Batch Verification for Statistical Zero-Knowledge Proofs

Inbar Kaslasi
Technion

Guy N. Rothblum
Weizmann Institute

Ron D. Rothblum
Technion

Adam Sealfon
UC Berkeley

Prashant N. Vasudevan
UC Berkeley

Statistical Zero-Knowledge Proofs

- Zero-knowledge proofs [GMR89] are an amazing and incredibly influential notion
- ZK proof lets a prover P to convince a verifier V of the validity of some statement without revealing any additional information
- We focus on SZK
  - ZK and soundness are information theoretic
  - Contains many problems studied in cryptography (e.g., variants of QR, dlog, LWE)
  - Has rich structure (see e.g. [Vad99])

Statistical Zero-Knowledge Proofs

Def: (P, V) is a **statistical zero-knowledge (SZK)** proof if

- $x \in \mathrm{YES} \rightarrow \mathrm{V}$ accepts w.h.p. when interacting with P
- $x \notin \mathrm{NO} \rightarrow \mathrm{V}$ rejects w.h.p. when interacting with any prover P^*
- For every poly-time verifier V^* there exists a poly-time Sim s.t. for any $x \in \mathrm{YES}$

$$\Delta\big((\mathrm{P}, \mathrm{V}^*)(x), Sim(x)\big) \leq neg$$

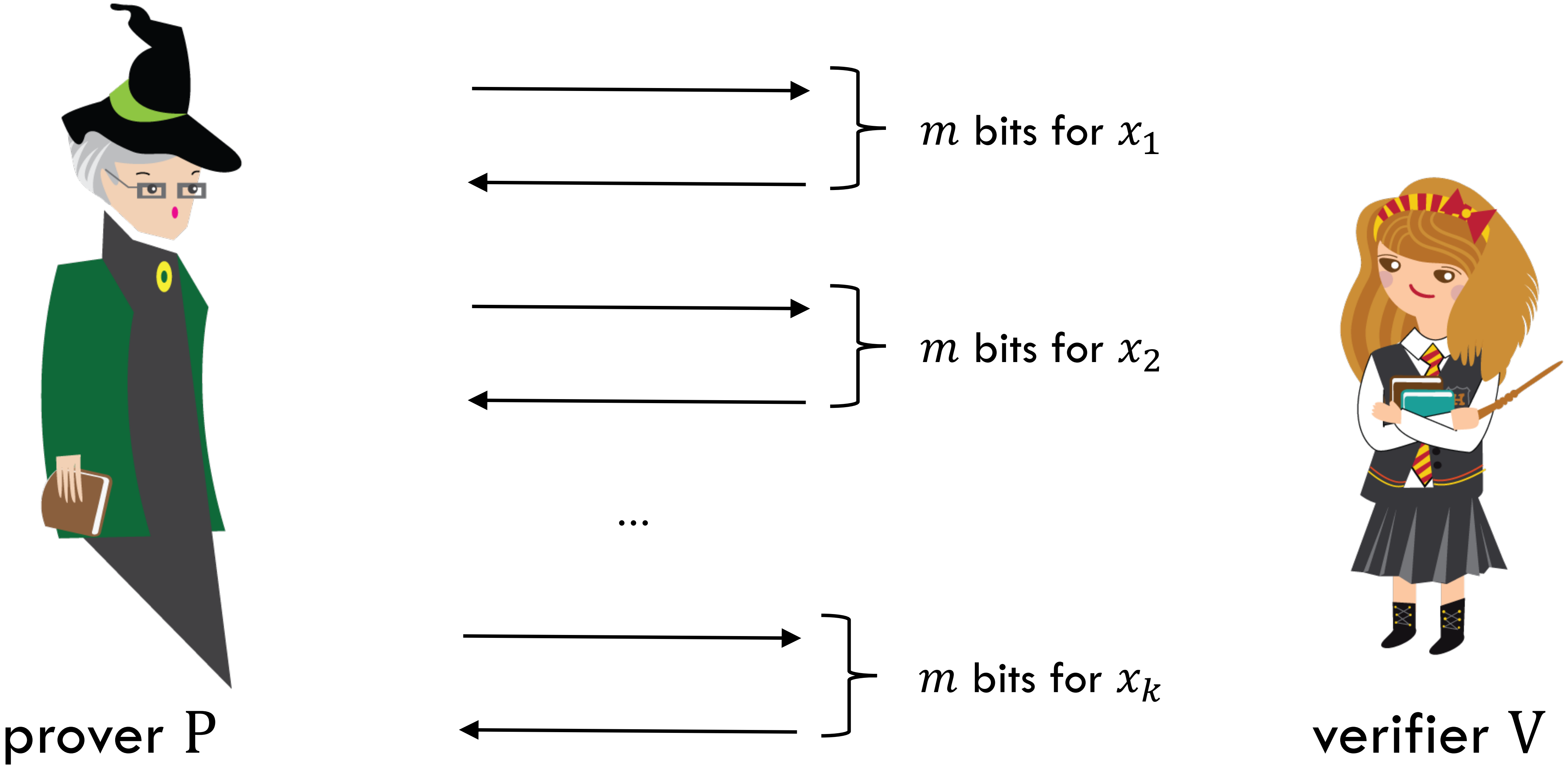
- We also consider a weaker notion of honest verifier SZK

