Near-Optimal Pseudorandom Generators for Constant-Depth Read-Once Formulas

Dean Doron[1]
UT Austin → Stanford

Pooya Hatami[2]
UT Austin → Ohio State

William M. Hoza[3]
UT Austin

July 19
CCC 2019

[1] Supported by NSF Grant CCF-1705028

[2] Supported by a Simons Investigator Award (#409864, David Zuckerman)

[3] Supported by the NSF GRFP under Grant DGE-1610403 and by a Harrington Fellowship from UT Austin

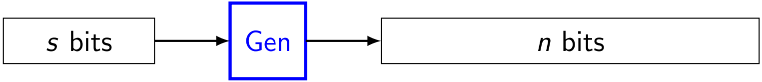
Randomness as a scarce resource

- Randomization is a popular algorithmic technique
- But randomness is costly

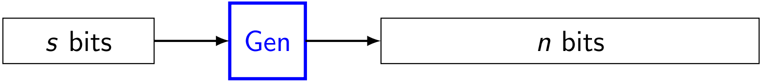
Randomness as a scarce resource

- Randomization is a popular algorithmic technique
- But randomness is costly
- An algorithm that uses fewer random bits is better

Pseudorandom generators (PRGs)



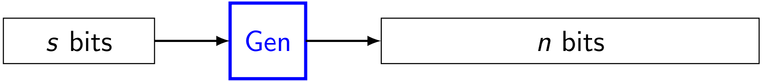
Pseudorandom generators (PRGs)



- Gen "fools" $f : \{0,1\}^n \to \{0,1\}$ if

$$\mathbb{E}[f(\mathrm{Gen}(U))] = \mathbb{E}[f(U)] \pm \varepsilon$$

Pseudorandom generators (PRGs)

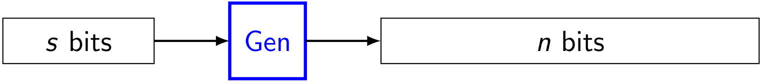


- Gen "fools" $f : \{0,1\}^n \to \{0,1\}$ if

$$\mathbb{E}[f(\mathsf{Gen}(U))] = \mathbb{E}[f(U)] \pm \varepsilon$$

- Goal: Design PRG that fools an interesting class of functions f

Pseudorandom generators (PRGs)

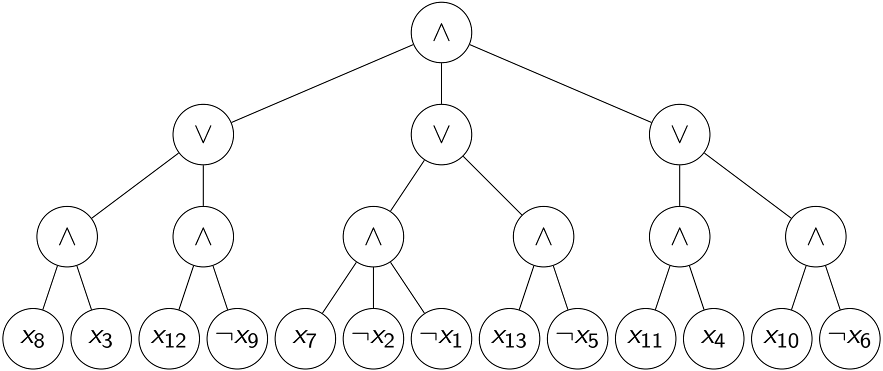


- Gen "fools" $f : \{0,1\}^n \to \{0,1\}$ if

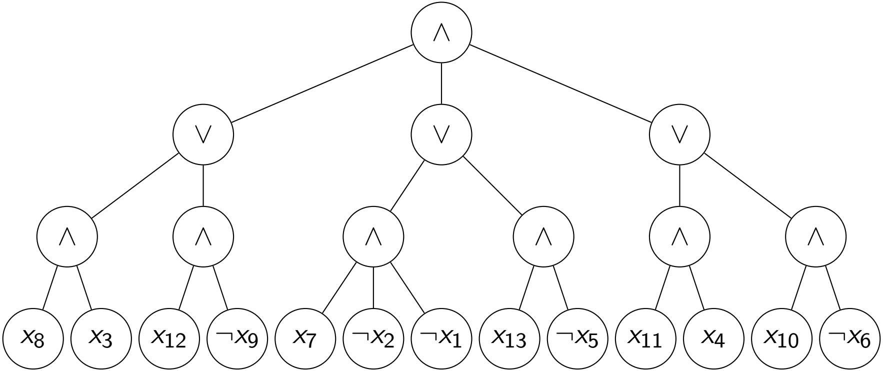
$$\mathbb{E}[f(\mathrm{Gen}(U))] = \mathbb{E}[f(U)] \pm \varepsilon$$

