Zero Knowledge Proofs

Introduction to Zero Knowledge Interactive Proofs

Dan Boneh, Shafi Goldwasser, Dawn Song, Justin Thaler, Yupeng Zhang
Classical Proofs

Prime-Number Thm

\[
a^2 + b^2
\]
Proofs

Prover → Proof → Verifier

Claim

accept/reject
Efficiently Verifiable Proofs (NP-proofs)

Prover: Works Hard

Verifier: Polynomial Time

Claim: short proof → accept/reject
Efficiently Verifiable Proofs (NP-proofs)

**Prover P**

- **Claim x**

**Verifier V**

- Accepts x if $V(x, w) = 1$
- Else reject

$|w| = \text{polynomial in } |x|$ is unbounded

$V$ takes time polynomial in $|x|$
Claim: N is a product of 2 large primes

proof=$((p, q))

If N=pq, V accepts
Else V rejects

After interaction, V knows:
1) N is product of 2 primes
2) The two primes p and q
Claim: $y$ is a quadratic residue mod $N$
(i.e. $\exists x \text{ in } \mathbb{Z}_N^* \text{ s.t. } y = x^2 \mod N$)

If $y = x^2 \mod N$, $V$ accepts.
Else $V$ rejects.

After interaction, $V$ knows:

1. $y$ is a quadratic residue mod $N$
2. Square root of $y$ (hard problem equivalent to factoring $N$)
Claim: the two graphs are isomorphic

After interaction, V knows:
1) \( G_0 \) is isomorphic to \( G_1 \)
2) The isomorphism \( \pi \)

Accept if \( \forall i, j: (\pi(i), \pi(j)) \in E_1 \) iff \( (i, j) \in E_0 \).
Efficiently Verifiable Proofs (NP-Languages)

Prover $P$

Claim $x$

Verifier $V$

Works Hard

$V$ Polynomial time

Accepts $x$ If $V(x, w) = 1$ Else reject

Def: A language $L$ is a set of binary strings $x$. 
Efficiently Verifiable Proofs (NP-Languages)

**Def:** $\mathcal{L}$ is an NP-language (or NP-decision problem), if there is a poly ($|x|$) time verifier $V$ where

- **Completeness** [True claims have (short) proofs].
  
  if $x \in \mathcal{L}$, there is a poly($|x|$)-long witness $w \in \{0,1\}^*$ s.t. $V(x, w) = 1$.

- **Soundness** [False theorems have no proofs].
  
  if $x \notin \mathcal{L}$, there is no witness. That is, for all $w \in \{0,1\}^*$, $V(x, w) = 0$. 

Claim $x$  

$\xrightarrow{w} \rightarrow$ accept/reject
Theorem: $y$ is a quadratic residue mod $N$

Proof: $\sqrt{y} \mod N \in Z_N^*$
Zero Knowledge Proofs: Yes

Main Idea:
Prove that
I could prove it
If I felt like it
Zero Knowledge Interactive Proofs
Two New Ingredients

Interactive and Probabilistic Proofs

**Interaction:** rather than passively “reading” proof, verifier engages in a non-trivial interaction with the prover.

**Randomness:** verifier is randomized (tosses coins as a primitive operation), and can err in accept/reject with small probability.
Interactive Proof Model

Comp. Unbounded

Probabilistic Polynomial-time (PPT)
Here is the idea:
How to prove colors are different to a **blind verifier**

**Claim:** This page contains 2 colors

Toss a coin to decide if to flip page over or not.
- Heads: flip page over.
- Tails: don’t flip page over.

Sends resulting page.

- If page is flipped:
  - Set coin’=heads
  - Else coin’=tails

I guess you tossed coin’.

