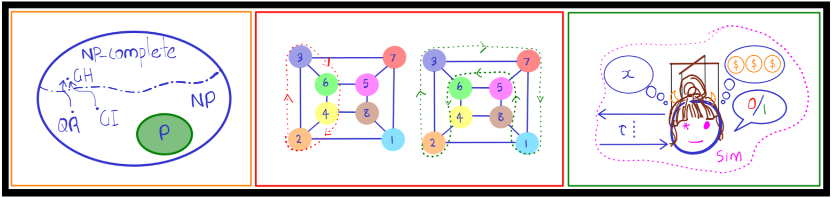


CS783: Theoretical Foundations of Cryptography

Lecture 15 (01/Oct/24)

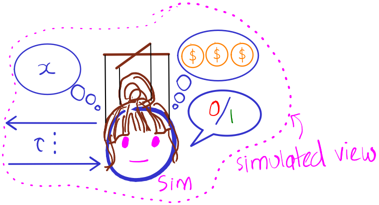
Instructor: Chethan Kamath

Recall from Last Lecture...

- *Interactive* proof (IP)
  - Compared to traditional "NP" proof
  - IP is powerful: IP for GNI

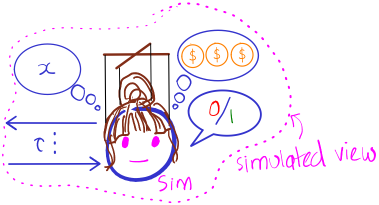
Recall from Last Lecture...

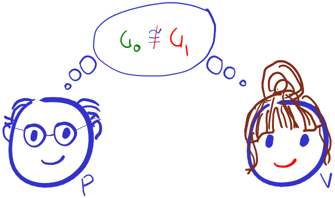
- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm



Recall from Last Lecture...

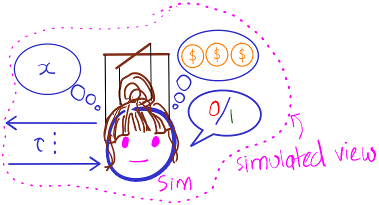
- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QNR**)

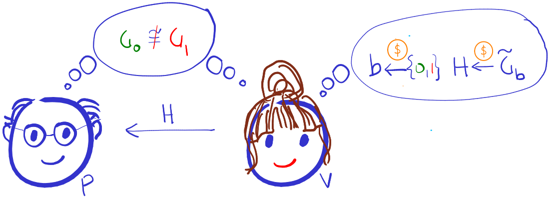




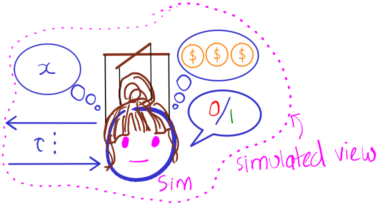
Recall from Last Lecture...

- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QNR**)

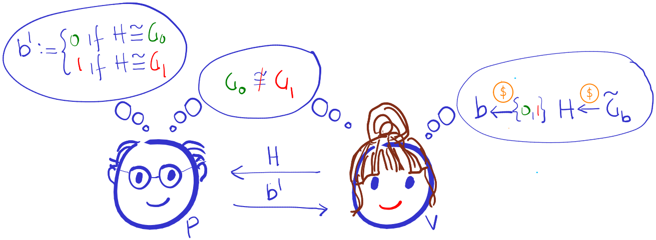


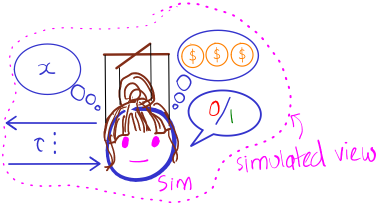


- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**

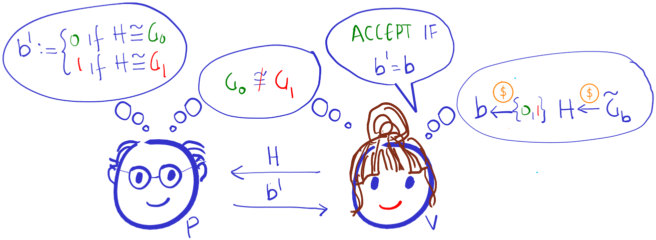


- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QNR**)

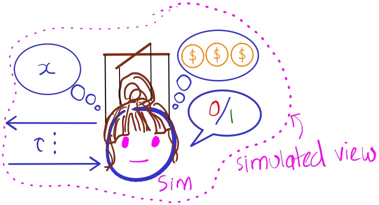


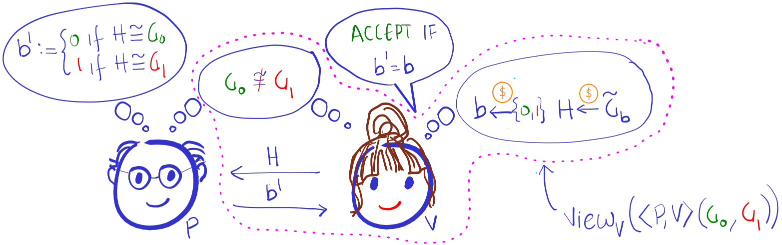


- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QNR**)



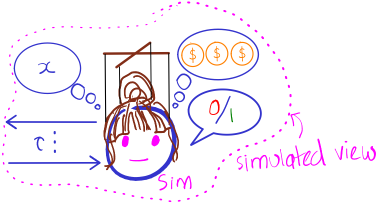
- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QNR**)

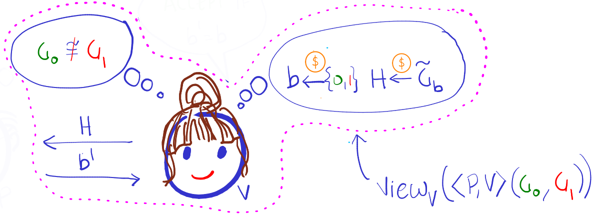




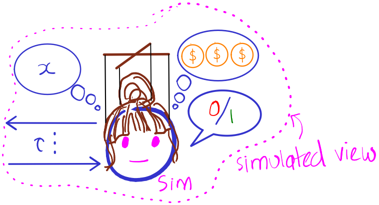
Recall from Last Lecture...

- *Interactive* proof (IP)
  - Compared to traditional "NP" proof
  - IP is powerful: IP for GNI
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for GNI (Exercise 3: QNR)

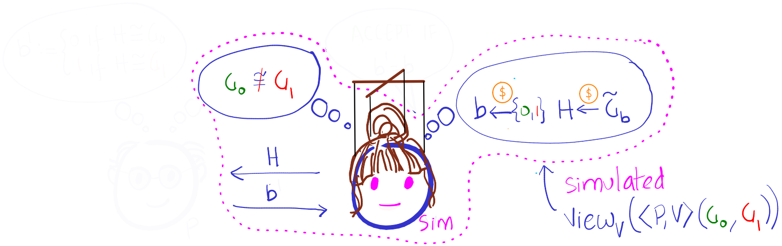




- *Interactive* proof (IP)
  - Compared to traditional "**NP**" proof
  - IP is powerful: IP for **GNI**

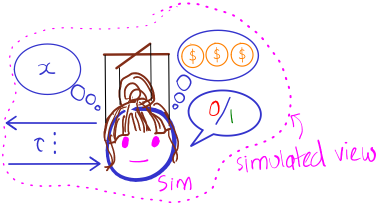


- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled "zero knowledge" via simulation paradigm
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QNR**)

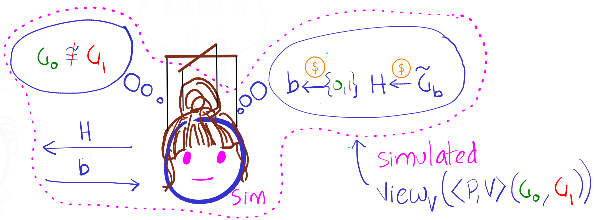


Recall from Last Lecture...

- *Interactive* proof (IP)
  - Compared to traditional “**NP**” proof
  - IP is powerful: IP for **GNI**
- Zero-knowledge proof
  - Knowledge vs. information
  - Modelled “zero knowledge” via simulation paradigm



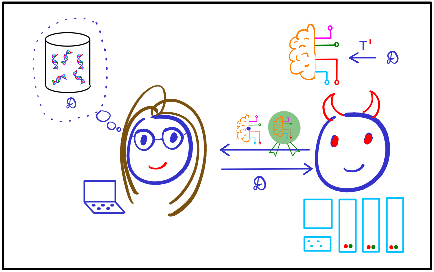
- *Honest-verifier* ZKP for **GNI** (Exercise 3: **QR**)



- *Honest-verifier* ZKP for **GI** (Exercise 4: **QR**)

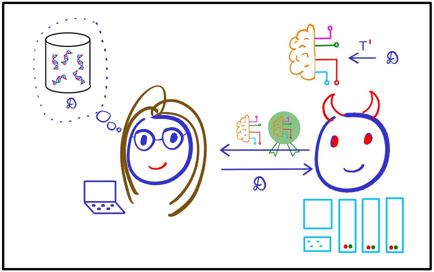
(ZK)IPs are Useful!

- Applications of IP: Verifiable outsourcing



(ZK)IPs are Useful!

- Applications of IP: Verifiable outsourcing

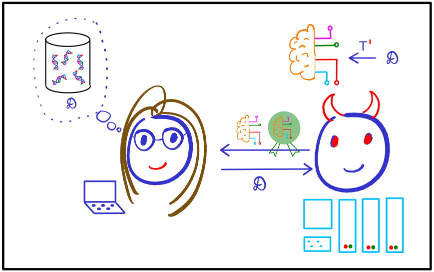


- Applications of ZKP:
  - Cryptocurrency: prove validity of a transaction without revealing information

    Zcash MONERO
  - Digital signatures: next lecture

(ZK)IPs are Useful!

- Applications of IP: Verifiable outsourcing



- Applications of ZKP:
  - Cryptocurrency: prove validity of a transaction without revealing information

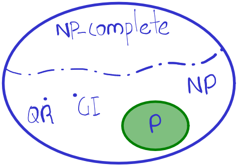
    Zcash MONERO
  - Digital signatures: next lecture
  - NIST is currently standardising ZKP (projects/pec/zkproof)

Plan for Today's Lecture...

- *Malicious-verifier* ZKP for GI

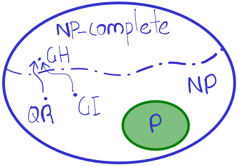
Plan for Today's Lecture...

- *Malicious-verifier* ZKP for **GI**



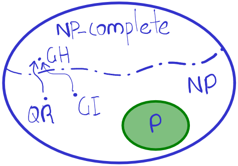
Plan for Today's Lecture...

- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle



Plan for Today's Lecture...

