

Pseudorandom Finite Models

Jamie Tucker-Foltz

Harvard University

Joint work with Jan Dreier, TU Wien

LICS 2023

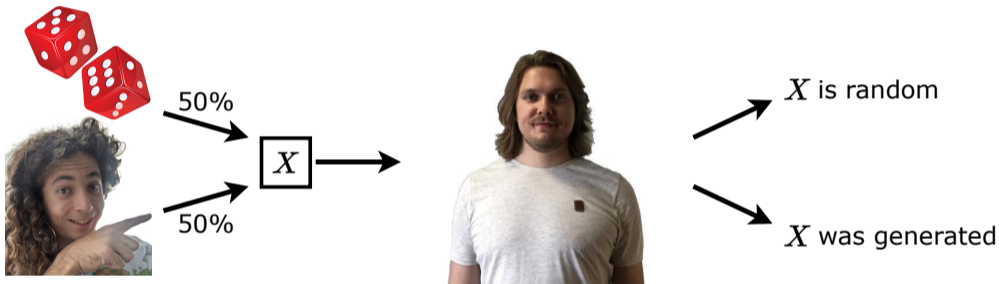
Boston University

Boston, MA, June 27, 2023

An object is pseudorandom if it is indistinguishable from one that is truly random.

Overview

An object is pseudorandom if it is indistinguishable from one that is truly random.



Me, the Generator

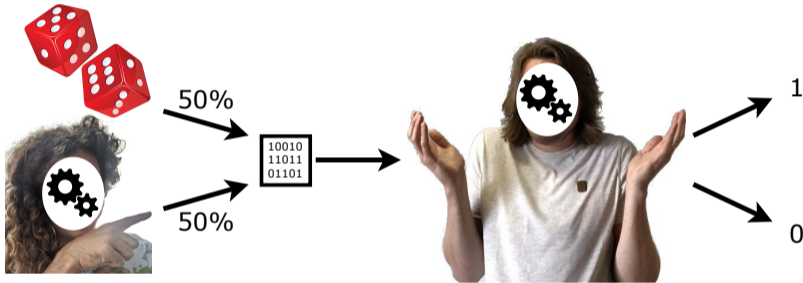
Jan, the Adversary

Overview

An object is pseudorandom if it is indistinguishable from one that is truly random.



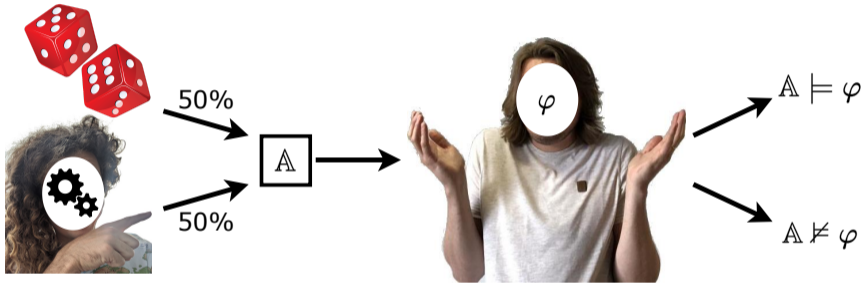
An object is pseudorandom if it is indistinguishable from one that is truly random.



Classic cryptography:

- Object is a binary string
- Generator is a poly-time TM with access to fewer random bits
- Adversary is a poly-size circuit

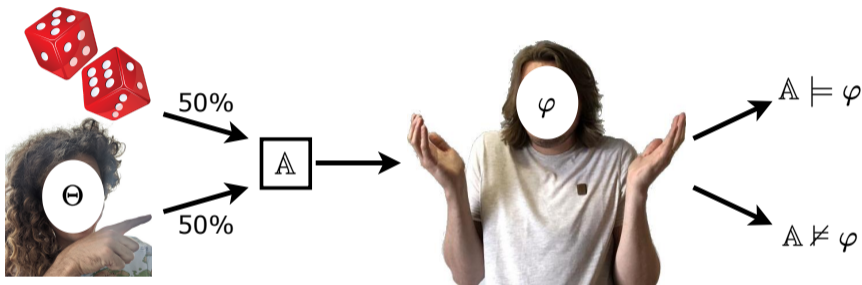
An object is pseudorandom if it is indistinguishable from one that is truly random.



Part I:

- Object is a **finite, relational structure**
- Generator is a poly-time TM
- Adversary is a **logical sentence**

An object is pseudorandom if it is indistinguishable from one that is truly random.



Part I:

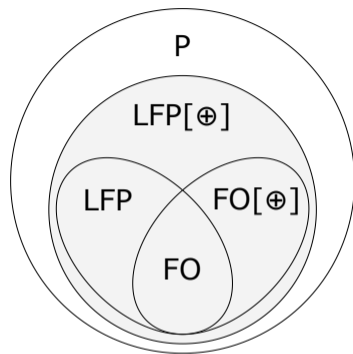
- Object is a **finite, relational structure**
- Generator is a poly-time TM
- Adversary is a **logical sentence**

Part II:

- Object is a finite, relational structure
- Generator is **logical transduction**
- Adversary is a logical sentence

Extensions of FO logic:

- LFP: *Least Fixed Point* operator (inductive definitions)
- \oplus : *Parity* operator (there are an odd number of elements such that...)

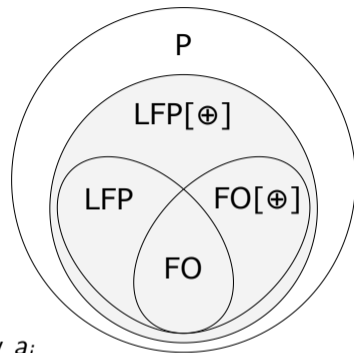


Extensions of FO logic:

- LFP: *Least Fixed Point* operator (inductive definitions)
- \oplus : *Parity* operator (there are an odd number of elements such that...)

