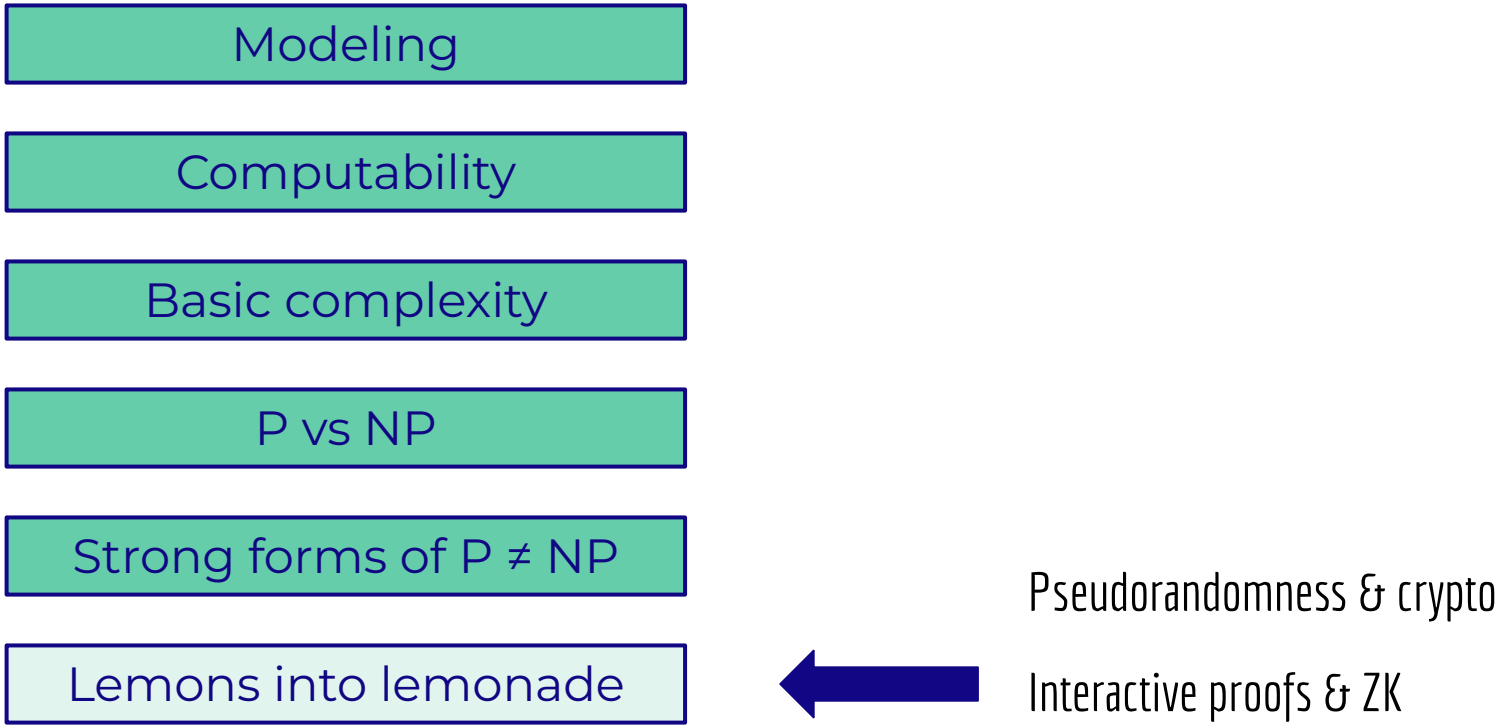
Complexity Theory

Instructor: Roei Tell

Lemons into lemonade

Course outline



Lemons into lemonade

› Throwback:

Lower bounds are *useful* for building algorithms and protocols.

Lemons into lemonade

› Throwback:

Lower bounds are *useful* for building algorithms and protocols.

› Build an algorithm/protocol Π based on a hard function f

⇒ no efficient adversary can compute f

⇒ the protocol Π is resilient against efficient adversaries

› Mainstay of complexity and cryptography

Lemons into lemonade

› Proofs of security:

If an efficient algorithm A can break our protocol Π, then another efficient algorithm B can compute f.

› Observation:

This is just a standard reduction!

computing f ≤ breaking Π

Pseudorandomness

& cryptography

Randomness

› Question:

Is there randomness in the world?

⇒ a question for physics

Randomness

- Question:

  Is there randomness in the world?

  ⇒ a question for physics

- Question:

  Do efficient algorithms behave like there is a randomness?

  ⇒ a question for computer science

Randomness

- Example 1:

  Is a coin toss random?