Batch Verification

V wants to check that k statements $x_1, x_2, \ldots, x_k$ are all true,

- Accept if $x_1, x_2, \ldots, x_k$ are all YES instances
- Reject if at least one x_i is a NO instance

Naïve Solution



Batch Verification

Communication is a key resource in modern networks

Verifying one instance takes m communication. Can we verify k instances with less than $m \cdot k$ communication?

Prior Work

[LFKN92, Sha92] Batching for IP via $\mathrm{IP} = \mathrm{PSPACE}$

- Inefficient prover

[RRR16, RRR18, RR20] Batching for UP with communication $\mathrm{poly}(m, \log k)$ (m = witness length)

- Efficient prover

[Kil92, BHK17] Batching with computational soundness (under crypto assumptions)

This Work: Batch Verification for SZK

Main question: suppose $\Pi \in \mathrm{SZK}$, can we verify that $x_1, \ldots, x_k \in \Pi_{\mathrm{YES}}$ in zero knowledge with non trivial communication?

Why? natural problem, also batch verification of signatures, public-keys

Main results: We give a partial positive answer.

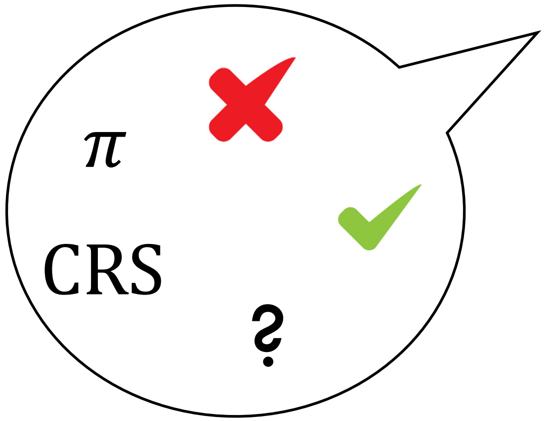
Non-Interactive Statistical Zero-Knowledge [BFM88]

CRS:

0	0	1	0	1	1	1	0	1	0

Prover

π →



Verifier

$\exists$ poly-time Sim s.t. for any $x \in \text{YES}$: $\Delta\big((\text{CRS}, \pi), Sim(x)\big) \leq neg$

Our Results

Main Thm: Every $\Pi \in \mathrm{NISZK}$ has an HVSZK batch-verification protocol with $k + \mathrm{poly}(n)$ communication

Shown via two steps:

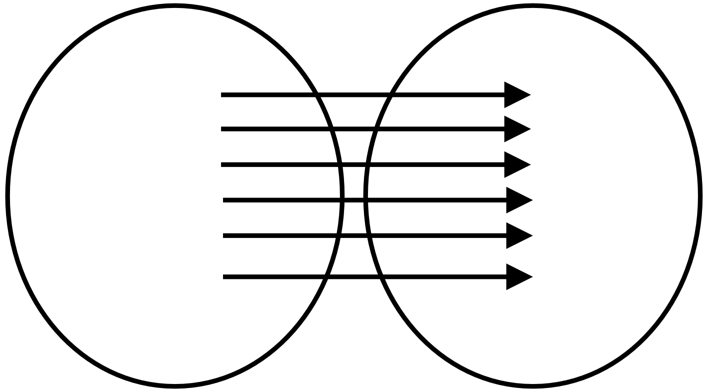
- New NISZK complete problem: *Approximate Injectivity* (AI)
- HVSZK batch-verification protocol for AI