Relational structures:

- Signatures: $\sigma, \tau, \rho = \langle R_1, R_2, \dots, R_t \rangle$, R_i has arity a_i
- Structures: $\mathbb{A} = \langle A, R_1^{\mathbb{A}}, R_2^{\mathbb{A}}, \dots, R_t^{\mathbb{A}} \rangle$; $\text{Str}[\sigma, n] := \{n\text{-element } \sigma\text{-structures}\}$



Extensions of FO logic:

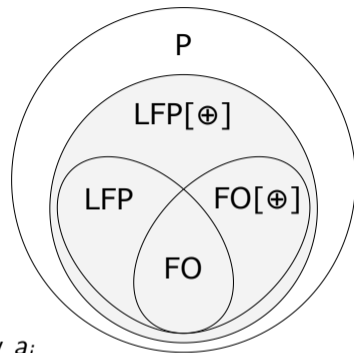
- LFP: *Least Fixed Point* operator (inductive definitions)
- \oplus : *Parity* operator (there are an odd number of elements such that...)

Relational structures:

- Signatures: $\sigma, \tau, \rho = \langle R_1, R_2, \dots, R_t \rangle$, R_i has arity a_i
- Structures: $\mathbb{A} = \langle A, R_1^{\mathbb{A}}, R_2^{\mathbb{A}}, \dots, R_t^{\mathbb{A}} \rangle$; $\text{Str}[\sigma, n] := \{n\text{-element } \sigma\text{-structures}\}$

Graphs/hypergraphs:

- Only one relation symbol, understood to always be symmetric.



Sampling $\mathbb{A} \sim \text{Str}[\sigma, n]$ means:

- 1 Fix an n -element set A .
- 2 Put each a_i -tuple of elements from A in $R_i^{\mathbb{A}}$ with probability $\frac{1}{2}$, independently.

Sampling $\mathbb{A} \sim \text{Str}[\sigma, n]$ means:

- 1 Fix an n -element set A .
- 2 Put each a_i -tuple of elements from A in $R_i^{\mathbb{A}}$ with probability $\frac{1}{2}$, independently.

Theorem [Fagin, 1976] [Blass, Gurevich, and Kozen, 1985]

For any LFP σ -sentence φ ,

$$\lim_{n \rightarrow \infty} \Pr_{\mathbb{A} \sim \text{Str}[\sigma, n]} [\mathbb{A} \models \varphi] = 0 \text{ or } 1.$$

Moreover, convergence is exponentially fast.

Sampling $\mathbb{A} \sim \text{Str}[\sigma, n]$ means:

- 1 Fix an n -element set A .
- 2 Put each a_i -tuple of elements from A in $R_i^{\mathbb{A}}$ with probability $\frac{1}{2}$, independently.

Theorem [Fagin, 1976] [Blass, Gurevich, and Kozen, 1985]

For any LFP σ -sentence φ ,

$$\lim_{n \rightarrow \infty} \Pr_{\mathbb{A} \sim \text{Str}[\sigma, n]} [\mathbb{A} \models \varphi] = 0 \text{ or } 1.$$

Moreover, convergence is exponentially fast.

Theorem [Kolaitis and Kopparty, 2009]

For $\text{FO}[\oplus]$, there are two limits, one for odd n and one for even n , both of the form $\frac{a}{2^b}$.

Theorem [Hella, Kolaitis, Luosto, 1996]

There is an $\text{LFP}[\oplus]$ formula with 2 free variables defining a total order on the universe with high probability over random graphs.

Theorem [Hella, Kolaitis, Luosto, 1996]

There is an $\text{LFP}[\oplus]$ formula with 2 free variables defining a total order on the universe with high probability over random graphs.

Theorem [Immerman, 1982], [Vardi, 1982]

If we have access to a total order, then LFP captures polynomial time.

Theorem [Hella, Kolaitis, Luosto, 1996]

There is an $\text{LFP}[\oplus]$ formula with 2 free variables defining a total order on the universe with high probability over random graphs.

Theorem [Immerman, 1982], [Vardi, 1982]

If we have access to a total order, then LFP captures polynomial time.

\implies Over truly random structures, $\text{LFP}[\oplus]$ is almost as powerful as polynomial time

Theorem [Hella, Kolaitis, Luosto, 1996]

There is an $LFP[\oplus]$ formula with 2 free variables defining a total order on the universe with high probability over random graphs.

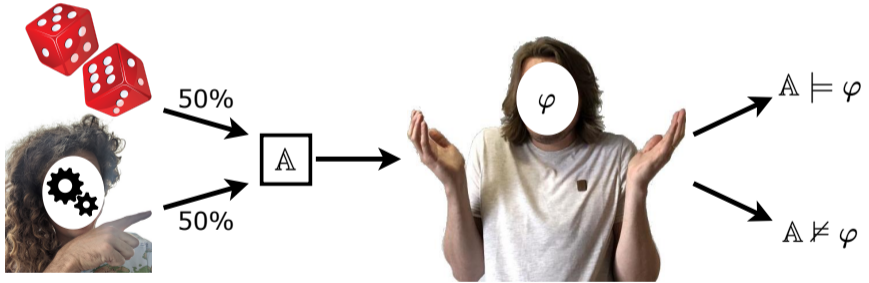
Theorem [Immerman, 1982], [Vardi, 1982]

If we have access to a total order, then LFP captures polynomial time.

\implies Over truly random structures, $LFP[\oplus]$ is almost as powerful as polynomial time

\implies No such limits exist

Part I



Central definition

Let $\{\mathcal{D}_n\}_{n \geq 1}$ be a collection of probability distributions, where \mathcal{D}_n ranges over $\text{Str}[\tau, n]$.

Central definition

Let $\{\mathcal{D}_n\}_{n \geq 1}$ be a collection of probability distributions, where \mathcal{D}_n ranges over $\text{Str}[\tau, n]$.

