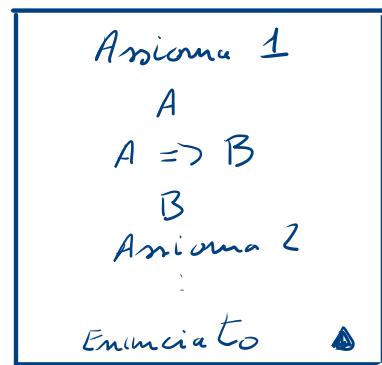


Concetto "classico" di prova

Thm : enunciato

Prova : Assomi \rightarrow derivazioni \rightarrow enunciato

Prova
formale :



\Leftarrow oggetto
sintattico

Critica : leggere l'intera prova può richiedere molto tempo

la prova può essere molto lunga

efficienza

Possiamo sempre immaginare una prova al modo seguente

Then : X



Prova π



Mago Proveratore

Potenza di calcolo infinita

Verificatore

Pigno-man Comp. limitato

- Il proveratore lavora alacremente per produrre una prova π breve che Pigno-man possa verificare velocemente

Quali tm ammettono prove efficienti?

Thm: Il grafo G è isomorfo al grafo H

$G = (V_1, E_1)$ e $H = (V_2, E_2)$ sono isomorfi

se esiste un mapping π

- 1-1 - 1 (iniettivo)

- suriettivo

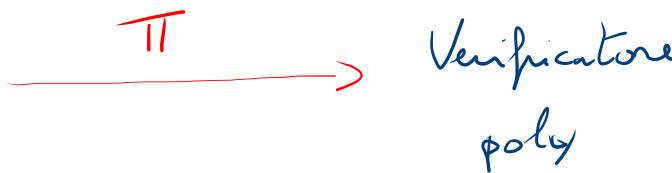
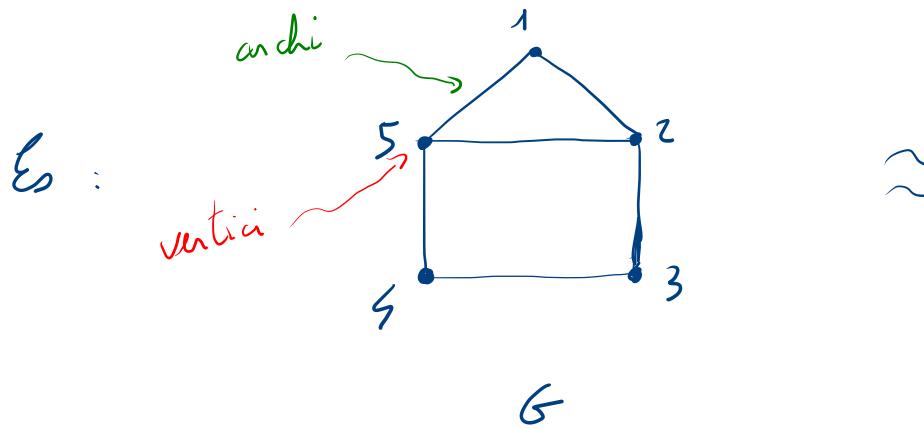
dall'insieme dei vertici V_1 all'insieme dei vertici V_2 tale che

$$(u, v) \in E_1 \iff \pi(u), \pi(v) \in V_2 \quad \text{e} \quad (\pi(u), \pi(v)) \in E_2$$

π è un isomorfismo tra i grafî.

Quali thm ammettono prove efficienti?

Thm: Il grafo G è isomorfo ad H



Prova

$\boxed{\text{TT} : 1 \rightarrow 5, 2 \rightarrow 3, 3 \rightarrow 4, 4 \rightarrow 1, 5 \rightarrow 2}$

isomorfismo tra G e H

corrispondenza
tra i vertici
di G e quelli
di H che
PRESERVA
gli archi

Per i grafi d'esempio il problema è banale (si vede ad occhio...)

Per grafi molto grandi non lo è: non si conoscono algoritmi efficienti.

Il provatore ha però potenza infinita e può calcolare Π^* .

