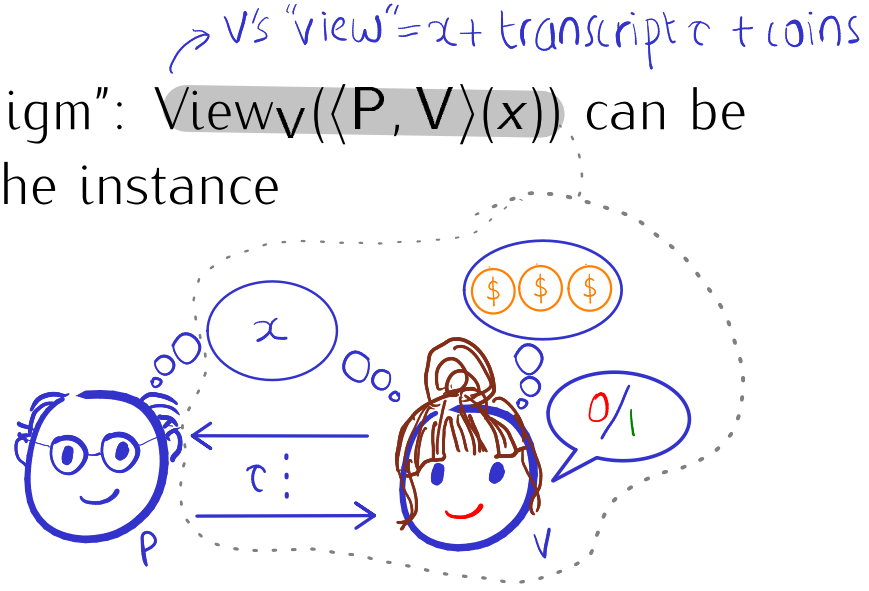
V's "view" = x + transcript τ + coins

- Formalised via "simulation paradigm": $\mathsf{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V} \rangle(x))$ can be *efficiently* simulated given only the instance



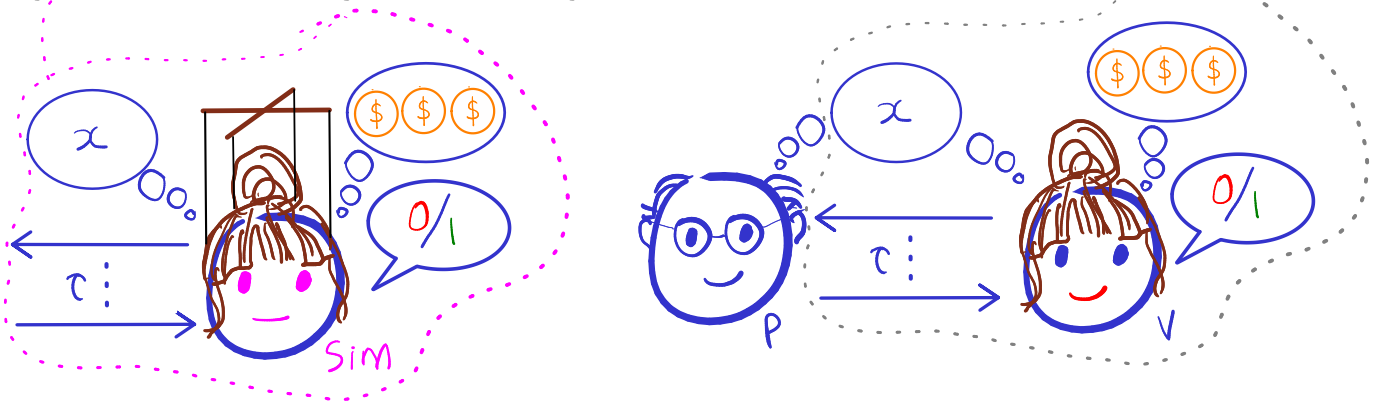
Defining Zero Knowledge via Simulators...

- Formalised via "simulation paradigm": $\mathsf{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V} \rangle(x))$ can be *efficiently* simulated given only the instance



Defining Zero Knowledge via Simulators...

- Formalised via "simulation paradigm": $\mathsf{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V} \rangle(x))$ can be *efficiently* simulated given only the instance



Defintion 2 (Honest-Verifier *Perfect* ZK)

An IP Π is honest-verifier perfect ZK if there exists a PPT simulator Sim *such that for all distinguishers* D *and all* $x \in \mathcal{L}$*, the following is zero*

$$\Pr[\mathsf{D}(\mathsf{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V} \rangle(x))) = 1] - \Pr[\mathsf{D}(\mathsf{Sim}(x)) = 1]$$

Defining Zero Knowledge via Simulators...

Defintion 2 (~~Honest-Verifier~~ Malicious-verifier *Perfect* ZK)

An IP Π is honest-verifier perfect ZK if ($\forall V^*$) there exists a PPT simulator Sim *such that for all distinguishers* D *and all* $x \in \mathcal{L}$*, the following is zero*

$$\Pr[\mathsf{D}(\mathrm{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V}^* \rangle(x))) = 1] - \Pr[\mathsf{D}(\mathsf{Sim}(x)) = 1]$$

- Malicious-Verifier ZK: honest verifier $\mathsf{V} \rightarrow$ all verifiers V^*
  - For every V^* there exists a PPT simulator Sim

Defining Zero Knowledge via Simulators...

Defintion 2 (Honest-Verifier ~~Perfect~~ Computational ZK)

An IP Π is honest-verifier ~~perfect~~ computational ZK if there exists a PPT simulator Sim *such that for ~~all~~ PPT distinguishers* D *and all* $x \in \mathcal{L}$*, the following is ~~zero~~ negligible*

$$\left|\Pr[\mathsf{D}(\mathrm{View}_{\mathsf{V}}(\langle \mathsf{P}, \mathsf{V}\rangle(x))) = 1] - \Pr[\mathsf{D}(\mathsf{Sim}(x)) = 1]\right|$$

- Malicious-Verifier ZK: honest verifier $\mathsf{V} \to$ all verifiers V^*
  - For every V^* there exists a PPT simulator Sim
- Computational ZK: relax
  - all distinguishers $\mathsf{D} \to$ PPT distinguishers
  - zero $\to$ negligible