Definition (uniform version)

$\{\mathcal{D}_n\}_{n \geq 1}$ is *pseudorandom* for a logic L if, for every τ -sentence $\varphi \in L$,

$$\left| \Pr_{\mathbb{A} \sim \mathcal{D}_n} [\mathbb{A} \models \varphi] - \Pr_{\mathbb{A} \sim \text{Str}[\tau, n]} [\mathbb{A} \models \varphi] \right| = \text{negl}(n).$$

*A function is $\text{negl}(n)$ if it is eventually less than $\frac{1}{p(n)}$ for any polynomial $p(n)$.

Central definition

Let $\{\mathcal{D}_n\}_{n \geq 1}$ be a collection of probability distributions, where \mathcal{D}_n ranges over $\text{Str}[\tau, n]$.

Definition (uniform version)

$\{\mathcal{D}_n\}_{n \geq 1}$ is *pseudorandom* for a logic L if, for every τ -sentence $\varphi \in L$,

$$\left| \Pr_{\mathbb{A} \sim \mathcal{D}_n} [\mathbb{A} \models \varphi] - \Pr_{\mathbb{A} \sim \text{Str}[\tau, n]} [\mathbb{A} \models \varphi] \right| = \text{negl}(n).$$

*A function is $\text{negl}(n)$ if it is eventually less than $\frac{1}{p(n)}$ for any polynomial $p(n)$.

Definition (nonuniform version)

$\{\mathcal{D}_n\}_{n \geq 1}$ is *pseudorandom* for L if, for every sequence of “advice” structures $\{\mathbb{X}_n\}_{n \geq 1}$ in some signature ρ , where $|\mathbb{X}_n| = n$, and every $(\tau \cup \rho)$ -sentence $\varphi \in L$,

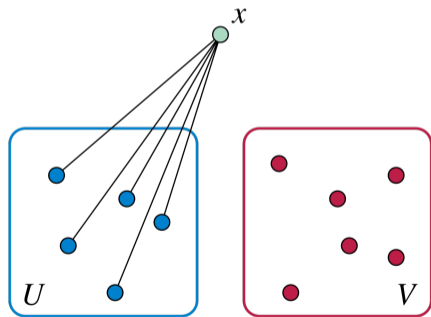
$$\left| \Pr_{\mathbb{A} \sim \mathcal{D}_n} [(\mathbb{A}, \mathbb{X}_n) \models \varphi] - \Pr_{\mathbb{A} \sim \text{Str}[\tau, n]} [(\mathbb{A}, \mathbb{X}_n) \models \varphi] \right| = \text{negl}(n).$$

Theorem

There are deterministic polynomial-time algorithms to construct graphs or relational structures that are pseudorandom for LFP.

Theorem

There are deterministic polynomial-time algorithms to construct graphs or relational structures that are pseudorandom for LFP.



Proof approach (for graphs):

- A graph satisfies the k -extension axioms, denoted EA_k , if, for all U and V s.t. $|U \cup V| \leq k$, there $\exists x$ adjacent to all vertices in U and none in V .

Theorem

There are deterministic polynomial-time algorithms to construct graphs or relational structures that are pseudorandom for LFP.

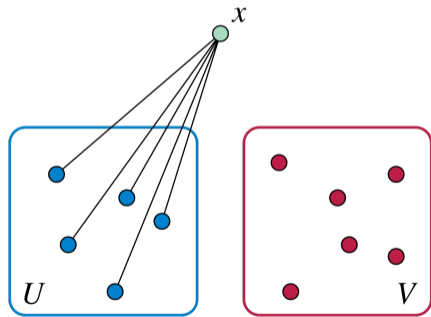


Image credit: David Eppstein

Proof approach (for graphs):

- A graph satisfies the k -extension axioms, denoted EA_k , if, for all U and V s.t. $|U \cup V| \leq k$, there $\exists x$ adjacent to all vertices in U and none in V .
- For any $G_1, G_2 \models EA_k$ and φ with k variables, $G_1 \models \varphi \iff G_2 \models \varphi$.

Theorem

There are deterministic polynomial-time algorithms to construct graphs or relational structures that are pseudorandom for LFP.

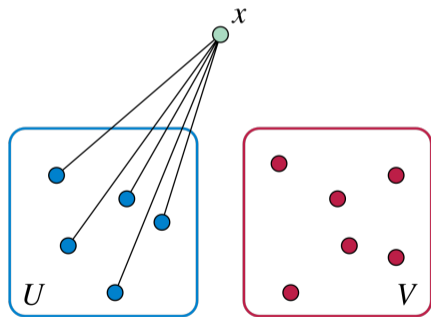


Image credit: David Eppstein

Proof approach (for graphs):

- A graph satisfies the k -extension axioms, denoted EA_k , if, for all U and V s.t. $|U \cup V| \leq k$, there $\exists x$ adjacent to all vertices in U and none in V .
- For any $G_1, G_2 \models EA_k$ and φ with k variables, $G_1 \models \varphi \iff G_2 \models \varphi$.
- Almost all graphs satisfy EA_k .

Theorem

There are deterministic polynomial-time algorithms to construct graphs or relational structures that are pseudorandom for LFP.

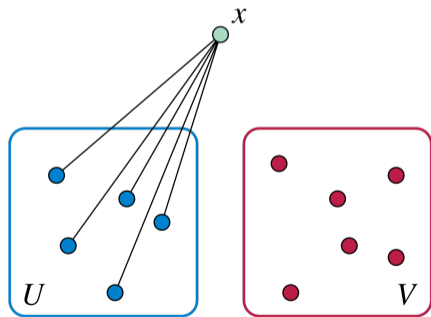


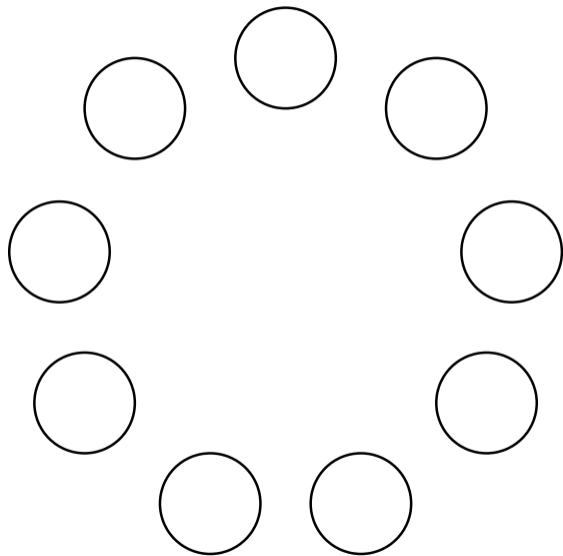
Image credit: David Eppstein

Proof approach (for graphs):