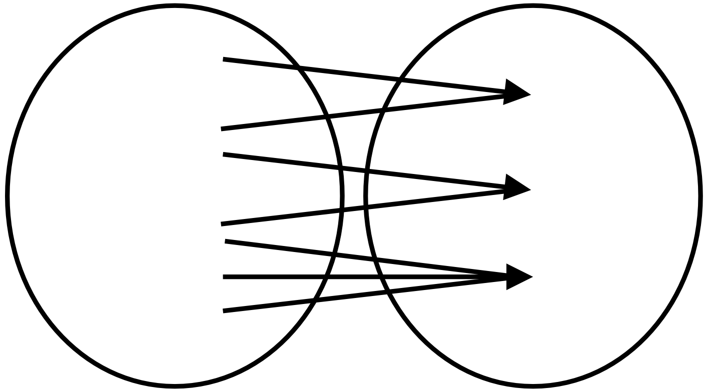
- In this talk we ignore polylog factors

Warmup: Batch Verification for Permutations

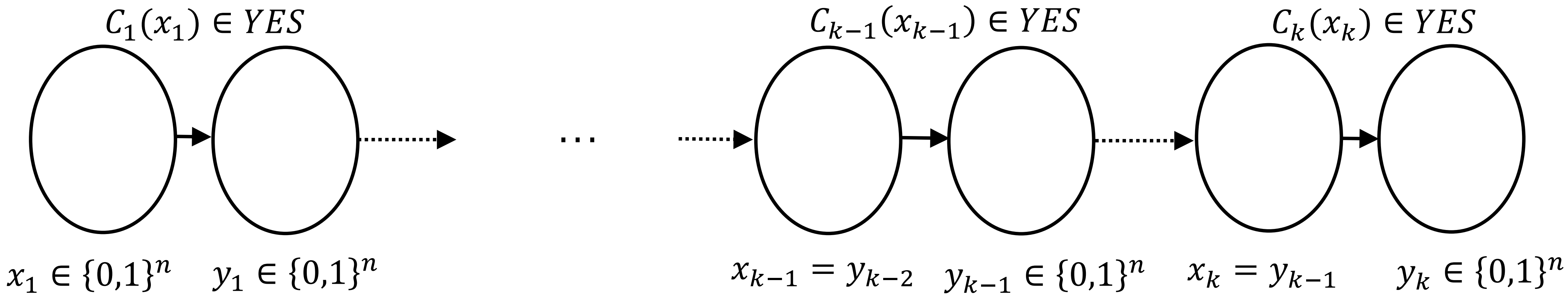
Input: length-preserving circuit $C: \{0,1\}^n \to \{0,1\}^n$

- **YES case:** circuit defines a permutation
- **NO case:** every image has at least two preimages





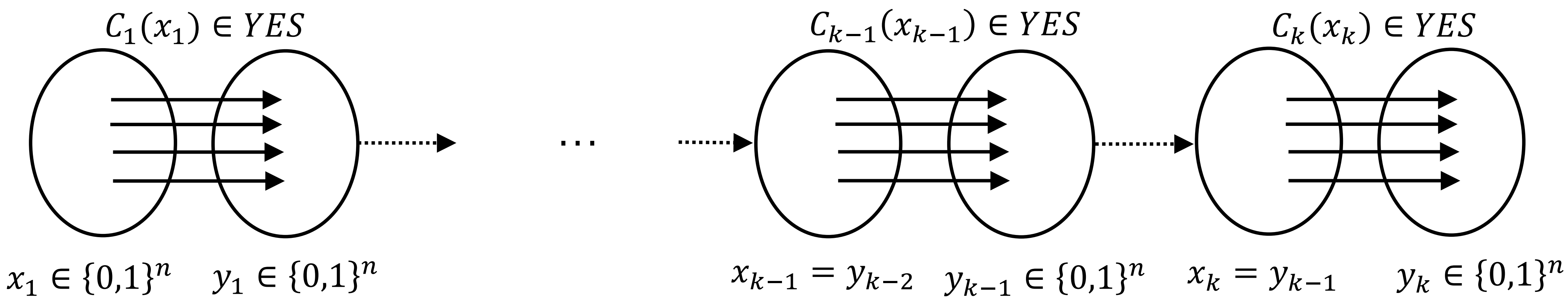
HV Public-Coin Batching for PERM



HVSZK batching protocol:

- V samples $x_1 \in \{0,1\}^n$ and sends $y_k = C_k\big(C_{k-1} \ldots (C_1(x_1)\big)$ to P
- P sends x_1' s.t. $y_k = C_k\big(C_{k-1} \ldots (C_1(x_1')\big)$
- V checks that $x_1 = x_1'$

