How to Simulate Random Oracles with Auxiliary Input

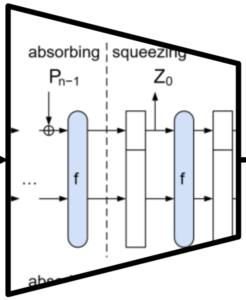
Yevgeniy Dodis Aayush Jain Rachel Lin Ji Luo 罗辑 Daniel Wichs

NYU Carnegie Mellon University UNIVERSITY of WASHINGTON BOSTON UNIVERSITY Northeastern University

Random Oracle Model [BR93]

Standard-Model (Real-World) Hash Functions

$\{0,1\}^n$ or $\{0,1\}^*$ →



→ $\{0,1\}^\lambda$

✗ no control over how code is used
⟹ difficulty in analyzing security

Random Oracle Model

x ←

$\mathcal{O}(x)$ →

$\mathcal{O}: \{0,1\}^{n/*} \to \{0,1\}^\lambda$
random truth table

✓ easier security analysis
- **practical schemes**
- **qualitative** – what property?
- **quantitative** – (T, ε)-security

✗ overly optimistic
against non-uniform adversaries

Non-Uniformity, Examples of Over-Optimism

Non-Uniform Adversary = machine A, sequence $\{z_\lambda\}_{\lambda\in\mathbb{N}}$ of strings (advice)
on security parameter λ: $A(1^\lambda, z_\lambda) \leftrightarrow$ scheme
potential benefits of reusable preprocessing (e.g., rainbow tables [O03])

⚠ Advice depends on H.
ROM does **not** capture $\mathcal{O}$-dependent advice.

Qualitative

ROM. $\mathcal{O}$ is a keyless collision-resistant (CR) hash function.

Fact. Keyless functions cannot be CR against non-uniform adversaries.
(advice = smallest collision)

Quantitative

ROM. to invert $y = \mathcal{O}(x)$ w.p. $\Omega(1)$ given y, need $Q = \Omega(2^\lambda)$ queries.

(heuristic) $\Longrightarrow$

to invert $y = H(x)$ for "good" H, need $T = \Omega(2^\lambda)$.

Fact. to invert $y = H(x)$, (S = advice length) $S, T = O(2^{3\lambda/4})$ suffices [FN91].

Auxiliary-Input Random Oracle Model [U07]

adversary = machine A, function ai

non-computable ai allowed; output length S depends on λ.

on security parameter λ:

- $z \leftarrow \mathrm{ai}(\lambda, \mathcal{O})$
- $A(1^\lambda, z) \leftrightarrow \mathcal{O}$ and scheme

✓ **avoids overly optimistic heuristics**

- **qualitative** – no seedless CRHF (advice = collision, depends on $\mathcal{O}$)
- **quantitative** – [FN91] attack carries over to AI-ROM

✗ **difficulty in analyzing security**

No clue even for proving DDH in AI-ROM from DDH in non-uniform standard model!

- ROM – can lazy-sample/program truth table
- AI-ROM – $A(1^\lambda, z) \leftrightarrow \mathcal{O}$ how to handle **arbitrary correlation**?

Primary Goal, Previous/Our Results

"Develop methods for showing (tight) security in AI-ROM."

Known Result. Presampling method [U07,CDGS17]
to "simulate" AI-ROM information-theoretically,
as tight as it can be, but not tight.

Our Main Result. A new technique
to simulate AI-ROM computationally,
much tighter!

tight. $T_{\text{reduction}} \sim T_{\text{underlying}}$; qualitatively – **same/stronger** security from **weaker/same** assumptions.

Why Two Colors for Simulation?

open since AI-ROM conception [U07]

Two-Stage Definition (Distinguisher-Independent Simulation)

1. let $A(z) \leftrightarrow \mathcal{O}$ and record transcript τ of A
2. $D(\tau) \rightarrow \{0,1\}$ (no $\mathcal{O}$!)

Requirement is $\forall(A, \mathrm{ai})\ \exists(S, \mathrm{nu})\ \forall D\ \ldots$

- "nu" for (standard-model) **n**on-**u**niformity
- **zero-knowledge** uses **two-stage** definition

Simulated Step 1.