D'altra parte:

- Π è corta (n vertici, lunga $O(n)$)
- Π è facile da verificare (archi presunti, al più n^2)
(tempo richiesto $O(n^2)$)

Il verificatore può memorizzarla e verificarla facilmente.

Anni '70

Classe NP →

Può essere vista come la classe
dei tm per cui \exists prove efficienti.

In generale : NP contiene tm rappresentabili da equazioni
di molte variabili per cui si può mostrare che
 \exists una soluzione a $\text{eq}(x_1, x_2, \dots, x_n) = 0$

Provatore ∞

Trova la soluzione
e la invia

$$\Pi = x_1, x_2, \dots, x_n$$

Verificatore poly
controlla se
sia corretta

Ma esistono anche domande "più difficili" a cui rispondere

Co-NP

- come faccio a dimostrare che NON esistono soluzioni per $\text{eq}(x_1, x_2, \dots, x_n) = 0$?

P

- o che esistono 2^{151} soluzioni ?



non posso neanche scivolare ed inviare né escludere da me 3 altre

Non disponiamo di prove "brevi" ed "efficienti"
da verificare per questo tipo di domande.

Nota #1 . Ma che cos'è una prova ?

"A proof is whatever convinces me"
(Shimon Even)

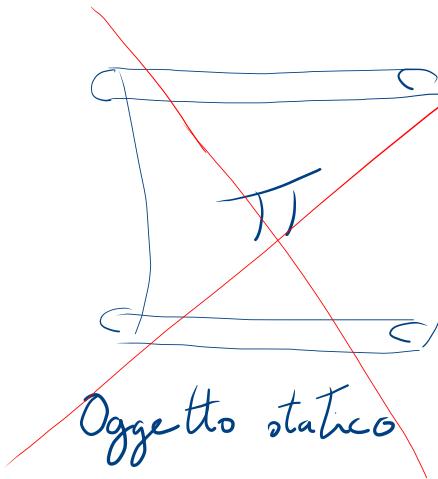
- l'esame incrociato di un testimone dinanzi alla corte è considerata una prova in legge
- l'incapacità di rispondere all'azione di un rivale in Filosofia ed in Politica a volte sono considerate prove
 - ... in situazioni della vita reale le prove hanno una natura dinamica, vengono stabilite tramite interazione

Per cercare di "dare prove" efficienti anche per le altre domande
introduciamo una nuova nozione di prova

IP (Interactive Proof) = Prova Interattiva

Anni '80

- ① Goldwasser, Micali, Rackoff (private coin)
- ② Babai, Moran (public coin)



Processo interattivo



Prova "come gioco" in cui P convince V

Precisamente, il processo deve essere tale che

- P convince V se il Thm è vero, e ci riesce sempre
- se il Thm è falso, P convince V con prob al più $\frac{1}{2}$

Ovviamente, se riusciamo a realizzare un processo del genere, allora:

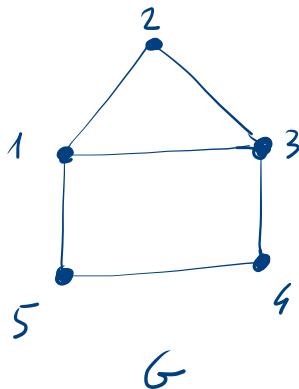
se P riesce a convincere V 100 volte

la probabilità che ci sia riuscito sempre e il Thm

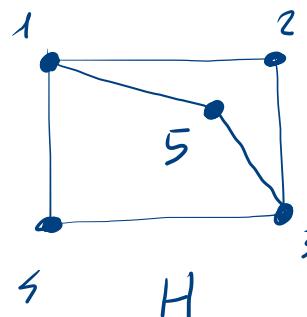
$$\text{è falso} \quad \text{risulta} \quad \leq \frac{1}{2^{100}} !$$

Thm: G e H non sono isomorfi

E.



\cancel{X}



Non c'è nessun isomorfismo tra G ed H

Come può P convincere V ?