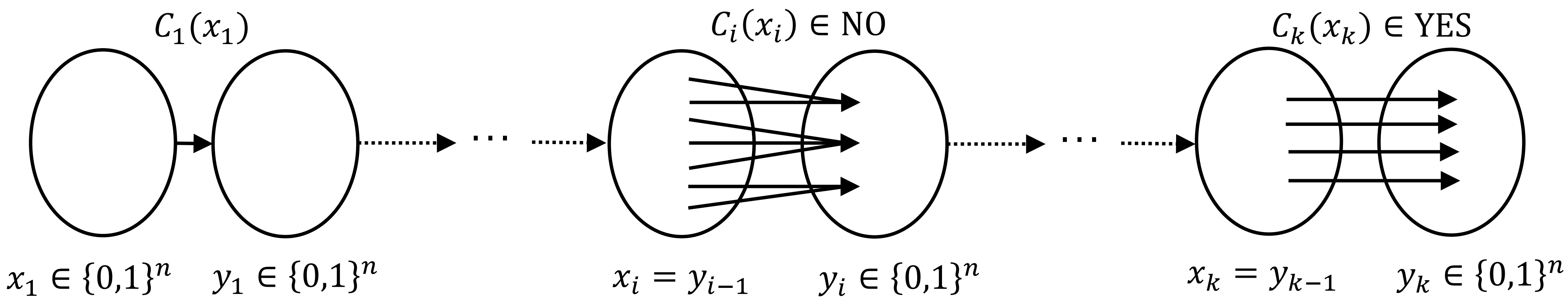
PERM-YES Cases



HVSZK batching protocol:

- V samples $x_1 \in \{0,1\}^n$ and sends $y_k = C_k\big(C_{k-1} \ldots (C_1(x_1)\big)$ to P
- P sends x_1' s.t. $y_k = C_k\big(C_{k-1} \ldots (C_1(x_1')\big)$
- V checks that $x_1 = x_1'$

PERM-NO Cases



HVSZK batching protocol:

- V samples $x_1 \in \{0,1\}^n$ and sends $y_k = C_k\big(C_{k-1} \ldots (C_1(x_1)\big)$ to P
- P sends x_1' s.t. $y_k = C_k\big(C_{k-1} \ldots (C_1(x_1')\big)$
- V checks that $x_1 = x_1'$

The Approximate Injectivity Problem AI_δ

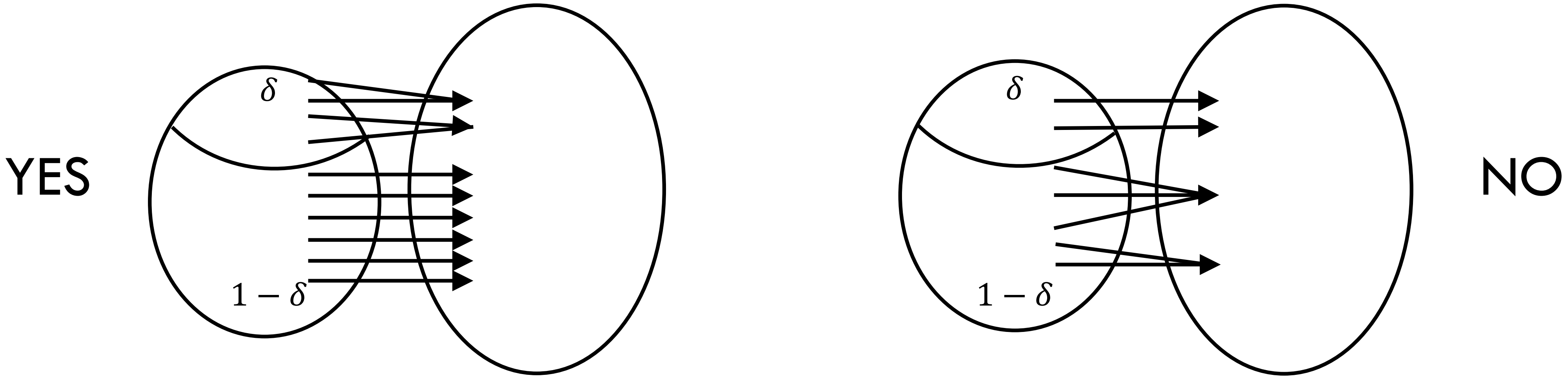
- **Input:** circuit $C: \{0,1\}^n \to \{0,1\}^m$
- **YES cases:** All but δ fraction of inputs are mapped injectively by C

$$\Pr_{x \leftarrow \{0,1\}^n}\left[\left|C^{-1}(C(x))\right| > 1\right] \leq \delta(n)$$

