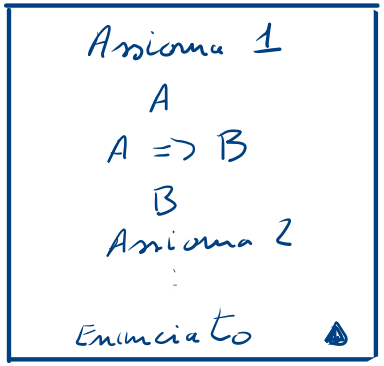
Concetto "classico" di prova

Thm : enunciato

Prova : Assiomi → derivazioni → enunciato

Prova formale :



<= 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

Thm : X

Prova Π ⟶

Mago Provatore

Potenza di calcolo infinita

Verificatore

Pigro-man Comp. limitato

. Il provatore lavora alacremente per produrre una prova Π breve che Pigro-man possa verificare velocemente

Quali thm 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

- $1-a-1$ (iniettivo)
- suriettivo

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

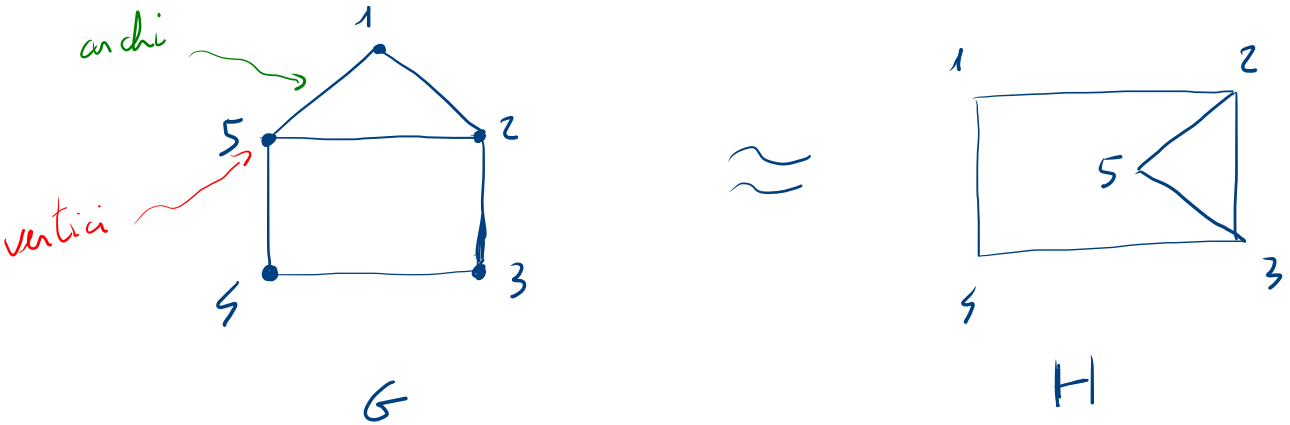
$$(u, v) \in E_1 \text{ se e solo se } (\Pi(u), \Pi(v)) \in E_2$$

Π è un isomorfismo tra i grafi.

Quali thm ammettono prove efficienti?

Thm: Il grafo G è isomorfo ad H

Es:



G $\qquad\qquad$ H

Provatore $\xrightarrow{\Pi}$ Verificatore

∞ $\qquad\qquad$ poly

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

Prova $\boxed{\Pi: 1 \to 5,\ 2 \to 3,\ 3 \to 4,\ 4 \to 1,\ 5 \to 2}$

isomorfismo tra G e H

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!

↳ (può provare tutte le possibili permutazioni)

D'altra parte:

- Π è corta (n vertici, lunga $O(n)$)
- Π è facile da verificare (archi presenti, 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 thm per cui ∃ prove efficienti

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

Provatore ∞
Trova la soluzione e la invia

$\Pi = x_1, x_2, \dots x_n$ ⟶

Verificatore poly
controlla che sia corretta

Ma esistono anche domande "più difficili" a cui rispondere

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

#P
- o che esistono 2^{151} soluzioni ?