- If coin ≠ coin’, reject, else accept.
Here is the idea:

How to prove colors are different to a blind verifier

Claim: This page contains 2 colors

Toss coin to decide if to flip page over or not

Heads flip, Tails don’t

Sends resulting page

If page is flipped
Set coin’=heads
Else coin’=tails

I guess you tossed coin’

If coin ≠ coin’, reject, else accept

- If there are 2 colors, then Verifier will accept
- If there is a single color, ∀provers Prob_{coins}(Verifier accept) ≤ 1/2
- If repeat i=1..k times and V accept if coin_{i}'=coin_{i} every repetition,
  ∀provers Prob_{coins}(Verifier accept) ≤ 1/2^k
Interactive Proof for $\text{QR} = \{(N, y): \exists x \text{ s.t. } y = x^2 \mod N\}$

Choose random $1 \leq r \leq N$ s.t. $\gcd(r, N) = 1$

Send $s = r^2 \mod n$ and say
- If I gave you square roots of both $s$ and $sy \mod N$, you would be convinced that the claim is true (but also know $\sqrt{y} \mod N$)
- Instead, I will give you a square root of either $s$ or $sy \mod N$ but you get to choose which!

Flip a $b = \text{coin}$ to choose

If $b = 1$: send $z = r$
If $b = 0$: send $z = r \sqrt{y} \mod N$

Accepts only if $z^2 = sy^{1-b} \mod N$
Interactive Proof for $\text{QR} = \{1, \ldots, N\}$

- Sends $s = r^2 \mod n$ and says:
  - If I gave you square roots of both $s$ and $sy \mod N$, you would be convinced that the claim is true (but also know $\sqrt{y} \mod N$).
  - Instead, I will give you a square root of $s$ or of $sy \mod N$ but you get to choose which!

Choose random $1 \leq r \leq N$ s.t. $\gcd(r, N) = 1$

Flip a $b$ to choose:
- If $b = 1$: send $z = r$
- If $b = 0$: send $z = r \sqrt{y} \mod N$

Accepts only if $z^2 = sy^{1-b} \mod N$

- Completeness: If Claim is true, then Verifier will accept.
- Soundness: If Claim is false, $\forall$ provers $\operatorname{Pr}_{\text{coins}}(\text{Verifier accept}) \leq 1/2$.
- Prover only needs to know $x = \sqrt{y}$.
**Interactive Proof for QR**

Repeat 100 times

- **Completeness:** If Claim is true, then Verifier will accept
- **Soundness:** If Claim is false, \(\forall\) provers
  \[ \text{Prob}_{\text{coins}}(\text{Verifier accept}) \leq \left( \frac{1}{2} \right)^{100} \]
- **Prover only needs to know** \(x = \sqrt{y}\)

Choose random \(1 \leq r \leq N\) s.t. \(\gcd(r, N) = 1\)

Send \(s = r^2 \mod n\) and \(s\):

- If I gave you square roots of both \(s\) and \(sy \mod N\) you would be convinced that the claim is true (but also know \(\sqrt{y} \mod N\))
- Instead, I will give you a square root of \(s\) or of \(sy \mod N\) but you get to choose which!
- The fact that I **COULD** (in principle) do both, should convince you

Flip a \(b = \) to choose

If \(b = 1\): send \(z = r\)
If \(b = 0\): send \(z = r \sqrt{y} \mod N\)

Accepts only if \(z^2 = sy^{1-b} \mod N\)
What Made it possible?

- The statement to be proven has **many possible proofs** of which the prover chooses one **at random**.

- Each such proof is made up of exactly 2 parts: seeing either part on its own gives the verifier no knowledge; seeing both parts imply 100% correctness.

- Verifier chooses **at random** which of the two parts of the proof he wants the prover to give him. The ability of the prover to provide either part, convinces the verifier
Definitions:

of Zero Knowledge

Interactive Proofs
**Def:** $(P, V)$ is an interactive proof for $L$, if $V$ is probabilistic poly ($|x|$) time &

- **Completeness:** If $x \in L$, $V$ always accepts.
- **Soundness:** If $x \notin L$, for all cheating prover strategy, $V$ will not accept except with negligible probability.
**Interactive Proofs: Notation**

**Def:** \((P, V)\) is an interactive proof for \(\mathcal{L}\), if \(V\) is probabilistic poly \((|x|)\) and

- **Completeness:** If \(x \in \mathcal{L}\), \(Pr[(P, V)(x) = \text{accept}] = 1\).
- **Soundness:** If \(x \notin \mathcal{L}\), for every \(P^*\), \(Pr[(P^*, V)(x) = \text{accept}] = \text{negl}(|x|)\)

where \(\text{negl}(\lambda) < \frac{1}{\text{polynomial}(\lambda)}\) for all polynomial functions.
Interactive Proofs: Notation