- A graph satisfies the k -extension axioms, denoted EA_k , if, for all U and V s.t. $|U \cup V| \leq k$, there $\exists x$ adjacent to all vertices in U and none in V .
- For any $G_1, G_2 \models EA_k$ and φ with k variables, $G_1 \models \varphi \iff G_2 \models \varphi$.
- Almost all graphs satisfy EA_k .
- It thus suffices to construct G_1, G_2, \dots eventually satisfying EA_k for each k .

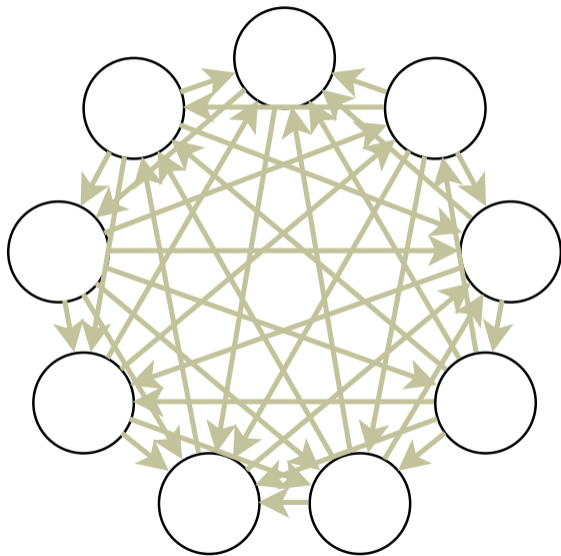
To construct G_n :

- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.



To construct G_n :

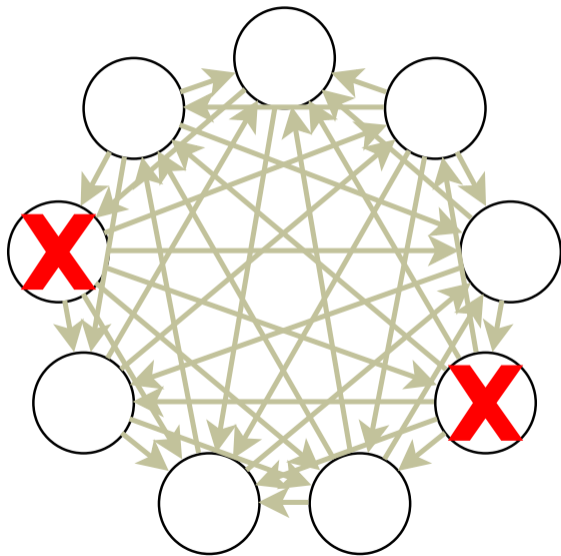
- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.



Proof (continued)

To construct G_n :

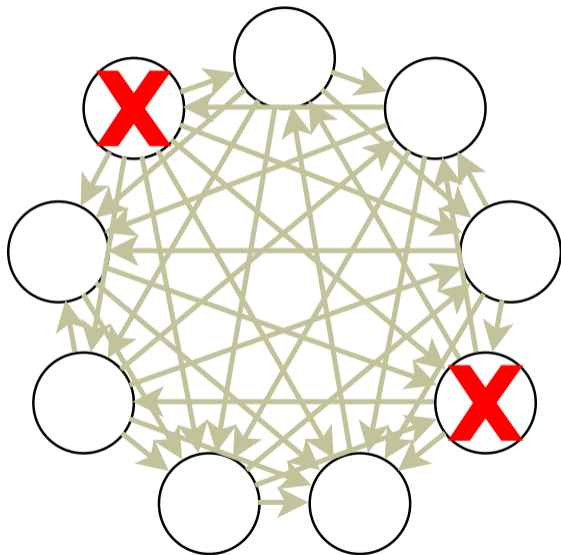
- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.



Proof (continued)

To construct G_n :

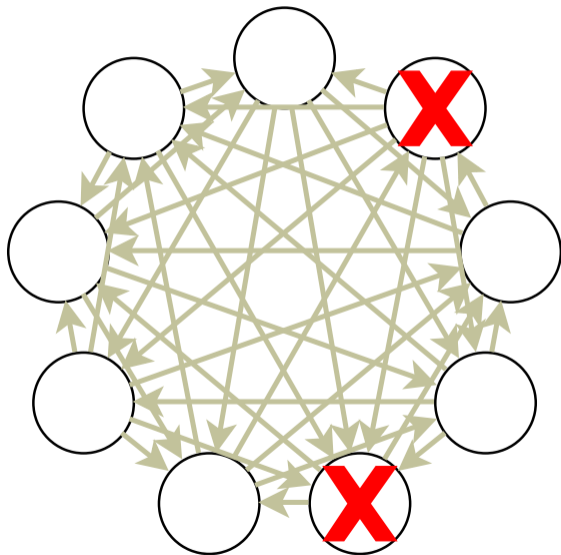
- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.



