NON-BLACK-BOX ZK
(Barak’s Protocol)

ALON ROSEN
IDC HERZLIYA
**Goal**: construct CZK argument $\forall L \in \text{NP}$

- with negligible soundness
- a constant number of rounds
- and public-coin

**Need to address:**

- How to use $V^*$’s code (BB impossibility)
- $V^*$’s running time is not a-priori bounded
• No $L \not\in \text{BPP}$ has a \textbf{black-box ZK} protocol that is:
  • constant-round
  • negligible-soundness
  • public-coin
• So for $L \not\in \text{BPP}$ must use a \textbf{non-black box simulator}
• On the one hand, $\forall V^* \exists S$ should be easier than $\exists S \forall V^*$
• On the other hand, where do we even begin?
  • Reverse engineering $V^*$ is difficult!
  • \textbf{Key insight}: there is no need to reverse engineer
  • Enough for $S$ to prove that he possesses $V^*$’s code
Theorem [B’01]: If CRH exist, every \( L \in \text{NP} \) has a constant-round, public-coin, negligible-soundness, ZK argument.

- **Idea**: enable usage of verifier’s code as a “fake” witness
- In the real proof, the code is \( V \)’s random tape
- In simulation, the code is \( V^* \)’s “next-message function”
- Since \( P \) does not have access to \( V \)’s random tape in real interactions, this will not harm soundness
- The simulator \( S \), on the other hand, will be always able to make verifier accept since it obtains \( V^* \)’s code as input
Collision-Resistant Hash Functions

**Definition:** \( H_k : \{0,1\}^* \rightarrow \{0,1\}^k \) is \((t, \varepsilon)\)-CRH if \( \forall \) time-\( t \) \( A \)

\[ Pr[A \text{ finds a collision in } h \in_R H_k] \leq \varepsilon \]

**Collision:** \( x \neq x' \) such that \( h(x) = h(x') \)

**Candidate CRHs:**

- **Discrete-log-based:** \( g^{x_L} h^{x_R} \mod P \)
- **SIS:** \( Ax \mod q \)
- **SHA:** \( h(x_L, x_R) \)

**Later:** \( H_k : \{0,1\}^* \rightarrow \{0,1\}^k \) from \( h : \{0,1\}^{2k} \rightarrow \{0,1\}^k \)
Constant-Round ZK Arguments for NP
The Basic Idea

witness \( w \)  \[ P \]  \( x \in L \)  \[ V \]

\[ \begin{align*} c &= \text{Com}(0^k) \\ r &\in R \{0,1\}^{2n} \end{align*} \]

WIAOK statement: \( \exists w, \pi, z \text{ s.t.} \)

1. \( (x, w) \in R_L \text{ or} \)
2. "\( c \) is a commitment to a program \( \pi \text{ s.t. } \pi(z) = r \) within \( t(n) \) steps"

NTIME(\( t(n) \)) statement

Intuition:

- In the real interaction \( P \) cannot predict the random string \( r \)
- In simulation, \( r = V^*(c) \) so \( S \) can set \( \pi = V^* \) and \( z = c \)
Completeness

Use \( w \) to prove

\[
\text{WIAOK statement: } \exists w, \pi, z \text{ s.t.}
\begin{align*}
1. \quad (x, w) &\in R_L \text{ or} \\
2. \quad \text{“} c \text{ is a commitment to a program } \pi \text{ s.t. } \pi(z) = r \text{ within } t(n) \text{ steps”}
\end{align*}
\]

\[
\begin{align*}
\text{witness } &w \quad \text{P} \quad x \in L \quad \text{V} \\
\end{align*}
\]

\[
\begin{align*}
c = \text{Com}(0^k) \\
r
\end{align*}
\]

ACCEPT
Soundness

WIAOK statement: \( \exists w, \pi, z \text{ s.t.} \)

1. \( (x, w) \in R_L \) or
2. "\( c \) is a commitment to a program \( \pi \) s.t. \( \pi(z) = r \) within \( t(n) \) steps"

\[
\forall \pi, Pr_r[\exists z \in \{0,1\}^n, \pi(z) = r] \leq 2^n \cdot 2^{-2n} = 2^{-n}
\]
Zero-Knowledge

Simulator $S$ \hspace{1cm} $x \notin L$ \hspace{1cm} $V^*$

\[
\begin{align*}
&c = \text{Com}(V^*) \\
r &\leftarrow r = V^*(c)
\end{align*}
\]

Use
$\pi = V^*$
$z = c$

to prove

WIAOK statement: $\exists w, \pi, z$ s.t.
1. $(x, w) \in R_L$ \textbf{or}
2. “$c$ is a commitment to a program $\pi$ s.t. $\pi(z) = r$ within $t(n)$ steps”

Cannot distinguish if 1 or 2

By definition, $\pi(z) = V^*(c) = r$
• Simulator runs in strict polynomial time
• Possession of $V^*$ is sufficient. No reverse engineering!