**Def**: $(P, V)$ is an interactive proof for $L$, if $V$ is probabilistic poly ($|x|$) and

- **Completeness**: If $x \in L$, $Pr_{P, V} x = accept = 1$.
- **Soundness**: If $x \notin L$, for every $P^*$, $Pr_{P^*, V} x = accept \leq negl(x)$ where $negl(\lambda) < \frac{1}{\lambda}$ for all polynomial functions.

This is what a proof ultimately is!
Interactive Proofs for a Language $\mathcal{L}$: Notation

**Claim/Theorem x**

Prover

$\begin{align*}
a_1 & \\
q_1 & \\
a_2 & \\
... & \\
\end{align*}$

Verifier

accept/reject

**Def:** $(P, V)$ is an interactive proof for $\mathcal{L}$, if $V$ is probabilistic poly ($|x|$) and

- **Completeness:** If $x \in \mathcal{L}$, $\Pr[(P, V)(x) = \text{accept}] \geq c$
- **Soundness:** If $x \notin \mathcal{L}$, for every $P^*$, $\Pr[(P^*, V)(x) = \text{accept}] \leq s$

Equivalent as long as $c - s \geq 1/\text{poly}(|x|)$
The class of Interactive Proofs (IP)

**Def**: class of languages \( \text{IP} = \{ \text{L for which there is an interactive proof} \} \)
What is zero-knowledge?

For true Statements,

What the verifier can compute
after the interaction =
What the verifier could have computed
before interaction

How do we capture this mathematically?

for every verifier
The Verifier’s View

- After interactive proof, V “learned”:
  - T is true (or $x \in \mathcal{L}$)
  - A view of interaction (= transcript + coins V tossed)

Def: $\text{view}_V(P, V)[x] = \{(q_1, a_1, q_2, a_2, ..., \text{coins of } V)\}$. (probability distribution over coins of V and P)
V’s view gives him nothing new, if he could have simulated it its own s.t `simulated view’ and `real-view’ are computationally-Indistinguishable
Computational Indistinguishability

If no “distinguisher” can tell apart two different probability distributions they are “effectively the same”.

For all distinguisher algorithms $D$, even after receiving a polynomial number of samples from $D_b$, $\text{Prob}[D \text{ guesses } b] < 1/2 + \text{negl}(k)$
Zero Knowledge: Definition

An Interactive Protocol \((P,V)\) is zero-knowledge for a language \(L\) if there exists a \textbf{PPT} algorithm \(\text{Sim}\) (a simulator) such that \textbf{for every} \(x \in L\), the following two probability distributions are \textbf{poly-time} indistinguishable:

1. \(\text{view}_V(P,V)[x] = \{(q_1,a_1,q_2,a_2,\ldots,\text{coins of } V)\}\) (over coins of \(V\) and \(P\))
2. \(\text{Sim}(x)\)

\textbf{Def:} \((P,V)\) is a zero-knowledge interactive protocol if it is \textit{complete, sound and zero-knowledge}
Zero Knowledge: Definition

An Interactive Protocol (P,V) is zero-knowledge for a language $L$ if there exists a PPT algorithm Sim (a simulator) such that for every $x \in L$, the following two probability distributions are poly-time indistinguishable:

1. $\text{view}_V(P,V)[x,1^\lambda] = \{(q_1,a_1,q_2,a_2,...,\text{coins of V})\}$ (over coins of V and P)
2. $\text{Sim}(x,1^\lambda)$

Def: (P,V) is a zero-knowledge interactive protocol if it is complete, sound and zero-knowledge.
What if V is NOT HONEST

An Interactive Protocol (P,V) is **honest-verifier** zero-knowledge for a language $L$ if there exists a PPT simulator $Sim$ such that for every $x \in L$,

$$\text{view}_V(P,V)[x] \approx Sim(x, 1^\lambda)$$

An Interactive Protocol (P,V) is **zero-knowledge** for a language $L$ if **for every PPT $V^*$**, there exists a poly time simulator $Sim$ s.t. for every $x \in L$,

$$\text{view}_V(P,V)[x] \approx Sim(x, 1^\lambda)$$
Flavors of Zero Knowledge

- Computationally indistinguishable distributions = CZK
- Perfectly identical distributions = PZK
- Statistically close distributions = SZK

\[ \text{view}_V(P, V)[x] \approx \text{Sim}(x, 1^\lambda) \]

REAL \[\approx\] SIMULATED
Special Case: **Perfect** Zero Knowledge

The verifier’s view can be exactly efficiently simulated.