Proof (continued)

To construct G_n :

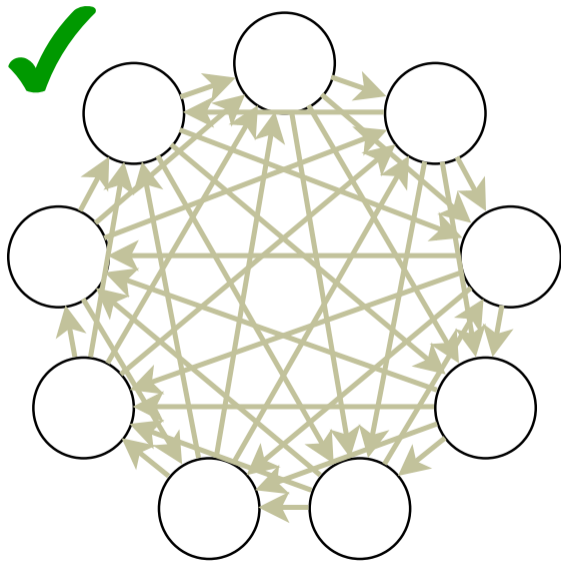
- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.



Proof (continued)

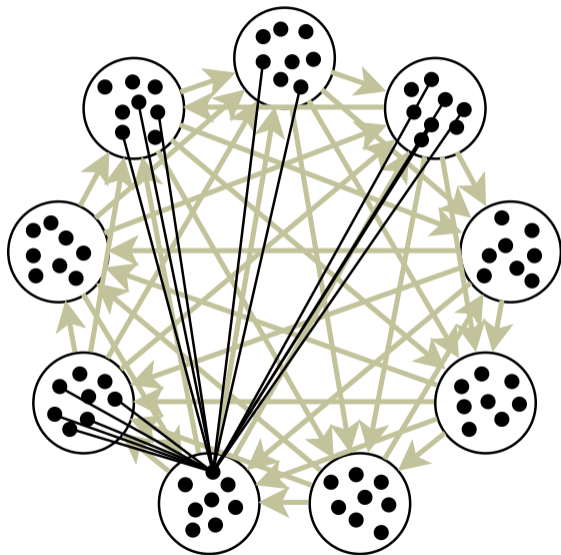
To construct G_n :

- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.



To construct G_n :

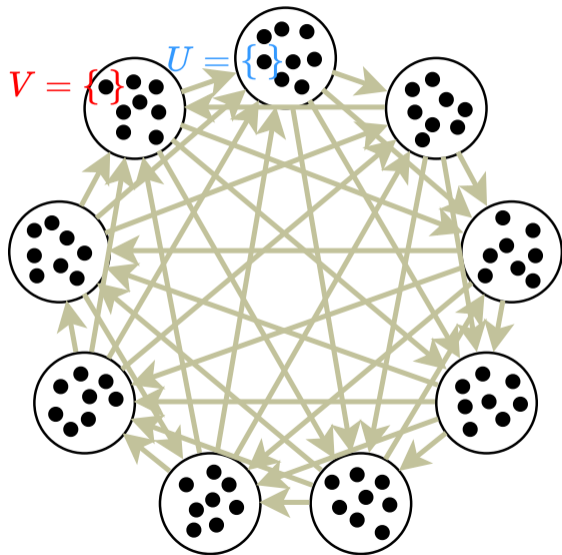
- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.
- 2 Define adjacency from parents to children via *universal sets* [Naor, Schulman, and Srinivasan, 1995].



To construct G_n :

- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.
- 2 Define adjacency from parents to children via *universal sets* [Naor, Schulman, and Srinivasan, 1995].

Given any U, V , to find x :



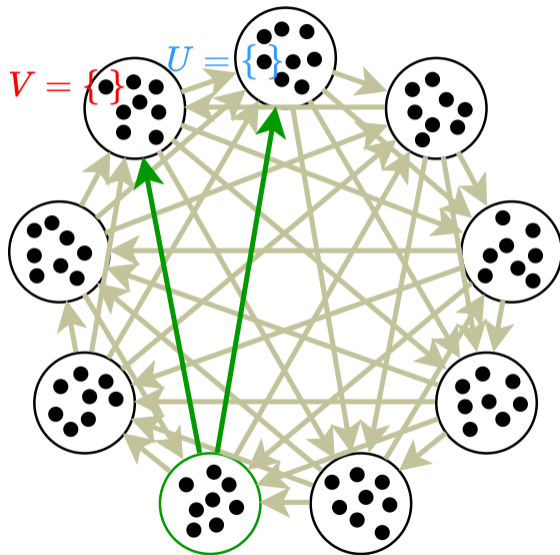
Proof (continued)

To construct G_n :

- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.
- 2 Define adjacency from parents to children via *universal sets* [Naor, Schulman, and Srinivasan, 1995].

Given any U, V , to find x :

- 1 Find common parent.



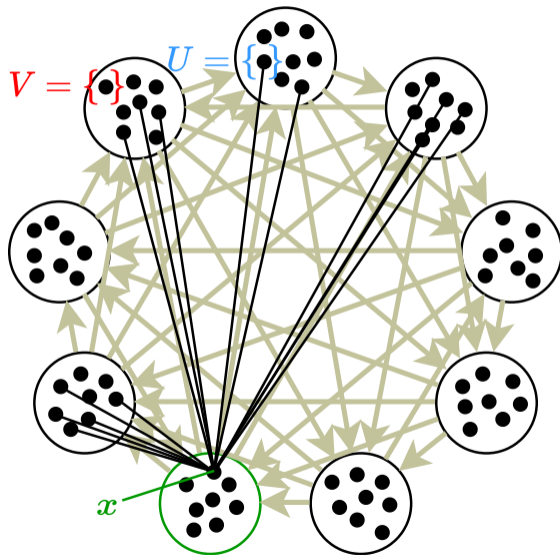
Proof (continued)

To construct G_n :

- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.
- 2 Define adjacency from parents to children via *universal sets* [Naor, Schulman, and Srinivasan, 1995].

Given any U, V , to find x :

- 1 Find common parent.
- 2 Universal sets guarantee $\exists x$ adjacent to all of U , none of V .