Nell'esempio è semplice:

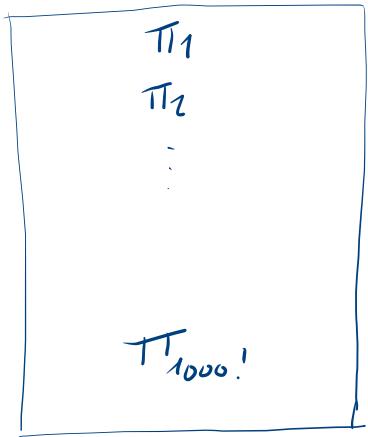
- il grafo G ha un ciclo di lunghezza 4
- il grafo H ha tre cicli di lunghezza 4

Per grafi molto grandi non lo è.

Ovviamente una prova TT sarebbe elencare tutte le possibili permutazioni e verificare che NON sono isomorfismi

$$n = 1000$$

\uparrow
di vertici



... il verificatore poly limitato non riuscirebbe neanche a "leggere" tutte ...

Non è efficiente !

$G \not\propto H$



Provatore ∞

Controlla $x \in G$

oppure $x \in H$

Invia la risposta
(G in questo caso)

$\leftarrow C$

\xrightarrow{G}

Verificatore poly(n)

Sceglie unif. a caso tra $G \in H$.

Sia G . Sceglie una permutazione π
a caso di G . Calcola $C = \pi(G)$
(isomorfo a G per costituzione)

Controlla se la risposta
è corretta

Il verificatore può ripetere il processo K volte scegliendo

- uniform. a caso tra \mathcal{G} ed H
- uniform. a caso una permutazione π

Se il provatore risponde sempre correttamente,
allora il verificatore può essere certo che $\mathcal{G} \not\approx H$

Infatti, se fosse $\mathcal{G} \approx H$, allora C sarebbe isomorfo
ad entrambi.cioé, $\mathcal{G} \approx H \Rightarrow \exists \varphi : \mathcal{G} = \varphi(H)$

$$\Rightarrow \begin{cases} C = \underline{\pi}(\mathcal{G}) \\ C = \pi(\varphi(H)) = \underline{\pi \circ \varphi}(H) \end{cases}$$

Pertanto, il provatore, può non calcolare un isomorfismo tra C e G , ma uno tra C e H , e ---

--- non avere assolutamente idea di quale dei due grafi, G o H , il verificatore abbia scelto per costituire C



Può solo lanciare una moneta e convincere il verificatore con probabilità $1/2$.

E può convincere il verificatore, se il protocollo viene ripetuto K volte, con probabilità $1/2^K$.

Condizione

Tutti gli enunciati veri ($\vdash \varphi \vdash t$) sono dimostrabili (Completeness)
Gli enunciati falsi no (Soundness)

Usando interazione e random bit sono in grado
di produrre prove efficienti per them per cui non dispongo
di prove tradizionali efficienti (concise)