`Simulated views' = 'real views'

\[
\text{view}_V(P,V)[x] = \text{Sim}(x)
\]
Working through a Simulation for QR Protocol
Recall the Simulation Paradigm

\( \text{view}_V(P, V): \)

Transcript = \((s, b, z),\)

Coins = \(b\)

\[ s = r^2 \ (\text{mod } N) \]

\[ b \leftarrow \{0, 1\} \]

Check:

\[ z^2 = sy^b \ (\text{mod } N) \]
Recall the Simulation Paradigm

\[ \text{view}_V(P, V) : (s, b, z) \]

\[ \text{sim} : (s, b, z) \]

\[ s = r^2 \pmod{N} \]

\[ b \leftarrow \{0, 1\} \]

If \( b = 0 \): \( z = r \)
If \( b = 1 \): \( z = rx \)

Check:
\[ z^2 = sy^b \pmod{N} \]

PPT “simulator” \( \text{Sim} \)

\( (N, y) \)
Claim: The QR protocol is perfect zero knowledge.

Simulator $S$ works as follows:

1. First pick a random bit $b$.
2. Pick a random $z \in \mathbb{Z}_N^*$.
3. Compute $s = z^2 / y^b$.
4. Output $(s, b, z)$.

view$_V(P, V)$: 

Claim: The simulated transcript is identically distributed as the real transcript.
**Perfect Zero Knowledge: for all V**

**Claim:** The QR protocol is perfect zero knowledge.

**Simulator S works as follows:**

1. First pick a random bit \( b \).
2. pick a random \( z \in \mathbb{Z}_N^* \).
3. compute \( s = z^2 / y^b \).
4. If \( V^*((N, y), s) = b \) output \( (s, b, z) \) if not goto 1 and repeat

\[ s = r^2 \pmod{N} \]

\[ b \leftarrow \{0, 1\} \]

If \( b=0 \): \( z = r \)
If \( b=1 \): \( z = rx \)

Check: \[ z^2 \equiv sy^b \pmod{N} \]

**Claim:** Expected number of repetitions is two

\[ \text{view}_V(P, V) = (s, b, z) \]
ZK proof of Knowledge
Prover seems to have proved more: theorem is correct and that she “knows” a square root mod N

Consider \( L_R = \{ x : \exists w \text{ s.t. } R(x, w) = \text{ accept} \} \) for poly-time relation R.

**Def:** \((P, V)\) is a proof of knowledge (POK) for \( L_R \) if:

\[ \exists \text{ PPT (knowledge) extractor algorithm } E \text{ s.t. } \forall x \text{ in } L, \]
\[ \text{in expected poly-time } E^P(x) \text{ outputs } w \text{ s.t. } R(x, w) = \text{accept}. \]

\( E^P(x) \) (E may run P repeatedly on the same randomness) possibly asking different questions in every executions
This is called the **rewinding technique**
Prover seems to have proved more not only that theorem is correct, but that she “knows” a square root mod N

Consider $L_R = \{x: \exists w \text{ s.t. } R(x, w) = \text{accept}\}$ for poly-time relation $R$.

Def: $(P,V)$ is a proof of knowledge (POK) for $L_R$ if:

$\exists$ PPT (knowledge) extractor algorithm $E$ s.t. $\forall x$ in $L$,
in expected poly-time $E^P(x)$ outputs $w$ s.t. $R(x, w) = \text{accept}$.