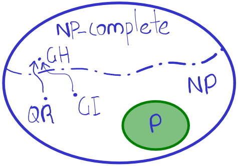
- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle

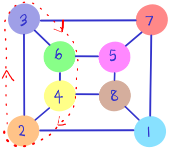


cycle that visits every vertex exactly once

- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle

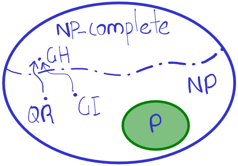
cycle that visits every vertex exactly once



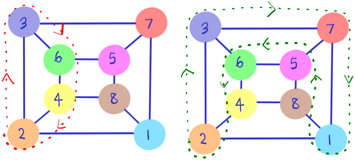


Plan for Today's Lecture...

- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle

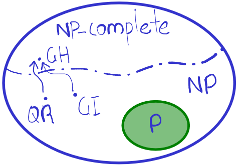


cycle that visits every vertex exactly once

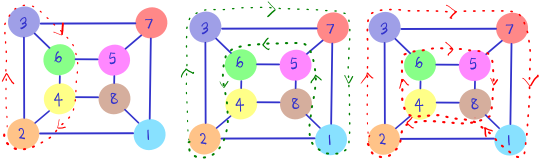


Plan for Today's Lecture...

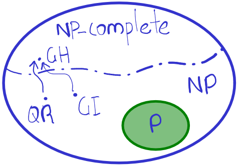
- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle



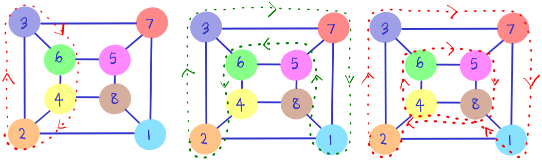
cycle that visits every vertex exactly once



- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle



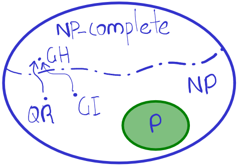
cycle that visits every vertex exactly once



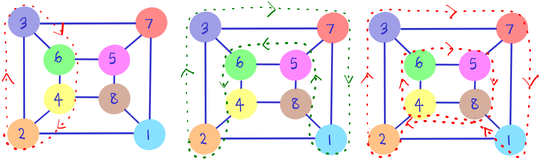
$\mathcal{L}_{GH} = \{G : G \text{ has a Hamiltonian cycle}\}$

Plan for Today's Lecture...

- *Malicious-verifier* ZKP for **GI**
- ZKP for all of **NP**
  - Blum's protocol for Graph Hamiltonicity (**GH**)
    - Given a graph G, decide whether it has a Hamiltonian cycle



cycle that visits every vertex exactly once



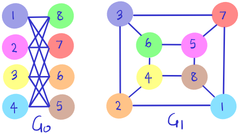
$\mathcal{L}_{GH} = \{G : G \text{ has a Hamiltonian cycle}\}$

- Commitment scheme

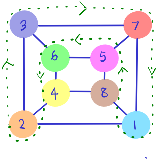
  - Digital analogues of lockers
  - OWP $\rightarrow$ (non-interactive) commitment scheme

Plan for Today's Lecture

1 Malicious-Verifier ZKP for Graph Isomorphism



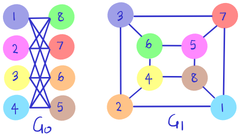
2 (Computational) ZKP for NP



3 Commitment Scheme

Plan for Today's Lecture

1 Malicious-Verifier ZKP for Graph Isomorphism



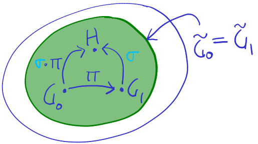
2 (Computational) ZKP for NP

3 Commitment Scheme

Recall Π_{GI}: Honest-Verifier ZK for GI...

Observation: transitivity of isomorphism

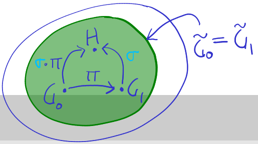
- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma\circ\pi}{\cong} H$



Recall Π_{GI}: Honest-Verifier ZK for GI...

Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma \circ \pi}{\cong} H$



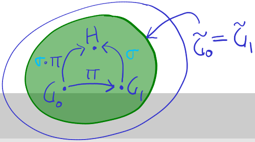
Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

1. P *"commits" by sending a random* H *s.t.* $G_1 \cong H$
2. *For* $b \leftarrow \{0, 1\}$, V *challenges* P *to "reveal"* $G_b \cong H$
3. V *accepts if the revealed permutation is valid*

Recall Π_{GI}: Honest-Verifier ZK for GI...

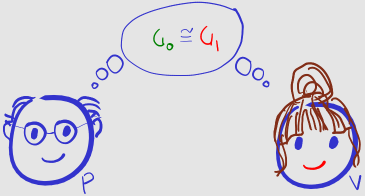
Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma\circ\pi}{\cong} H$



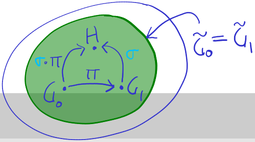
Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

1. P *"commits" by sending a random* H *s.t.* $G_1 \cong H$
2. *For* $b \leftarrow \{0,1\}$, V *challenges* P *to "reveal"* $G_b \cong H$
3. V *accepts if the revealed permutation is valid*



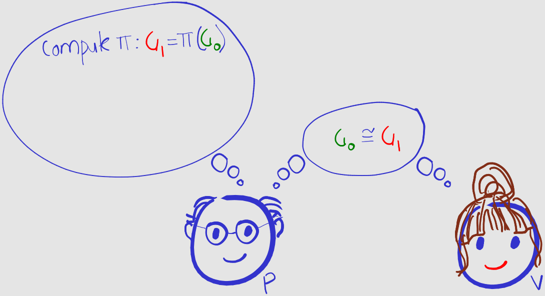
Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma \circ \pi}{\cong} H$



Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

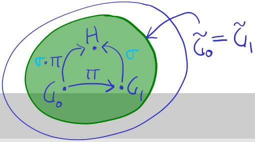
1. P *"commits" by sending a random* H *s.t.* $G_1 \cong H$
2. *For* $b \leftarrow \{0, 1\}$, V *challenges* P *to "reveal"* $G_b \cong H$
3. V *accepts if the revealed permutation is valid*



Recall Π_{GI}: Honest-Verifier ZK for GI...

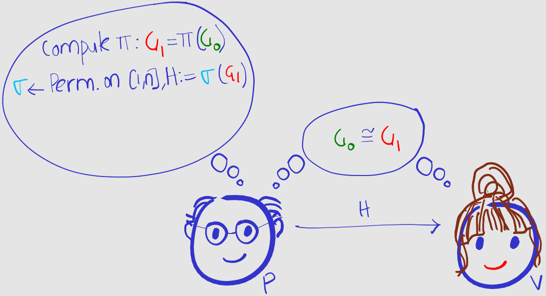
Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma\circ\pi}{\cong} H$



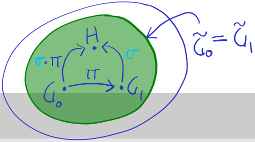
Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

1. P *"commits" by sending a random H s.t. $G_1 \cong H$*
2. *For $b \leftarrow \{0,1\}$,* V *challenges* P *to "reveal" $G_b \cong H$*
3. V *accepts if the revealed permutation is valid*



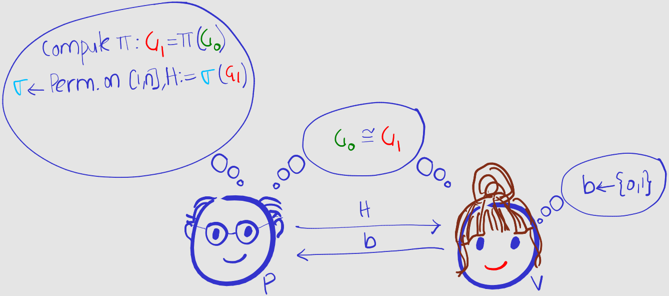
Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma\circ\pi}{\cong} H$



Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

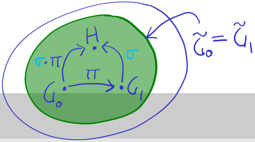
1. P *"commits" by sending a random* H *s.t.* $G_1 \cong H$
2. *For* $b \leftarrow \{0, 1\}$, V *challenges* P *to "reveal"* $G_b \cong H$
3. V *accepts if the revealed permutation is valid*



Recall Π_{GI}: Honest-Verifier ZK for GI...

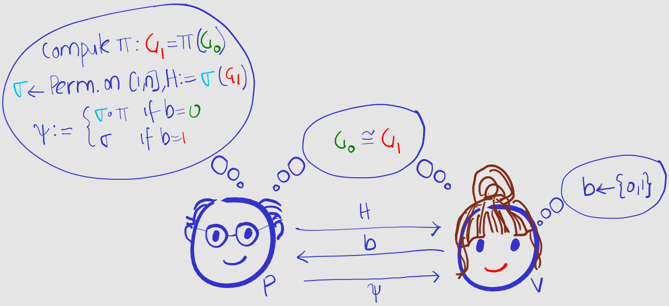
Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma\circ\pi}{\cong} H$



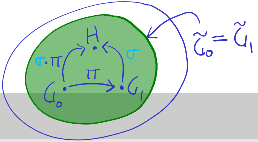
Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

1. P *"commits" by sending a random H s.t. $G_1 \cong H$*
2. *For $b \leftarrow \{0, 1\}$,* V *challenges* P *to "reveal" $G_b \cong H$*
3. V *accepts if the revealed permutation is valid*



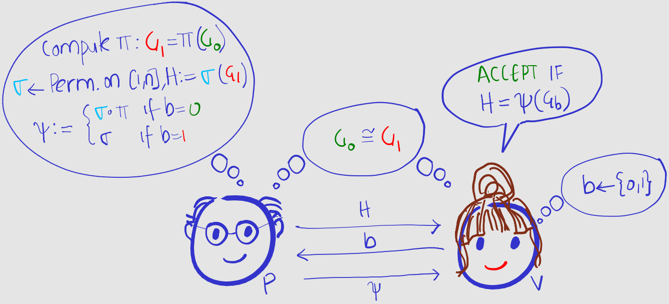
Observation: transitivity of isomorphism

- $G_0 \overset{\pi}{\cong} G_1 \Rightarrow$ if $G_1 \overset{\sigma}{\cong} H$ then $G_0 \overset{\sigma\circ\pi}{\cong} H$



Protocol 1 ($\Pi_{\mathsf{GI}} = (\mathsf{P}, \mathsf{V})$: IP for GI)

1. P *"commits" by sending a random* H *s.t.* $G_1 \cong H$
2. *For* $b \leftarrow \{0,1\}$, V *challenges* P *to "reveal"* $G_b \cong H$
3. V *accepts if the revealed permutation is valid*



Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can reveal on either challenge $\Rightarrow$ V always accepts $\Rightarrow \epsilon_c = 0$
- Soundness: $G_0 \ncong G_1 \Rightarrow$ for *any* H, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P^* can do is guess $b \Rightarrow \epsilon_s = 1/2$