Prove efficienti = Prove interattive

①

$$NP \subset IP$$

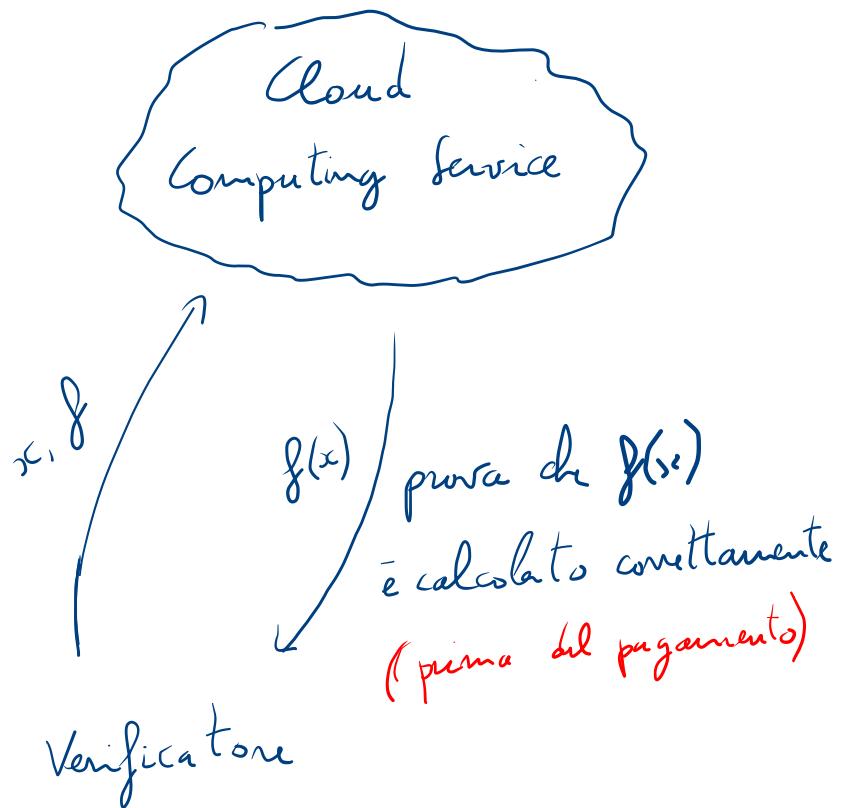
②

Riusciamo anche a dimostrare che $\text{eq}(x_1 \dots x_n) = 0$
non ha soluzioni ($\in NP$)

③

E riusciamo a dimostrare anche quante
soluzioni ammette ($\# P$)

Applicazioni



Una parola trasmette "conoscenza"

Quanta conoscenza è necessaria trasmettere per permettere ad un altro di verificare che un Θ è vero?

→ cos'è ?

Prove a conoscenza zero

non ce ne occupiamo



"Nient'altro de la verità"

Axiomi:

- la computazione efficiente "è gratis"
- la causalità (random bit) "è gratis"

Ritorniamo al problema dell'isomorfismo di grafi

$$G \approx H$$

Inviando l'isomorfismo π provo che $G \approx H$

Ma fornisco più conoscenza del dovuto

Invece di 1 bit



fornisco la corrispondenza precisa tra G e H .

Troppa conoscenza! Vogliamo una prova alternativa
che fornisca al verificatore soltanto un bit!

$$b_0 \approx b_1$$



$$P \propto$$

$$\checkmark \text{ poly}(n)$$

①

Sceglie π uniforme a caso

Costituisce $c \approx b_1$

Invia c

③

Se $b' = 1$, pone $\varphi = \pi$

Altrimenti, pone $\varphi = \pi \circ \gamma$

Invia φ



isomorfismo tra $b_0 \leftarrow b_1$

②

Sceglie $b' \in_{\mathbb{N}} \{0, 1\}$

Invia b'

$b' = 0$ "fammi vedere come
 c si ottiene da b_0 "

$b' = 1$ "fammi vedere come
 c si ottiene da b_1 "

④

Controlla x

$$c = \varphi(b')$$

Quando $g_0 \approx g_1$ il provatore vince sempre il gioco
D'altra parte, se il Thm è falso, il provatore vince solo
con prob. $1/2$. Infatti:

- se il verificatore chiede di vedere come $C \approx g_0$
ma il provatore l'ha costituito isomorfo a g_1 ,
allora non può rispondere!

~~F~~ q !

È un sistema di prova interattivo. Rispetto alla prova di
inizio lezione, che manda la corrispondenza tra g_0 e g_1 , lo chiamiamo
ripetere molte volte.

Ma cosa ha in più?

E' una prova del Thm a conoscenza zero!

Perche'?

Supponiamo che
il provatore non ci sia



poly(n)

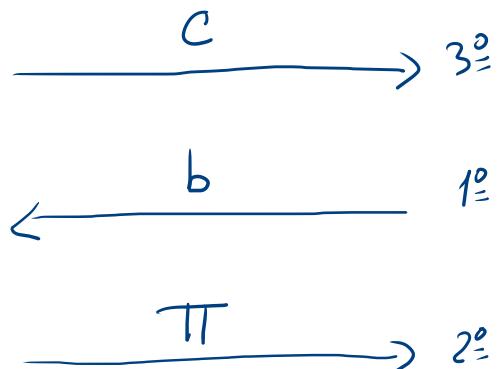
Supponiamo che il verificatore sappia che $\ell_0 \approx \ell_1$.