- Goal: Design PRG that fools an interesting class of functions f
- Minimize seed length $s = s(n, \varepsilon)$

Read-once formulas

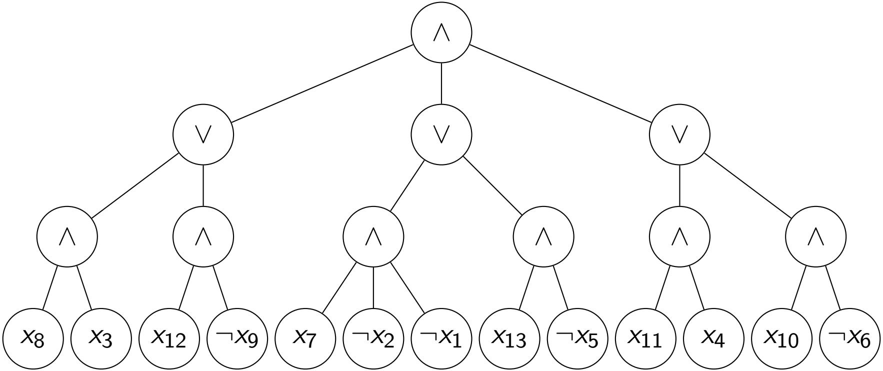


Read-once formulas



- This work: Fool depth-d read-once formulas for $d = O(1)$

Read-once formulas



- This work: Fool depth-d read-once formulas for $d = O(1)$
- Read-once version of $\mathbf{AC}^0$

Prior work and main result

Seed length	Model fooled	Reference
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson ’89

Prior work and main result

Seed length	Model fooled	Reference
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91

Prior work and main result

Seed length	Model fooled	Reference
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13

Prior work and main result

Seed length	**Model fooled**	**Reference**
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13
$\widetilde{O}(\log^{d+1} n)$	Read-once $\mathbf{AC}^0$	Chen, Steinke, Vadhan '15

Prior work and main result

Seed length	**Model fooled**	**Reference**
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13
$\widetilde{O}(\log^{d+1} n)$	Read-once $\mathbf{AC}^0$	Chen, Steinke, Vadhan '15
$\widetilde{O}(\log^2 n)$	Arbitrary-order width-$O(1)$ ROBPs	Forbes, Kelley '18

Prior work and main result

Seed length	**Model fooled**	**Reference**
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13
$\widetilde{O}(\log^{d+1} n)$	Read-once $\mathbf{AC}^0$	Chen, Steinke, Vadhan '15
$\widetilde{O}(\log^2 n)$	Arbitrary-order width-$O(1)$ ROBPs	Forbes, Kelley '18
$\widetilde{O}(\log n)$	Read-once $\mathbf{AC}^0$	This work

Prior work and main result

Seed length	Model fooled	Reference
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13
$\widetilde{O}(\log^{d+1} n)$	Read-once $\mathbf{AC}^0$	Chen, Steinke, Vadhan '15
$\widetilde{O}(\log^2 n)$	Arbitrary-order width-$O(1)$ ROBPs	Forbes, Kelley '18
$\widetilde{O}(\log n)$	Read-once $\mathbf{AC}^0$	This work

- **Main result:** PRG for read-once $\mathbf{AC}^0$ with seed length

$$\log(n/\varepsilon) \cdot O(d \log\log(n/\varepsilon))^{2d+2}.$$

Motivation: **L** vs. **BPL**

- Big open problem: Prove $\mathbf{L} = \mathbf{BPL}$

Motivation: **L** vs. **BPL**

- Big open problem: Prove $\mathbf{L} = \mathbf{BPL}$
  - "Randomness is not necessary for space-efficient computation"

Motivation: **L** vs. **BPL**

- Big open problem: Prove $\mathbf{L} = \mathbf{BPL}$
  - "Randomness is not necessary for space-efficient computation"
- Main approach: Design optimal PRG for "ROBPs"