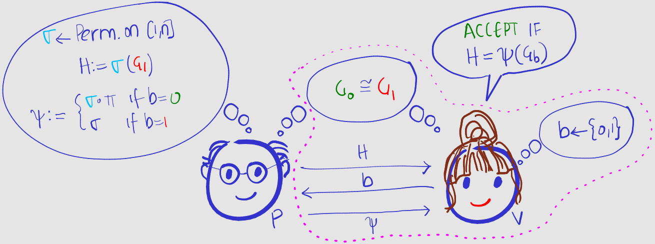
Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can reveal on either challenge $\Rightarrow$ V always accepts $\Rightarrow \epsilon_c = 0$
- Soundness: $G_0 \not\cong G_1 \Rightarrow$ for *any* H, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P* can do is guess $b \Rightarrow \epsilon_s = 1/2$
- Zero knowledge: sample *out of order* (info. vs knowledge)



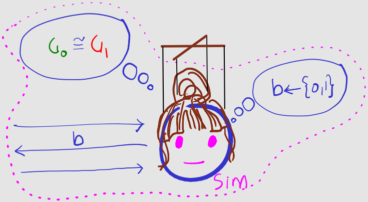
Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can reveal on either challenge $\Rightarrow$ V always accepts $\Rightarrow \epsilon_c = 0$
- Soundness: $G_0 \not\cong G_1 \Rightarrow$ for *any* H, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P^* can do is guess $b \Rightarrow \epsilon_s = 1/2$
- Zero knowledge: sample *out of order* (info. vs knowledge)



$\square$

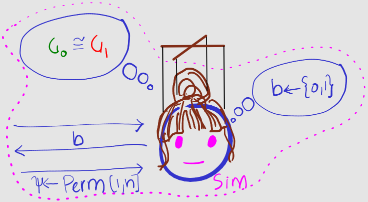
Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can reveal on either challenge $\Rightarrow$ V always accepts $\Rightarrow \epsilon_c = 0$
- Soundness: $G_0 \not\cong G_1 \Rightarrow$ for *any* H, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P^* can do is guess $b \Rightarrow \epsilon_s = 1/2$
- Zero knowledge: sample *out of order* (info. vs knowledge)



$\square$

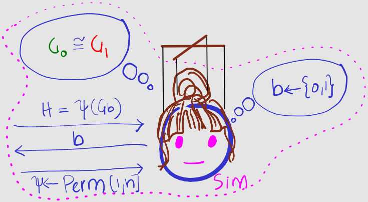
Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can reveal on either challenge $\Rightarrow$ V always accepts $\Rightarrow \epsilon_c = 0$
- Soundness: $G_0 \not\cong G_1 \Rightarrow$ for *any* H, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P^* can do is guess $b \Rightarrow \epsilon_s = 1/2$
- Zero knowledge: sample *out of order* (info. vs knowledge)



□

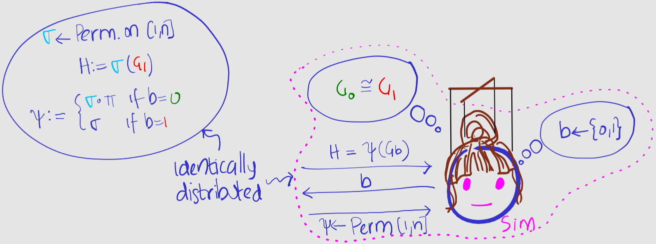
Recall Π_{GI}: Honest-Verifier ZK for GI...

Theorem 1

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

Proof.

- Completeness: $G_0 \cong G_1 \Rightarrow$ P can reveal on either challenge $\Rightarrow$ V always accepts $\Rightarrow \epsilon_c = 0$
- Soundness: $G_0 \not\cong G_1 \Rightarrow$ for *any* H, $G_0 \cong H$ *and* $G_1 \cong H$ *cannot both hold* $\Rightarrow$ best P^* can do is guess $b \Rightarrow \epsilon_s = 1/2$
- Zero knowledge: sample *out of order* (info. vs knowledge)



□

What about Malicious Verifiers? 

Defintion 1 ((*Malicious-Verifier*) Perfect ZK)

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

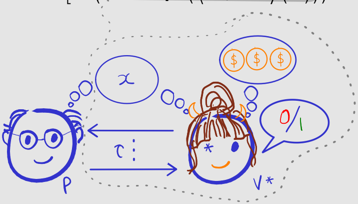
$$\Pr[\mathsf{D}(\mathsf{View}_{\mathsf{V}^*}(\langle \mathsf{P}, \mathsf{V}^* \rangle(x))) = 1] - \Pr[\mathsf{D}(\mathsf{Sim}^{\mathsf{V}^*}(x)) = 1]$$

What about Malicious Verifiers? 

Defintion 1 ((*Malicious-Verifier*) Perfect ZK)

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

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

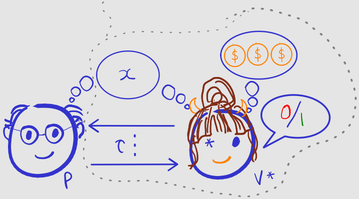


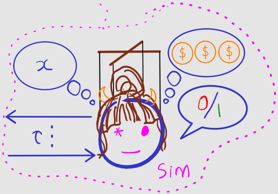
What about Malicious Verifiers?

Defintion 1 ((*Malicious-Verifier*) Perfect ZK)

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

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



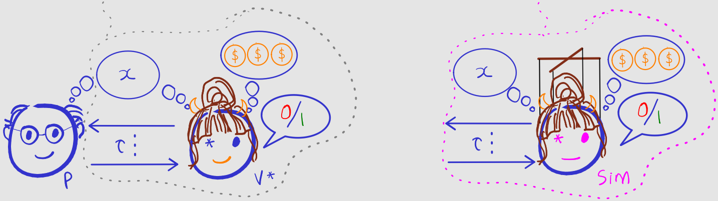


What about Malicious Verifiers?

Defintion 1 ((*Malicious-Verifier*) Perfect ZK)

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

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



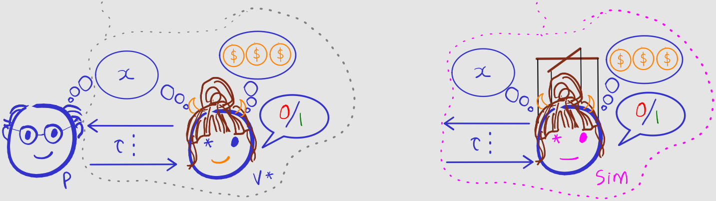
What happens if we use honest-verifier simulator Sim now?

What about Malicious Verifiers? 

Defintion 1 ((*Malicious-Verifier*) Perfect ZK)

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

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



What happens if we use honest-verifier simulator Sim now?

- The distribution of b generated by V^* may not be uniform
- It could depend arbitrarily on P's message H

Π_{GI} Works Also For Malicious Verifiers!...

Just need a different Sim

Theorem 2

Π_{GI} is a malicious-verifier perfect ZKP for $\mathcal{L}_{\mathsf{GI}}$

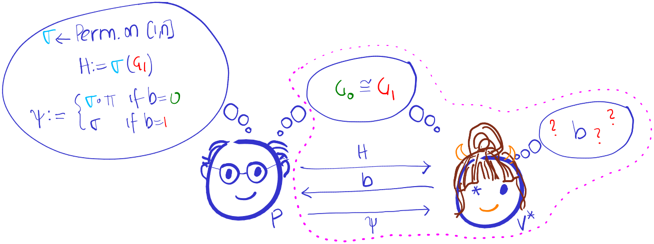
Π_{GI} Works Also For Malicious Verifiers!...

Just need a different Sim

Theorem 2

Π_{GI} *is a malicious-verifier perfect ZKP for* $\mathcal{L}_{\mathsf{GI}}$

Proof (of ZK) ?



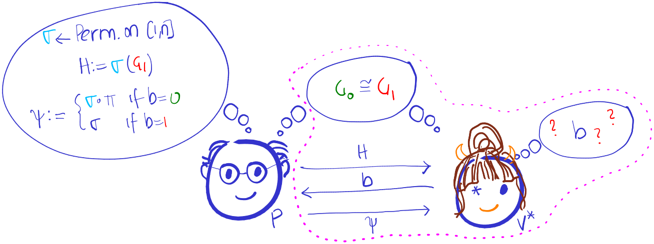
Π_{GI} Works Also For Malicious Verifiers!

Just need a different Sim

Theorem 2

Π_{GI} *is a malicious-verifier perfect ZKP for* $\mathcal{L}_{\mathsf{GI}}$

Proof (of ZK) ?



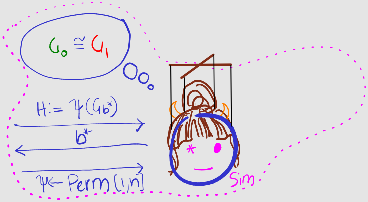
Π_{GI} Works Also For Malicious Verifiers!...

Just need a different Sim

Theorem 2

Π_{GI} is a malicious-verifier perfect ZKP for $\mathcal{L}_{\mathsf{GI}}$

Proof (of ZK) Idea: **Sim** invokes V^*!



- New simulator $\mathsf{Sim}^{\mathsf{V}^*}$: repeat till required
  1. Sample random $H \overset{\psi}{\cong} G_{b^*}$ for $b^* \leftarrow \{0,1\}$
  2. Invoke V^* on H to obtain challenge b (with fresh random coins)
  3. If $b^* = b$ output $((G_0, G_1), (H, b, \psi))$ $\square$

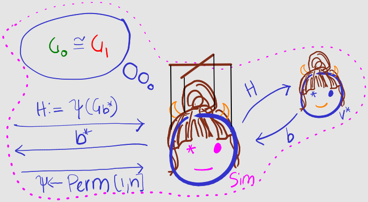
Π_{GI} Works Also For Malicious Verifiers!...

Just need a different Sim

Theorem 2

Π_{GI} is a malicious-verifier perfect ZKP for $\mathcal{L}_{\mathsf{GI}}$

Proof (of ZK) Idea: Sim invokes V^*!



- New simulator $\mathsf{Sim}^{\mathsf{V}^*}$: repeat till required
  1. Sample random $H \overset{\psi}{\cong} G_{b^*}$ for $b^* \leftarrow \{0, 1\}$
  2. Invoke V^* on H to obtain challenge b (with fresh random coins)
  3. If $b^* = b$ output $((G_0, G_1), (H, b, \psi))$ $\square$

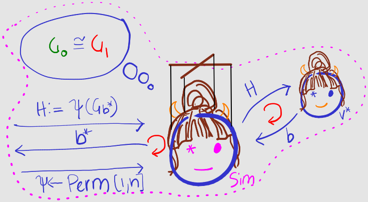
Π_{GI} Works Also For Malicious Verifiers!

Just need a different Sim

Theorem 2

Π_{GI} *is a malicious-verifier perfect ZKP for* $\mathcal{L}_{\mathsf{GI}}$

Proof (of ZK) Idea: Sim invokes V^*!