Pseudorandomness

- Mental experiment:

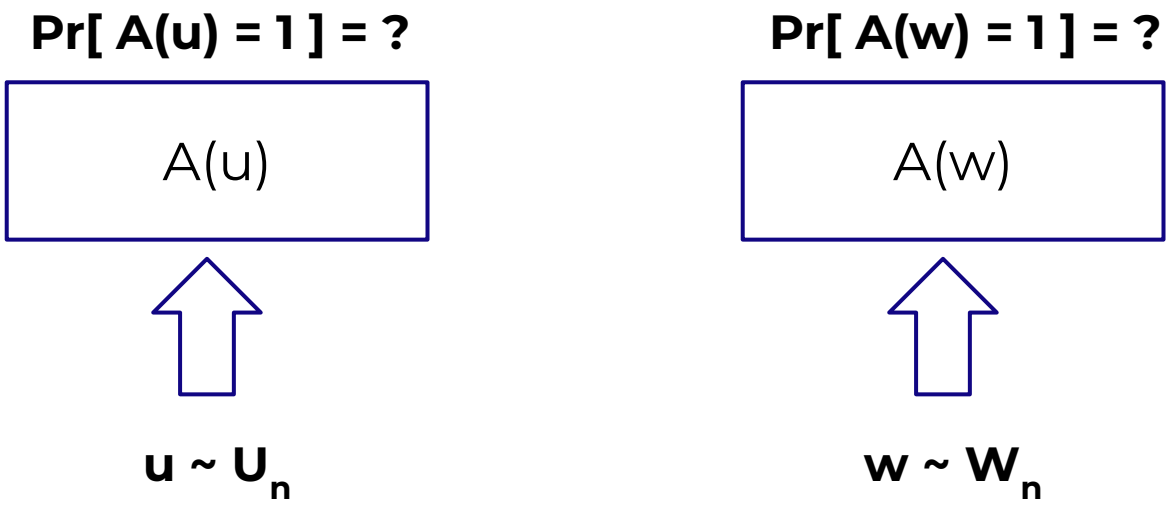
  An efficient algorithm A looks at a string $w \in \{0,1\}^n$, supposedly the result of random coin tosses, and either declares:

  - "the string w is the result of real coin tosses!"
  - "the string w is was artificially generated, not real coin tosses!"

  In other words, A tries to distinguish between the two cases.

Pseudorandomness

› Mental experiment:



Pseudorandomness

› Definition:

A probability distribution $\mathbf{W_n}$ **over** $\mathbf{\{0,1\}^n}$ is **ε-pseudorandom for an algorithm A** if

$$| \Pr_{w \sim W_n}[A(w) = 1] - \Pr_{u \sim U_n}[A(u) = 1] | \leq \varepsilon(n)$$

[1] some sources also consider A's that receive a polynomial-sized "auxiliary input" as advice; we omit this for simplicity

Pseudorandomness

- Pseudorandom generators (PRGs) in this context:
  - the output **looks random** to **all efficient observers**
  - relies on **computational limitations:** the output is not truly random (there are only 2^{ℓ} choices, where ℓ is the seed length)

Pseudorandomness

› Definition:

A **pseudorandom generator** G is an algorithm that gets as input a seed $s \in \{0,1\}^{\ell(n)}$ and outputs $G(s) \in \{0,1\}^n$ st for every polynomial time algorithm A (and large enough n)

the distribution $\mathbf{G(u_{\ell(n)})}$ **is ε-pseudorandom for A**

where $\varepsilon(n) = n^{-\omega(1)}$.

[1] this is often called a “cryptographic” PRG (to be distinguished from other types of PRGs)

Pseudorandomness

- Conjecture (PRGs exist):

  For every constant $\delta > 0$, there exists a polynomial-time pseudorandom generator with seed length n^{δ} .

Pseudorandomness

- Theorem:

  If the PRG conjecture is true, then $\mathbf{P} \neq \mathbf{NP}$.

Commitment schemes

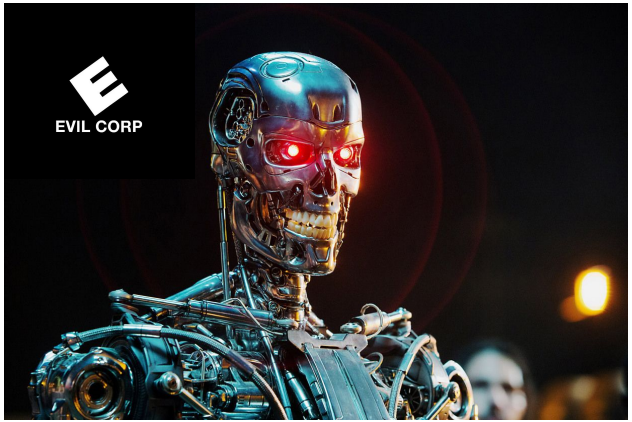
› Theorem:

If the PRG conjecture is true, then there exists an efficient **commitment scheme.**

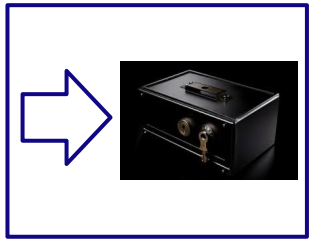
turn lemons into lemonade

(lower bounds ⇒ algorithms, crypto)

Commitment schemes



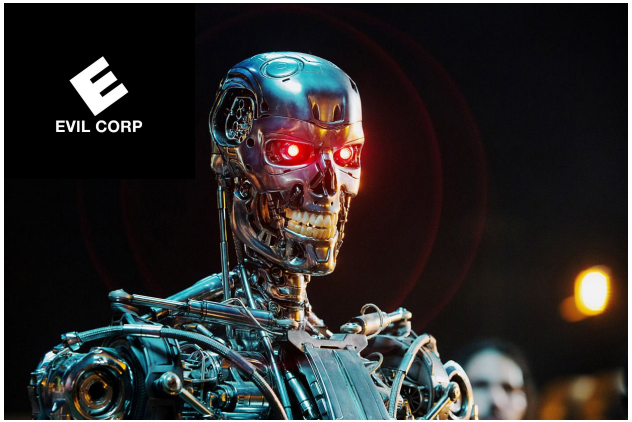
commitment protocol



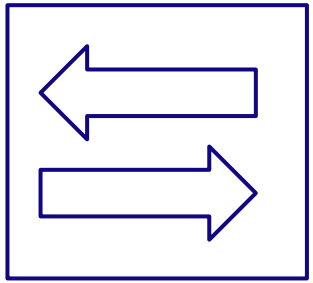
...