↑ non posso neanche scriverle ed enumerarle né escludere che ne ∃ 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'asserzione di un rivale in Filosofia ed in Politica a volte sono considerate prove

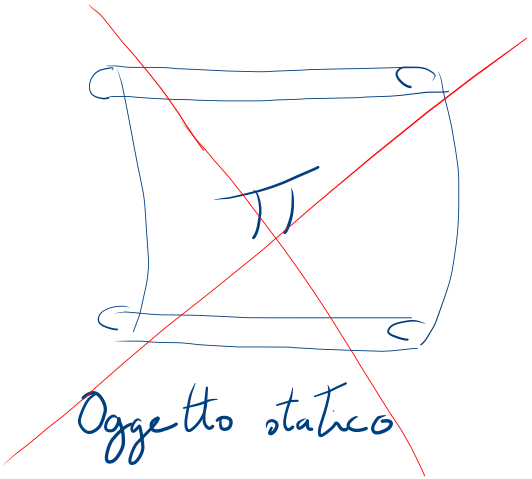
... in situazioni della vita reale le prove hanno una natura dinamica, vengono stabilite tramite interazione

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

IP (Interactive Proof) = Prova Interattiva

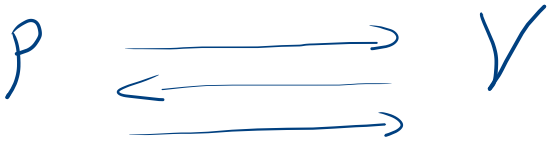
Anni '80 ① Goldwasser, Micali, Rackoff (private coin)

② Babai, Moran (public coin)



Oggetto statico

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 $1/2$

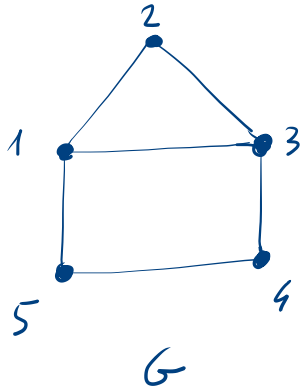
Ovviamente, se riusciamo a realizzare un processo del genere, allora:

se P riesce a convincere V 100 volte

la probabilità che ci sia riuscito sempre e il Thm è falso risulta $\leq 1/2^{100}$!

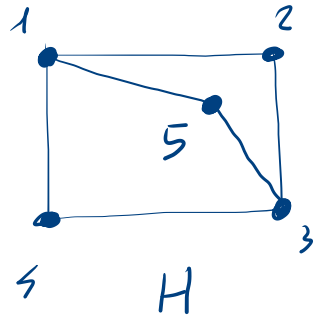
Thm: G e H non sono isomorfi

Es:



G

$\not\cong$



H

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

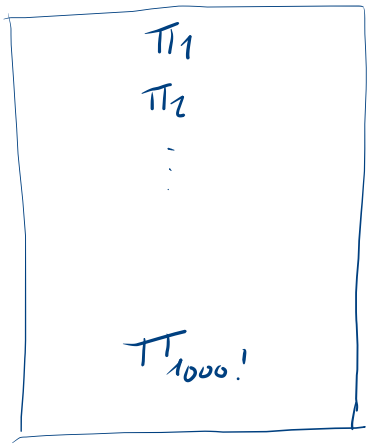
Per grafi molto grandi non lo è.

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

$n = 1000$

↑

di vertici



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

Non è efficiente!

$G \not\cong H$

Provatore ∞

Verificatore poly(n)

Sceglie unif. a caso tra G e H.
Sia G. Sceglie una permutazione π
a caso di G. Calcola $C = \pi(G)$
(isomorfo a G per costruzione)

C ⟵

Controlla se $C \cong G$
oppure se $C \cong H$
Invia la risposta
(G in questo caso)

G ⟶

Controlla che la risposta
sia corretta

Il verificatore può ripetere il processo k volte scegliendo

- uniform. a caso tra G ed H
- uniform. a caso una permutazione Π

Se il provatore risponde sempre correttamente, allora il verificatore può essere certo che $G \not\simeq H$

Infatti, se fosse $G \simeq H$, allora C sarebbe isomorfo ad entrambi. Cioé, $G \simeq H \Rightarrow \exists \varphi : G = \varphi(H)$

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

Pertanto, il provatore, può sia calcolare un isomorfismo tra C e G, sia uno tra C e H, e ...