- **NO cases:** At most δ fraction of inputs are mapped injectively by C

$$\Pr_{x \leftarrow \{0,1\}^n}\left[\left|C^{-1}(C(x))\right| > 1\right] \geq 1 - \delta(n)$$

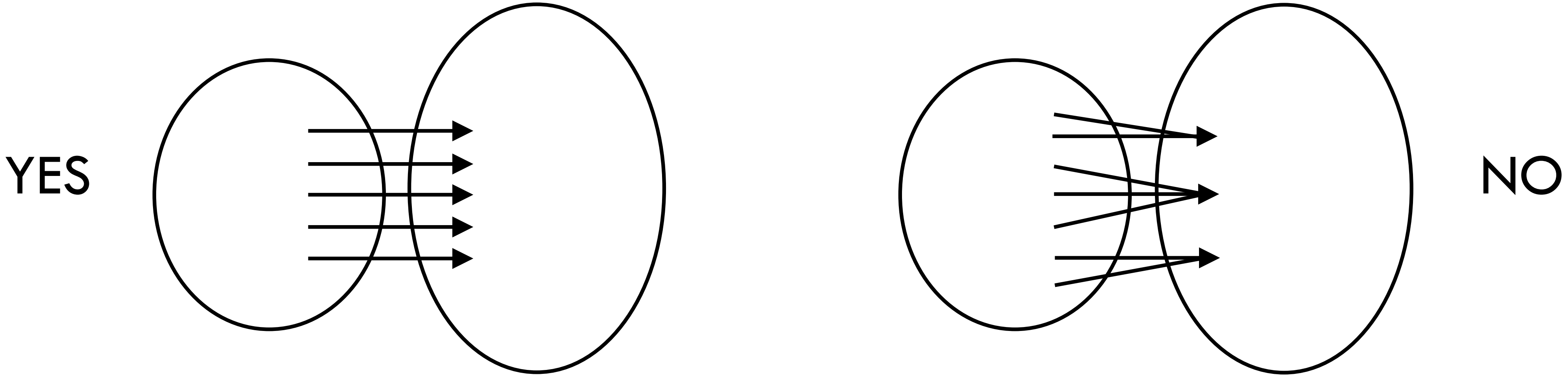
Later: AI_δ is NISZK-hard



This Talk: Exact Injectivity (AI_0)