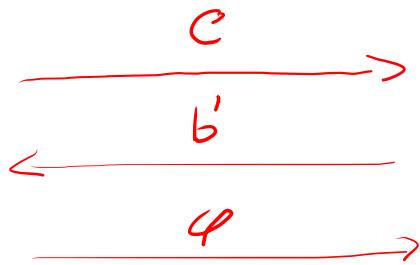
Allora, come esperimento mentale, il verificatore:

- lancia una moneta, sceglie ℓ_b
- lancia "più" monete, sceglie Π
- sostiene $C = \Pi(\ell_b)$

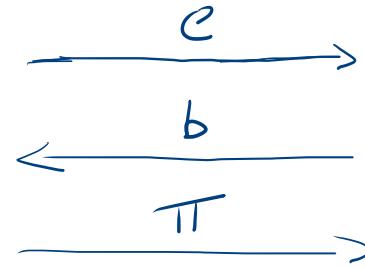
e poi scrive

(transcript)

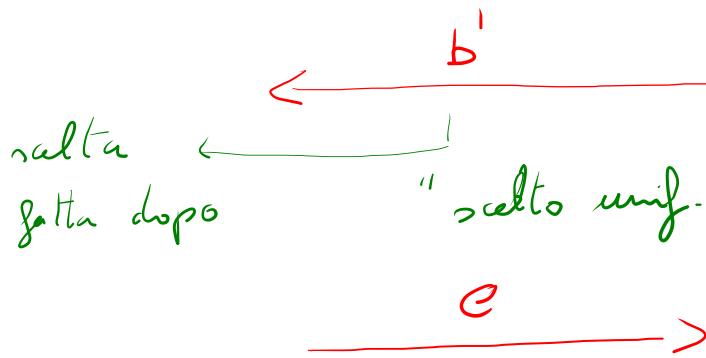




transcript reale



Confrontiamo
i due transcript
(e supp. $b' = b$)



salta ←
fatta dopo →

"scelta unf. a caso in entrambi i transcript"



"grado isomorfia a δ_b in entrambi i casi, scelta unf. a caso"



"

"corrispondenza casuale salta tra tutte le corrispondenze che
mappano c in δ_b in entrambi i casi"



In sintesi:

il Verificatore riesce da solo a costruire
l'interazione che avrebbe con il Provatore
nel caso in cui il them fosse vero

Manca qualche dettaglio alla prova (e.g., ho supposto che
il Verificatore sia onesto mentre potrebbe non scegliere $b \in \{0,1\}$
(unif. a caso) ma, in buona sostanza, l'idea è questa.
Se il Verificatore riesce a costruire da solo ciò che sarebbe
in una interazione reale, allora nell'interazione non c'è nulla
che gli permetta di acquisire ulteriore conoscenza al di là del
fatto che $b_0 \neq b_1$.

In conclusione:

(P , V)

\downarrow
 ∞

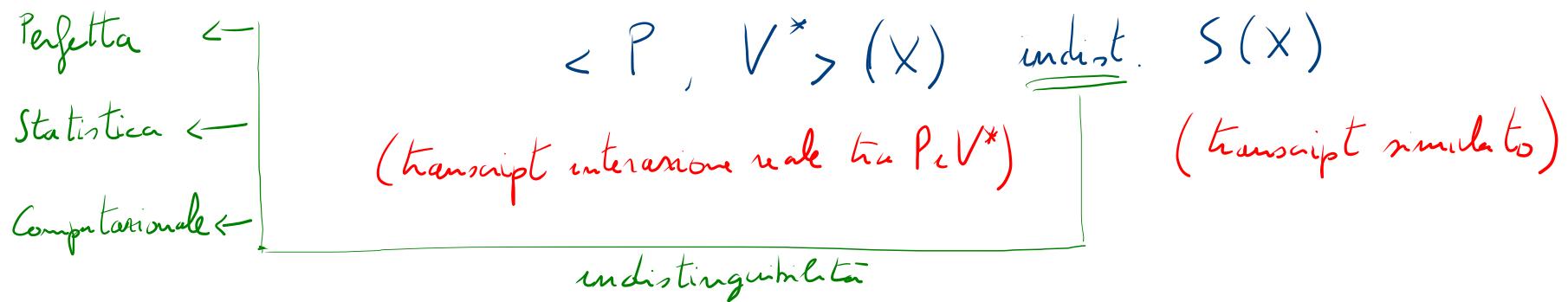
Sistema di prova a conoscenza zero

\downarrow
 $\text{poly}(n)$

- Completeness : them veri SI
- Soundness : them falso NO
- Zero Knowledge :