[if $\text{Prob}[(P,V)(x) = \text{accept}] > \alpha$, then $E^P(x)$ runs in expected poly($|x|, 1/\alpha$) time]

$E^P(x)$ (may run $P$ repeatedly on the same randomness)
Possibly asking different questions in every executions
This is called the rewinding technique

Extractor $E$

same msg
ZKPOK that Prover knows a square root $x$ of $y \mod N$

**Extractor Algorithm**

**Input:** $(y,N)$

1. Run prover & receive $s$
2. Set verifier message to $\text{head}$; Store $r$

$$s = r^2 \mod N$$

$\text{head}$

$r$
The Rewinding Method

Input: \((y,N)\)

Extractor Algorithm

- \(s=r^2 \mod N\)
- \(rx \mod N\)
- \(\text{tail}\)

Extractor:
On input \((y,N)\)
1. Run prover & receive \(s\)
2. Set verifier message to head; receive and store \(r\)
3. Rewind and 2\textsuperscript{nd} time set verifier message to tail receive \(rx\)
4. Output \(rx/r=x \mod N\)
Recall:

$G_0$ is isomorphic to $G_1$

If there exists an isomorphism $\pi: [N] \rightarrow [N]$, $\forall i, j: (\pi(i), \pi(j)) \in E_1$ if and only if $(i, j) \in E_0$. 
I will produce a random graph $H$ for which

1: I can give an isomorphism $\gamma_0$ from $G_0$ to $H$
OR
2: I can give an isomorphism $\gamma_1$ from $G_1$ to $H$

Thus, $\exists$ isomorphism $\sigma$ from $G_0$ to $G_1$

Verifier, please randomly choose if I should demonstrate my ability to do #1 or #2.

POINT IS: If I can do both, there exists an isomorphism from $G_0$ to $G_1$
**Claims:**

1. Statement true: can answer correctly for $b = 0$ and $1$
2. Statement false: $\text{prob}_b(\text{catch a mistake}) \geq 1 - 1/2^k$
3. Perfect ZK [Exercise]
ZKPOK that **Prover knows an isomorphism** from $G_1$ to $G_2$

**Extractor Algorithm**

1) On input $H$
   - set $coin=\text{head}$
   - Store $\gamma_0$
2) **Rewind** and 2nd time
   - set $coin=\text{tail}$
   - Store $\gamma_1$
3) Output $\gamma_1^{-1}(\gamma_0)$
The first application: Identity Theft [FS86]

For Settings:
- Alice = Smart Card.
- Over the Net
- Breaking ins at Bob/Amazon are possible

Passwords are no good
Zero Knowledge: Preventing Identity Theft

To identify itself prover proves a hard theorem.

PROVER

Smart Card

Hard Theorem: I know a Square root of $y \mod N$

Proof: zero knowledge proof

VERIFIER

ATM/Main Frame
Interesting examples, one application

But, do all NP Languages have Zero Knowledge Interactive Proofs?
Yes: All of NP is in Zero Knowledge

Theorem [GMW86, Naor]: If one-way functions exist, then every L in NP has computational zero knowledge interactive proofs

Ideas of the proof:

1. Show that an NP-Complete Problem has a ZK interactive Proof

 [GMW87] Showed ZK interactive proof for G3-COLOR using bit-commitments

⇒ For any other L in NP, L \neq_p G3-COLOR (due to NPC reducibility)

⇒ Every instance x can be reduced to graph G_x such that

- if x in L then G_x is 3 colorable
- if x not in L then G_x is not 3 colorable
Can you show Zero Knowledge for all of NP [GMW87]

Theorem [GMW86, Naor]: If one-way functions exist, then every language $L$ in NP has computational ZK interactive proofs.

Ideas of the proof:
1. [GMW87] Show that an NP-Complete Problem has a ZK interactive Proof if bit commitments exist.
2. [Naor] One Way functions exist and a bit commitment protocol exists.