reveal phase

Commitment schemes

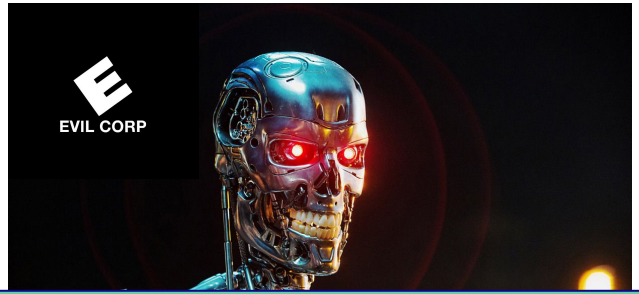


commitment protocol

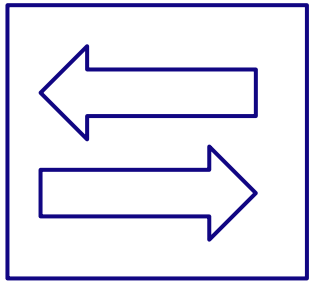


reveal phase

Commitment schemes

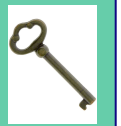


commitment protocol



- **Hiding:** The initial commitment isn't revealing

- **Binding:** A commitment can't be opened in two ways



reveal phase

Indistinguishability

› Definition:

Two probability distributions **W_n and Q_n over $\{0,1\}^n$** are **ε-indistinguishable by an algorithm A** if

$$| \Pr_{w \sim W\_n}[A(w) = 1] - \Pr_{q \sim Q\_n}[A(q) = 1] | \leq \varepsilon(n)$$

Commitment schemes

› Definition:

A commitment scheme is a two-player protocol

C(b) // "committer" with a bit $b \in \{0,1\}$

R // "receiver"

with a "commit" phase and a "reveal" phase.

We model both players as Turing machines.

Commitment schemes

› Definition (continued):

At any given point the receiver can output “error” and abort.

After the “reveal” phase the receiver should output a bit.

The honest receiver & honest committer are polytime machines. When the honest machines two interact, with probability 1 over both phases, the receiver outputs C(b).

Commitment schemes

› Definition (continued):

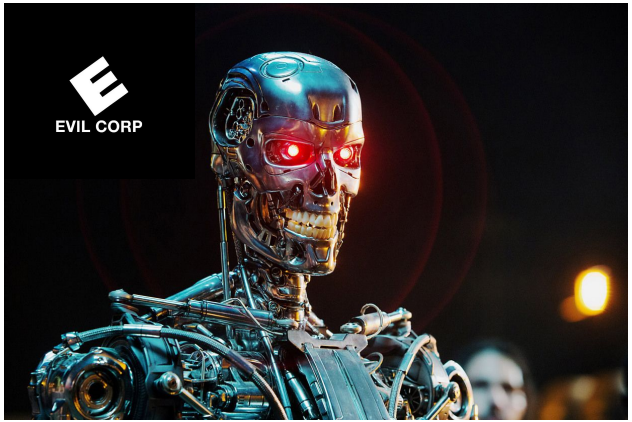
For security, we require that:

Hiding: The distributions of transcripts W_n^0 and W_n^1 when the honest prover commits to 0 and to 1, respectively, are $n^{-\omega(1)}$-indistinguishable (by any polytime algorithm).

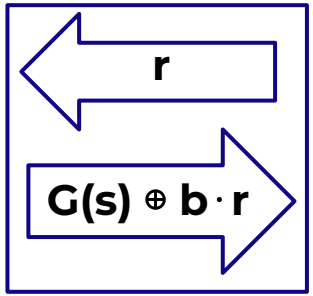
Binding: With probability $1-n^{-\omega(1)}$ over the commit phase, there is a single bit $b \in \{0,1\}$ st on any "reveal" message, the receiver outputs either b or "error".

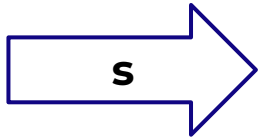
Commitment schemes

› A construction:



commitment protocol

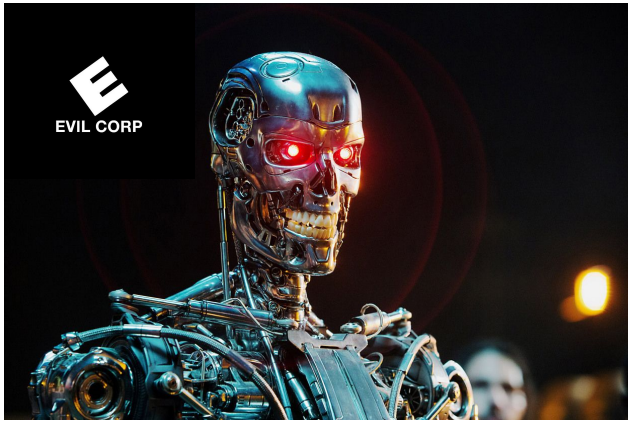




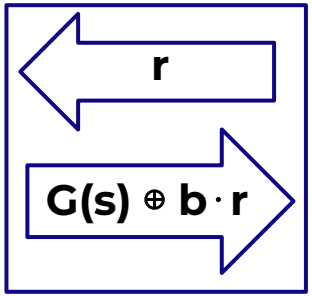
reveal phase

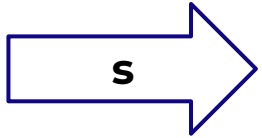
Commitment schemes

› A construction:



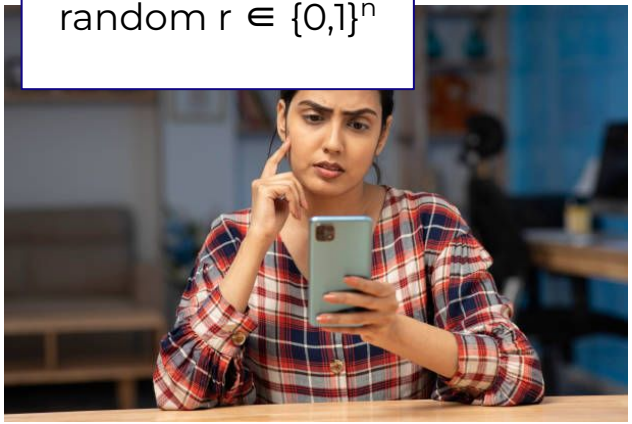
commitment protocol





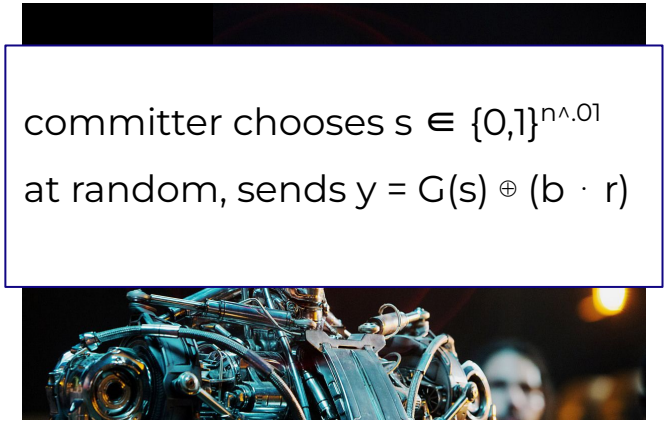
reveal phase

receiver sends a random $r \in \{0,1\}^n$



Commitment schemes

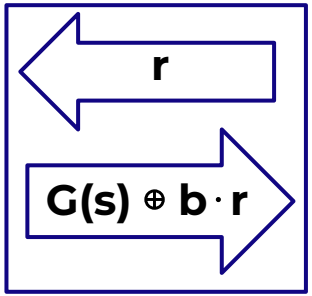
› A construction:

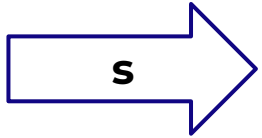


committer chooses $s \in \{0,1\}^{n^{.01}}$

at random, sends $y = G(s) \oplus (b \cdot r)$

commitment protocol



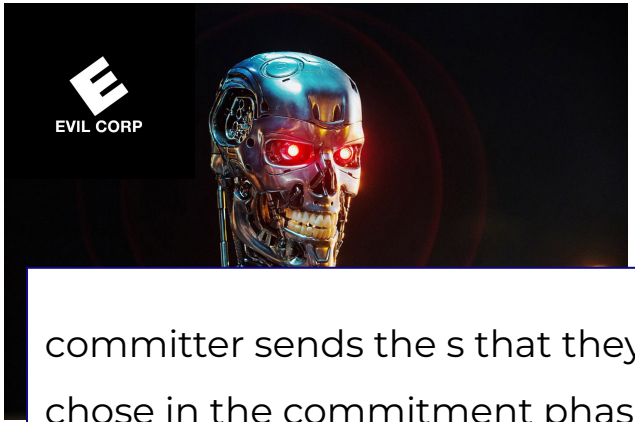


reveal phase

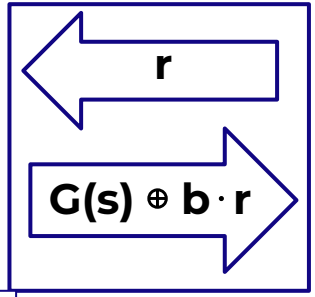
Commitment schemes

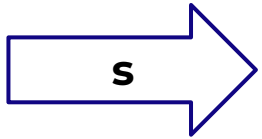
› A construction:

commitment protocol



committer sends the s that they chose in the commitment phase





reveal phase

Commitment schemes

- Lemma:

  With probability $1-n^{-\omega(1)}$, after the commitment phase there is at most one $b \in \{0,1\}$ st on any “reveal” message, the receiver outputs either b or “error”.