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

$\Downarrow$

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 $1/2^k$.

Conclusione

Tutti gli enunciati veri ($\phi \notin HL$) sono dimostrabili (Completeness)

Gli enunciati falsi no (Soundness)

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

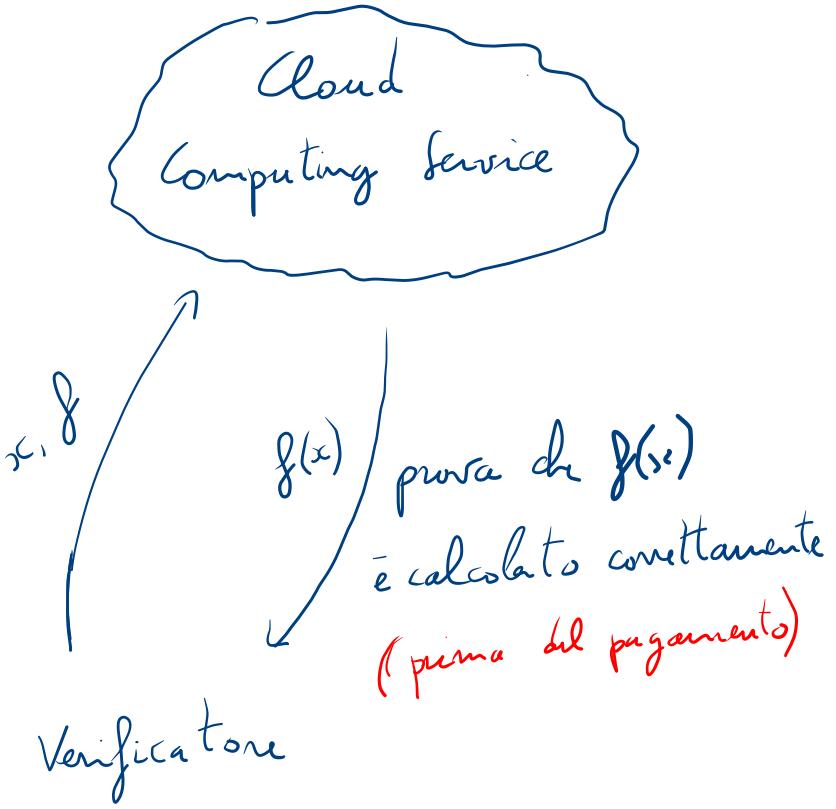
Prove efficienti = Prove interattive

(1) $NP \subset IP$

(2) Riusciamo anche a dimostrare che $eq(x_1 \dots x_n) = 0$ non ha soluzioni (coNP)

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

Applicazioni



Una prova trasmette "conoscenza"

Quanta conoscenza è necessario trasmettere per permettere ad un altro di verificare che un Thm è vero?

→ cos'è?

Prove a conoscenza zero — non ce ne occupiamo

↓

"Nient'altro che la verità"

Assiomi:

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

Ritorniamo al problema dell'isomorfismo di grafi

$$G \cong H$$

Inviando l'isomorfismo Π provo che $G \cong H$

Ma fornisco più conoscenza del dovuto

Invece di 1 bit $\begin{cases} \cong \\ \ncong \end{cases}$

fornisco la corrispondenza precisa tra G e H.

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

$G_0 \approx G_1$

$P \quad \infty$

V poly(n)

(1) Sceglie π unif. a caso

Costruisce $C \approx G_1$

Invia C

$C \longrightarrow$

(2) Sceglie $b \in_R \{0,1\}$

Invia b

$b = 0$ "fammi vedere come C si ottiene da G_0"

$b = 1$ "fammi vedere come C si ottiene da G_1"

$\longleftarrow b$

(3) Se $b = 1$, pone $\varphi = \pi$

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

Invia φ

$\varphi \longrightarrow$

γ: isomorfismo tra G_0 e G_1

(4) Controlla se

$C = \varphi(G_b)$

Quando $G_0 \not\cong G_1$ 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 $C \not\cong G_1$
  ma il provatore l'ha costruito isomorfo a G_0,
  allora non può rispondere!