Commit(m) → Decommit
Properties of a Bit Commitment Protocol (Commit, Decommit) between Sender S and Receiver R

**Hiding:** \( \forall \text{ receiver } R^* \), after commit stage \( \forall \ b, b' \in \{0,1\} \), view of sender \( R^* \)

\[
\{\text{View}_{R^*}(\text{Sender}(b),R^*)(1^k)\} \approx_c \{\text{View}_{R^*}(\text{Sender}(b'),R^*)(1^k)\} \quad [k=\text{sec. param}]
\]

**Binding:** \( \forall \text{ sender } S^* \), after commit and decommit stage

\[
\text{Prob}[\text{R will accept two different values } b \text{ and } b'] < \text{negl}(k)
\]

K-security parameter

**Ex:** Use (semantically) secure probabilistic encryption scheme Enc

Commit(b) = “sender chooses r and sends c=Enc(b;r)”

Decommit(c) = “sender sends r and b. Receiver rejects unless c=Enc(b;r)”
All of NP is in Zero Knowledge

Theorem [GMW86, Naor]: If one-way functions exist, then every L in NP has computational zero knowledge interactive proofs.

Ideas of the proof:

1. Show that an G3-COLOR has a ZK interactive Proof

Theorem: A graph is 3-color.
Theorem: \text{is G3-COLORABLE}

On common input graph $G = (V,E)$ & prover input coloring $\pi: V \rightarrow \{0,1,2\}$

1. **Prover:** pick a random permutation $\sigma$ of colors $\{0,1,2\}$ & color the graph with coloring $\phi(v) := \sigma(\pi(v))$, and **commit** to each color of each vertex $v$ by running $\text{Commit}(\phi(v))$ protocol

2. **Verifier:** select a random edge $e = (a, b)$ to send to Prover

3. **Prover:** **Decommit** colors $\phi(a)$ & $\phi(b)$ of vertices $a$ and $b$

Decision: Verifier rejects if $\phi(a) \neq \phi(b)$, otherwise Verifier repeats steps 1-3 and accepts after $k$ iterations
Completeness and Soundness

- **Completeness:** if G is 3-colorable, then the honest prover uses a proper 3-coloring & the verifier always accept.

- **Soundness:** If G is not 3-colorable, then for all P*,
  \[ \text{Prob[ Verifier accepts]} < \left(1 - \frac{1}{|E|}\right)^k < \frac{1}{e^{|E|}} \]
  for \( k = |E|^2 \).

- **Zero Knowledge:** Easy to see informally, Messy to prove formally
Simulator $S$ in input $G=(V,E)$: choose at random in advance a challenge $(a,b)$ of the honest verifier $V$.

- Choose random edge $(a,b)$ in $G$
- Choose colors $\phi_a, \phi_b$ in $\{0,1,2\}$ s.t $\phi_a \neq \phi_b$ at random and for all other $v \neq a,b$ set $\phi_a = 2$. Output simulated-view = (commit-transcript to $\phi(v)$ for all $v$, edge =$(a, b)$, decommit-transcript to colors $\phi_a, \phi_b$)
Computational ZK: Simulation for any Verifier V*

Simulator S on input G and verifier V*: For \( i = 1 \) to \( |E|^2 \):

- Choose random edge \((a, b)\) and generate commitments \(com\) to colors as in honest verifier simulation.
- Run \(V^*\) on \(com\) to obtain challenge \((a^*, b^*)\);
  - if \((a^*, b^*) = (a, b)\), then output simulation as honest verifier case,

If all iterations fail, then output \(\perp\).

Claim: If Commitment scheme is Hiding & Binding, then
\[
\forall G, \pi \text{ (a true coloring)} : \text{prob}[\perp \text{ output}] = \text{neg}(|E|) \text{ and if } \perp \text{ is not output, then simulated-view } \approx_c \text{real-view}
\]
Now, we have as many CZK examples as NP-languages

- $n$ is the product of 2 primes
- $x$ is a square mod $n$
- $(G_0, G_1)$ are isomorphic
- Any SAT Boolean Formula has satisfying assignment
- Given encrypted inputs $E(x)$ & program $PROG$, $y = PROG(x)$
- Given encrypted inputs $E(x)$ & encrypted program $E(PROG)$, $y = PROG(x)$

} Stronger Guarantee: PZK
Applications in practice and in theory
 Protocol design applications

• Can prove relationships between \( m_1 \) and \( m_2 \) never revealing either one, only commit(\( m_1 \)) and commit(\( m_2 \)).