Commitment schemes

› Lemma:

With probability $1-n^{-\omega(1)}$, after the commitment phase there is at most one $b \in \{0,1\}$ st on any "reveal" message, the receiver outputs either b or "error".

› Main observation:

If $y = G(s) \oplus r$ and $y = G(s')$ then $r = G(s) \oplus G(s')$.

Let $B = \{ r \in \{0,1\}^n : \exists\ s,s',\ r = G(s) \oplus G(s') \}$. Then $|B| \le 2^{2n \wedge .01} \ll 2^n$.

Commitment schemes

- Lemma:

  If there is a polytime algorithm that distinguishes between W_n^0 and W_n^1 then the PRG conjecture is false.

- Intuition:

  What the receiver sees is either G(s) or “shifted” G(s) $\oplus$ r, for a random s, and that should look random.

Commitment schemes

› Proof:

This is a reduction!

distinguishing the PRG ≤ distinguishing W_n^0 from W_n^1

Given an algorithm distinguishing between W_n^0 and W_n^1, we construct an algorithm that distinguishes the PRG's outputs from a random n-bit string.

(proof as optional reading material)

Interactive proofs

& zero-knowledge

Interactive proofs

- Example 1:

  The color-blind instructor

Interactive proofs

› Example 1 (cont'd):

Common input: two markers (x_1, x_2)

Prover sends a random labeling of x_1, x_2 by 0 and 1, verifier writes the labels secretly on the markers

Verifier randomly chooses $b \in \{1,2\}$, presents x_b to the prover and asks it to name the label of b

Prover sends a label, verifier check that it matches

Interactive proofs

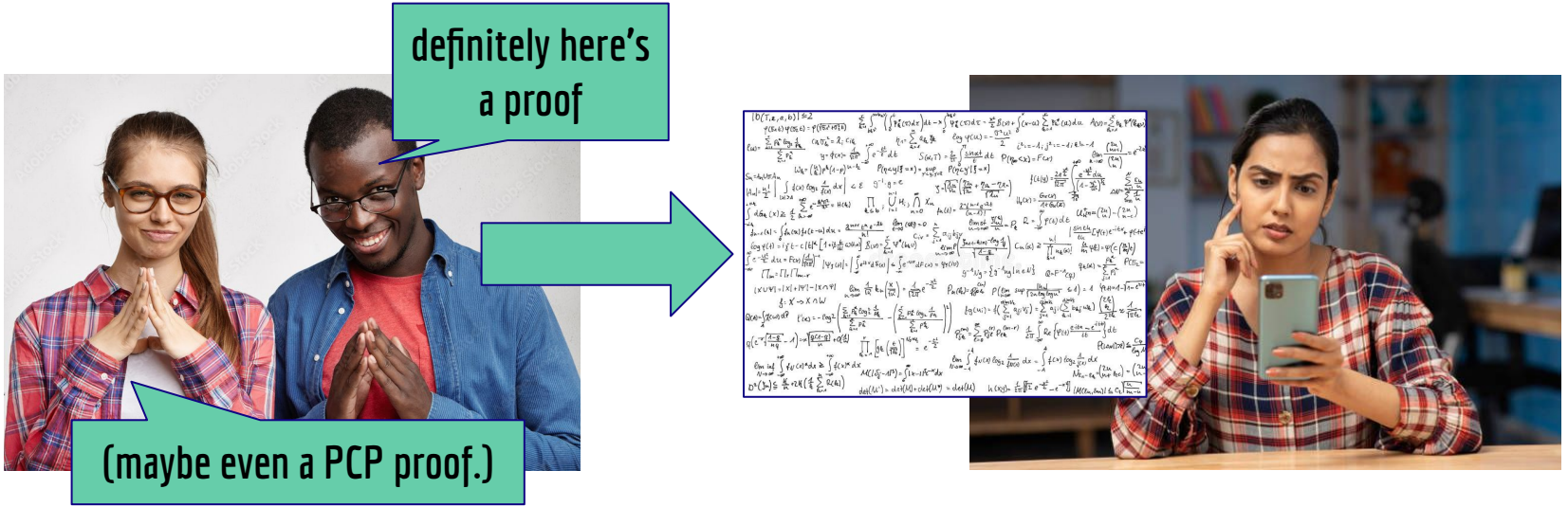
$x \in \{0,1\}^n$

Interactive proofs

seems wrong… are you sure?