- New simulator $\mathsf{Sim}^{\mathsf{V}^*}$: repeat till required
  1. Sample random $H \overset{\psi}{\cong} G_{b^*}$ for $b^* \leftarrow \{0,1\}$
  2. Invoke V^* on H to obtain challenge b (with fresh random coins)
  3. If $b^* = b$ output $((G_0, G_1), (H, b, \psi))$ $\square$

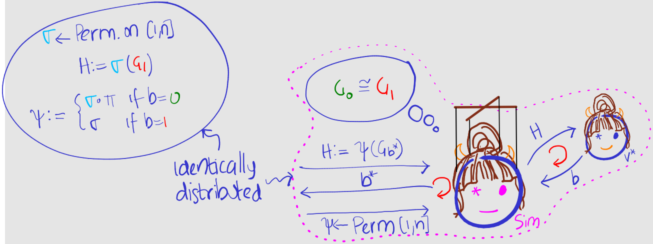
Π_{GI} Works Also For Malicious Verifiers!

Just need a different Sim

Theorem 2

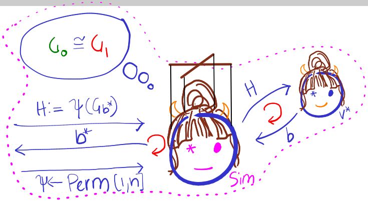
Π_{GI} *is a malicious-verifier perfect ZKP for* $\mathcal{L}_{\mathsf{GI}}$

Proof (of ZK) Idea: **Sim** invokes V^*!



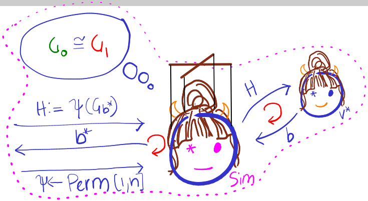
- New simulator $\mathsf{Sim}^{\mathsf{V}^*}$: repeat till required
  1. Sample random $H \overset{\psi}{\cong} G_{b^*}$ for $b^* \leftarrow \{0,1\}$
  2. Invoke V^* on H to obtain challenge b (with fresh random coins)
  3. If $b^* = b$ output $((G_0, G_1), (H, b, \psi))$ $\square$

Π_{GI} Works Also For Malicious Verifiers!...



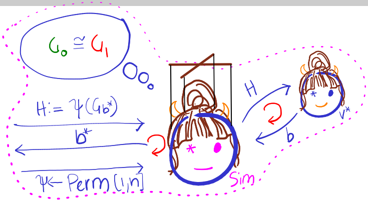
Why is b independent of b^*?

Π_{GI} Works Also For Malicious Verifiers!...



- Why is b independent of b^*? H hides b^*
- What is the run-time of the new simulator $\mathsf{Sim}^{\mathsf{V}^*}$?

Π_{GI} Works Also For Malicious Verifiers!...

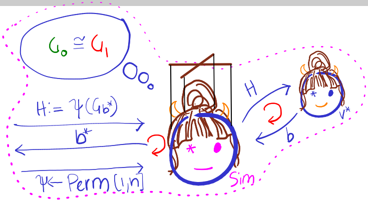


- Why is b independent of b^*? H hides b^*
- What is the run-time of the new simulator $\mathsf{Sim}^{\mathsf{V}^*}$?
  - In expectation: polynomial time
  - Worst case: exponential time

Exercise 1

Can you come up with a strict PPT simulator?

Π_{GI} Works Also For Malicious Verifiers!...



Why is b independent of b^*? H hides b^*

What is the run-time of the new simulator $\mathsf{Sim}^{\mathsf{V}^*}$?

- In expectation: polynomial time
- Worst case: exponential time

Exercise 1

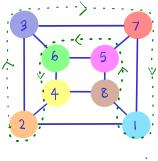
Can you come up with a strict PPT simulator?

Exercise 2

1. *Design malicious-verifier perfect ZKP for $\mathcal{L}_{\mathsf{QR}}$*
2. *Think about malicious-verifier perfect ZKP for $\mathcal{L}_{\mathsf{GNI}}$*
   - *Hint: you need to somehow use Π_{GI} as sub-routine*

Plan for Today's Lecture

1 Malicious-Verifier ZKP for Graph Isomorphism

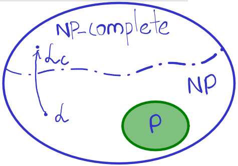
2 (Computational) ZKP for NP 

3 Commitment Scheme

ZKP for Any Problem in NP

Claim 1

ZKP for an NP*-complete language* $\mathcal{L}_c$ *implies ZKP for any* $\mathcal{L} \in$ NP

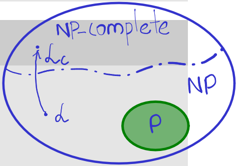


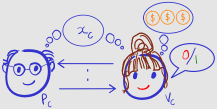
ZKP for Any Problem in NP

Claim 1

ZKP for an NP*-complete language* $\mathcal{L}_c$ *implies ZKP for any* $\mathcal{L} \in$ NP

Construction 1 $(\Pi_c = (P_c, V_c) \rightarrow \Pi = (P, V))$





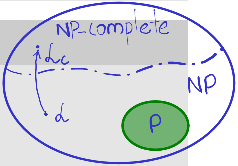
1. *Encode* $x \in \mathcal{L}$ *by Karp-reducing to* $x_c \in \mathcal{L}_c$
2. *Use ZKP for* $\mathcal{L}_c$ *on* x_c

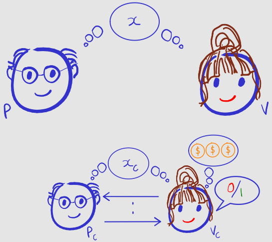
ZKP for Any Problem in NP

Claim 1

ZKP for an NP*-complete language* $\mathcal{L}_c$ *implies ZKP for any* $\mathcal{L} \in$ NP

Construction 1 ($\Pi_c = (P_c, V_c) \to \Pi = (P, V)$)





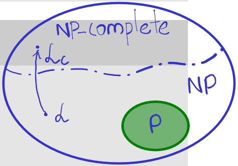
1. *Encode* $x \in \mathcal{L}$ *by Karp-reducing to* $x_c \in \mathcal{L}_c$
2. *Use ZKP for* $\mathcal{L}_c$ *on* x_c

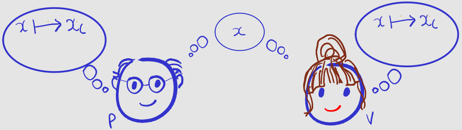
ZKP for Any Problem in NP

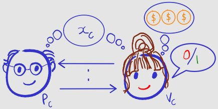
Claim 1

ZKP for an NP*-complete language* $\mathcal{L}_c$ *implies ZKP for any* $\mathcal{L} \in$ NP

Construction 1 ($\Pi_c = (P_c, V_c) \to \Pi = (P, V)$)







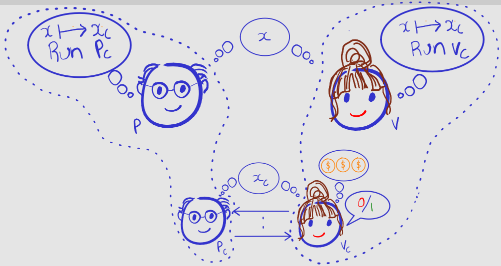
1. *Encode* $x \in \mathcal{L}$ *by Karp-reducing to* $x_c \in \mathcal{L}_c$
2. *Use ZKP for* $\mathcal{L}_c$ *on* x_c

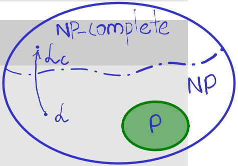
ZKP for Any Problem in NP

Claim 1

ZKP for an NP*-complete language* $\mathcal{L}_c$ *implies ZKP for any* $\mathcal{L} \in$ NP

Construction 1 ($\Pi_c = (P_c, V_c) \to \Pi = (P, V)$)





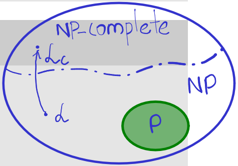
1. *Encode* $x \in \mathcal{L}$ *by Karp-reducing to* $x_c \in \mathcal{L}_c$
2. *Use ZKP for* $\mathcal{L}_c$ *on* x_c

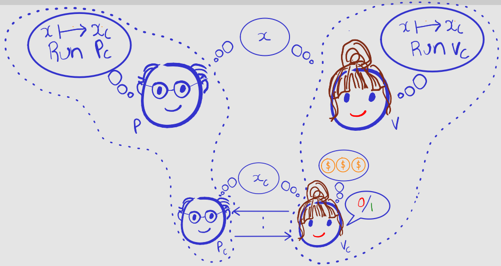
ZKP for Any Problem in NP

Claim 1

ZKP for an NP*-complete language* $\mathcal{L}_c$ *implies ZKP for any* $\mathcal{L} \in$ NP

Construction 1 ($\Pi_c = (P_c, V_c) \to \Pi = (P, V)$)





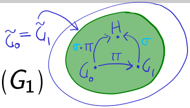
1. *Encode* $x \in \mathcal{L}$ *by Karp-reducing to* $x_c \in \mathcal{L}_c$
2. *Use ZKP for* $\mathcal{L}_c$ *on* x_c

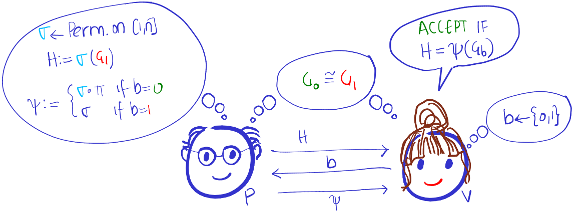
Exercise 3

Show that if Π_c *is a ZKP for* $\mathcal{L}_c$ *then* Π *is a ZKP for* $\mathcal{L}$

Let's Construct ZKP for Graph Hamiltonicity...

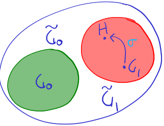
- Let's recall/rephrase Π_{GI}:
  - Honest P "commits" to G_0 *and* G_1 by sending $H = \sigma(G_1)$

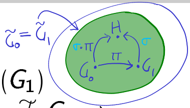


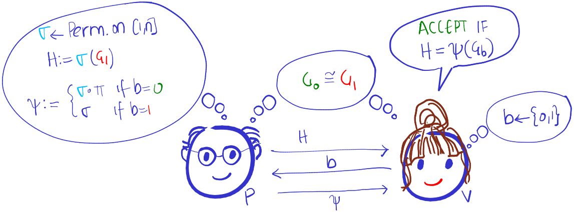


Let's Construct ZKP for Graph Hamiltonicity...

- Let's recall/rephrase Π_{GI}:
  - Honest P "commits" to G_0 *and* G_1 by sending $H = \sigma(G_1)$
  - Soundness: commitment H is "perfectly binding" if $G_0 \not\cong G_1 \Rightarrow$ malicious P* can commit to only one of G_0 *or* G_1 in advance

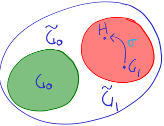


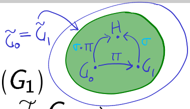




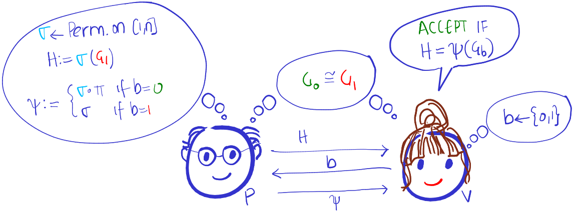
Let's Construct ZKP for Graph Hamiltonicity...