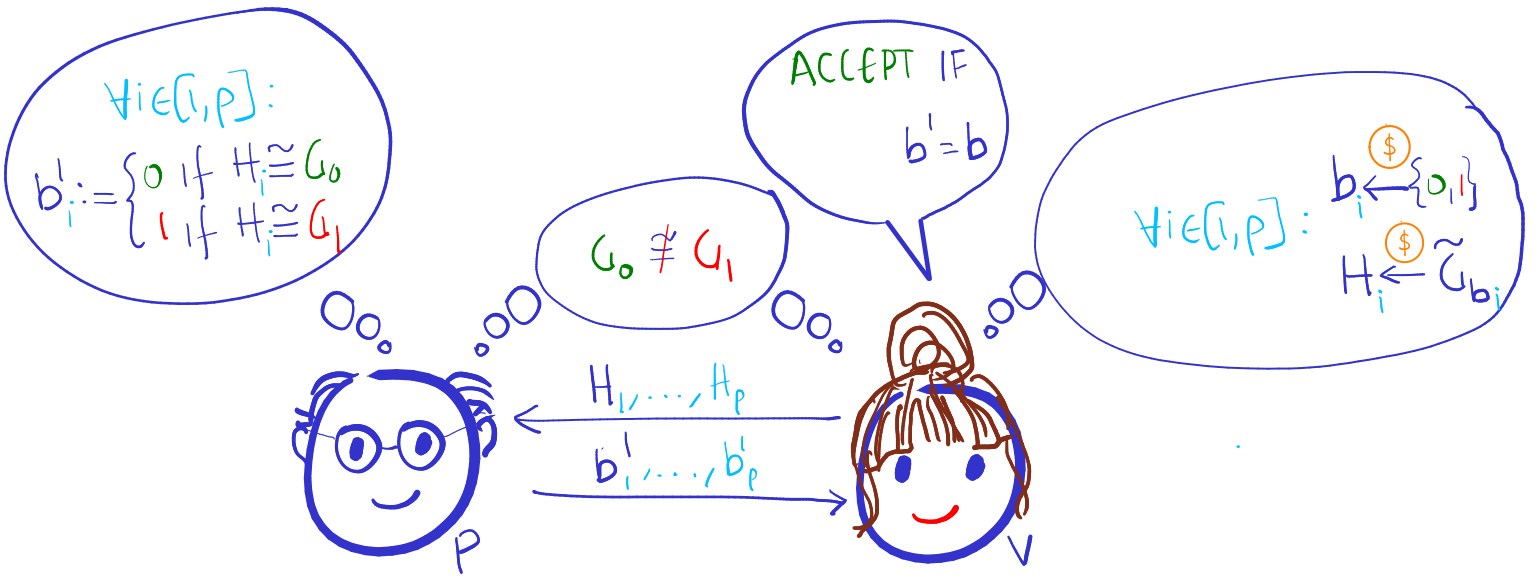
Plan for Today's Lecture

1. Interactive Proof (IP)
2. Zero Knowledge (Interactive) Proof (ZKP)
3. Honest Verifier ZKP for Graph (Non-) Isomorphism

Π_{GNI} is *Honest-Verifier* ZK

Theorem 2

Π_{GNI} is honest-verifier perfect zero-knowledge IP for $\mathcal{L}_{GNI}$

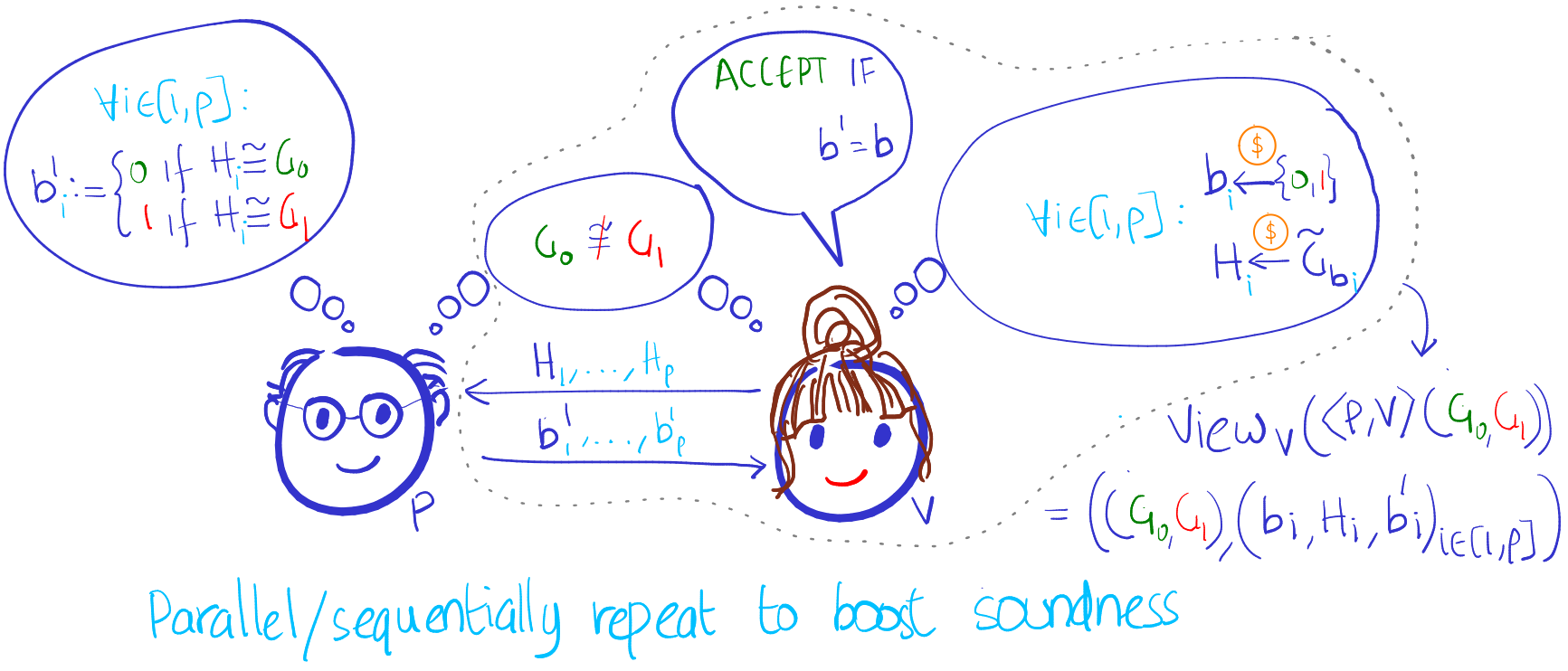


Parallel/sequentially repeat to boost soundness

Π_{GNI} is *Honest-Verifier* ZK

Theorem 2

Π_{GNI} is honest-verifier perfect zero-knowledge IP for $\mathcal{L}_{GNI}$



Π_{GNI} is *Honest-Verifier* ZK

Theorem 2

Π_{GNI} is honest-verifier perfect zero-knowledge IP for $\mathcal{L}_{GNI}$

Proof.

$$\text{view}_V(\langle P, V\rangle(G_0, G_1)) := \left((G_0, G_1), \left(b_i, H_i, b_i'\right)_{i\in[1,\rho]}\right)$$

$\forall\, G_0 \not\cong G_1$:

Π_{GNI} is *Honest-Verifier* ZK

Theorem 2

Π_{GNI} is honest-verifier perfect zero-knowledge IP for $\mathcal{L}_{GNI}$

Proof.

$$\text{view}_V(\langle P,V\rangle(G_0,G_1)) := \left((G_0,G_1),\left(b_i,H_i,b_i'\right)_{i\in[1,\rho]}\right)$$

$\forall\, G_0 \not\cong G_1$:

$\text{Sim}(G_0,G_1) :=$?

Π_{GNI} is *Honest-Verifier* ZK

Theorem 2

Π_{GNI} is honest-verifier perfect zero-knowledge IP for $\mathcal{L}_{GNI}$

Proof.

$$\text{view}_V(\langle P,V\rangle(G_0,G_1)) := \left((G_0,G_1),\left(b_i,H_i,b_i'\right)_{i\in[1,p]}\right)$$

$\forall\ G_0 \not\cong G_1$:

$\text{Sim}(G_0,G_1)$:= $\forall i\in[1,p]$: sample $b_i \leftarrow \{0,1\}$ and $H_i \leftarrow \tilde{G}_{b_i}$

o/p $\left((G_0,G_1),\left(b_i,H_i,b_i\right)_{i\in[1,p]}\right)$

$\forall\ G_0 \not\cong G_1$: $\text{view}_V(\langle P,V\rangle(G_0,G_1))$ identically distributed to $\text{Sim}(G_0,G_1)$.