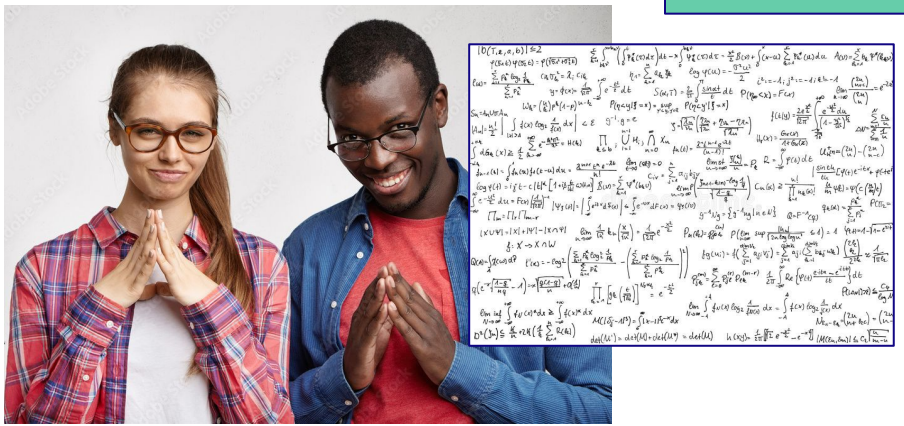
$x \in \{0,1\}^n$

Interactive proofs

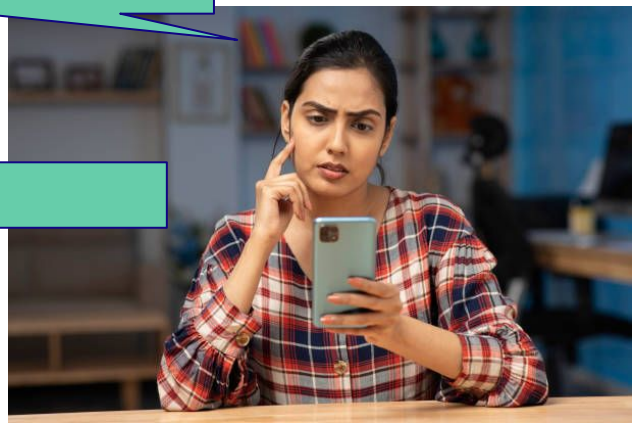


$x \in \{0,1\}^n$

Interactive proofs

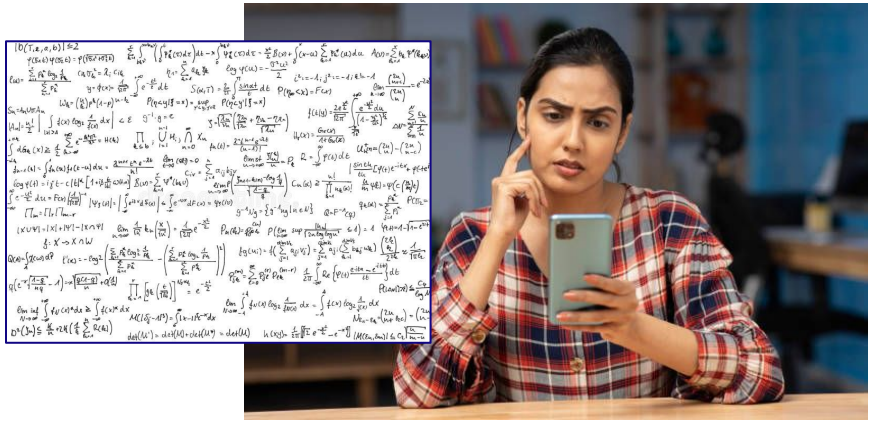


there's a bug on page 129



$x \in \{0,1\}^n$

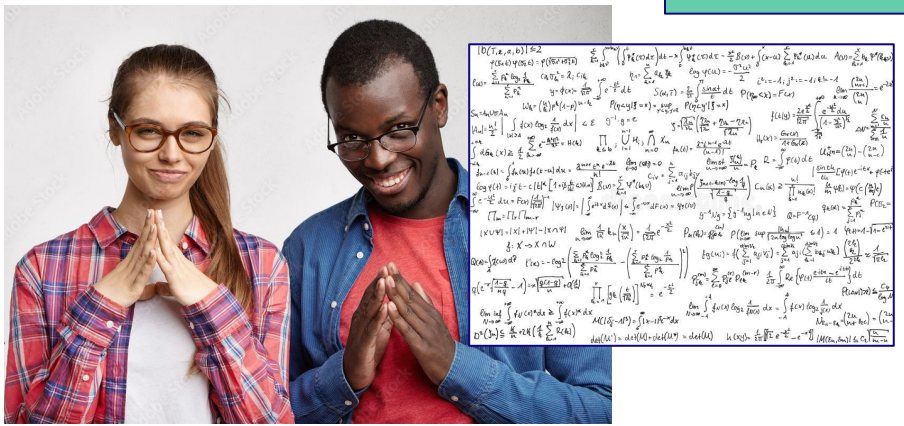
Interactive proofs

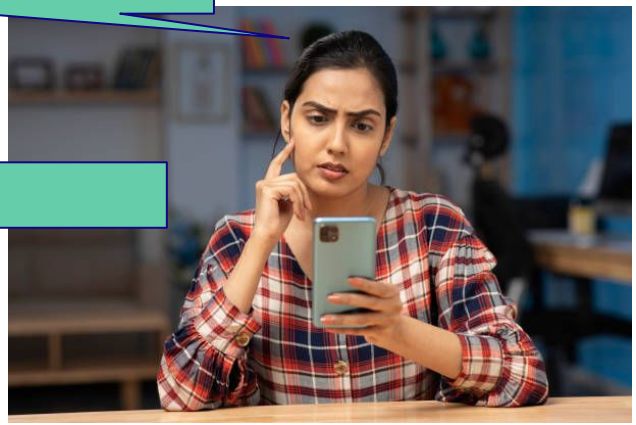


$x \in \{0,1\}^n$

Interactive proofs

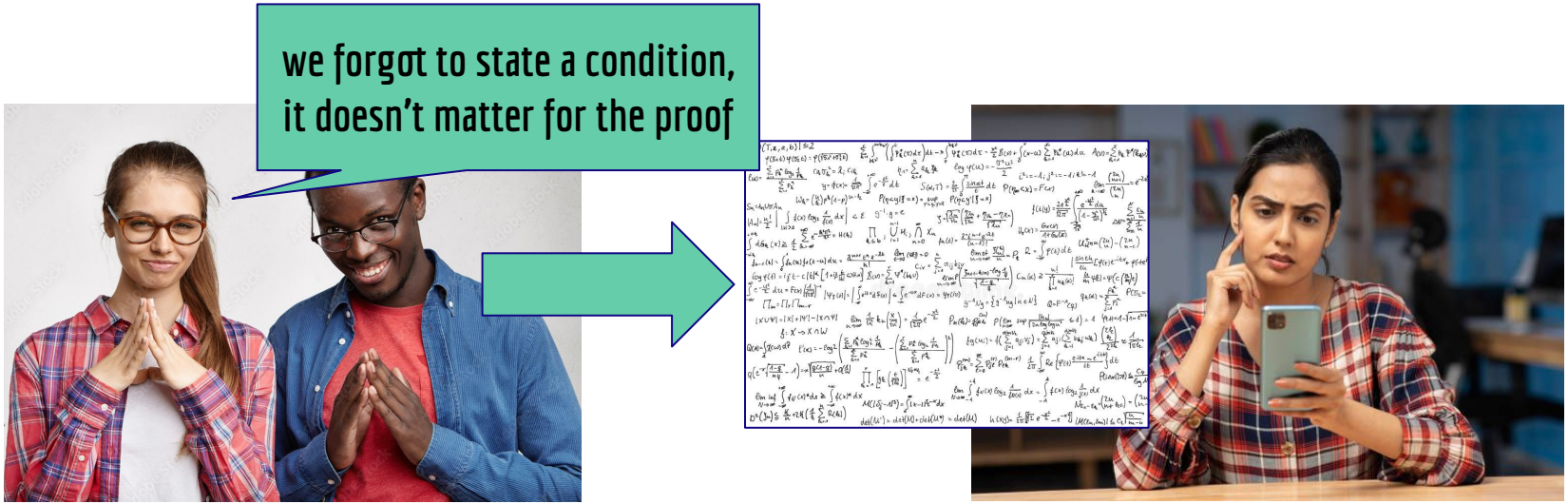
found a counter-example
for Lemma 23.5





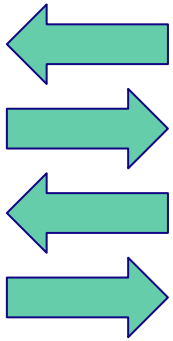
$x \in \{0,1\}^n$

Interactive proofs

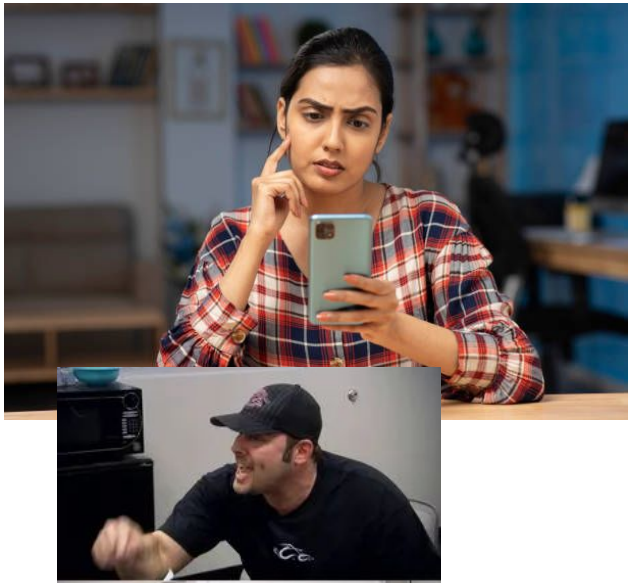


$x \in \{0,1\}^n$

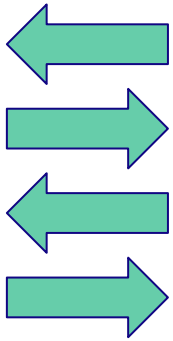
Interactive proofs



$x \in \{0,1\}^n$



Interactive proofs



$x \in \{0,1\}^n$

Interactive proofs

Wiles's proof of Fermat's Last Theorem

During 21–23 June 1993, Wiles announced and presented his proof of the Taniyama–Shimura conjecture for semistable elliptic curves, and hence of Fermat's Last Theorem, over the course of three lectures delivered at the Isaac Newton Institute for Mathematical Sciences in Cambridge, England.[2] There was a relatively large amount of press coverage afterwards.[13]

After the announcement, Nick Katz was appointed as one of the referees to review Wiles's manuscript. In the course of his review, he asked Wiles a series of clarifying questions that led Wiles to recognise that the proof contained a gap. There was an error in one critical portion of the proof which gave a bound for the order of a particular group: the Euler system used to extend Kolyvagin and Flach's method was incomplete. The error would not have rendered his work worthless—each part of Wiles's work was highly significant and innovative by itself, as were the many developments and techniques he had created in the course of his work, and only one part was affected.[1]:289,296–297 Without this part proved, however, there was no actual proof of Fermat's Last Theorem.