- Let's recall/rephrase Π_{GI}:



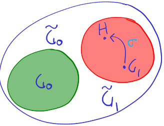


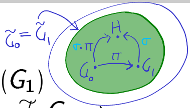
  - Honest P "commits" to G_0 *and* G_1 by sending $H = \sigma(G_1)$
  - Soundness: commitment H is "perfectly binding" if $G_0 \not\cong G_1 \Rightarrow$ malicious P* can commit to only one of G_0 *or* G_1 in advance
  - ZK: commitment is "perfectly hiding" if $G_0 \cong G_1 \Rightarrow H$ hides information about G_0/G_1

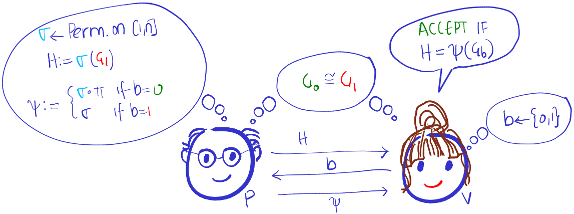


Let's Construct ZKP for Graph Hamiltonicity...

- Let's recall/rephrase Π_{GI}:
  - Honest P "commits" to G_0 *and* G_1 by sending $H = \sigma(G_1)$
  - Soundness: commitment H is "perfectly binding" if $G_0 \not\cong G_1 \Rightarrow$ malicious P* can commit to only one of G_0 *or* G_1 in advance
  - ZK: commitment is "perfectly hiding" if $G_0 \cong G_1 \Rightarrow H$ hides information about G_0/G_1
  - Possible because of GI's structure: isomorphisms are transitive

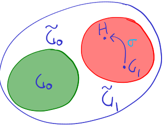


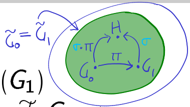


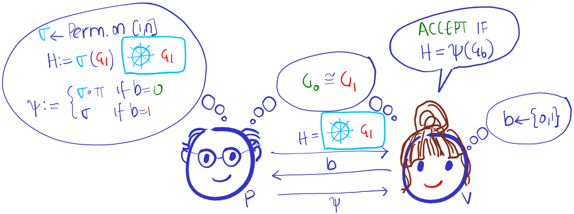


Let's Construct ZKP for Graph Hamiltonicity...

- Let's recall/rephrase Π_{GI}:
  - Honest P "commits" to G_0 *and* G_1 by sending $H = \sigma(G_1)$
  - Soundness: commitment H is "perfectly binding" if $G_0 \not\cong G_1 \Rightarrow$ malicious P^* can commit to only one of G_0 *or* G_1 in advance
  - ZK: commitment is "perfectly hiding" if $G_0 \cong G_1 \Rightarrow H$ hides information about G_0/G_1
  - Possible because of GI's structure: isomorphisms are transitive



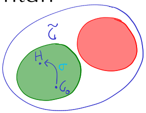




- Physical analogy: H acts as a secure "locker"
  1. *Hides* its contents from the verifier V
  2. *Binds* P^* by forcing it to store either G_0 or G_1 *before* seeing challenge b

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers 

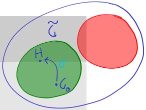
Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

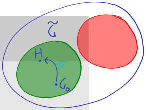
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

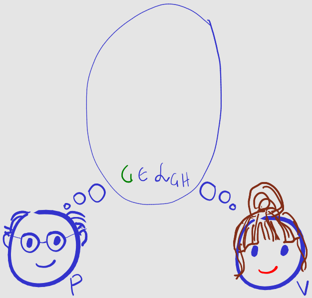


Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for **GH**)

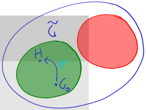


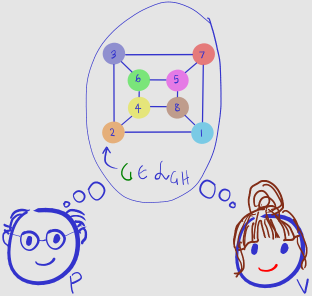


Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)

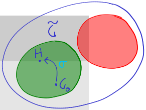




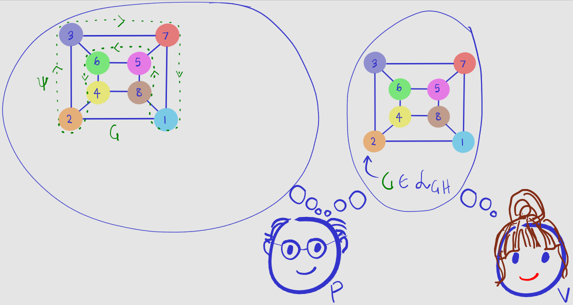
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

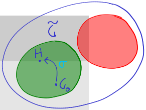


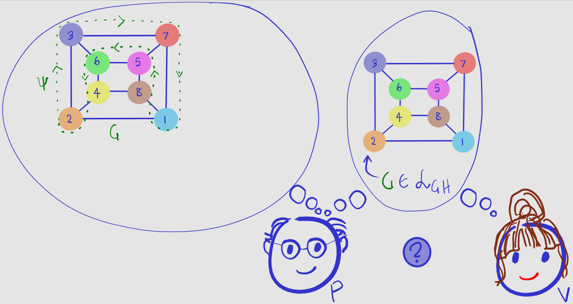
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

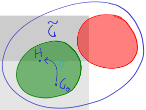
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)



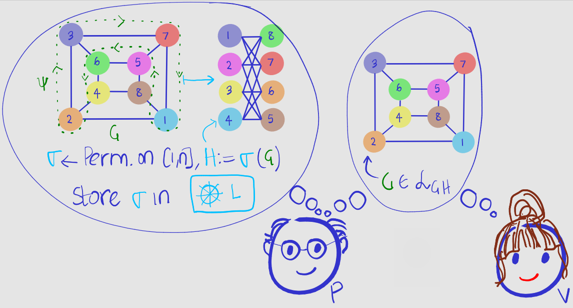


Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

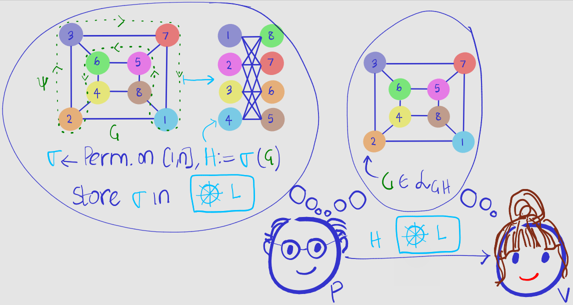


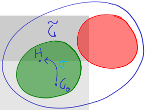
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for **GH**)



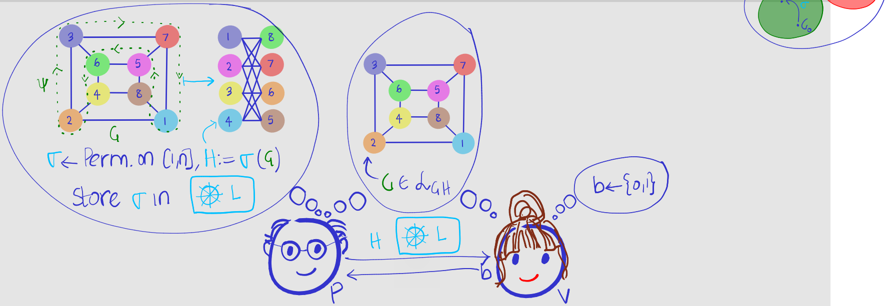


Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

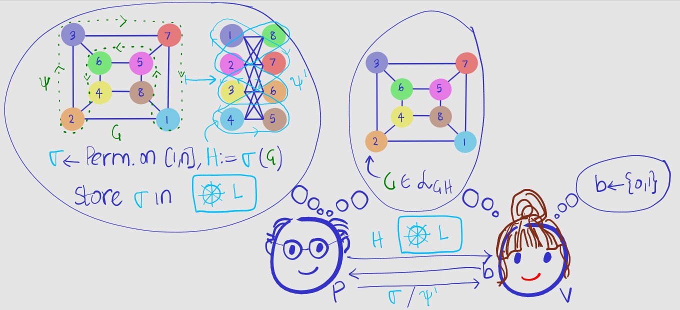
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)



Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

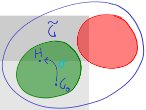
Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)

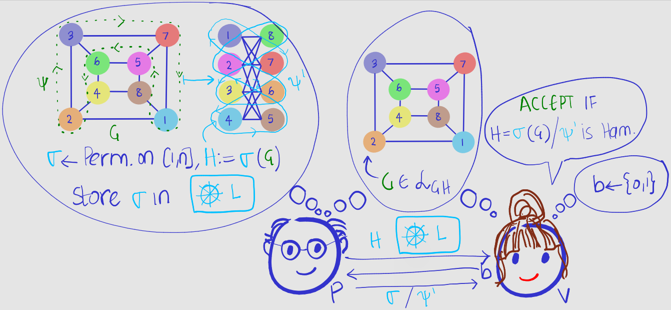


Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for **GH**)

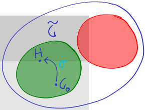


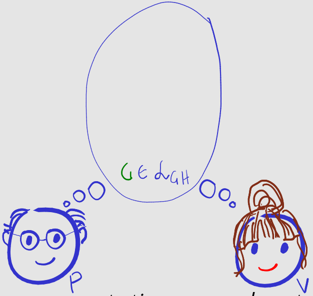
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian

Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)

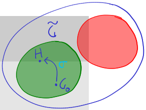




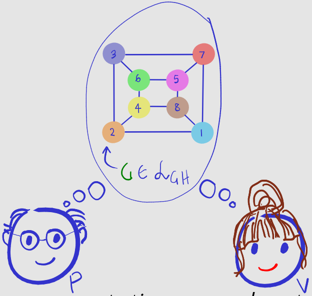
1. P *samples random permutation σ and puts it in locker L*
2. P *commits by sending L and $H := \sigma(G)$ to* V
3. V *challenges* P *to reveal* 0) *σ by opening L or* 1) *Hamiltonian cycle $\sigma(\psi)$ in H*

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)

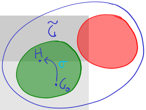


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

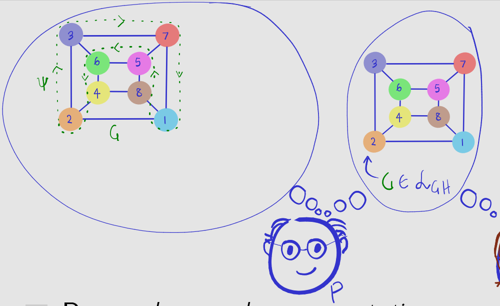
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)