Π_{GNI} is *Honest-Verifier* ZK

Theorem 2

Π_{GNI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GNI}$

Proof.

$$\text{view}_V(\langle P,V\rangle(G_0,G_1)) := \left((G_0,G_1),\left(b_i,H_i,b_i'\right)_{i\in[1,p]}\right)$$

$\forall\ G_0 \not\cong G_1$:

$\text{Sim}(G_0,G_1)$:= $\forall i\in[1,p]$: sample $b_i \leftarrow \{0,1\}$ and $H_i \leftarrow \tilde{G}_{b_i}$

o/p $\left((G_0,G_1),\left(b_i,H_i,b_i\right)_{i\in[1,p]}\right)$

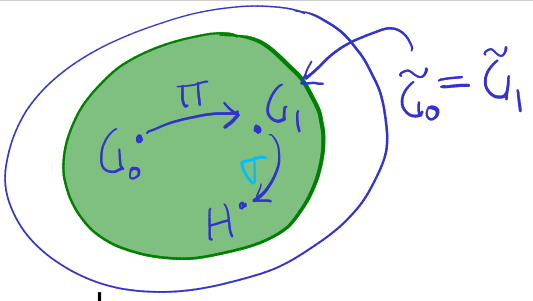
$\forall\ G_0 \not\cong G_1$: $\text{view}_V(\langle P,V\rangle(G_0,G_1))$ identically distributed to $\text{Sim}(G_0,G_1)$. □

Exercise 3

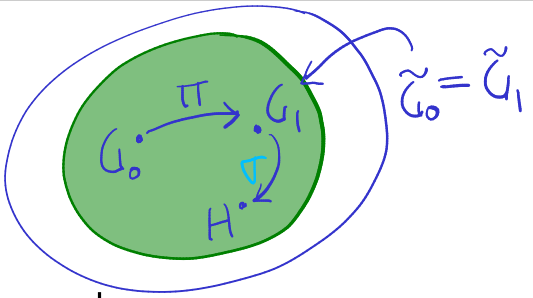
1. *What happens if* V *is malicious and can deviate from protocol?*
2. *Using ideas from* Π_{GNI}*, build honest-verifier ZKP for* $\mathcal{L}_{QNR}$

Honest-Verifier ZKP for GI...

- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\cdot\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\cdot\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random



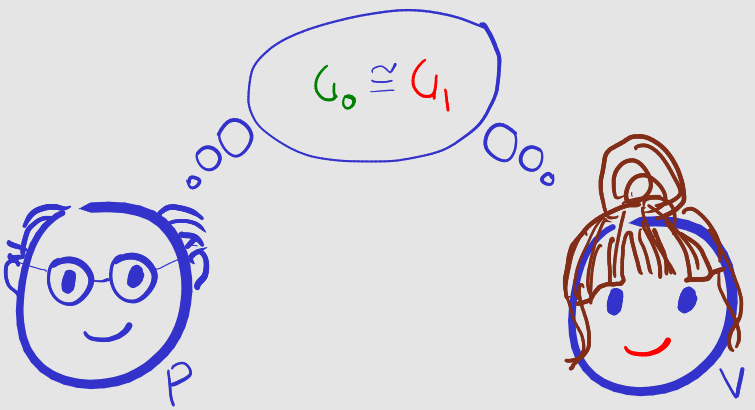
Honest-Verifier ZKP for GI...



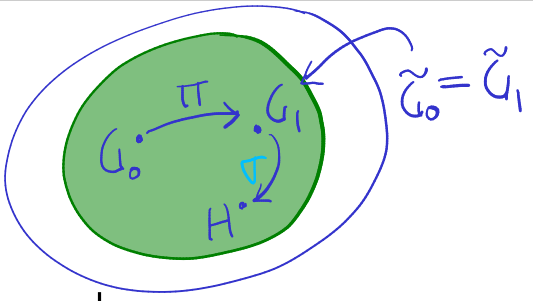
- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

Protocol 2 (Π_{GI}: IP for GI)

$G_0 \simeq G_1$ if $G_1 = H(G_0)$

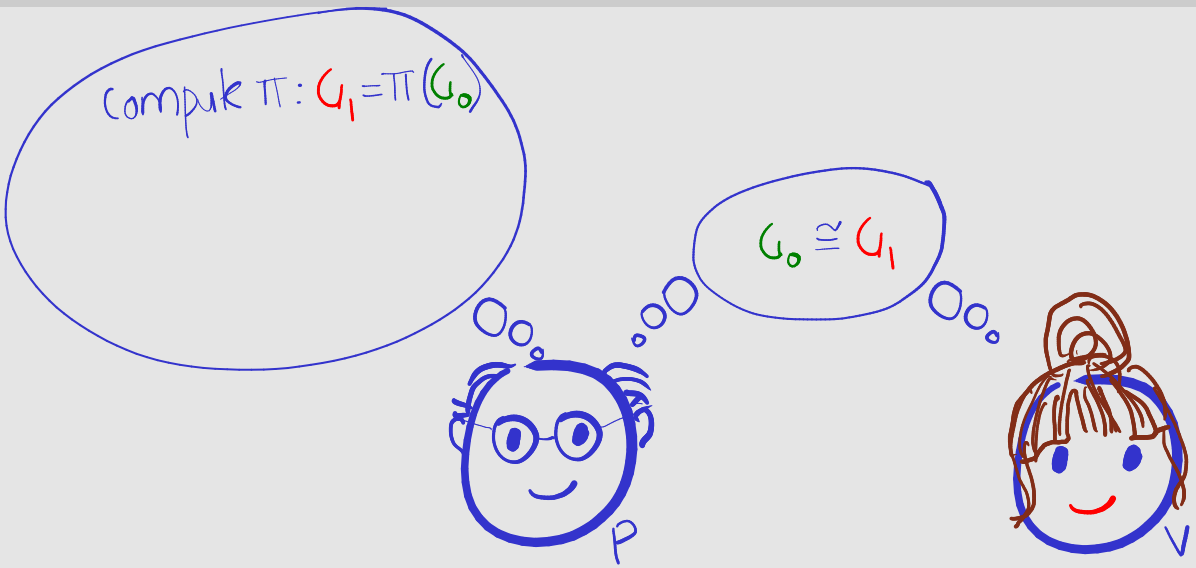


Honest-Verifier ZKP for GI...



- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

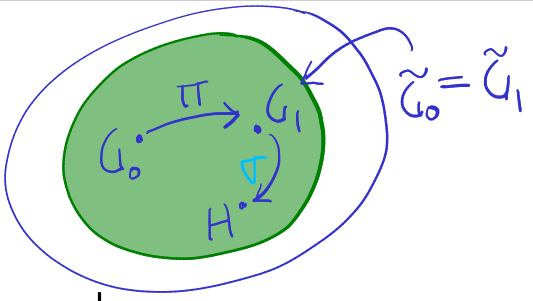
Protocol 2 (Π_{GI}: IP for GI)