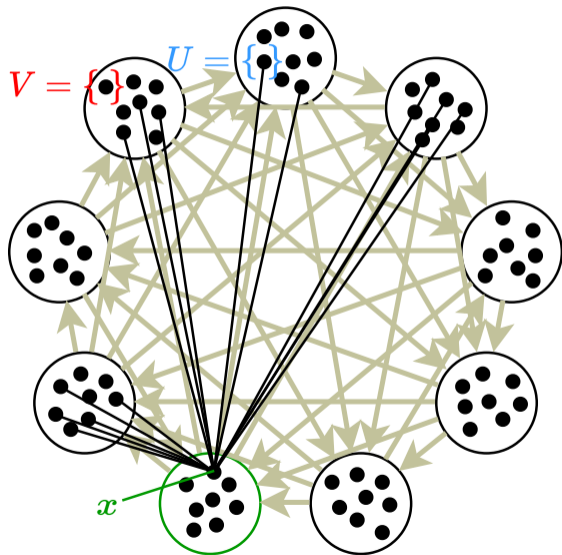
To construct G_n :

- 1 Brute force search all tournaments on $\sim \log(n)$ vertices to find one where every $k \sim \log(\log(n))$ vertices has a common parent.
- 2 Define adjacency from parents to children via *universal sets* [Naor, Schulman, and Srinivasan, 1995].

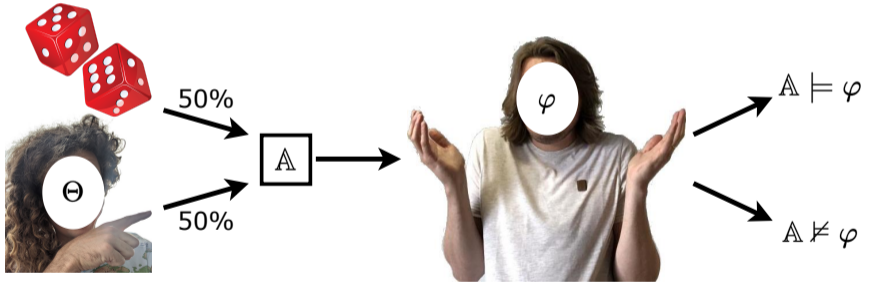
Given any U, V , to find x :

- 1 Find common parent.
- 2 Universal sets guarantee $\exists x$ adjacent to all of U , none of V .

For relational structures, we use *universal hash functions* instead.



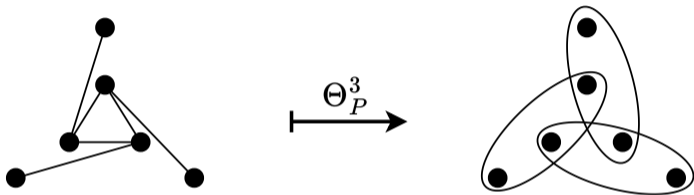
Part II



An *L-transduction* from σ to τ is a sequence of σ -formulas in L defining each relation of τ from the relations in σ . This gives a map $\Theta : \text{Str}[\sigma, n] \rightarrow \text{Str}[\tau, n]$.

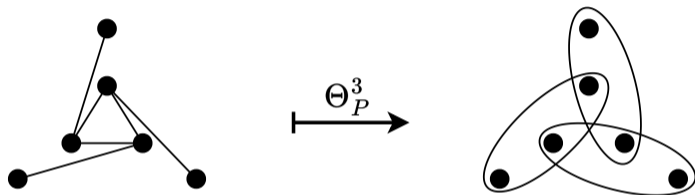
Transductions

An L -transduction from σ to τ is a sequence of σ -formulas in L defining each relation of τ from the relations in σ . This gives a map $\Theta : \text{Str}[\sigma, n] \rightarrow \text{Str}[\tau, n]$.



Example of a $\text{FO}[\oplus]$ -transduction: $\Theta_P^3 = \langle \theta^3 \rangle$ from $\langle E^2 \rangle$ to $\langle H^3 \rangle$, where $\theta^3(x, y, z) \equiv (\oplus w)E(x, w) \wedge E(y, w) \wedge E(z, w)$

An L -transduction from σ to τ is a sequence of σ -formulas in L defining each relation of τ from the relations in σ . This gives a map $\Theta : \text{Str}[\sigma, n] \rightarrow \text{Str}[\tau, n]$.



Example of a $\text{FO}[\oplus]$ -transduction: $\Theta_P^3 = \langle \theta^3 \rangle$ from $\langle E^2 \rangle$ to $\langle H^3 \rangle$, where $\theta^3(x, y, z) \equiv (\oplus w)E(x, w) \wedge E(y, w) \wedge E(z, w)$

Definition

An (L_1, L_2) -pseudorandom generator from σ to τ is an L_1 -transduction $\Theta : \text{Str}[\sigma] \rightarrow \text{Str}[\tau]$ such that $\Theta(\mathbb{A}_n \sim \text{Str}[\sigma, n])$ is pseudorandom for L_2 .

Characterization of when (L_1, L_2) -pseudorandom generators exist for FO, LFP, LFP[\oplus]

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]			

Characterization of when (L_1, L_2) -pseudorandom generators exist for FO, LFP, LFP[\oplus]

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]	✓	✓	

Corollary

For $L_2 \in \{FO, LFP\}$, there exists an $(LFP[\oplus], L_2)$ -pseudorandom generator from σ to τ if and only if σ has at least one non-unary relation (or σ and τ both have only unary relations, with σ containing at least as many as τ).

Characterization of when (L_1, L_2) -pseudorandom generators exist for FO, LFP, LFP[\oplus]

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]	✓	✓	

Corollary

For $L_2 \in \{FO, LFP\}$, there exists an $(LFP[\oplus], L_2)$ -pseudorandom generator from σ to τ if and only if σ has at least one non-unary relation (or σ and τ both have only unary relations, with σ containing at least as many as τ).

Proof:

- 1 Use randomness to canonize input.

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]	✓	✓	

Corollary

For $L_2 \in \{FO, LFP\}$, there exists an $(LFP[\oplus], L_2)$ -pseudorandom generator from σ to τ if and only if σ has at least one non-unary relation (or σ and τ both have only unary relations, with σ containing at least as many as τ).