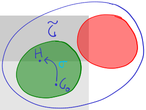


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

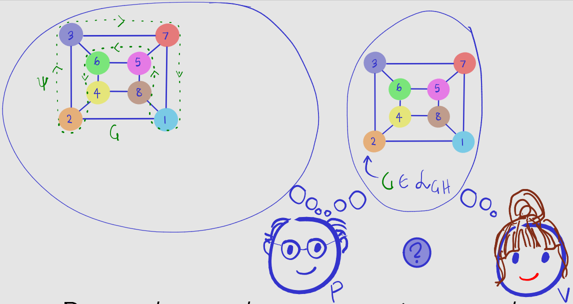
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



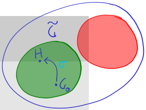
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)



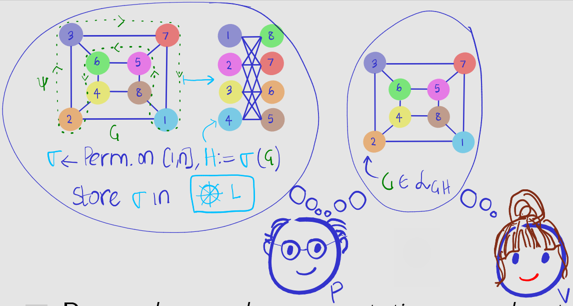
1. P *samples random permutation σ and puts it in locker L*
2. P *commits by sending L and $H := \sigma(G)$ to* V
3. V *challenges* P *to reveal* 0) *σ by opening L or* 1) *Hamiltonian cycle $\sigma(\psi)$ in H*

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

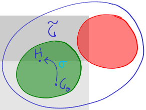


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

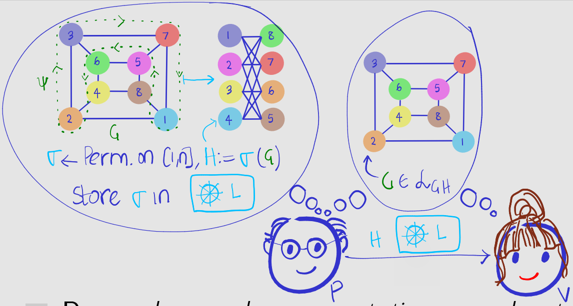
Let's Construct ZKP for Graph Hamiltonicity...

Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



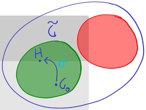
Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)



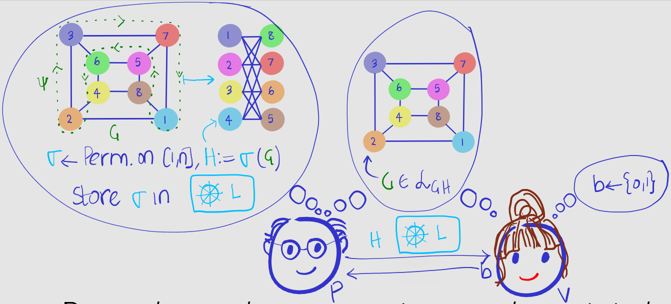
1. P *samples random permutation σ and puts it in locker L*
2. P *commits by sending L and $H := \sigma(G)$ to* V
3. V *challenges* P *to reveal* 0) *σ by opening L or* 1) *Hamiltonian cycle $\sigma(\psi)$ in H*

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



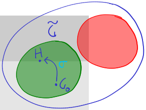
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)



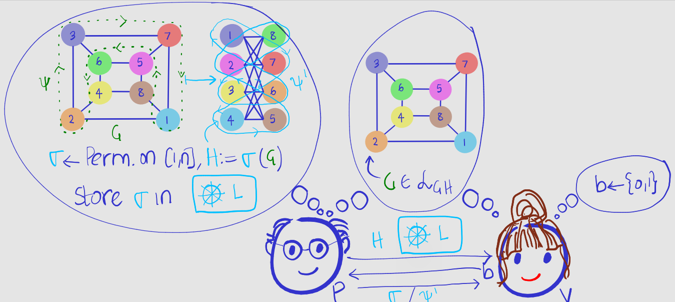
1. P *samples random permutation σ and puts it in locker L*
2. P *commits by sending L and $H := \sigma(G)$ to* V
3. V *challenges* P *to reveal* 0*)* σ *by opening L or* 1*) Hamiltonian cycle $\sigma(\psi)$ in H*

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



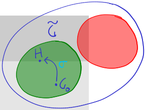
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)



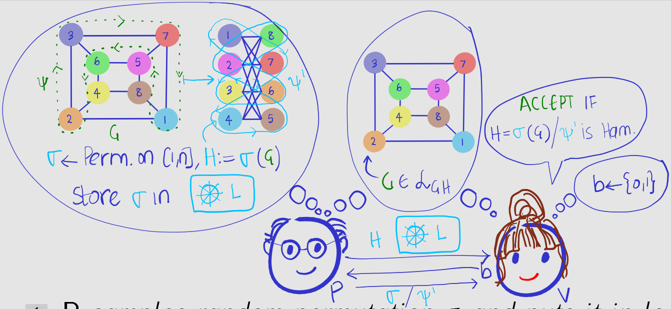
1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



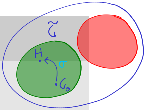
Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)



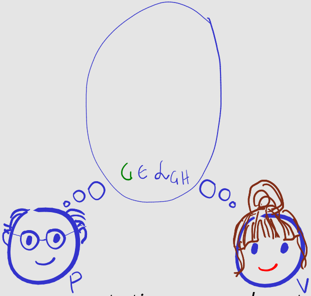
1. P *samples random permutation σ and puts it in locker L*
2. P *commits by sending L and $H := \sigma(G)$ to* V
3. V *challenges* P *to reveal* 0) *σ by opening L or* 1) *Hamiltonian cycle $\sigma(\psi)$ in H*

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

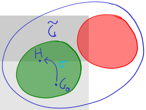


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

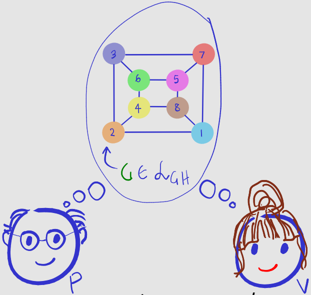
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

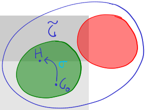


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

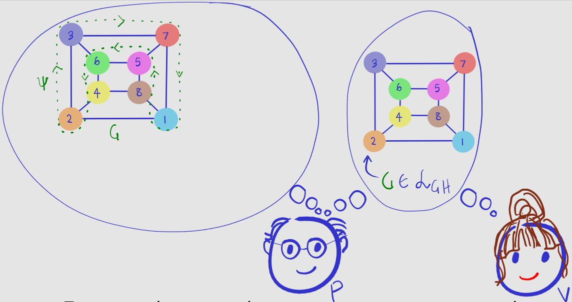
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for **GH**)

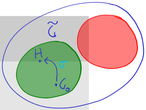


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

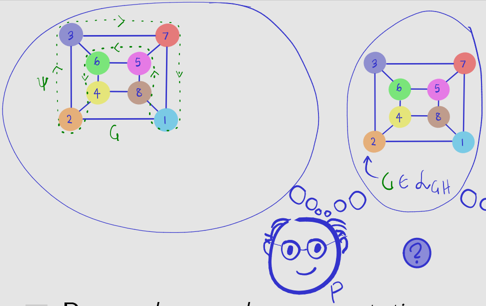
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\simeq} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for **GH**)

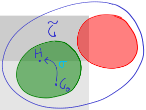


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

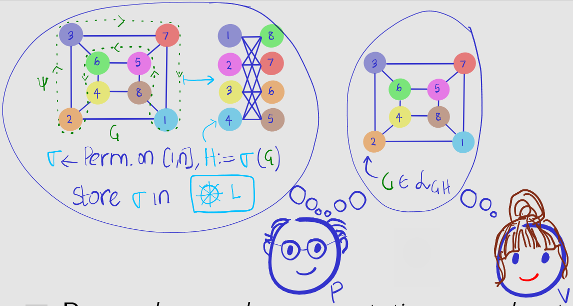
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): First attempt at ZKP for GH)

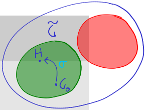


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

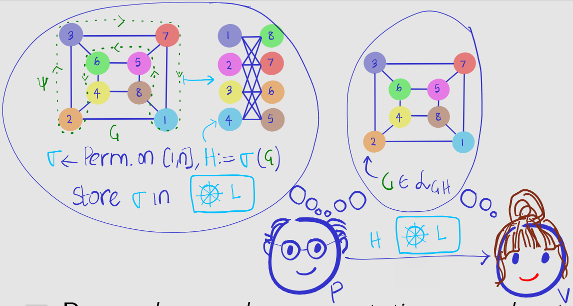
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

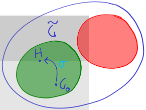


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

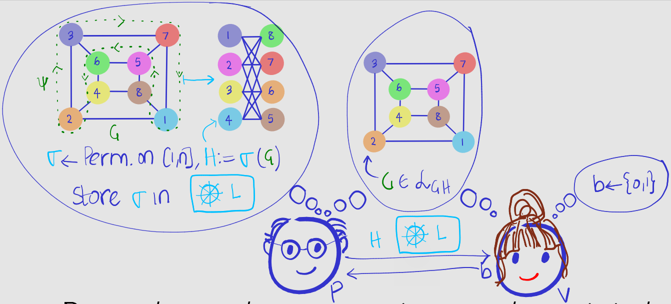
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\cong} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for $\mathbf{GH}$)

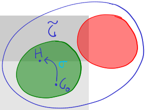


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

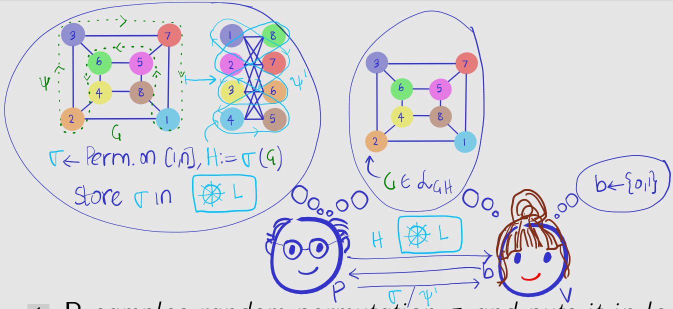
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\simeq} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

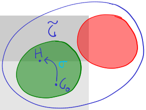


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

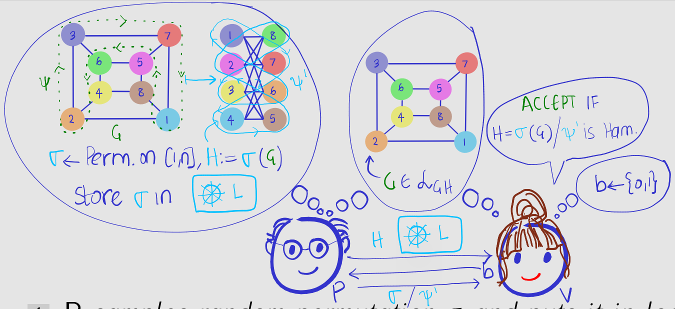
- Problem: not clear if zero knowledge. How to simulate?