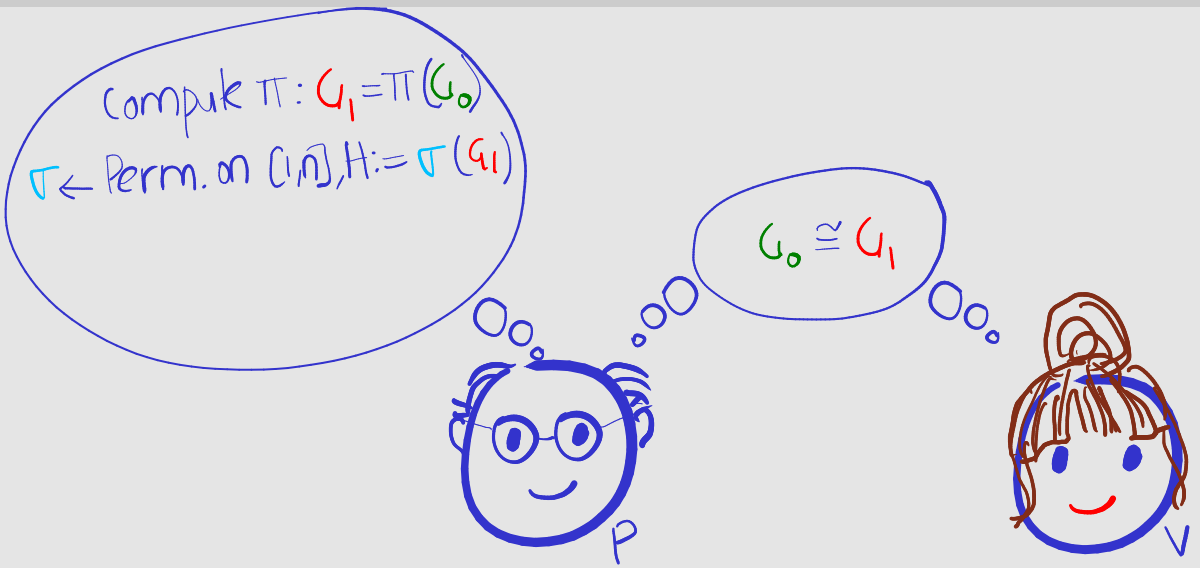
$G_0 \simeq G_1$ if $G_1 = H(G_0)$

Honest-Verifier ZKP for GI...

- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\cdot\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\cdot\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

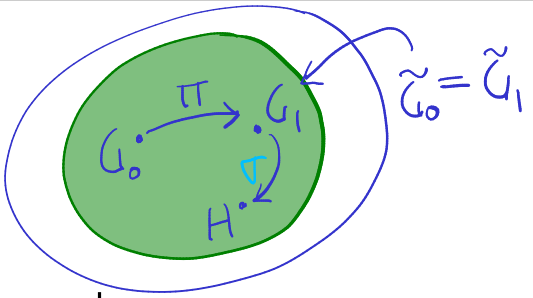


Protocol 2 (Π_{GI}: IP for GI)



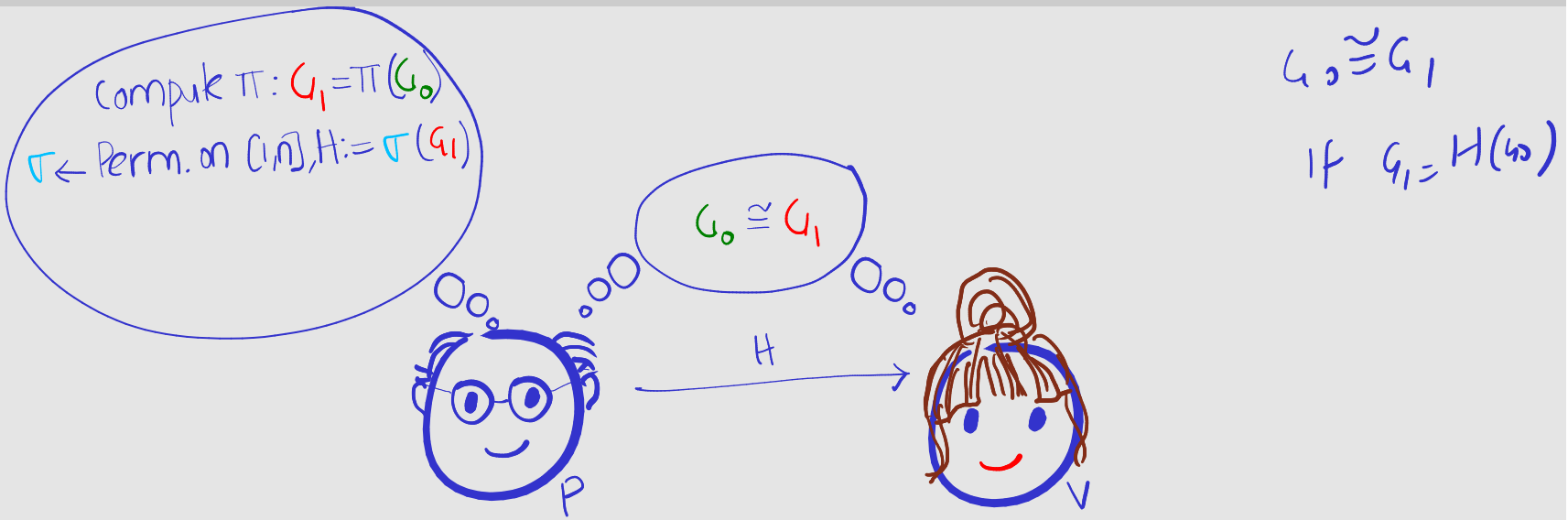
$G_0 \cong G_1$ if $G_1 = H(G_0)$

Honest-Verifier ZKP for GI...

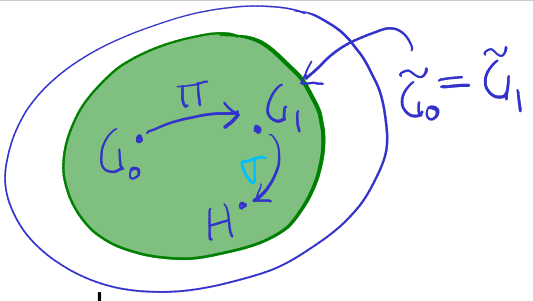


- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

Protocol 2 (Π_{GI}: IP for GI)

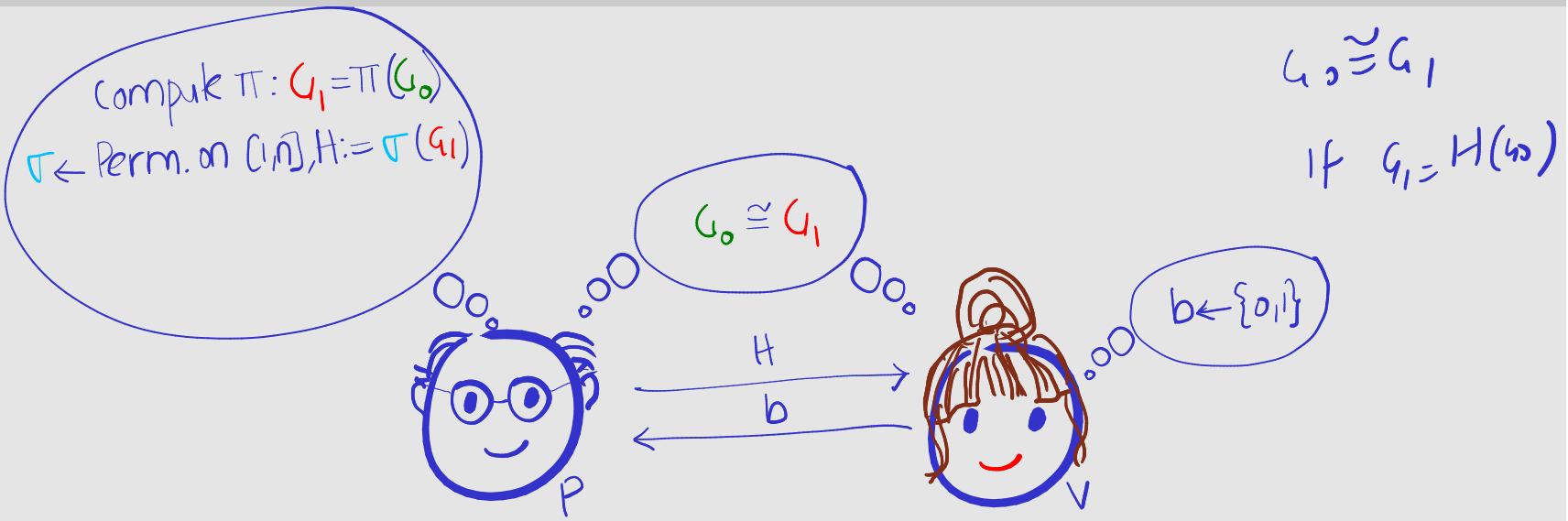


Honest-Verifier ZKP for GI...