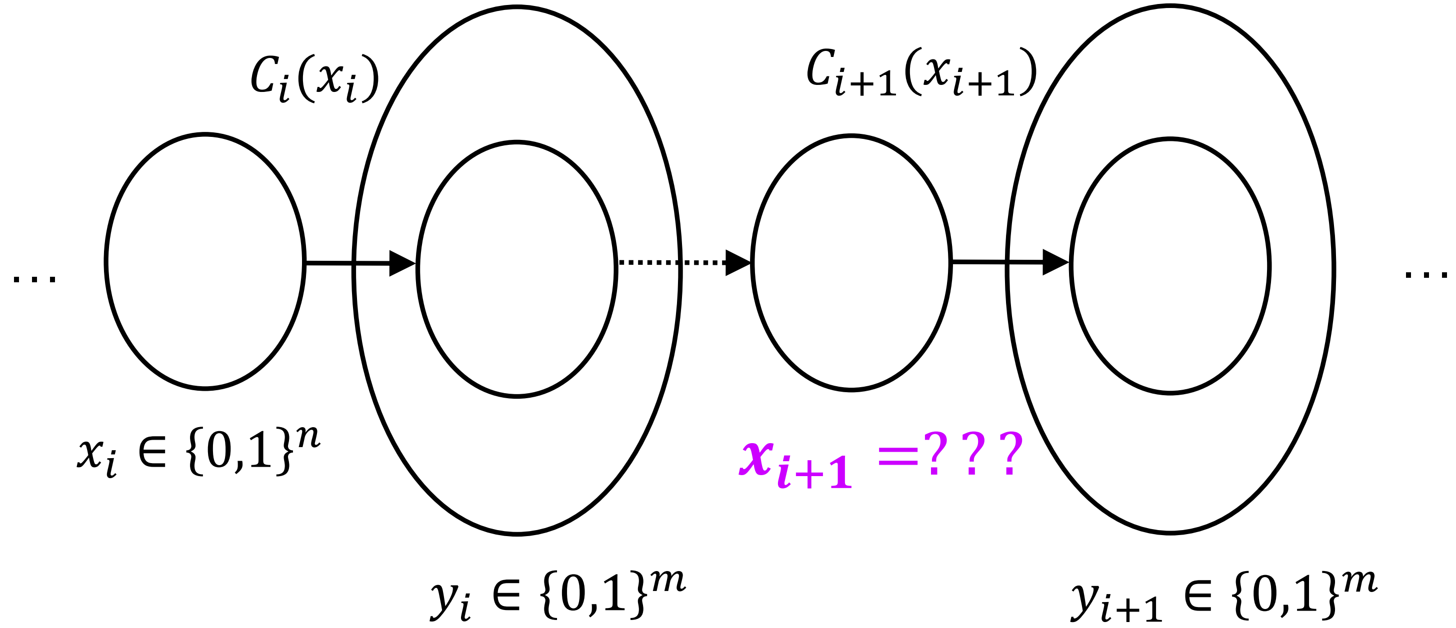
- For simplicity, $\delta = 0$

➢ **Goal:** Distinguish circuits that are injective from those in which every image has at least two preimages



Batch Verification for AI_0

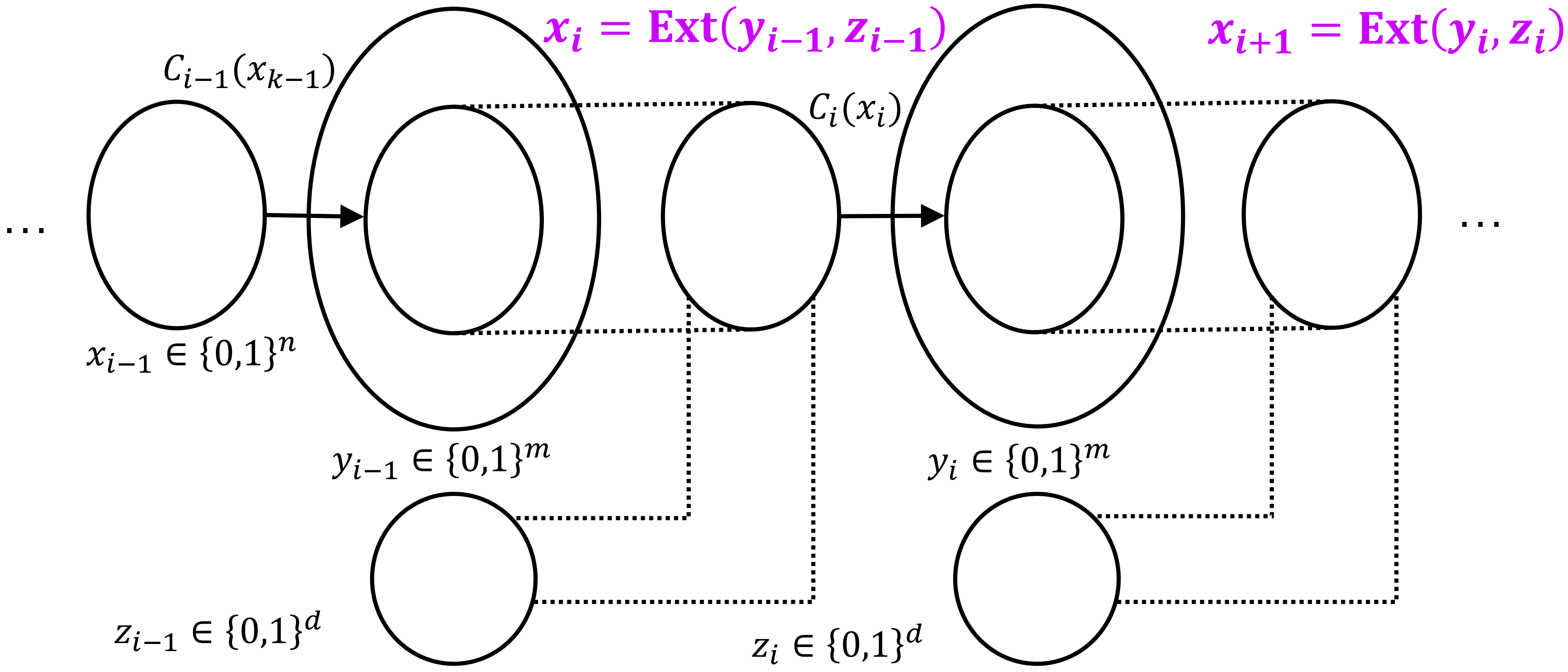
- **Difficulty:** output size is not the same as input size - cannot directly compose
- **Idea:** hash each circuit output to the next circuit input
- **Want:** each x_i to be close to uniform, for soundness



Batch Verification for AI_0

Natural Idea: use extractors

Need extractors that extract (almost) all entropy with $d = \mathrm{polylog}(n)$ seed [GUV07]