$\forall V^*$ (verificatore, anche malizioso) PPT

$\exists S$ (simulatore) PPT, tale che, \forall them vero X

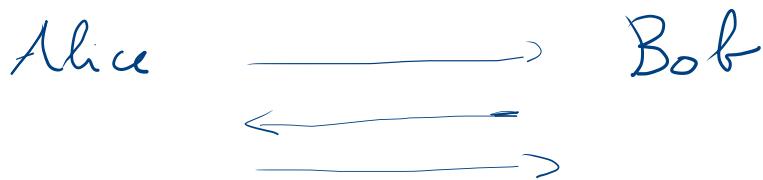


Vale solo per l'isomorfismo tra grafi?
Fortunatamente NO !

Tutto il "probabile" è probabile a conoscenza zero!

Applicazioni

Identificazione sicura



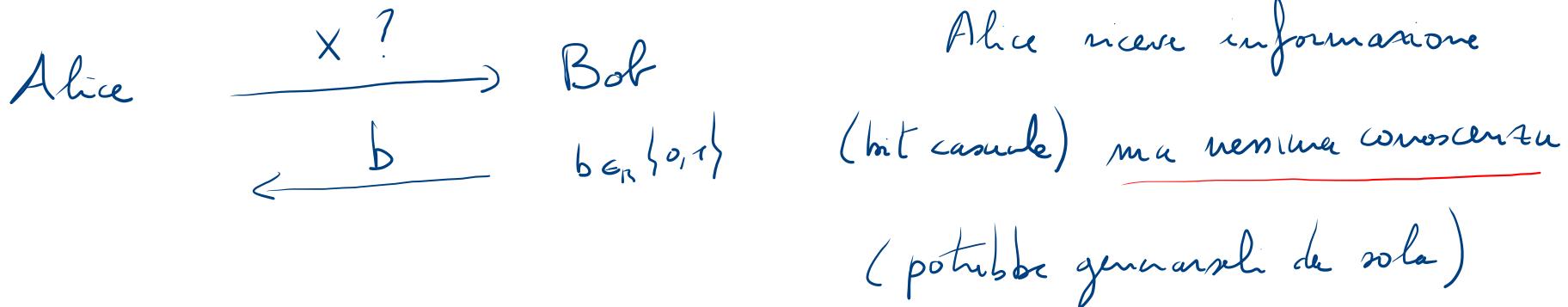
Alice riesce a farsi riconoscere da Bob

- senza condividerne con Bob una chiave segreta / pswol
- senza rilasciare ulteriori informazioni

Note : "Conoscenza" contro "Informazione"

↳ nel senso della teoria
dell'informazione

- Conoscenza: ha a che fare con la difficoltà computazionale (l'informazione no)
- Conoscenza: fa principalmente riferimento ad oggetti pubblicamente noti, mentre l'informazione ad oggetti di cui si hanno informazioni parziali



Nota: Simulatore: vi ricordo qualche cosa?
(tecnica)

Zero knowledge: $\forall V^* \text{ ppt}, \exists S \text{ ppt}$ tale che
 \forall them vero \times

Transcript reale \approx Transcript simulato
(indistinguishable)