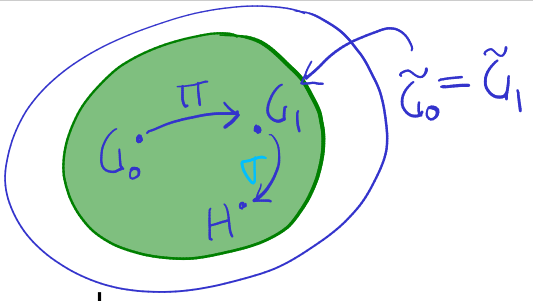


- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

Protocol 2 (Π_{GI}: IP for GI)

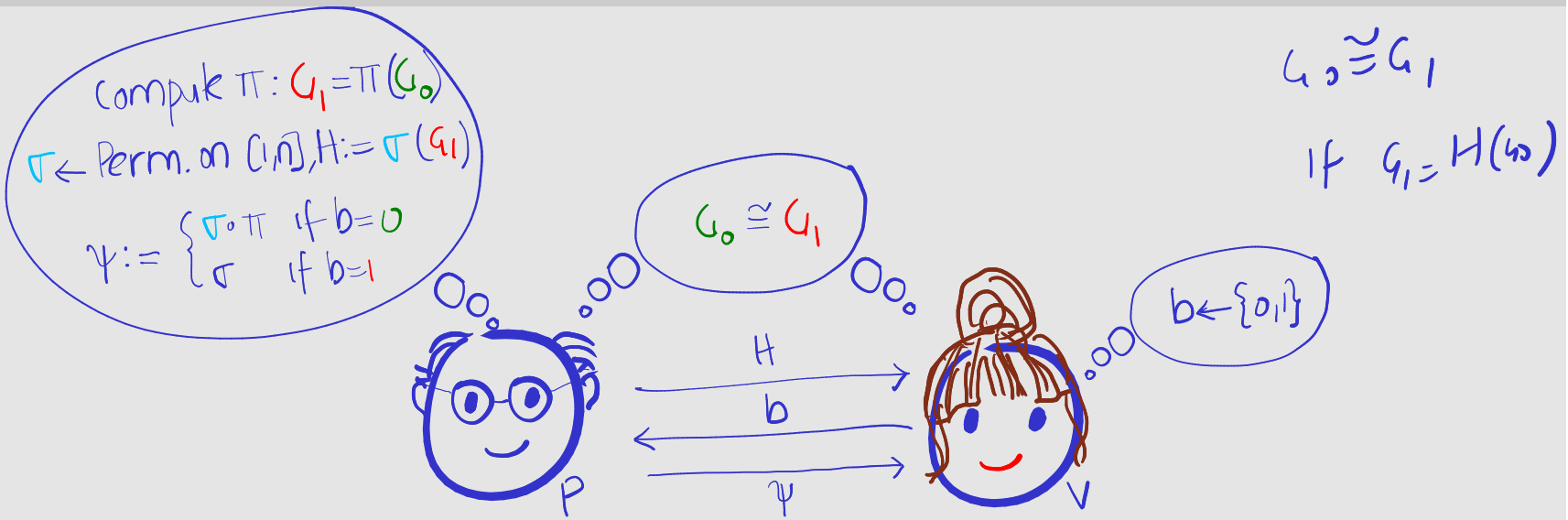


Honest-Verifier ZKP for GI...

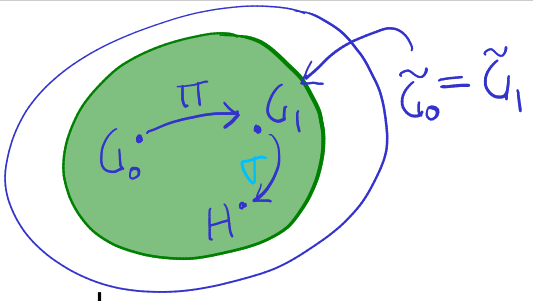


- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

Protocol 2 (Π_{GI}: IP for GI)

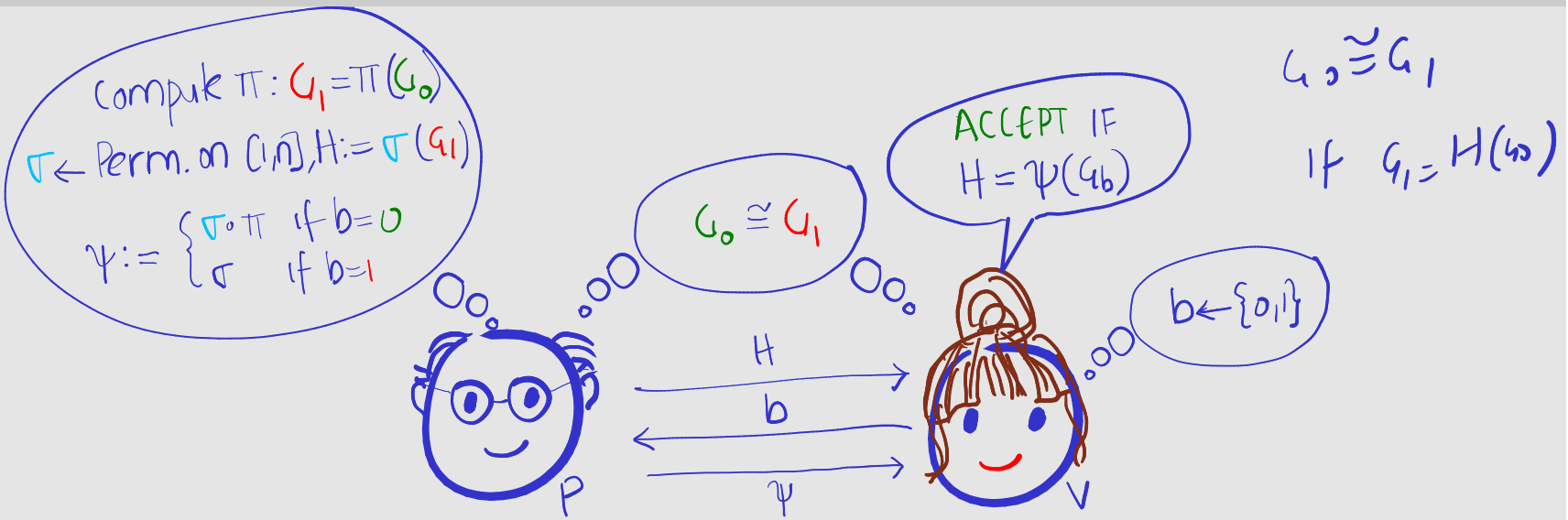


Honest-Verifier ZKP for GI...