Motivation: **L** vs. **BPL**

- Big open problem: Prove $\mathbf{L} = \mathbf{BPL}$
  - "Randomness is not necessary for space-efficient computation"
- Main approach: Design optimal PRG for "ROBPs"
- Bad news: Seed length $O(\log^2 n)$ has not been improved for decades [Nisan '92]

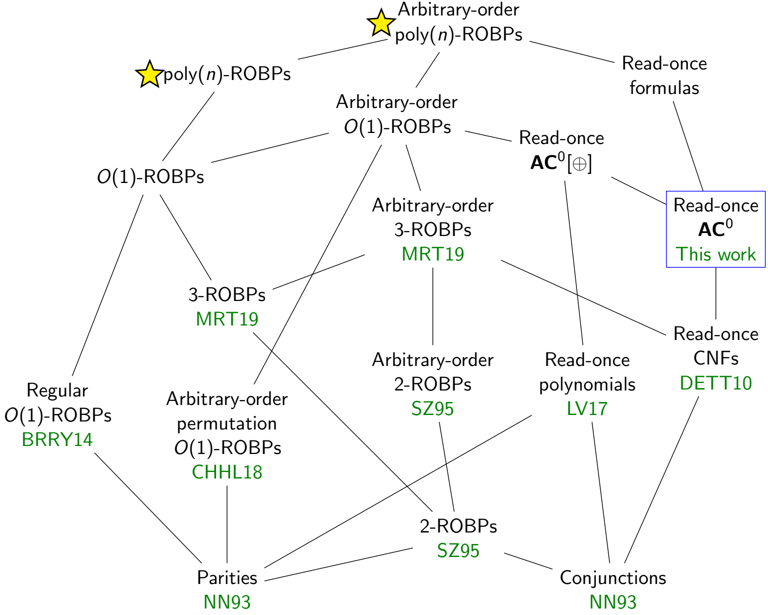
Motivation: $\mathbf{L}$ vs. $\mathbf{BPL}$

- Big open problem: Prove $\mathbf{L} = \mathbf{BPL}$
  - "Randomness is not necessary for space-efficient computation"
- Main approach: Design optimal PRG for "ROBPs"
- Bad news: Seed length $O(\log^2 n)$ has not been improved for decades [Nisan '92]
- Good news: Can achieve seed length $\widetilde{O}(\log n)$ for increasingly powerful restricted models

Motivation: **L** vs. **BPL**

- Big open problem: Prove $\mathbf{L} = \mathbf{BPL}$
  - "Randomness is not necessary for space-efficient computation"
- Main approach: Design optimal PRG for "ROBPs"
- Bad news: Seed length $O(\log^2 n)$ has not been improved for decades [Nisan '92]
- Good news: Can achieve seed length $\widetilde{O}(\log n)$ for increasingly powerful restricted models
- Read-once $\mathbf{AC}^0$ is one of the frontiers of this progress

Seed length $\widetilde{O}(\log n)$



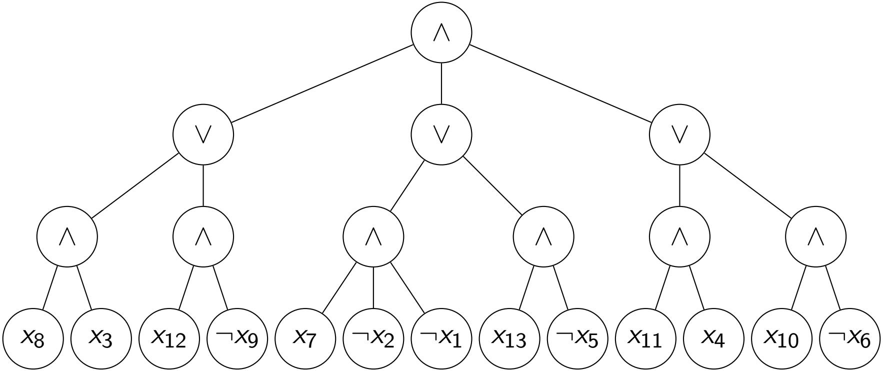
Starting point: Forbes-Kelley PRG

Seed length	**Model fooled**	**Reference**
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13
$\widetilde{O}(\log^{d+1} n)$	Read-once $\mathbf{AC}^0$	Chen, Steinke, Vadhan '15
$\widetilde{O}(\log^2 n)$	Arbitrary-order width-$O(1)$ ROBPs	Forbes, Kelley '18
$\widetilde{O}(\log n)$	Read-once $\mathbf{AC}^0$	This work