$\not\exists \varphi$!

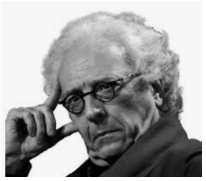
È un sistema di prova interattivo. Rispetto alla prova di inizio lezione, che manda la corrispondenza tra G_0 e G_1, lo dobbiamo ripetere molte volte.

Ma cosa ha in più?

E' una prova del Thm a conoscenza zero!

Perché?

Supponiamo che il proratore non ci sia

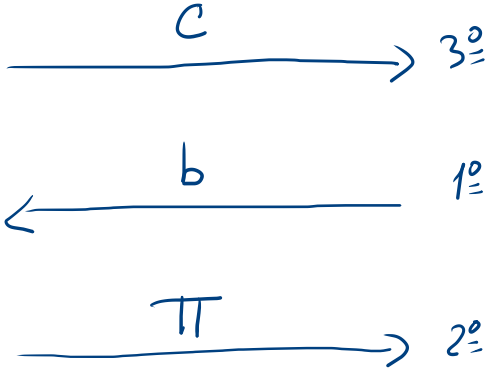


poly(n)

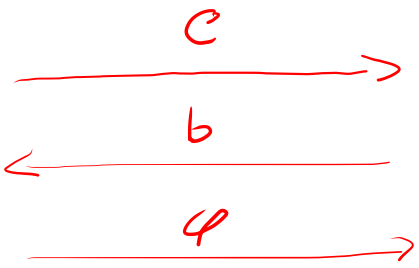
Supponiamo che il verificatore sappia che $G_0 \not\simeq G_1$.

Allora, come esperimento mentale, il verificatore:

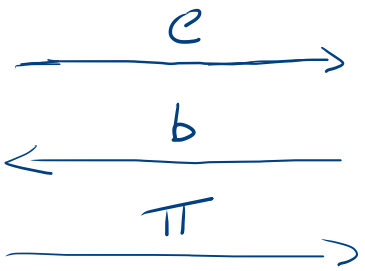
- lancia una moneta, ottiene b
- lancia "più" monete, sceglie Π
- costruisce $C = \Pi(G_b)$
- e poi scrive



(transcript)



transcript reale



transcript simulato

Confrontiamo i due transcript

scelta fatta dopo

"scelto uniformemente a caso in entrambi i transcript"

scelta fatta subito

"un grafo casuale isomorfo a G_1" (ma anche a G_0)

"un grafo casuale isomorfo a G_0 o G_1"

"corrispondenza casuale scelta tra tutte le corrispondenze che mappano G_b in C in entrambi i casi"

In sintesi:

il Verificatore riesce da solo a costruirsi l'interazione che avrebbe con il Provatore nel caso in cui il thm 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 vedrebbe in una interazione reale, allora nell'interazione non c'è nulla che gli permetta di acquisire ulteriore conoscenza al di là del fatto che $G_0 \not\simeq G_1$.

In conclusione:

(P, V) Sistema di prova a conoscenza zero

∞ $\quad$ $poly(n)$

- Completeness: thm veri SI
- Soundness: thm falsi NO
- Zero Knowledge:

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

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

Perfetta ←
Statistica ←
Computazionale ←

$\langle P, V^* \rangle(X)$ indist. $S(X)$

(transcript interazione reale tra P e V^*) (transcript simulato)

indistinguibilità

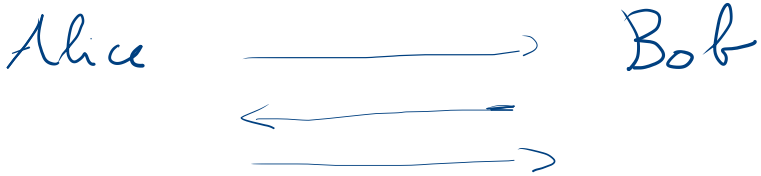
Vale solo per l'isomorfismo tra grafi?