First technical issue:
• $V^*$’s size is $\text{poly}(n)$, but not a-priori bounded
• In particular, how can $c = \text{Com}(V^*)$ accommodate $V^*$?
• **Solution**: use $h: \{0,1\}^* \rightarrow \{0,1\}^k$ to compute $\text{Com}(h(V^*))$

Second technical issue:
• Running time $t(n)$ of $V^*$ not bounded by any fixed $\text{poly}(n)$
• So $\text{NTIME}(t(n))$ relation in WIAOK is not an NP-relation
• **Solution**: WIAOK that handles $\text{NTIME}(n^{\omega(1)})$ relations
A constant-round ZK Argument

**Witness** $w$  $P$  $x \in L$  $V$

$h \leftarrow h$

$c = \text{Com}(0^n) \rightarrow c$

$r \leftarrow r$

**WIAOK Statement:** $\exists w, \pi, z \text{ s.t.}$

1. $(x, w) \in R_L$ **or**
2. “$c$ is a commitment to $h(\pi)$ where $\pi$ is a program s.t. $\pi(z) = r$ within $t(n)$ steps”

$H_k : \{0,1\}^* \rightarrow \{0,1\}^k$

$h \in R H_k$

$r \in R \{0,1\}^{2n}$
The Relation $R_{SIM}$

**WIAOK statement**: \( \exists w, \langle \pi, s, z \rangle \) s.t.

1. \((x, w) \in R_L\) \textbf{or}
2. \((\langle h, c, r \rangle, \langle \pi, s, z \rangle) \in R_{SIM}\)

\((\langle h, c, r \rangle, \langle \pi, s, z \rangle) \in R_{SIM}:

1. \(|z| \leq |r| - n\)
2. \(c = \text{Com}(h(\pi), s)\) \textbf{and}
3. \(\pi(z) = r\) \textbf{within } t(n) \textbf{ steps}
The Universal Language $L_U$

**Goal:** handling $\text{NTIME}(t(n))$ statements for $t(n) = n^{\omega(1)}$

Consider the universal language $L_U$:

$$y = (M, x, t) \in L_U$$

$$\uparrow$$

$$\exists w, M(x, w) = \text{ACCEPT within } t \text{ steps}$$

- Every $L \in \text{NP}$ is linear-time reducible to $L_U$
- A proof system for $L_U$ enables to handle all $\text{NP}$-statements
- More importantly, a proof system for $L_U$ enables to handle $\text{NTIME}(n^{\omega(1)})$ statements and even beyond ($\text{NEXP}$)
Universal Arguments
Universal Argument Systems

\[ y = (M, x, t) \in L_U \iff \exists w, M(x, w) = \text{ACCEPT in } t \text{ steps} \]

**Definition [K’91, M’91, BG’02]:** A universal argument system for \( L_U \) is a pair \((P, V)\) such that \(\forall y = (M, x, t)\):

**Efficient verification:** \(V\) runs in \(\text{poly}(|y|)\) time

**Completeness:** If \( y \in L_U \), then \( \Pr[(P, V) \text{ accepts } y] = 1 \)

Moreover, \( P \) runs in time \(\text{poly}(t)\)

**Computational soundness:** If \( y \notin L_U \), then \( \forall \text{PPT } P^* \)

\[ \Pr[(P^*, V) \text{ accepts } x] \leq \text{neg}(n) \]

**Theorem:** If CRH exist, \( L_U \) has a universal argument
Building block: PCP Proof System

Makes use of a PCP[\(O(\log), \text{poly}\)] system for \(L_U\)

What is a PCP[\(O(\log), \text{poly}\)] proof system?

• It is a PPT \(V_{PCP}\) with access to an oracle \(\pi_y\) that represents a proof for \(y \in L_U\) in redundant form

• \(V_{PCP}\) (non-adaptively) queries \(q\) oracle bits of \(\pi_y\) where

\[
q = \text{poly}(|y|)
\]

• the bit positions are determined by \(V_{PCP}\)‘s coin tosses

• the number of coins tossed by \(V_{PCP}\) is \(O(\log t)\)

• and the length of \(\pi_y\) is

\[
\exp(O(\log t)) = \text{poly}(t)
\]
PCP Reduction

\[ y = (M, x, t), w \]

\[ \pi_y \]

length = \(\text{poly}(t)\)

\[ q = \text{poly}(|y|) \text{ queries} \]

P’s complexity

V’s complexity

the \(q\) queries are determined by

\[ V_{\text{PCP}}(r) \text{ where } r \in \{0,1\}^{0(\log t)} \]
Commitment with Local Decommitment

Problem: the PCP is too long to be sent to $V$ in its entirety

Solution: commit to $\pi_y$ and allow “local decommitment”

$H$ is computationally binding - built using CRH $h$
The Protocol

\[ y = (M, x, t) \in L_U \]

\[ H_k: \{0,1\}^* \rightarrow \{0,1\}^k \]

\[ h \in_R H_k \]

\[ r \in_R \{0,1\}^{O(\log t)} \]

Authenticated replies to \( q \) queries \( V_{PCP}(r) \) with respect to \( c \)