- $(z, w) \leftarrow \mathrm{nu}(\lambda)$
- let $A(z) \leftrightarrow S(w)$

***w* = extra advice**

Single-Stage Definition (Distinguisher-Dependent Simulation)

- let $A(z) \leftrightarrow \mathcal{O}$ and A outputs a bit

Requirement is $\forall(A, \mathrm{ai})\ \exists(S, \mathrm{nu})\ \ldots$

(think of D merged inside A)

Comparison of Simulation Methods

method / assumption	extra advice	time per query
presampling [U07,CDGS18]	$\Theta(\lambda n S\, Q/\varepsilon)$	$\Theta(\lambda n S\, Q/\varepsilon)$ [circuit] or $\tilde{O}(n)$ [RAM]
subexp.sec. PRF	$\mathrm{poly}(\lambda, n, S)$	$\mathrm{poly}(\lambda, n, S)$
exp.sec. quasi.lin.time PRG	$O(\lambda + S)$	$n \cdot \tilde{O}(\lambda + S)$
faster-than-secure PRF	$O(n^{1+\gamma}(\lambda + S)^{1+\gamma})$	$n^{1+\gamma} \cdot \mathrm{polylog}(\lambda, n, S)$ [RAM]

$\gamma > 0$ arbitrarily tunable

- Showing $\mathcal{O}: \{0,1\}^n \to \{0,1\}^\lambda$.
- Presampling requires knowing Q, ε in advance
  - (due to ε,) **fails** two-stage definitions (e.g., zero-knowledge).
    - **^ would need super-poly.-time sim. for negl. ε**
- **All ours satisfy two-stage definition.**
- S, Q are (adversarial resources) large $\mathrm{poly}(\lambda)$ [even $2^{\Theta(\lambda)}$ in concrete security].
  - **Optimization Priority.** $S, T, Q, \varepsilon \succ n \succ \lambda$ – **Last 2 of ours are tighter.**

Key Idea

✓ **first NIZK in AI-ROM**

Q. The simulator is **efficient** and **looks like a random function.** What can it be?

A. A pseudorandom function (PRF).

✓ **resolves open problem of AI-ROM simulation** (two-stage definition)

even better – does not depend on A

Let's try it… [$\langle \cdots \rangle$ = transcript]

$$\langle A(\mathrm{ai}(\mathcal{O})) \leftrightarrow \mathcal{O} \rangle \overset{?}{\approx} \langle A(\mathrm{ai}(F(k,\cdot))) \leftrightarrow F(k,\cdot) \rangle$$

✗ ai: arbitrarily complex

$$\overset{(1)}{\approx} \langle A(\mathrm{ai}'(\mathcal{O})) \leftrightarrow \mathcal{O} \rangle$$

$$\overset{(2)}{\approx} \langle A(\mathrm{ai}'(F(k,\cdot))) \leftrightarrow F(k,\cdot) \rangle$$

Fix (leakage simulation lemma [CCL18]). Let

$$X = \mathcal{O}, \quad Z = \mathrm{ai}(X) \text{ (of length } S\text{)},$$

then $\exists \mathrm{ai}'$ of complexity $\sim 2^S ST/\varepsilon^2$:

$$(X, Z) \approx_{T,\varepsilon} (X, \mathrm{ai}'(X)).$$

"function of output length S, complexity controlled at $\sim 2^S$"

(1) set $T = 2^\lambda > T_A + T_D$ and $\varepsilon = 2^{-\lambda}$

(2) tune up λ_{prf} for security against total time ($\sim 2^{S+\lambda}$)

requires subexp.-secure PRF

Tight Reduction = Fast and Secure PRF

Convention. at λ_{prf}, the PRF is ε-secure in $2^{\lambda_{\mathrm{prf}}}$ time.