The Protocol - First Attempt

Finds consistent x_1',
i.e., s.t. the following
yields same y_k:

- For i=1,...k-1:
  - $y_i = C_i(x_i')$
  - $x_{i+1}' = \mathrm{Ext}(\mathrm{y_i}, \mathrm{z_i})$
- $y_k = C_k(x_k')$

Prover

$y_k, z_1, \dots, z_k$ ←

x_1' →

- Samples $x_1 \leftarrow \{0,1\}^n$ and $z_1, \dots, z_k \in \{0,1\}^d$
- Computes for $i = 1, \dots k - 1$:
  - $y_i = C_i(x_i)$
  - $x_{i+1} = \mathrm{Ext}(y_i, z_i)$
- Computes $y_k = C_k(x_k)$

Verifiers that $x_1 = x_1'$

Verifier

Protocol - First Attempt

- **Problem:** even if $\mathrm{Ext}(\cdot, z_i)$ were a random function:
  - Constant fraction of the x_{i+1} has > 1 preimages
  - ➢ P's chances to guess the correct x_1 are negligible

- **Idea:** give P additional information about x_1

- **New Problem:** additional information can help the malicious prover

- **Solution:** use interaction
  - The verifier gradually reveals information about the y_i's

The Protocol - Second Attempt

- Samples $x_1 \leftarrow \{0,1\}^n$ and $z_1, \dots, z_k \in \{0,1\}^d$
- Computes for $i = 1, \dots k-1$:
  - $y_i = C_i(x_i)$
  - $x_{i+1} = \mathrm{Ext}(y_i, z_i)$
- Computes $y_k = C_k(x'_k)$

For $i = k, \dots, 1$

y_i ←

Finds x'_i s.t. $y_i = C_i(x'_i)$

x'_i →

Verifiers $x_i = x'_i$

Prover

Verifier

The Protocol - Second Attempt

- Samples $x_1 \leftarrow \{0,1\}^n$ and $z_1, \dots, z_k \in \{0,1\}^d$
- Computes for $i = 1, \dots k - 1$:
  - $y_i = C_i(x_i)$
  - $x_{i+1} = \mathrm{Ext}(y_i, z_i)$
- Computes $y_k = C_k(x'_k)$

For $i = k, \dots, 1$

y_i ←

Finds x'_i s.t. $y_i = C_i(x'_i)$

x'_i →

Verifiers $x_i = x'_i$

Prover

Communication Overhead

Verifier

Protocol Analysis

- **Completeness:** for each injective C_i, P can guess correctly x_i
- **Soundness:** for the first No instance C_{i^*}, the input x_{i^*} is close to uniform $\Rightarrow$ $\tilde{\mathrm{P}}$'s chances to guess the correct x_{i^*} given y_{i^*} is roughly $\leq \frac{1}{2}$
- **Zero-knowledge:** simulator that generates $x_1, \dots, x_k, z_1, \dots, z_k, y_1, \dots y_k$ similarly to the verifier.

The Protocol - Second Attempt

- Samples $x_1 \leftarrow \{0,1\}^n$ and $z_1, \ldots, z_k \in \{0,1\}^d$
- Computes for $i = 1, \ldots k-1$:
  - $y_i = C_i(x_i)$
  - $x_{i+1} = \text{Ext}(y_i, z_i)$
- Computes $y_k = C_k(x_k')$

For $i = k, \ldots, 1$

y_i

Finds x_i' s.t. $y_i = C_i(x_i')$

x_i'

Verifiers $x_i = x_i'$

Communication Overhead

Prover

Verifier

The Protocol - Second Attempt

- Samples $x_1 \leftarrow \{0,1\}^n$ and $z_1, \dots, z_k \in \{0,1\}^d$
- Computes for $i = 1, \dots k-1$:
  - $y_i = C_i(x_i)$
  - $x_{i+1} = \mathrm{Ext}(y_i, z_i)$
- Computes $y_k = C_k(x_k')$

y_k ←

For $i = k, \dots, 1$

~~y_i~~ **Hint about y_i, z_i** ←

Finds x_i' s.t. $y_i = C_i(x_i')$ and $\mathrm{Ext}(y_i, z_i) = x_{i+1}$

~~x_i'~~ **Hint about x_i'** →

Verifiers $x_i = x_i'$

x'_1 →

Prover

Verifier

AI is NISZK-complete

- *Entropy Approximation* (EA) is NISZK-complete [GSV99]

- To show AI $\in$ NISZK: reduction from AI to EA
- To show AI is NISZK-hard: reduction from EA to AI

