De Cifris Trends
in
Cryptographic Protocols

University of Trento and *De Componendis Cifris*

October 2023

Lecture 3

Luisa Siniscalchi Technical University of Denmark

Zero-Knowledge Protocols

Luisa Siniscalchi

Technical University of Denmark

Two parties for a protocol

- Prover has unbounded resources
- Verifier has limited resources

Verifier

Theorem x

π

Prover

The proof is efficient:
x is an NP statement that π is its certificate/witness/proof

Graph Isomorphism

An isomorphism of graphs G and H is a bijection (permutation) f between the vertex sets of G and H

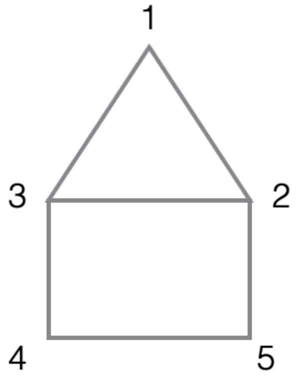
$$\pi: V(G) \longrightarrow V(H)$$

such that any two vertices u and v of G are adjacent in G if and only if $\pi(u)$ and $\pi(v)$ are adjacent in H.

Two parties for a protocol

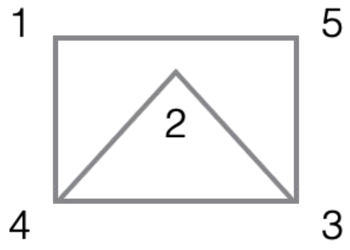
π

G	H
1	2
2	4
3	3
5	1
4	5



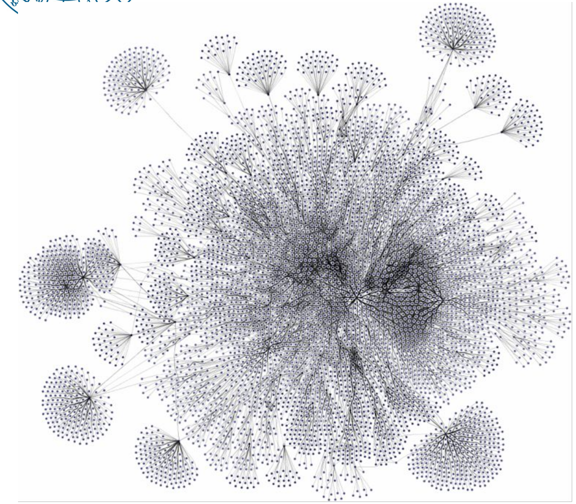
G

?
$\approx$

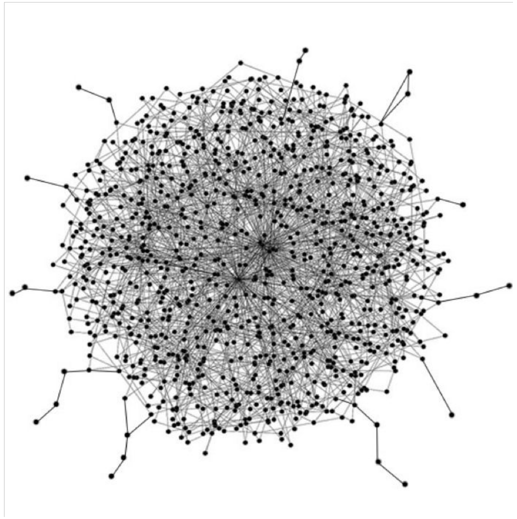


H

Graph Isomorphism



? $\approx$

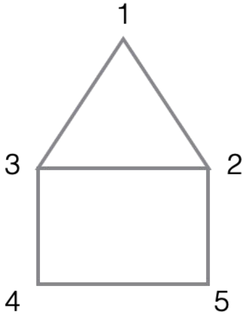


We do not know if it is in P: best-known algorithm is quasi-polynomial time

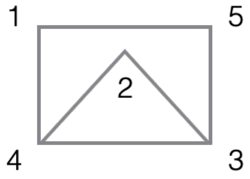
The problem belongs to NP

Graph Isomorphism

Thm:



≈



G

H

π: 1—>2, 4—>1,…

Interactive Proofs

- Suppose now that I want to prove that two graphs are <u>not</u> isomorphic.
- A proof is described as a game between a prover and a verifier
- The theorem is true if and only if the prover wins the game always.
- If the theorem is false then the prover loses the game with 50% probability

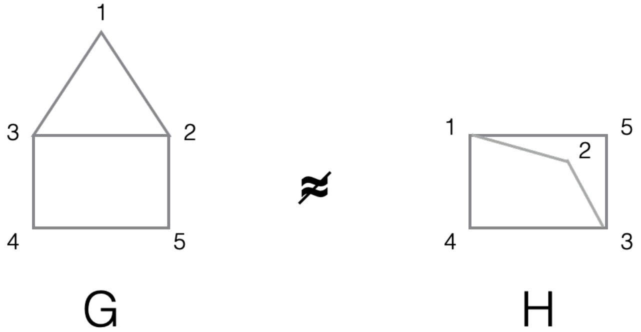
•Introduced by Goldwasser, Micali and Rackoff

Interactive Proofs

Example

- If the pencils are both red, then the prover convinces the verifier with 50% probability
- We can repeat the proof many times to make this probability small

Interactive Proofs



G ⇐ 

f ⇐ Random permutations

C ⇐ f(G)

$\xrightarrow{\quad C \quad}$

$\xleftarrow{\quad G \quad}$

Poly

Unbounded

Luisa Siniscalchi Technical University of Denmark

Interactive Proofs (formal definition)

Definition 4.2.6 (Generalized Interactive Proof): *Let* $\mathsf{c}, \mathsf{s} : \mathbb{N} \to \mathbb{R}$ *be functions satisfying* $\mathsf{c}(n) > \mathsf{s}(n) + \frac{1}{p(n)}$ *for some polynomial* $p(\cdot)$. *An interactive pair* (P, V) *is called a (generalized) interactive proof system for the language* L, *with* **completeness bound** $\mathsf{c}(\cdot)$ *and* **soundness bound** $\mathsf{s}(\cdot)$, *if*

- (modified) completeness: *for every* $x \in L$,

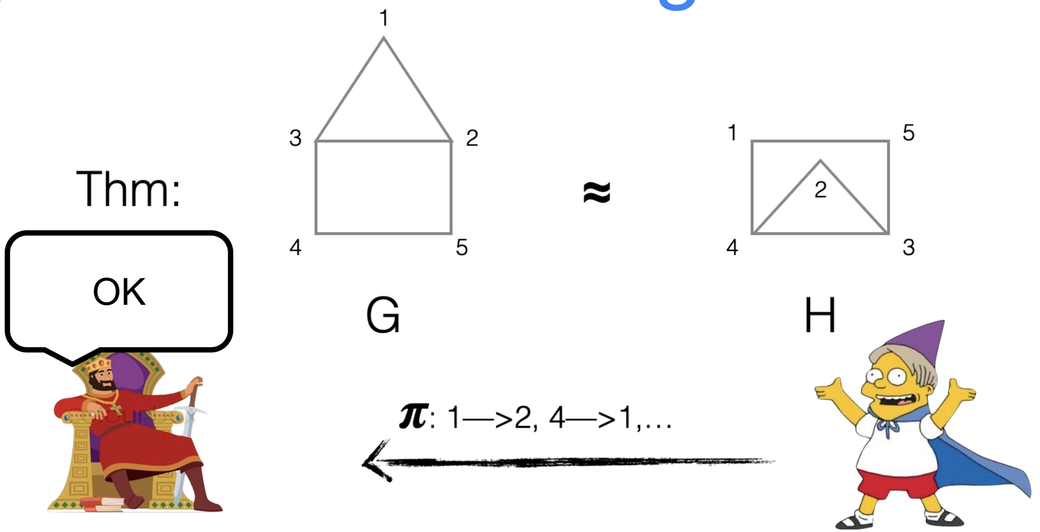
$$\Pr[\langle P, V\rangle(x) = 1] \geq \mathsf{c}(|x|)$$