Let's Construct ZKP for Graph Hamiltonicity... Using Lockers

Observation: G Hamiltonian and $G \overset{\sigma}{\simeq} H$ then H Hamiltonian



Protocol 2 ($\Pi_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: First attempt at ZKP for GH)

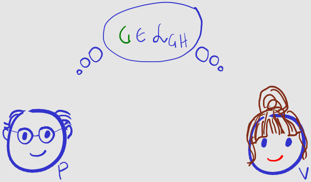


1. P *samples random permutation* σ *and puts it in locker* L
2. P *commits by sending* L *and* $H := \sigma(G)$ *to* V
3. V *challenges* P *to reveal* 0) σ *by opening* L *or* 1) *Hamiltonian cycle* $\sigma(\psi)$ *in* H

- Problem: not clear if zero knowledge. How to simulate?

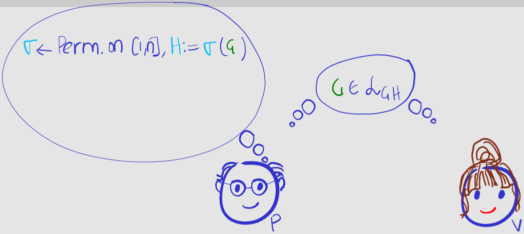
Let's Construct ZKP for Graph Hamiltonicity...

Protocol 3 ($\Pi'_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: Blum's IP for GH)



Let's Construct ZKP for Graph Hamiltonicity...

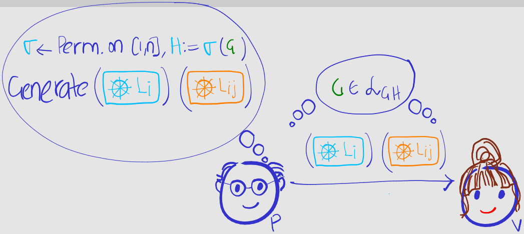
Protocol 3 ($\Pi'_{\mathbf{GH}} = (\mathsf{P}, \mathsf{V})$: Blum's IP for $\mathbf{GH}$)



1. P *samples random permutation* σ *and sets* $H := \sigma(G)$

Let's Construct ZKP for Graph Hamiltonicity...

Protocol 3 ($\Pi'_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$): Blum's IP for GH)



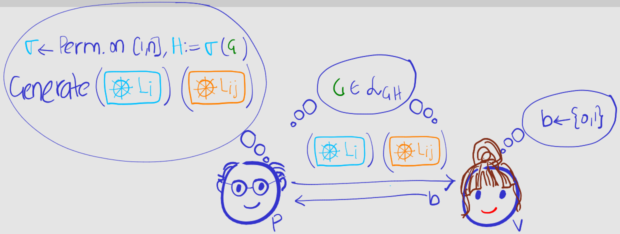
1. P *samples random permutation σ and sets $H := \sigma(G)$*
2. P *commits by sending σ and $H := \sigma(G)$ in lockers to* V

- *Lockers $(L_1, \ldots, L_n)$, where L_i stores $\sigma(i)$*

- *Lockers $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$ store H's adjacency matrix*

Let's Construct ZKP for Graph Hamiltonicity...

Protocol 3 ($\Pi'_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: Blum's IP for GH)



1. P *samples random permutation σ and sets $H := \sigma(G)$*
2. P *commits by sending σ and $H := \sigma(G)$ in lockers to* V

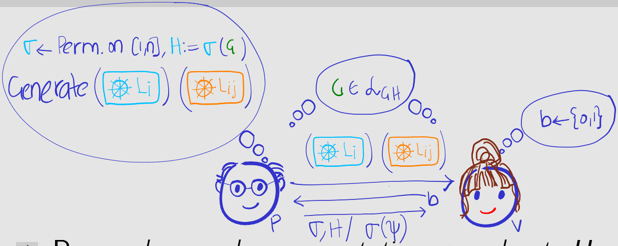
- *Lockers $(L_1, \ldots, L_n)$, where L_i stores $\sigma(i)$*

- *Lockers $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$ store H's adjacency matrix*

3. V *challenges* P *to reveal either* 0) *all lockers; or* 1) *lockers $L_{i,j}, L_{j,k}, \cdots, L_{\ell,i}$ corresponding to Ham. cycle $\sigma(\psi)$ in H*

Let's Construct ZKP for Graph Hamiltonicity...

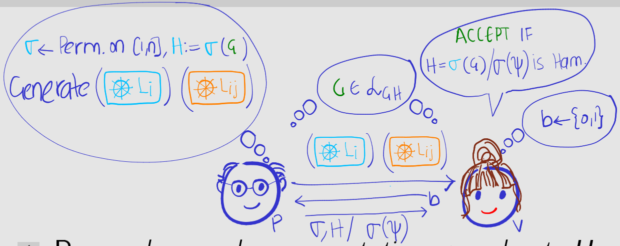
Protocol 3 ($\Pi'_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: Blum's IP for GH)



1. P *samples random permutation* σ *and sets* $H := \sigma(G)$
2. P *commits by sending* σ *and* $H := \sigma(G)$ *in lockers to* V
   - *Lockers* $(L_1, \ldots, L_n)$, *where* L_i *stores* $\sigma(i)$
   - *Lockers* $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$ *store* H*'s adjacency matrix*
3. V *challenges* P *to reveal either* 0) *all lockers; or* 1) *lockers* $L_{i,j}, L_{j,k}, \cdots, L_{\ell,i}$ *corresponding to Ham. cycle* $\sigma(\psi)$ *in* H

Let's Construct ZKP for Graph Hamiltonicity...

Protocol 3 ($\Pi'_{\mathsf{GH}} = (\mathsf{P}, \mathsf{V})$: Blum's IP for $\mathbf{GH}$)



1. P *samples random permutation* σ *and sets* $H := \sigma(G)$
2. P *commits by sending* σ *and* $H := \sigma(G)$ *in lockers to* V

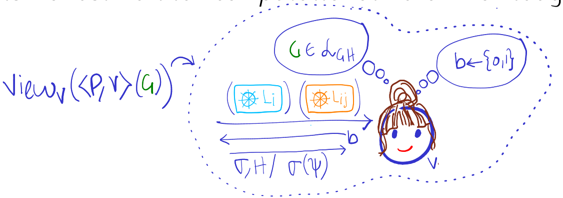
- *Lockers* $(L_1, \ldots, L_n)$, *where* L_i *stores* $\sigma(i)$

- *Lockers* $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$ *store* H*'s adjacency matrix*

3. V *challenges* P *to reveal either* 0) *all lockers; or* 1) *lockers* $L_{i,j}, L_{j,k}, \cdots, L_{\ell,i}$ *corresponding to Ham. cycle* $\sigma(\psi)$ *in* H
4. V *accepts if* 0) $H = \sigma(G)$ *or* 1) $L_{i,j}, L_{j,k}, \cdots, L_{\ell,i}$ *correspond to a Ham. cycle.*

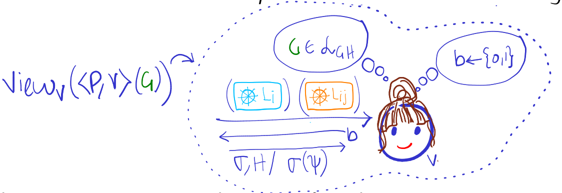
Π'_{GH} is Computational ZKP for Graph Hamiltonicity...

- Soundness: locker binding $\Rightarrow$ Π'_{GH} is sound
- Zero-knowledge: locker "computationally" hides its content $\Rightarrow$ Π'_{GH} is honest-verifier *computational* zero-knowledge for $\mathcal{L}_{\mathsf{GH}}$



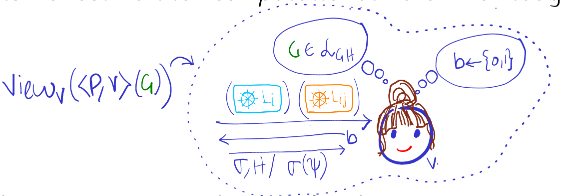
Π'_{GH} is Computational ZKP for Graph Hamiltonicity...

- Soundness: locker binding $\Rightarrow$ Π'_{GH} is sound
- Zero-knowledge: locker "computationally" hides its content $\Rightarrow$ Π'_{GH} is honest-verifier *computational* zero-knowledge for $\mathcal{L}_{\mathsf{GH}}$

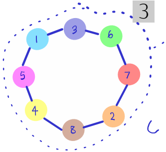


- Simulator: again, sample out of order
  1. Sample random $b \leftarrow \{0, 1\}$
  2. If $b = 0$
     - Sample random permutation σ and set $H := \sigma(G)$
     - Prepare lockers $(L_1, \ldots, L_n)$ and $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$ as in protocol

- Soundness: locker binding $\Rightarrow$ Π'_{GH} is sound
- Zero-knowledge: locker "computationally" hides its content $\Rightarrow$ Π'_{GH} is honest-verifier *computational* zero-knowledge for $\mathcal{L}_{\mathsf{GH}}$



- Simulator: again, sample out of order
  1. Sample random $b \leftarrow \{0,1\}$
  2. If $b = 0$
     - Sample random permutation σ and set $H := \sigma(G)$
     - Prepare lockers $(L_1, \ldots, L_n)$ and $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$ as in protocol
  3. If $b = 1$
     - Sample random cycle C over $[1, n]$
     - Leave lockers $(L_1, \ldots, L_n)$ empty and store C's adjacency matrix in $\left(L_{i,j}\right)_{(i,j)\in\binom{n}{2}}$



Π'_{GH} is Computational ZKP for Graph Hamiltonicity...