Examples: \( m_1 = m_2 \), \( m_1 \neq m_2 \) or more generally \( v = f(m_1, m_2) \) for any poly-time \( f \)

Generally: A tool to enforce honest behavior in protocols without revealing any information. Idea: protocol players sends along with each next-msg, a ZK proof that next-msg = Protocol(history h, randomness r) on history h & c=commit(r)
Possible since \( L = \{ \exists r \text{ s.t. } \text{next - msg} = \text{Protocol}(h, r) \text{ and } c=\text{commit}(r) \} \) in NP.
### Uses for Zero Knowledge Proofs 90-onwards

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation Delegation</td>
<td>Kalai, Rothblum x 2, Tromer,...</td>
</tr>
<tr>
<td>Zero Knowledge and Nuclear Disarmament</td>
<td>Barak et al</td>
</tr>
<tr>
<td>Zero Knowledge and Forensics</td>
<td>Naor et al</td>
</tr>
<tr>
<td>Zcash: Bit Coin with privacy and anonymity</td>
<td>BenSasson, Chiesa et al</td>
</tr>
<tr>
<td>Zero Knowledge and Verification Dilemmas in the Law</td>
<td>Bamberger et al</td>
</tr>
</tbody>
</table>
### Complexity Theory: Randomized Analogue to NP

<table>
<thead>
<tr>
<th>Efficiently Solvable</th>
<th>P</th>
<th>BPP (randomized poly time)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efficiently Verifiable</td>
<td>NP</td>
<td>IP</td>
</tr>
</tbody>
</table>

**Q:** Is IP greater than NP?
Claim: $G_0$ is **Not Isomorphic** to $G_1$
(in co-NP, not known to be in NP)

Shortest classical proof:
$\approx$ exponential $n!$ for $n$ vertices

But can convince with an efficient interactive proof
Graph **Non-Isomorphism in IP**

**Claim:** Completeness & Soundness hold

Input: \((G_0, G_1)\)

- Flip coin \(c \in \{0, 1\}\)
- Pick random \(\gamma\)

- If \(H\) isomorphic to \(G_0\) then \(b = 0\), else \(b = 1\)
- Reject if \(b \neq c\)
- Accept after \(n\) repetitions
Graph Non-Isomorphism in IP

Input: \((G_0, G_1)\)

1. Flip coin \(c \in \{0,1\}\) and pick random \(g\).
2. \(H = \gamma(G_c)\)
3. If \(H\) isomorphic to \(G_0\) then \(b = 0\), else \(b = 1\).
4. Reject if \(b \neq c\), Accept otherwise.

Not ZK! \(V^*\) can learn if graph \(H\) of its choice is isomorphic to \(G_0\) or \(G_1\).

Idea for fix: \(V\) proves to \(P\) in ZK that he knows an isomorphism \(\gamma\).
Arthur-Merlin Games [BaM85]

GNI requires verifier to keep its coins secret as in IP protocols
Is coin privacy necessary?

**Theorem [GoldwasserSipser86]:** \( \text{AM (protocols with Public Coins)} = \text{IP} \)

**Idea:** Merlin proves to Arthur “the set of private coin executions that would make Verifer accept” is large. **Technique**= prove lower bound on size of sets
AM Protocols enable “in practice” removal of interaction: the Fiat-Shamir Paradigm [FS87]

- Let $H: \{0,1\}^* \rightarrow \{0,1\}^k$ be a cryptographic Hash function
- Can take an AM protocol
- Replace by

Fiat-Shamir Heuristic:
If $H$ is random-oracle, then completeness & soundness hold, Use $H$ as hash function
AM Protocols suggest “in practice” removal of interaction: the Fiat-Shamir Paradigm[FS87]

Warning: this does **NOT** mean every interactive ZK proof can transform to AM protocols and then use Fiat-Shamir heuristic,
Since IP = AM transformation requires extra super-polynomial powers from Merlin
And for Fiat-Shamir heuristic to work, Prover must be computationally bounded so not to be able to invert H
Yet, many specific protocols, can benefit from this heuristic

Fiat-Shamir Heuristic:
If H is random-oracle, then completeness & soundness hold

\[(a_1, H(a_1), a_2) \rightarrow V(x, a_1, H(x, a_1), a_2) = \text{Accept or Rejects}\]
AM Protocols suggest “in practice” removal of interaction: the Fiat-Shamir Paradigm [FS87]

- Let $H: \{0,1\}^* \rightarrow \{0,1\}^k$ be a cryptographic Hash function
- Can take an AM protocol

Q: What if first message are coins from Arthur?