Time

\[ poly(q) = poly(|y|) \]
Completeness

Completeness of PCP

Authenticated replies to $q$ queries $V_{PCP}(r)$ with respect to $c$

$\pi_y \downarrow$ completeness of PCP

$\{\}$

$w$ witness

$P$ $y \in L_U$

$V$

$\{\}$

$h$

$c = H(\pi_y)$

$r$

ACCEPT
Computational Soundness

soundness of PCP and binding of $H$

Authenticated replies to $q$ queries $V_{PCP}(r)$ with respect to $c$

Recall: binding of $H$ is computational - built using CRH $h$
Interlude: Merkle Trees
Merkle Tree

\[ h: \{0,1\}^{2k} \rightarrow \{0,1\}^k \quad H(x) \quad H: \{0,1\}^{Nk} \rightarrow \{0,1\}^k \]

\[ h(x_L, x_R) \]

\[ N = x2^n \quad \log N = n \]
Merkle Tree: Collision Resistance

\[ x \neq x', \quad H(x) = H(x') \]

\[ x_i \neq x'_i \]

Computationally (globally) binding
Merkle Tree: Local Decommitment

Authentication path:
2 log \( N \) − 1 labels

Computationally (locally) binding

\[ H(x) \]
Back to ZK
Arguments for NP
Recall: Barak’s Protocol

\[ x \in L \]

\[ h \in_R H_k \]

\[ r \in_R \{0,1\}^{2n} \]

WIUAOK statement: \( \exists w, \pi, z \text{ s.t.} \)

1. \((x, w) \in R_L\) or
2. “\(c\) is a commitment to \(h(\pi)\) where \(\pi\) is a program s.t. \(\pi(z) = r\) within \(t(n)\) steps”

So far: we only saw how to build UAOK. What about WI?
WI Universal Arguments

\[ y \in L_U \]

\[ c = \beta \pi_y \]

\[ V \]

\[ \alpha \]

\[ \gamma \]

\[ PCF_\delta \text{plies} \]

\[ P \]

\[ V \]

\[ P_{WI} \]

\[ y \in L_U \]

\[ c = Com(\beta) \]

\[ \alpha \]

\[ \gamma \]

\[ d = Com(\delta) \]

\[ V_{WI} \]

WIAOK statement: \( \exists \beta, \delta \text{ s.t.} \)

1. \( c = Com(\beta) \)

2. \( d = Com(\delta) \)

3. \( V(\alpha, \beta, \gamma, \delta) = \text{ACCEPT} \)

Subtle point: actually run \( k \) parallel copies of ZKPOK with constant soundness error
Summary

Saw:
- CZK argument $\forall L \in \text{NP}$
- with negligible soundness
- a constant number of rounds
- and public-coin

Tools:
- Non-black-box simulation
- WI universal arguments
Follow-up Work (2001-2012)

- Resettably-sound ZK [BGGL’01, CPS’13, COPVV’13]
- Constant-round bounded-conc. ZK and MPC [B’01, PR’03]
- Constant-round ZK with strict poly-time sim. [BL’02]
- Simultaneously resettable ZK and MPC [DGS’09, GM’11]
- Constant-round covert MPC [GJ’10]
- Constant-round public-coin parallel ZK [PRT’11]
- Simultaneously resettable WI-POK [COSV’12]
- Constant-round conc. ZK from iO [CLP’13, PPS’13, CLP’15]
- Concurrent secure computation [GGS’15]
New non-BB Techniques

[BP’12]:
- Impossibility for obfuscation $\rightarrow$ non BB simulation
- In particular, no use of PCP

[BKP’15]:
- Homomorphic trapdoors
- Enables to break all Black-Box barriers for e.g. WH
Food for Thought
Efficiency of universal arguments depends on:

- **Number** \( q \) **of oracle queries made by** \( V_{PCP} \) **to** \( \pi_y \)

\[ q = poly(|y|) \]

- **Length of** \( \pi_y \) **- depends on number of coins tossed by** \( V_{PCP} \)

\[ exp(O(log t)) = poly(t) \]

- **Optimizing params:**
  - Larger alphabet size
  - Trading off prover/verifier time

- **Less modular design and/or other models:**
  - Interactive PCPs/oracle IPs
  - Using homomorphism of commitments
• Can turn Merkle-tree into statistically hiding:
  • Generically
  • Assuming $h$ is a random oracle

Open questions:
• Is $O(qk \log N)$ optimal?
• In practice $N$ can be quite large
• Bulletproofs is $O(q + k \log N)$ but verifier space is $N$
• Lattices/amortization gets $O(q + k\sqrt{N})$

• Ideally $O(q + k \log N)$ size and verification time
• Define what it means to be secure
• Build a protocol/scheme
• Prove that protocol/scheme satisfies definition

• First feasibility then efficiency
• Relax definitions
History

Rafael Pass
Nir Bitansky
Dakshita Khurana
Omer Paneth

Rachel Lin
Kai-Min Chung
Dustin Tseng
Muthuramakrishnan Venkitasubramaniam

Vipul Goyal
Abhishek Jain
Ivan Visconti
The End

Questions?