Summary and Open Problems

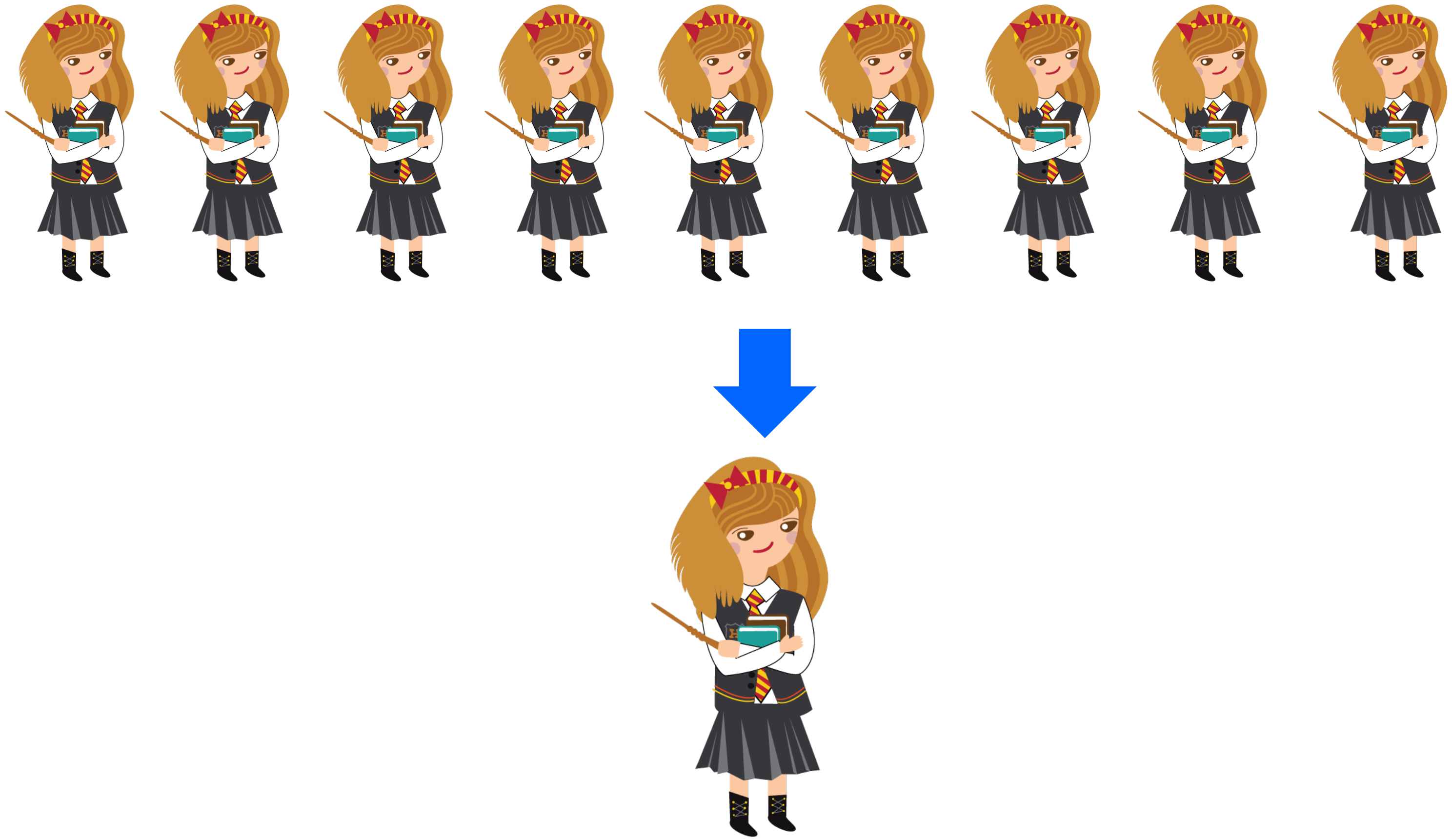
Main Thm: Every $\Pi \in \mathrm{NISZK}$ has an HVSZK batch-verification protocol with $k + \mathrm{poly}(n)$ communication

No longer open problems: SZK protocol, public-coin protocol [KRV2?]

Open problems:

- Batch verification for SZK
- Communication $\mathrm{poly}(n, \log k)$
- Constant number of rounds
- Efficient prover (for $\Pi \in \mathrm{SZK} \cap \mathrm{NP}$, see also [NV06])

Thank You!



Harry Potter images by Sarit Evrani