Proof:

- 1 Use randomness to canonize input.
- 2 Use Immerman-Vardi Theorem to simulate deterministic generator.

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]	✓	✓	

Theorem

Let $\sigma = \langle R_1, R_2, \dots, R_t \rangle$ and $\tau = \langle R'_1, R'_2, \dots, R'_{t'} \rangle$ be relational signatures, where R_i has arity a_i and R'_i has arity a'_i . The following statements are equivalent:

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]	✓	✓	

Theorem

Let $\sigma = \langle R_1, R_2, \dots, R_t \rangle$ and $\tau = \langle R'_1, R'_2, \dots, R'_{t'} \rangle$ be relational signatures, where R_i has arity a_i and R'_i has arity a'_i . The following statements are equivalent:

- 1 There exists an (LFP, FO)-pseudorandom generator from σ to τ .

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP			
LFP[\oplus]	✓	✓	

Theorem

Let $\sigma = \langle R_1, R_2, \dots, R_t \rangle$ and $\tau = \langle R'_1, R'_2, \dots, R'_{t'} \rangle$ be relational signatures, where R_i has arity a_i and R'_i has arity a'_i . The following statements are equivalent:

- 1 There exists an (LFP, FO)-pseudorandom generator from σ to τ .
- 2 There exists a *truly random* quantifier-free FO transduction from σ to τ .

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO			
LFP	\times		
LFP[\oplus]	\checkmark	\checkmark	

Theorem

Let $\sigma = \langle R_1, R_2, \dots, R_t \rangle$ and $\tau = \langle R'_1, R'_2, \dots, R'_{t'} \rangle$ be relational signatures, where R_i has arity a_i and R'_i has arity a'_i . The following statements are equivalent:

- 1 There exists an (LFP, FO)-pseudorandom generator from σ to τ .
- 2 There exists a *truly random* quantifier-free FO transduction from σ to τ .

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO	✗	✗	✗
LFP	✗	✗	✗
LFP[\oplus]	✓	✓	

Theorem

Let $\sigma = \langle R_1, R_2, \dots, R_t \rangle$ and $\tau = \langle R'_1, R'_2, \dots, R'_{t'} \rangle$ be relational signatures, where R_i has arity a_i and R'_i has arity a'_i . The following statements are equivalent:

- 1 There exists an (LFP, FO)-pseudorandom generator from σ to τ .
- 2 There exists a *truly random* quantifier-free FO transduction from σ to τ .

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO	✗	✗	✗
LFP	✗	✗	✗
LFP[\oplus]	✓	✓	

Theorem

Let $\sigma = \langle R_1, R_2, \dots, R_t \rangle$ and $\tau = \langle R'_1, R'_2, \dots, R'_{t'} \rangle$ be relational signatures, where R_i has arity a_i and R'_i has arity a'_i . The following statements are equivalent:

- 1 There exists an (LFP, FO)-pseudorandom generator from σ to τ .
- 2 There exists a *truly random* quantifier-free FO transduction from σ to τ .
- 3 For every positive integer k ,

$$\sum_{i=1}^t T(a_i, k) \geq \sum_{i=1}^{t'} T(a'_i, k) \quad \text{where} \quad T(a, k) := \sum_{j=0}^k (-1)^{k-j} j^a \binom{k}{j}.$$

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO	✗	✗	✗
LFP	✗	✗	✗
LFP[\oplus]	✓	✓	

Theorem

Let σ and τ be relational signatures.

- 1 If the relations in σ have lexicographically higher arities than the relations in τ , then there exists an $(\text{LFP}[\oplus], \text{LFP}[\oplus])$ -pseudorandom generator from σ to τ .

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO	✗	✗	✗
LFP	✗	✗	✗
LFP[\oplus]	✓	✓	

Theorem

Let σ and τ be relational signatures.

- 1 If the relations in σ have lexicographically higher arities than the relations in τ , then there exists an $(\text{LFP}[\oplus], \text{LFP}[\oplus])$ -pseudorandom generator from σ to τ .
- 2 Otherwise, if σ has only unary relations, there does not exist an $(\text{LFP}[\oplus], \text{LFP}[\oplus])$ -pseudorandom generator from σ to τ .

Generator logic $L_1 \setminus$ Adversary logic L_2	FO	LFP	LFP[\oplus]
FO	✗	✗	✗
LFP	✗	✗	✗
LFP[\oplus]	✓	✓	\iff OWFs exist

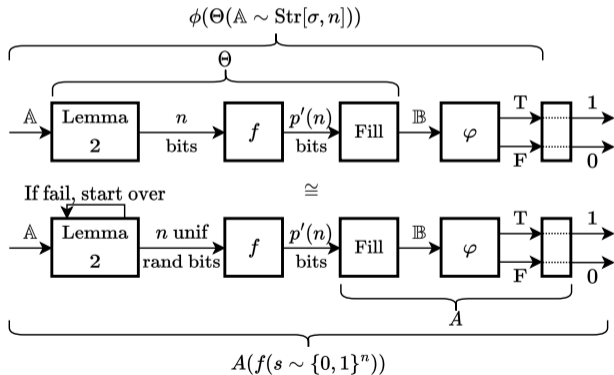
Theorem

Let σ and τ be relational signatures.

- 1 If the relations in σ have lexicographically higher arities than the relations in τ , then there exists an $(\text{LFP}[\oplus], \text{LFP}[\oplus])$ -pseudorandom generator from σ to τ .
- 2 Otherwise, if σ has only unary relations, there does not exist an $(\text{LFP}[\oplus], \text{LFP}[\oplus])$ -pseudorandom generator from σ to τ .
- 3 Otherwise, $(\text{LFP}[\oplus], \text{LFP}[\oplus])$ -pseudorandom generators from σ to τ exist if and only if length-increasing ordinary pseudorandom generators exist (a.k.a. One-Way Functions exist).