- fixed level of security
- compete for small T_{prf} (plus, short k = extra advice)

$T_{\mathrm{prf}} \in \mathrm{subpoly}(\lambda)$
"faster than secure"

Typical. $T_{\mathrm{prf}} = n \times \Omega(\lambda_{\mathrm{prf}})$

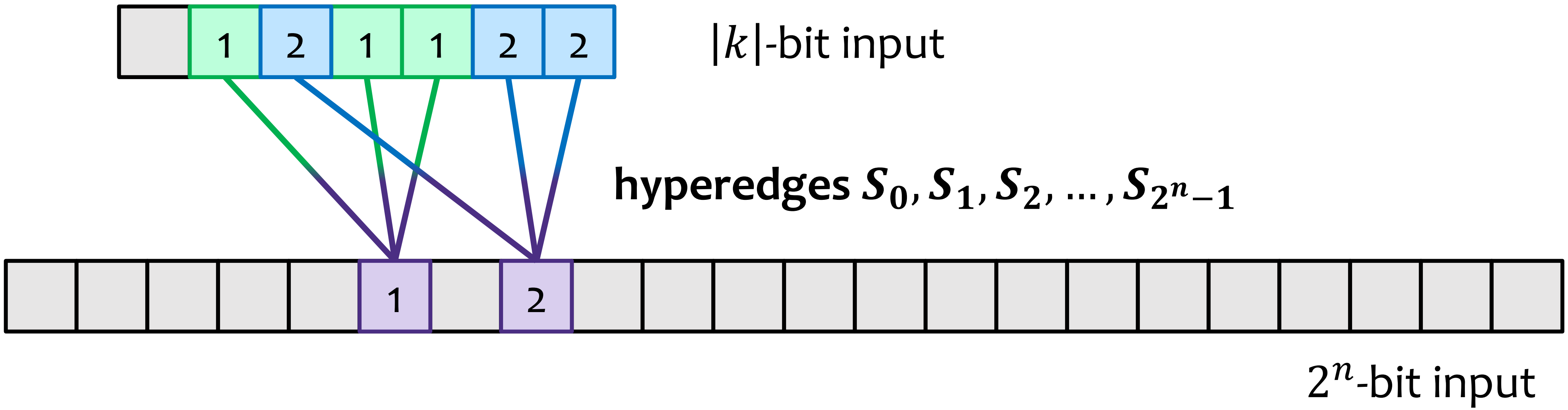
- $\lambda_{\mathrm{prf}} \geq S + \lambda$

adversary-dependent degrades quickly

Wish. What about $T_{\mathrm{prf}} = n \cdot \mathrm{polylog}(\lambda_{\mathrm{prf}})$?

[AR16] **weak** PRF with $T_{\mathrm{wprf}} = \tilde{O}(n)$ from Goldreich's PRG [Goo]

Faster-than-Secure PRF from Goldreich's PRG



$F(k, x) = P(k[S_x])$ with fixed function P [e.g., XOR-MAJ]

- **cryptanalysis.** exp.-PRF if $(S_x)_x$ is sufficiently expanding [e.g., $(t, 0.99)$ for $t \sim \lambda_{\text{prf}}^{1+\gamma}$]

✓ locality = P's input length = $\Theta(n)$

? represent exp.-size hypergraph succinctly (extra advice)

? compute S_x efficiently (time per query)

Representing and Accessing Expanders

2^n-size hypergraph must be $(t, 0.99)$-expanding, $t \sim \lambda_{\mathrm{prf}}^{1+\gamma}$.

- $S_x = h(x)$ is expanding w.h.p. with t-wise independent h
- let h be a degree-t polynomial

Evaluate $x \mapsto h(x) \mapsto S_x$ in time $n \cdot \mathrm{o}(\lambda_{\mathrm{prf}})$?

- fast polynomial evaluation [KU09]
- coefficients $\Longrightarrow$ preprocess $\Longrightarrow$ size $t^{1+\gamma} \log^{1+\gamma} p$ (p for field)
- **online.** evaluation time is $(\log^{1+\gamma} p) \cdot \mathrm{polylog}(t) \sim n^{1+\gamma}\, \mathrm{polylog}(\lambda_{\mathrm{prf}})$

first NIZK in AI-ROM

$\Uparrow$

Leakage Simulation + Subexponentially Secure PRF $\Longrightarrow$ AI-ROM simulation

"Faster-than-Secure" $\Longrightarrow$ very tight

Thanks!

luoji@bu.edu
luoji.bio

another trick: combine presampling + ours
for best of both worlds [see paper!]