Idea (used later in course extensively):
Post first message coins as a “publicly” chosen randomness for all to see and then apply Fiat-Shamir heuristics to get non-interactive proofs

$V(x, a_1, a_2) = \begin{cases} \text{Accept} & \text{or Rejects} \\ \end{cases}$
IP: Complexity Theory Catalyst

Decoupled “Correctness” from “Knowledge of the proof”

Ask new questions about nature of proof

Questions have been asked and answered in last 30+ years leading up to current research on Provably outsourcing computation
Classically: Can Efficiently Verify

<table>
<thead>
<tr>
<th>Complexity Class</th>
<th>Can Efficiently Verify</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NP</strong></td>
<td>∃ solution</td>
</tr>
<tr>
<td><strong>Co-NP</strong></td>
<td>0 solutions</td>
</tr>
<tr>
<td><strong>#P</strong></td>
<td>$2^{100} - 13$ solutions</td>
</tr>
<tr>
<td><strong>PSPACE</strong></td>
<td>$\forall \exists \forall ... \exists \forall$</td>
</tr>
</tbody>
</table>

Can you prove more via interactive proofs?
Interactively Provable = PSPACE

[FortnowKarloffLundNissan89, Shamir89]

Other Ways to define probabilistic proof systems?
The Arrival of the Second Prover (MIP)

[BenorGoldwasserKilianWigderson88]

Unconditionally: NP = PZK

NP

Co-NP

#P

PSPACE

Could two prove more than one?

Intuition: Can check consistency, Verifier catches provers if deviate

Accept/Reject
The Second Prover is a Game Changer (MIP)

<table>
<thead>
<tr>
<th>Complexity Class</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>✓</td>
</tr>
<tr>
<td>Co-NP</td>
<td>✓</td>
</tr>
<tr>
<td>#P</td>
<td>✓</td>
</tr>
<tr>
<td>PSPACE</td>
<td>✓</td>
</tr>
<tr>
<td>NEXPTIME</td>
<td>✓</td>
</tr>
</tbody>
</table>

PCP theorem \cite{FGLSS, AS, ALMSS91}: NP statements can be verified with high prob. by only reading a **constant** number of bits of the proof → NP-Hardness of Approximation Problems

Accept/ Reject
Q: Can the correctness of a Quantum polynomial time computation be checked by a classical verifier?

**Theorem [ReichardtUngerVazirani13]:**
A classical Verifier can verify the computation of two entangled but non-communicating poly-time quantum algorithms.
Quantum MIP is All Powerful

2020: $\text{MIP}^* = \text{Recursively Enumerable Languages}$

[Ji, Natarajan, Vidick, Wright, Yuen]
Aside:
The Resistance
An Interactive Non-deterministic Turing Machine (INTDM) is formed by two communicating modules: a guesser G and a checker C. The checker is a probabilistic Turing Machine. G and C share read-only tape in which the input is given. We need the application context, as C, the computa-

d to be 1983-1985 (The Resistance)
1983–1985 (The Resistance)

Revised Version, Dec 8, 198

The Information Content of Proof Systems

1. Introduction

1.1 The goal

The goal is contained in a

Example associated with graphs. A standard in exhibiting

This is the first of all weig

than B. Similar
1983–1985 (The Resistance)

Let me show you how to do it!

Dec 7, 1984

In traditional communication as much as possible, we have generally thought that the problem at all communicating the knowledge efficiently, but the whole problem is making sure not too much knowledge has been communicated.
1985 (The Acceptance)

We are very happy to inform you that your paper
“The Knowledge Complexity of Interactive Proof Systems”
has been selected for presentation at the 17th Symposium on Theory of
Computing
Broader Lessons

- Pay attention to good ideas

- It may take a long time (>30 years) to go from the basic idea to impact