On 6 October Wiles asked three colleagues (including Gerd Faltings) to review his new proof,[20] and on 24 October 1994 Wiles submitted two manuscripts, "Modular elliptic curves and Fermat's Last Theorem"[4] and "Ring theoretic properties of certain Hecke algebras",[5] the second of which Wiles had written with Taylor and proved that certain conditions were met which were needed to justify the corrected step in the main paper.

Interactive proofs

- Model:
  - We consider Turing machines with a special state s_{comm}
  - When moving to s_{comm}, the content of a dedicated worktape is sent to the other machine, and the response of other machine replaces it on that worktape

Interactive proofs

- Model:
  - Consider machines P and V with common input x.
  - The interaction is defined in the natural way:

    V(x) runs, at some point enters state s_{comm}, then P(x) runs with the message from V on its dedicated worktape, at some point enters s_{comm}, … , finally V moves to s_{end}
  - Let <P,V>(x) be the output of V at the end of the interaction.

Interactive proofs

- Model:
  - For two machines P and V interacting on common input x, **<P,V>(x) denotes the output** of V at the interaction's end.
  - The **transcript of <P,V>(x)** is the concatenation of x and all messages that appeared on the dedicated worktapes.

Interactive proofs

› Definition:

We say that L has an **interactive proof** if there is a probabilistic polytime V such that

- › $x \in L \quad \Rightarrow \quad \exists\ P, \quad \Pr[\ \langle P,V\rangle(x) = 1\ ] = 1$
- › $x \notin L \quad \Rightarrow \quad \forall\ P, \quad \Pr[\ \langle P,V\rangle(x) = 1\ ] \leq \frac{1}{2}$

(note: P above can be all-powerful)

[1] most sources even allow P to be any *function*, rather than just a machine, e.g. P can solve HALT

Interactive proofs

- Theorem (sanity check):

  Every $L \in NP$ has an interactive proof

Interactive proofs

- Theorem (sanity check):

  Every $L \in NP$ has an interactive proof

- Do you think that interaction “buys more”? Are there problems outside NP that have an interactive proof?

Zero-knowledge

› Example 1, revisited:

What did the color-blind instructor **learn** from the interactive proof?

Zero-knowledge

› Example 1, revisited:

What did the color-blind instructor **learn** from the interactive proof?

› Observation:

Given the bit “$x \in L$”, the instructor could’ve produced the transcript by himself, not needing any prover.

Interactive proofs

› Example 1, revisited (cont'd):

Produce an honest transcript **without a prover** when $x_1 \neq x_2$:

Choose a random labeling of x_1, x_2 by 0 and 1,
write the labels secretly on the markers

Randomly choose $b \in \{1,2\}$, write the label of "b" in the transcript

… what did we need the prover for then?

Interactive proofs

- Zero-knowledge, idea:

  Formalize

  > “V learned nothing other than $x \in L$”

  as

  > “V can produce a transcript that, if $x \in L$, is the correct transcript it would’ve gotten with P^*”

Zero-knowledge

› Definition:

An interactive proof is **zero-knowledge** if there is an honest prover P^* and a probabilistic polynomial-time machine Sim such that for every **$x \in L$,**

Sim(x) is $n^{-\omega(1)}$-indistinguishable from the transcript of $\langle P^*, V \rangle(x)$

by any polynomial-time machine.

[1] honest prover = convincing V to accept on $x \in L$

Zero-knowledge

- Theorem:

  Assuming the PRG conjecture, every problem in NP has a zero-knowledge interactive proof.

Zero-knowledge

- Proof idea (baby version, for intuition):

1. reduce to 3COLOR, let c be a coloring
2. prover P^* randomly re-labels c (chooses a permutation of {1,2,3})
3. prover P^* sends a commitment for the re-labelled c'
4. verifier chooses a random edge e = (u,v)
5. prover P^* reveals c'(u), c'(v), verifier accepts iff c'(u) ≠ c'(v)

Zero-knowledge

- Proof idea (baby version, fo

1. reduce to 3COLOR, let
2. prover P* randomly re-l
3. prover P* sends a comr
4. verifier chooses a random edge e = (u,v)
5. prover P* reveals c'(u), c'(v), verifier accepts iff c'(u) ≠ c'(v)

observation:

- if the graph is 3-colorable, all the verifier sees when interacting with P* is two random colors!
- it's possible to simulate that transcript without any prover, just choose two colors random

Zero-knowledge

› Proof idea (baby version, fo

1. reduce to 3COLOR, let
2. prover P^* randomly re-l
3. prover P^* sends a comr
4. verifier chooses a rando
5. prover P^* reveals c'(u), c'(v), verifier accepts iff c'(u) ≠ c'(v)

… (cont'd)

› why is this is a baby version?

when the graph isn't 3-colorable, the probability of catching an edge with two identical colors is 1/poly(n), which is low

the full version improves this probability to ½ using PCPs

Where next?

- You are the **next generation of all of this**
  - research conducted in academia & small parts of industry
- Advanced courses, research projects
  - grad courses at UTSG
  - faculty specializing in complexity

Course outline

- Modeling
- Computability
- Basic complexity
- P vs NP
- Strong forms of P ≠ NP
- Lemons into lemonade

Course outline

- Modeling
- Computability
- Basic complexity
- P vs NP
- Strong forms of P ≠ NP
- Lemons into lemonade