- (modified) soundness: *for every* $x \notin L$ *and every interactive machine* B,

$$\Pr[\langle B, V\rangle(x) = 1] \leq \mathsf{s}(|x|)$$

In the previous example c(|x|)=1 and s(|x|)=1/2

Zero-Knowledge Proofs



- How much knowledge is transmitted to the verifier?
- We would like to transmit only one bit: 1 if the theorem is true and 0 otherwise.
- E.g. For the case of graph isomorphism the prover does not want to disclose the witness

Zero-Knowledge Proofs

Thm: $G \approx H$

$C \approx G \approx H$

$G \Leftarrow$

$C \Leftarrow f(G)$

$\xleftarrow{\quad C \quad}$

$\xrightarrow{\quad G \quad}$

$\xleftarrow{\quad f \quad}$

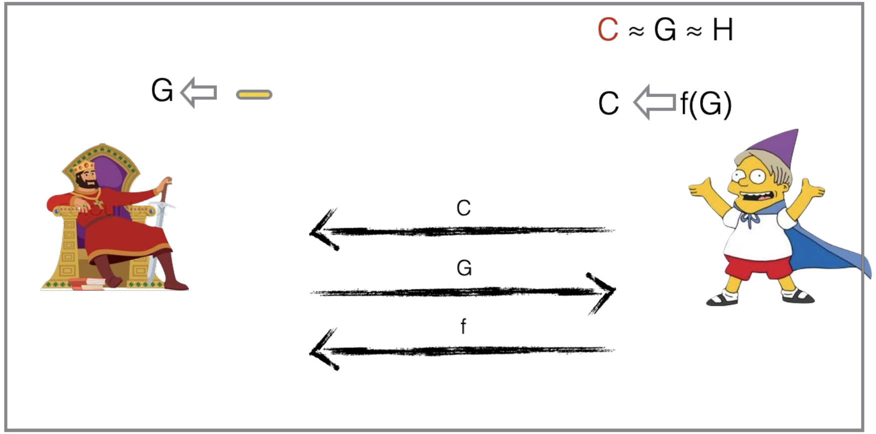
Luisa Siniscalchi Technical University of Denmark

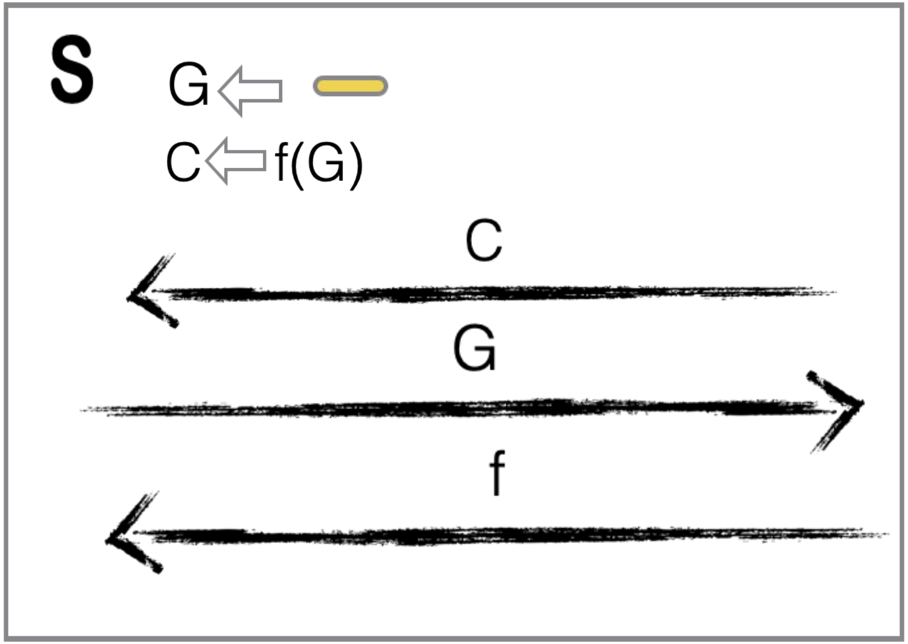
(Honest-Verifier) Zero-Knowledge Proofs

- The notion of (honest-verifier) zero-knowledge requires the existence of a simulator S that:
    - generates a transcript that is distributed similarly* to the real one (when the verifier is honest)
    - knows only that the theorem is true
    - is efficient (expected polynomial time)