Exercise 4

Describe the simulator for malicious-verifier ZK for Π'_{GH}

Exercise 5

Think of ZKP for other NP*-complete problems like* $n \times n$ *Sudoku and graph three-colouring*

Plan for Today's Lecture

1 Malicious-Verifier ZKP for Graph Isomorphism

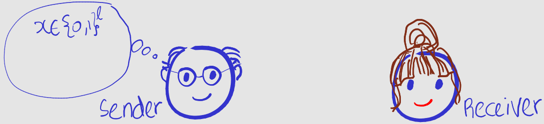
2 (Computational) ZKP for NP

3 Commitment Scheme

Commitment Schemes are Digital Lockers

Defintion 2

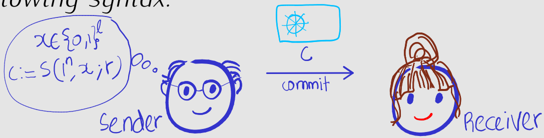
A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



Commitment Schemes are Digital Lockers

Defintion 2

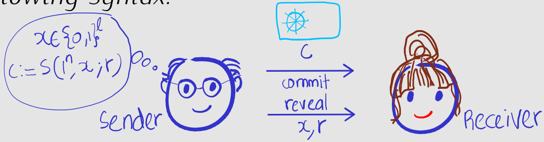
A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



Commitment Schemes are Digital Lockers

Defintion 2

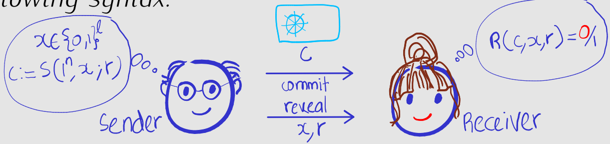
A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



Commitment Schemes are Digital Lockers

Defintion 2

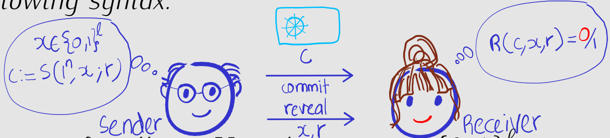
A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



Commitment Schemes are Digital Lockers

Defintion 2

A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



- *Correctness: for all* $n \in \mathbb{N}$ *and inputs* $x \in \{0,1\}^\ell$*:*

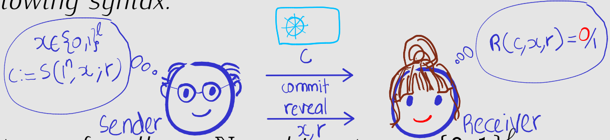
$$\Pr_r\left[R(S(1^n, x; r), x, r) = 1\right] = 1$$

- *Computational hiding:* c *reveals no information about* x *to PPT adversaries*

Commitment Schemes are Digital Lockers

Defintion 2

A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



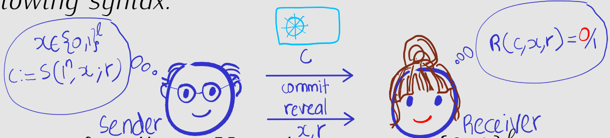
- *Correctness: for all* $n \in \mathbb{N}$ *and inputs* $x \in \{0,1\}^\ell$*:*

  $$\Pr_r\left[R(S(1^n, x; r), x, r) = 1\right] = 1$$

- *Computational hiding:* c *reveals no information about* x *to PPT adversaries*
- *Perfect binding: for any* $c \in \{0,1\}^*$*, there do not exist openings* $r_1, r_2 \in \{0,1\}^*$ *such that* $\mathsf{R}(c, r_1) \neq \mathsf{R}(c, r_2)$

Defintion 2

A (non-interactive) commitment scheme is a pair of algorithms (S, R) *with the following syntax:*



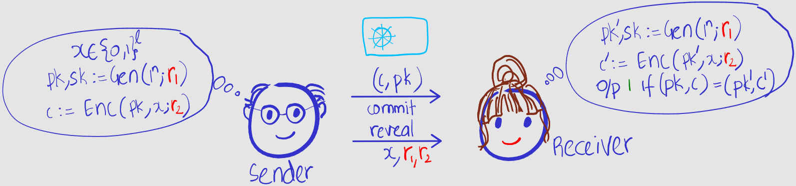
- *Correctness: for all* $n \in \mathbb{N}$ *and inputs* $x \in \{0,1\}^\ell$*:*

$$\Pr_r\left[R(S(1^n, x; r), x, r) = 1\right] = 1$$

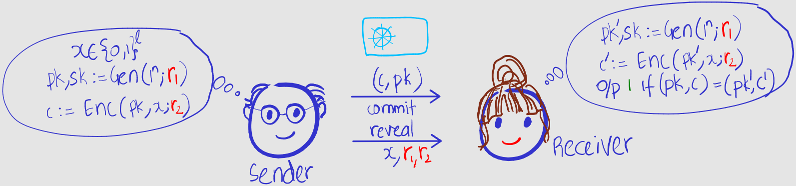
- *Computational hiding:* c *reveals no information about* x *to PPT adversaries*
- *Perfect binding: for any* $c \in \{0,1\}^*$*, there do not exist openings* $r_1, r_2 \in \{0,1\}^*$ *such that* $\mathsf{R}(c, r_1) \neq \mathsf{R}(c, r_2)$

- In general the commit phase can be interactive

Construction 2 (PKE $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}) \to$ commitment scheme Σ)



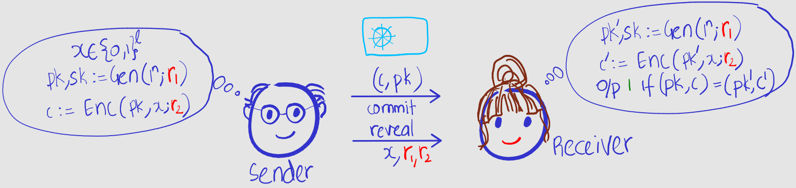
Construction 2 (PKE $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}) \rightarrow$ commitment scheme Σ)



What are the properties we require from Π?

How to Construct Commitment Schemes?

Construction 2 (PKE $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}) \rightarrow$ commitment scheme Σ)

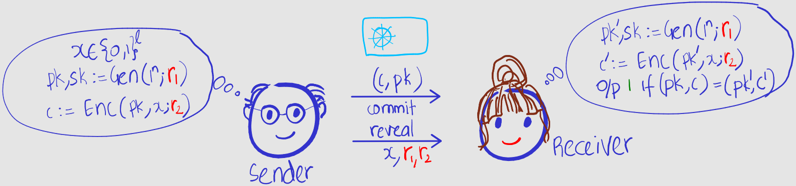


What are the properties we require from Π?

1. Recognise honestly sampled pks
2. Ciphertext-indistinguishability $\Rightarrow$ hiding
3. Perfect correctness of decryption $\Rightarrow$ binding

How to Construct Commitment Schemes?...

Construction 2 (PKE $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}) \rightarrow$ commitment scheme Σ)



What are the properties we require from Π?

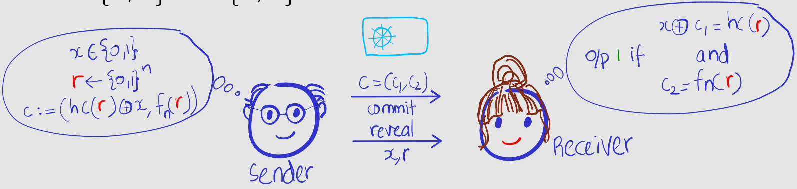
1. Recognise honestly sampled pks
2. Ciphertext-indistinguishability $\Rightarrow$ hiding
3. Perfect correctness of decryption $\Rightarrow$ binding

Exercise 6

Which of the PKEs we have seen satisfy the above properties?

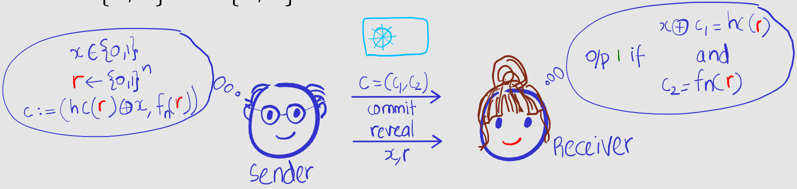
Construction 3 (OWP $f_n : \{0,1\}^n \to \{0,1\}^n \to$ *bit*-commitment Σ)

- *Recall: every (leaky)* f_n *has hard-core predicate* $hc : \{0,1\}^n \to \{0,1\}$



Construction 3 (OWP $f_n : \{0,1\}^n \to \{0,1\}^n \to$ *bit*-commitment Σ)

- *Recall: every (leaky)* f_n *has hard-core predicate* $hc : \{0,1\}^n \to \{0,1\}$

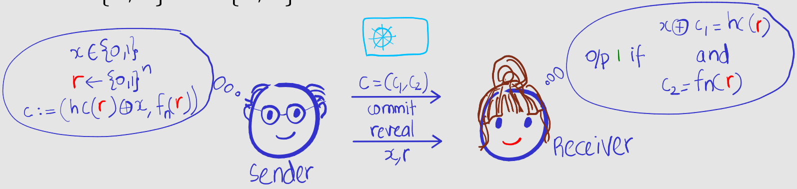


- Security of hard-core predicate hc $\Rightarrow$ computational hiding
- f permutation $\Rightarrow$ perfect binding

How to Construct Commitment Schemes?...

Construction 3 (OWP $f_n : \{0,1\}^n \to \{0,1\}^n \to$ *bit*-commitment Σ)

- *Recall: every (leaky)* f_n *has hard-core predicate* $\mathrm{hc} : \{0,1\}^n \to \{0,1\}$



- Security of hard-core predicate hc $\Rightarrow$ computational hiding
- f permutation $\Rightarrow$ perfect binding

Exercise 7

1. *Formally describe the construction, and write down the proof*
2. *Given a bit-commitment, construct a commitment for* $\{0,1\}^\ell$

How to Construct Commitment Schemes?...

Construction 3 (OWP $\mathsf{f}_n : \{0,1\}^n \to \{0,1\}^n \to$ *bit*-commitment Σ)

- *Recall: every (leaky)* f_n *has hard-core predicate* $\mathsf{hc} : \{0,1\}^n \to \{0,1\}$

- Security of hard-core predicate hc $\Rightarrow$ computational hiding
- f permutation $\Rightarrow$ perfect binding

How to Construct Commitment Schemes?...

Construction 3 (OWP $\mathsf{f}_n : \{0,1\}^n \to \{0,1\}^n \to$ *bit*-commitment Σ)

- *Recall: every (leaky)* f_n *has hard-core predicate* $\mathsf{hc} : \{0,1\}^n \to \{0,1\}$

- Security of hard-core predicate hc $\Rightarrow$ computational hiding
- f permutation $\Rightarrow$ perfect binding

Exercise 7

1. *Formally describe the construction, and write down the proof*
2. *Given a bit-commitment, construct a commitment for* $\{0,1\}^\ell$

To Recap Today's Lecture

- Malicious-verifier perfect ZKP for GI
  - Simulator was expected polynomial-time
  - Takeaway: Out of order sampling of transcript

To Recap Today's Lecture

- Malicious-verifier perfect ZKP for **GI**
  - Simulator was expected polynomial-time
  - Takeaway: Out of order sampling of transcript
- Computational ZKP for **NP**
  - Blum's protocol for Graph Hamiltonicity
  - What about perfect/statistical ZKP for **NP**?
    - Not possible (unless polynomial hierarchy collapses)!

To Recap Today's Lecture

- Malicious-verifier perfect ZKP for **GI**
  - Simulator was expected polynomial-time
  - Takeaway: Out of order sampling of transcript
- Computational ZKP for **NP**
  - Blum's protocol for Graph Hamiltonicity
  - What about perfect/statistical ZKP for **NP**?
    - Not possible (unless polynomial hierarchy collapses)!
- Commitment schemes
  - Non-interactive constructions from PKE and OWP
  - Two-message construction from PRG $\leftarrow$ OWF

Next Lecture

- Proofs of knowledge (PoK)
- PoK for the discrete-logarithm problem: Schnorr's protocol
- Fiat-Shamir Transform
  - Digital signatures from discrete-logarithm problem in the random-oracle model

References

1. [Gol01, Chapter 4] for details of today's lecture
2. [GMR89] for definitional and philosophical discussion on ZK
3. The ZKP for graph Hamiltonicity is due to Blum [Blu86]
4. The constructions of commitment scheme from OWP and PRG is from [GMW91] and [Nao90]