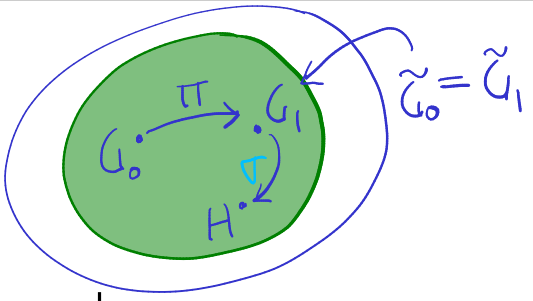
- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

Protocol 2 (Π_{GI}: IP for GI)

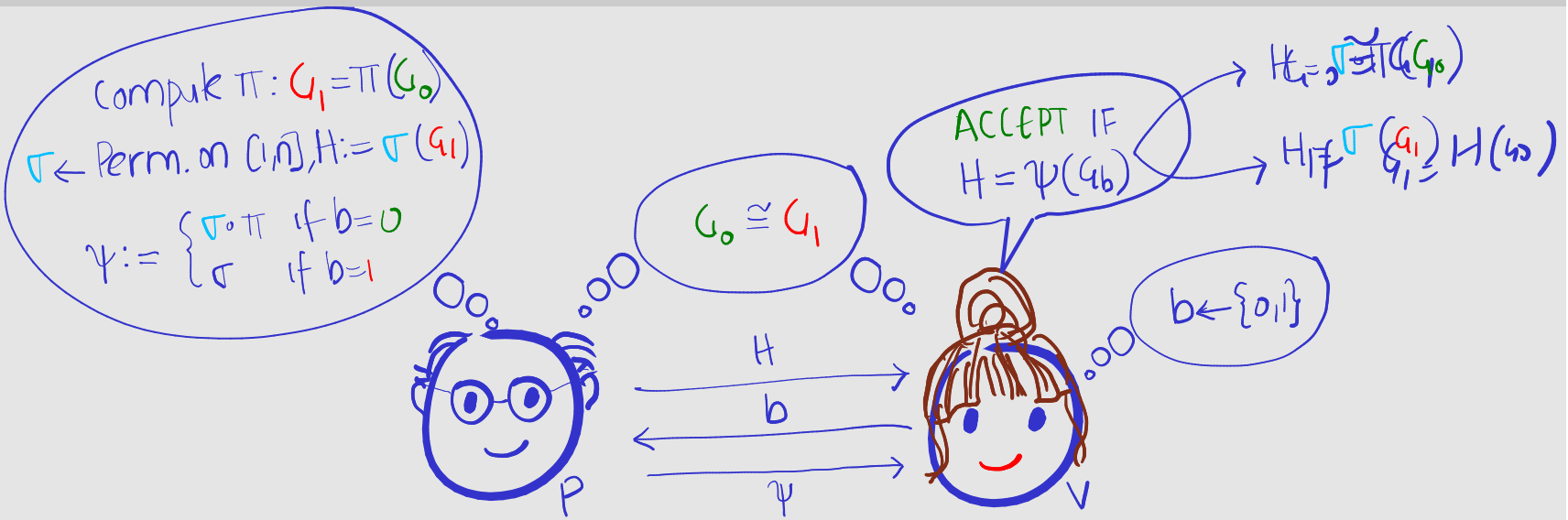


Honest-Verifier ZKP for GI...

- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random

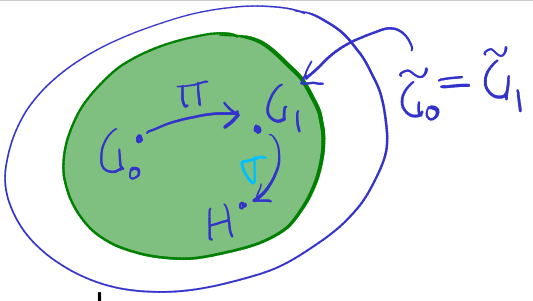


Protocol 2 (Π_{GI}: IP for GI)

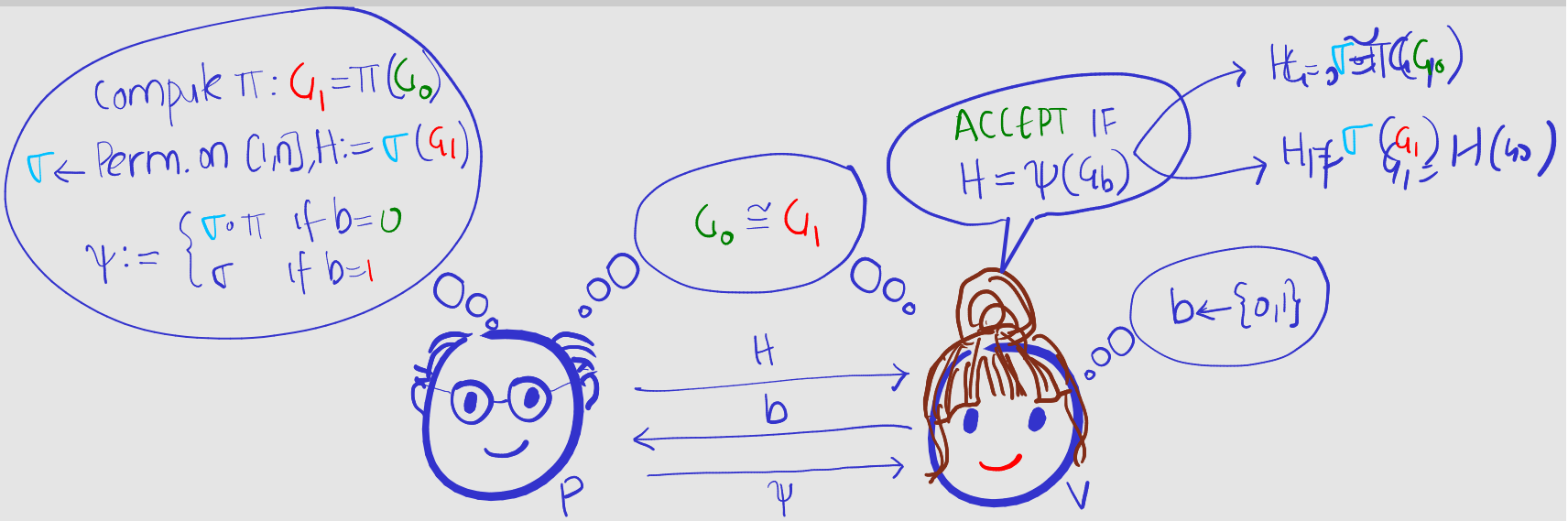


Honest-Verifier ZKP for GI...

- Idea for ZK:
  1. $G_0 \overset{\pi}{\simeq} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\simeq} H$ then $G_0 \overset{\sigma\circ\pi}{\simeq} H$
  2. Prover sends a random H s.t. $G_1 \overset{\sigma}{\simeq} H$
  3. Verifier asks to prove $G_0 \overset{\sigma\circ\pi}{\simeq} H$ or $G_1 \overset{\sigma}{\simeq} H$ at random



Protocol 2 (Π_{GI}: IP for GI)



(Parallel/sequentially repeat to boost soundness)

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} is honest-verifier perfect zero-knowledge IP for $\mathcal{L}_{GI}$

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GI}$

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can answer both challenges $\Rightarrow$ V always accepts
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H P^* commits to, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P^* can do is guess b

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GI}$

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can answer both challenges $\Rightarrow$ V always accepts
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H P* commits to, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P* can do is guess b
- Zero knowledge:

$$\text{view}_V(\langle P, V\rangle(G_0, G_1)) := \big((G_0, G_1), (H, b, \psi)\big)$$

$$\forall\, G_0 \overset{\pi}{\cong} G_1 :$$

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GI}$

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can answer both challenges $\Rightarrow$ V always accepts
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H P* commits to, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P* can do is guess b
- Zero knowledge:

$\sigma(G_1)$ → bit → $\begin{cases} \sigma \circ \pi & \text{if } b=0 \\ \sigma & \text{if } b=1 \end{cases}$

$\text{view}_V(\langle P, V\rangle(G_0, G_1)) := ((G_0, G_1), (H, b, \psi))$

$\forall\, G_0 \overset{\pi}{\cong} G_1$:

□

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GI}$

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can answer both challenges $\Rightarrow$ V always accepts
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H P* commits to, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P* can do is guess b
- Zero knowledge:

$\tau(G_1)$ ↑ bit → $\begin{cases} \sigma \circ \pi & \text{if } b=0 \\ \sigma & \text{if } b=1 \end{cases}$

$$\text{view}_V(\langle P,V\rangle(G_0,G_1)) := \big((G_0,G_1),(H,b,\psi)\big)$$

$\forall\, G_0 \overset{\pi}{\cong} G_1$: $\text{sim}(G_0,G_1)$:

□

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GI}$

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can answer both challenges $\Rightarrow$ V always accepts
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H P* commits to, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P* can do is guess b
- Zero knowledge:

$\tau(G_1)$ ↑ bit → $\begin{cases} \sigma \circ \pi & \text{if } b=0 \\ \sigma & \text{if } b=1 \end{cases}$

$\text{view}_V(\langle P,V\rangle(G_0,G_1)) := ((G_0,G_1),(H,b,\psi))$

$\forall\ G_0 \overset{\pi}{\cong} G_1$: $\text{Sim}(G_0,G_1)$: sample $b \leftarrow \{0,1\}$, $\psi \leftarrow$ permutation on $[1,n]$

set $H := \psi(G_b)$

o/p $((G_0,G_1),(H,b,\psi))$

□

Honest-Verifier ZKP for GI...

Theorem 3

Π_{GI} *is honest-verifier perfect zero-knowledge IP for* $\mathcal{L}_{GI}$

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can answer both challenges $\Rightarrow$ V always accepts
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H P* commits to, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P* can do is guess b
- Zero knowledge:

$\pi(G_1)$ ↑ bit ↑ $\begin{cases} \sigma\circ\pi & \text{if } b=0 \\ \sigma & \text{if } b=1 \end{cases}$

$\text{view}_V(\langle P,V\rangle(G_0,G_1)) := ((G_0,G_1),(H,b,\psi))$

$\forall\ G_0 \overset{\pi}{\cong} G_1$: $\text{Sim}(G_0,G_1)$: sample $b \leftarrow \{0,1\}$, $\psi \leftarrow$ permutation on $[1,n]$

set $H := \psi(G_b)$

o/p $((G_0,G_1),(H,b,\psi))$

$\forall\ G_0 \cong G_1$: $\text{view}_V(\langle P,V\rangle(G_0,G_1))$ identically distributed to $\text{Sim}(G_0,G_1)$. □

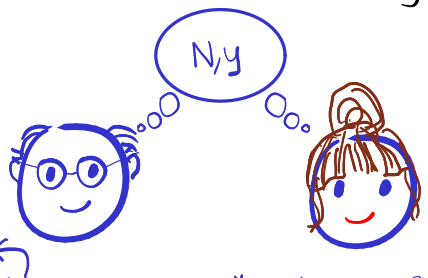
Honest-Verifier ZKP for GI...

Exercise 4

1. *What happens if* V *is malicious and can deviate from protocol?*
2. *Using ideas from* Π_{GI}*, build honest-verifier ZKP for* $\mathcal{L}_{QR}$

Which Languages have ZKPs?

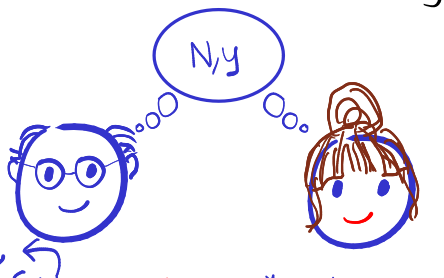
Quadratic residuosity (QR)



$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$

✓

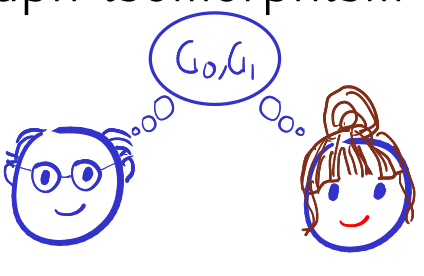
Quad. non-residuosity (QNR)



$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$

✓

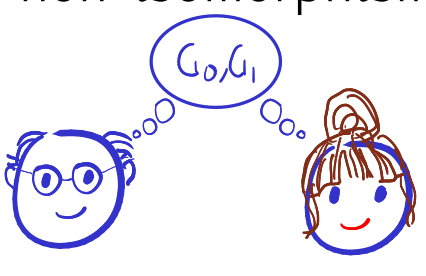
Graph isomorphism (GI)



$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

✓

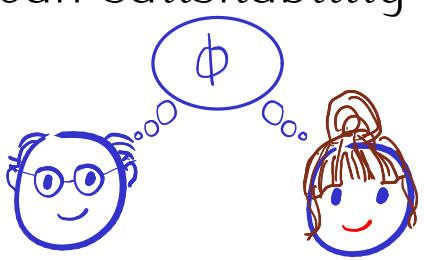
Graph non-isomorphism (GNI)



$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

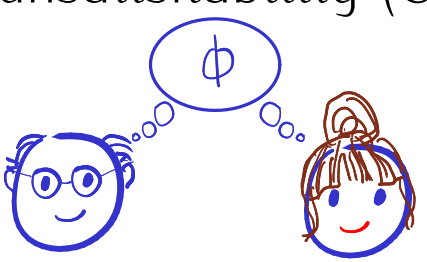
✓

Boolean satisfiability (SAT)



$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

Which Languages have ZKPs?

Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$

Quad. non-residuosity (QNR)

$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$

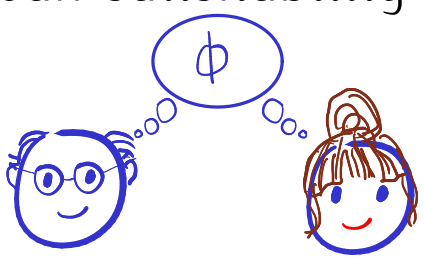
Graph isomorphism (GI)

$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

Graph non-isomorphism (GNI)

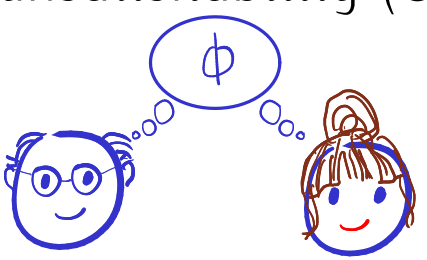
$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

Boolean satisfiability (SAT)



$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

L15

Which Languages have Computational ZKPs? PSPACE Languages

Quadratic residuosity (QR)

$\mathcal{L}_{QR} = \{(N,y) : \exists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$

✓

Quad. non-residuosity (QNR)

$\mathcal{L}_{QNR} = \{(N,y) : \nexists x \in \mathbb{Z}_N^* \text{ s.t. } y = x^2 \bmod N\}$

✓

Graph isomorphism (GI)

$\mathcal{L}_{GI} = \{(G_0, G_1) : \exists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

✓

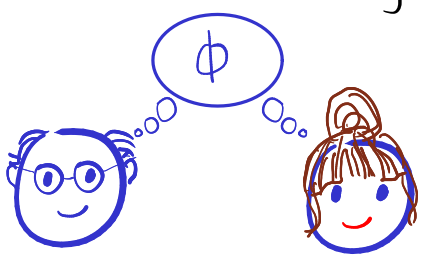
Graph non-isomorphism (GNI)