Fortunatamente NO!

Tutto il "provabile" è provabile a conoscenza zero!

Applicazioni

Identificazione sicura



Alice riesce a farsi riconoscere da Bob

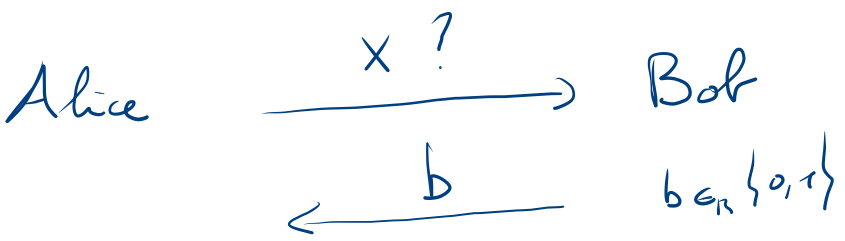
- senza condividere con Bob una chiave segreta / pwd
- senza rilasciare ulteriori informazioni

Nota: "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



Alice riceve informazione (bit casuale) ma nessuna conoscenza

(potrebbe generarseli da sola)

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

Zero knowledge: $\forall\ V^*$ ppt, $\exists\ S$ ppt tale che

$\forall$ thm vero x

Transcript reale $\approx$ Transcript simulato
(indistinguibile)

Quando abbiamo parlato di sicurezza semantica nel contesto della cifratura, abbiamo detto che cogliera l'idea che

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

$\forall\ A$ ppt, $\exists\ A'$ PPT tale che

"ciò che calcola A disponendo di c" è +/- "ciò che calcola A' senza c"

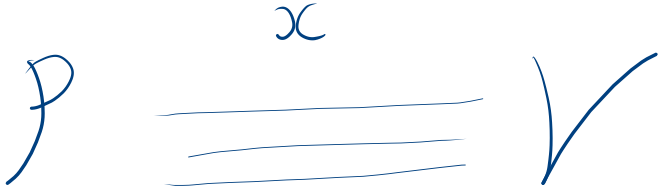
Cifratura

Alice ═══ Bob

↑

Adv (non ha input)

Schema di prova
a conoscenza zero

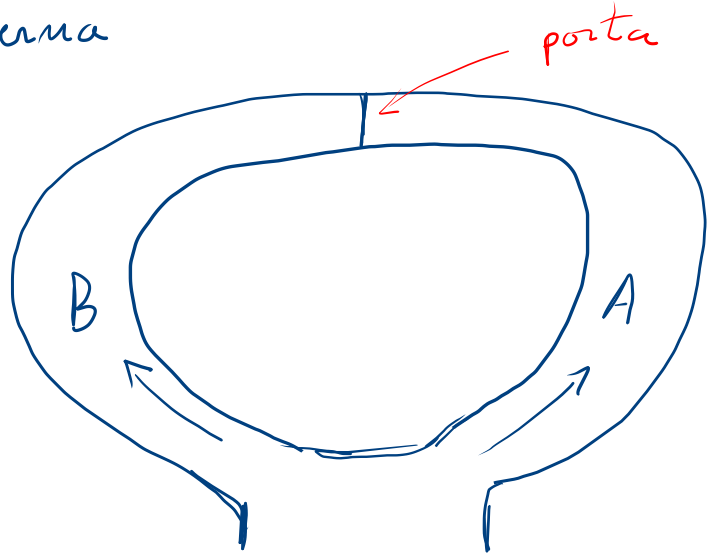


P cerca di convincere V che x è vero

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

In entrambi i casi l'esistenza di S (simulatore) garantisce "l'inutilità" 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 credo"

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! Perché?

Pensateci...

Riferimenti

Presentazione essenzialmente estratta (...rubata 😉) 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 (4.1 non tecnico)

③ A. Wigderson - "Mathematics and Computation". Capitolo 18.
Per una visione generale della Crit. Moderna e per avere un'idea di tutto ciò che non abbiamo visto.