(Honest-Verifier) Zero-Knowledge Proofs

Thm: $G \approx H$





Zero-Knowledge Proofs (formal definition)

Definition 4.3.2 (Computational Zero-Knowledge): *Let (P, V) be an interactive proof system for some language L. We say that (P, V) is* **computational zero-knowledge** *(or just* **zero-knowledge***) if for every probabilistic polynomial-time interactive machine V^* there exists a probabilistic polynomial-time algorithm M^* such that the following two ensembles are computationally indistinguishable:*

- $\{\langle P, V^* \rangle(x)\}_{x \in L}$ *(i.e., the output of the interactive machine V^* after it interacts with the interactive machine P on common input x)*
- $\{M^*(x)\}_{x \in L}$ *(i.e., the output of machine M^* on input x)*

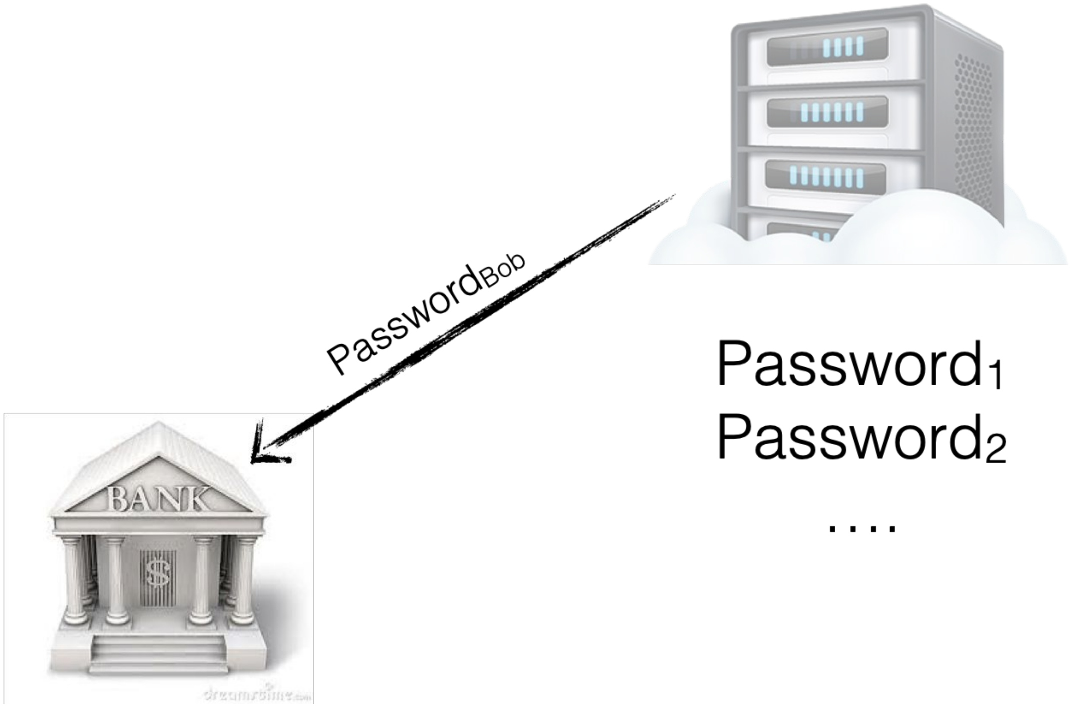
Machine M^* is called a simulator *for the interaction of V^* with P.*

ZK Application: Authentication

$\text{Password}_{\text{Bob}}$

Password_1
Password_2
....

ZK Application: Authentication



$Password_1$

$Password_2$

....

De Componendis Cifris

https:/www.decifris.it

Luisa Siniscalchi **Technical University of Denmark**