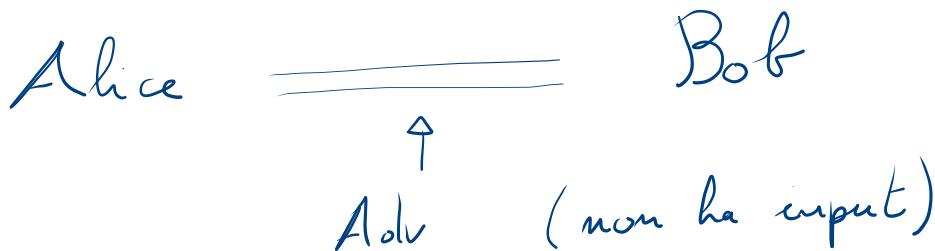
Quando abbiamo parlato di sicurezza semantică nel
contesto della cifratura, abbiamo detto che coglieva l'idea che

"Whatever is efficiently computable about the clear text
given the ciphertext, is also efficiently computable without"

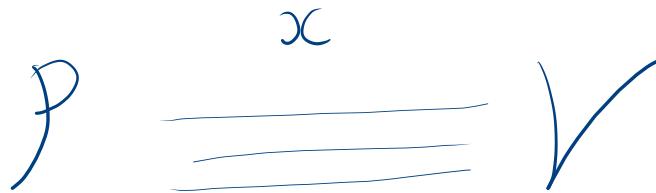
$\forall A \text{ ppt}, \exists A' \text{ ppt}$ tale che

"cioè di calcolare A dipendendo da c " è +/- "cioè di calcolare A' senza c "

Cifra tira



Schemi di prova
a conoscenza zero

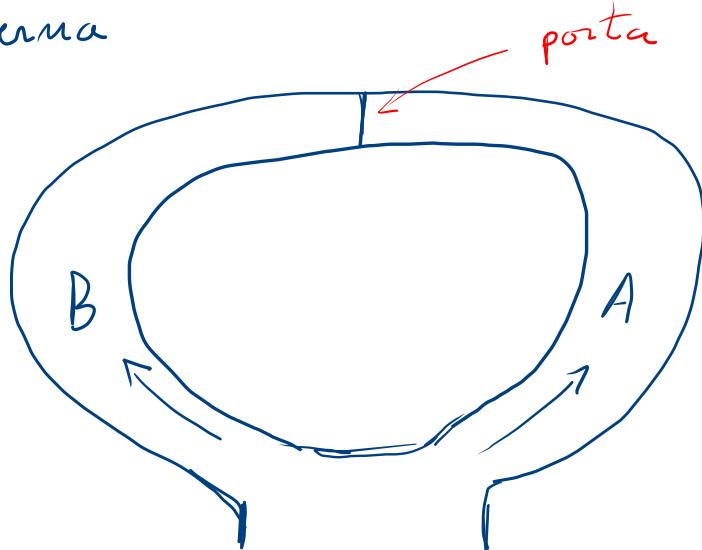


P cerca di convincere V che x è vero

- V ha un input (x , comune con P)
- V può essere malizioso (V^*)

In entrambi i casi l'esistenza di S (simulatore) garantisce "l'invitabilità" del transcript reale al fine di accrescere la conoscenza di Adv (V^*)

Gioco della caverna



Alice : "conosco la parola magica per aprire la porta"

Bob : "non ci andrò"

Protocollo (da ripetere K volte)

- Alice entra nella caverna (senza che Bob veda!) dal lato A oppure dal lato B
- Bob lancia una moneta e, a seconda del risultato, le chiede di uscire da A o da B

Sistema di prova a conoscenza zero ! Perche ?

Pensa tu ...

Riferimenti

Presentazione esemplificativa (---ribattezi) dalla lectio Magistralis del Prof. Silvio Micali (Turing Award, 2013) "Prove, segreti e computazione"
(la trovate su YOUTUBE)
(Eventuali errori li ho introdotti io!)

Ulteriori riferimenti

- ① D. Stinson - "Cryptography: theory and practice", 1st ed.
Il capitolo 13 offre una bella introduzione all'area,
chiara e concisa e non molto tecnica.
- ② O. Goldreich - "Foundations of Cryptography". Vol. I.
Capitolo 4. Tutto ciò che volete (s. i non tecnico)
- ③ A. Wigderson - "Mathematics and Computation". Capitolo 18.
Per una visione generale della Comp. Moderna e per avere
un'idea di tutto ciò che non abbiamo visto.