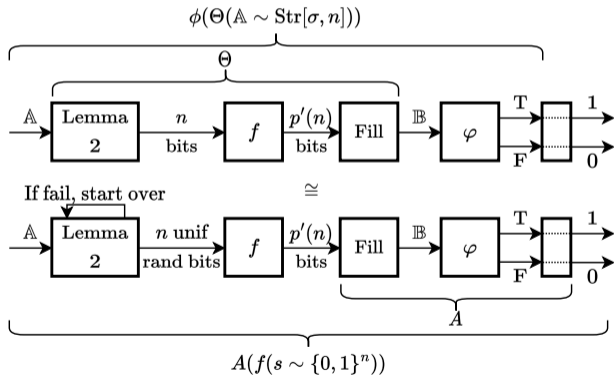
Proof ingredients for OWF equivalence

- Converting between random strings and structures



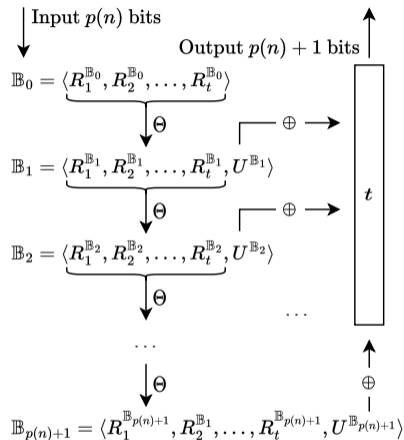
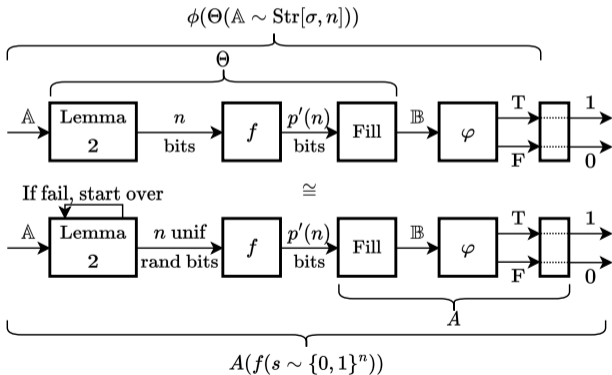
Proof ingredients for OWF equivalence

- Converting between random strings and structures
- Random canonization while saving $\Omega(n^2)$ bits



Proof ingredients for OWF equivalence

- Converting between random strings and structures
- Random canonization while saving $\Omega(n^2)$ bits
- Length-extension + hybrid argument from crypto



Recall the transduction $\Theta_P^t = \langle \theta^t \rangle$ from graphs to t -ary hypergraphs:

$$\theta^t(x_1, \dots, x_t) \equiv \oplus y \bigwedge_{i=1}^t E(y, x_i)$$

Recall the transduction $\Theta_P^t = \langle \theta^t \rangle$ from graphs to t -ary hypergraphs:

$$\theta^t(x_1, \dots, x_t) \equiv \oplus y \bigwedge_{i=1}^t E(y, x_i)$$

Theorem

For any t , Θ_P^t is an $(\text{FO}[\oplus], \text{LFP})$ -pseudorandom generator.

Recall the transduction $\Theta_P^t = \langle \theta^t \rangle$ from graphs to t -ary hypergraphs:

$$\theta^t(x_1, \dots, x_t) \equiv \oplus y \bigwedge_{i=1}^t E(y, x_i)$$

Theorem

For any t , Θ_P^t is an $(\text{FO}[\oplus], \text{LFP})$ -pseudorandom generator.

Conjecture

For any t , Θ_P^t is an $(\text{FO}[\oplus], \text{FO}[\oplus])$ -pseudorandom generator.

Is $\text{FO}[\oplus]$ secure against itself?

Recall the transduction $\Theta_P^t = \langle \theta^t \rangle$ from graphs to t -ary hypergraphs:

$$\theta^t(x_1, \dots, x_t) \equiv \oplus y \bigwedge_{i=1}^t E(y, x_i)$$

Theorem

For any t , Θ_P^t is an $(\text{FO}[\oplus], \text{LFP})$ -pseudorandom generator.

Conjecture

For any t , Θ_P^t is an $(\text{FO}[\oplus], \text{FO}[\oplus])$ -pseudorandom generator.

Suffices to show that Θ_P^t produces structures satisfying the limiting satisfaction probabilities of [Kolaitis and Kopparty, 2009].

Contributions:

- Framework to analyze cryptographic concepts in finite model theory

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP
- Complete characterization of when pseudorandom transductions exist for FO, LFP, and LFP[\oplus]

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP
- Complete characterization of when pseudorandom transductions exist for FO, LFP, and LFP[\oplus]
- Partial progress for FO[\oplus]

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP
- Complete characterization of when pseudorandom transductions exist for FO, LFP, and LFP[\oplus]
- Partial progress for FO[\oplus]

Potential future directions:

- Is FO[\oplus] secure against itself? Other natural logics?

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP
- Complete characterization of when pseudorandom transductions exist for FO, LFP, and LFP[\oplus]
- Partial progress for FO[\oplus]

Potential future directions:

- Is FO[\oplus] secure against itself? Other natural logics?
- Model-theoretic cryptosystems?

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP
- Complete characterization of when pseudorandom transductions exist for FO, LFP, and LFP[\oplus]
- Partial progress for FO[\oplus]

Potential future directions:

- Is FO[\oplus] secure against itself? Other natural logics?
- Model-theoretic cryptosystems?

We thank Boaz Barak for helpful pointers and feedback.

Contributions:

- Framework to analyze cryptographic concepts in finite model theory
- Efficient constructions of pseudorandom graphs and structures for FO and LFP
- Complete characterization of when pseudorandom transductions exist for FO, LFP, and LFP[\oplus]
- Partial progress for FO[\oplus]

Potential future directions:

- Is FO[\oplus] secure against itself? Other natural logics?
- Model-theoretic cryptosystems?

We thank Boaz Barak for helpful pointers and feedback.

Thank you for listening!