$\mathcal{L}_{GNI} = \{(G_0, G_1) : \nexists \text{ permutation } \pi \text{ s.t. } G_1 = \pi(G_0)\}$

✓

Boolean satisfiability (SAT)

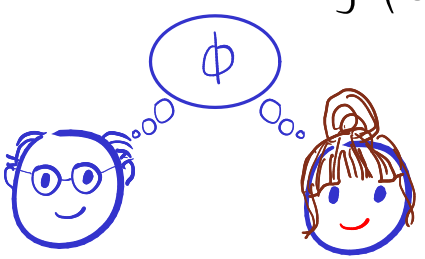
L15



$\mathcal{L}_{SAT} = \{\phi : \exists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

✓

Bool. unsatisfiability (UNSAT)



$\mathcal{L}_{UNSAT} = \{\phi : \nexists \text{ assignment } a \text{ s.t. } \phi(a) = 1\}$

✓

Are Randomness and Interaction Necessary?

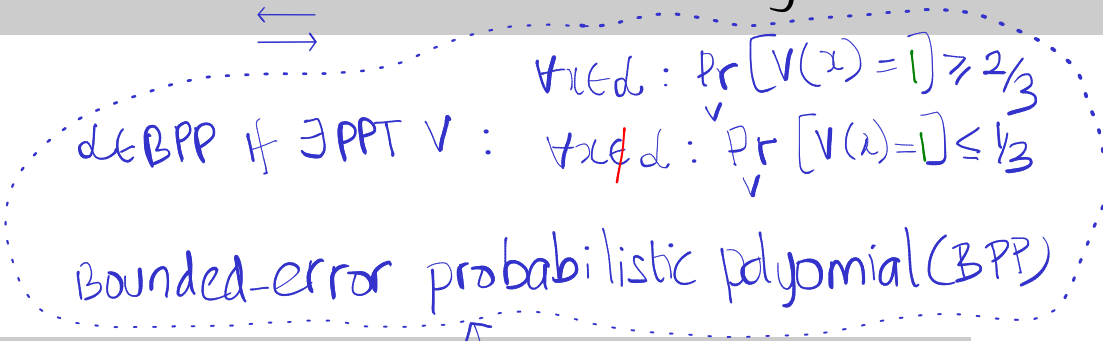
$\leftrightarrows$ Interaction is necessary

Exercise 5

If $\mathcal{L}$ has a non-interactive (i.e, one-message) ZKP then $\mathcal{L} \in$ BPP

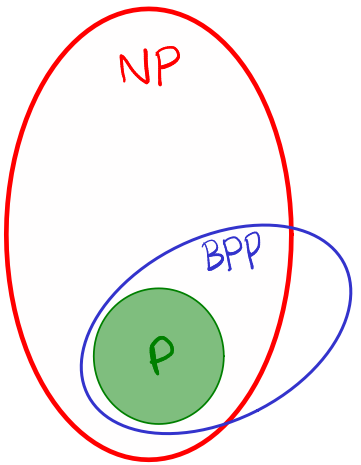
Are Randomness and Interaction Necessary?

$\leftrightarrows$ Interaction is necessary



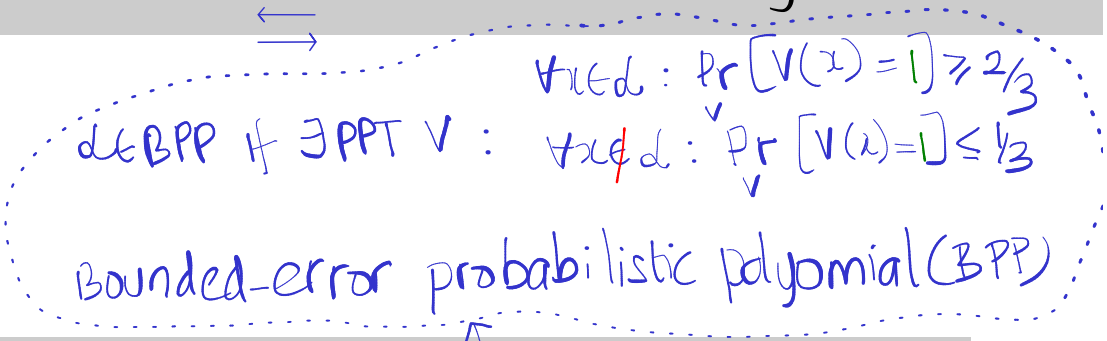
Exercise 5

If $\mathcal{L}$ has a non-interactive (i.e, one-message) ZKP then $\mathcal{L} \in$ BPP



Are Randomness and Interaction Necessary?

$\leftrightarrows$



$\leftrightarrows$ Interaction is necessary

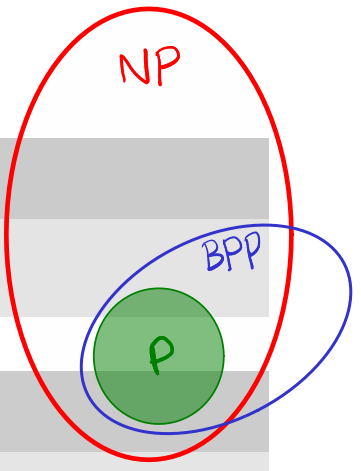
Exercise 5

If $\mathcal{L}$ has a non-interactive (i.e, one-message) ZKP then $\mathcal{L} \in$ BPP

Randomness is necessary

Exercise 6

If $\mathcal{L}$ has an IP with deterministic verifier then $\mathcal{L} \in$ NP



Exercise 7

If $\mathcal{L}$ has an ZKP with deterministic verifier then $\mathcal{L} \in$ BPP

To Recap Today's Lecture

- Traditional "NP" proofs vs *interactive* proofs
  - IP is more powerful: IP for GNI

To Recap Today's Lecture

- Traditional "NP" proofs vs *interactive* proofs
  - IP is more powerful: IP for GNI
- Zero-knowledge proofs
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm

To Recap Today's Lecture

- Traditional "NP" proofs vs *interactive* proofs
  - IP is more powerful: IP for GNI
- Zero-knowledge proofs
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for GNI (HW: QNR) and GI (HW: QR)

Next Lecture

- 27/Sep: Crib session for mid-term exam
- 01/Oct: More ZKP:
  - Malicious-verifier ZKP
  - Computational ZKP for all of NP!
    - New cryptographic primitive: commitment schemes

References

1. [Gol01, Chapter 4] for details of today's lecture
2. [GMR89] for definitional and philosophical discussion on ZK
3. The ZKPs for GI and GNI are taken from [GMR89, GMW91]
4. IP for all of PSPACE is due to [LFKN92, Sha90].
   Computational ZKP for all of PSPACE is due to [GMW91].


Shafi Goldwasser, Silvio Micali, and Charles Rackoff.
The knowledge complexity of interactive proof systems.
SIAM J. Comput., 18(1):186–208, 1989.


Oded Goldreich, Silvio Micali, and Avi Wigderson.
Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems.
J. ACM, 38(3):691–729, 1991.


Oded Goldreich.
The Foundations of Cryptography - Volume 1: Basic Techniques.
Cambridge University Press, 2001.


Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan.
Algebraic methods for interactive proof systems.
J. ACM, 39(4):859–868, October 1992.


Adi Shamir.
IP=PSPACE.
In *31st FOCS*, pages 11–15. IEEE Computer Society Press, October 1990.