Starting point: Forbes-Kelley PRG

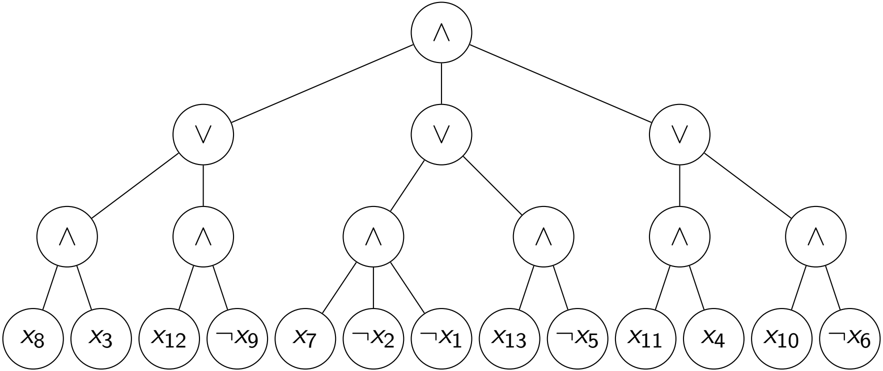
Seed length	Model fooled	Reference
$O(n^{0.001})$	$\mathbf{AC}^0$	Ajtai, Wigderson '89
$O(\log^{2d+6} n)$	$\mathbf{AC}^0$	Nisan '91
$\widetilde{O}(\log^{d+4} n)$	$\mathbf{AC}^0$	Trevisan, Xue '13
$\widetilde{O}(\log^{d+1} n)$	Read-once $\mathbf{AC}^0$	Chen, Steinke, Vadhan '15
$\widetilde{O}(\log^2 n)$	Arbitrary-order width-$O(1)$ ROBPs	Forbes, Kelley '18
$\widetilde{O}(\log n)$	Read-once $\mathbf{AC}^0$	This work

PRGs via pseudorandom restrictions [AW89]



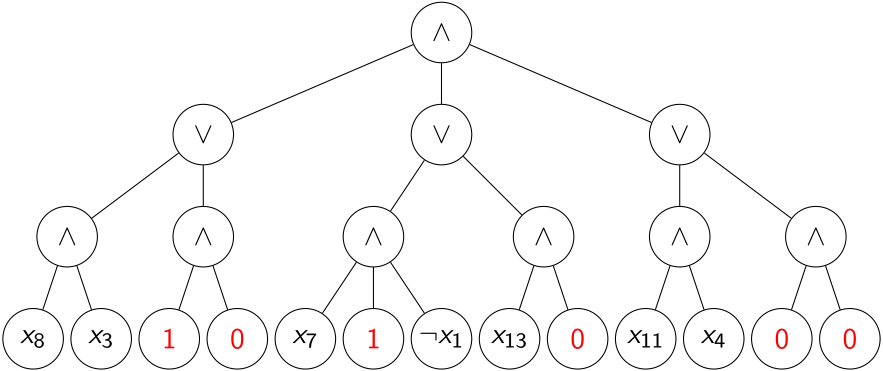
PRGs via pseudorandom restrictions [AW89]

- Start by sampling a pseudorandom restriction $X \in \{0, 1, \star\}^n$



PRGs via pseudorandom restrictions [AW89]

- Start by sampling a pseudorandom restriction $X \in \{0, 1, \star\}^n$



Restriction notation

- Define $\mathsf{Res}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1,\star\}^n$ by

$$\mathsf{Res}(y,z)_i = \begin{cases} \star & \text{if } y_i = 1 \\ z_i & \text{if } y_i = 0 \end{cases}$$

Restriction notation

- Define Res: $\{0,1\}^n \times \{0,1\}^n \to \{0,1,\star\}^n$ by

$$\mathrm{Res}(y,z)_i = \begin{cases} \star & \text{if } y_i = 1 \\ z_i & \text{if } y_i = 0 \end{cases}$$

$y =$	0	1	1	0	0	1	0	0
$z =$	0	0	1	1	1	1	0	1
$\mathrm{Res}(y,z) =$	0	$\star$	$\star$	1	1	$\star$	0	1

Forbes-Kelley pseudorandom restriction

- A distribution D over $\{0,1\}^n$ is ε-biased if it fools parities:

$$S \neq \varnothing \implies \left| \mathbb{E}\left[\bigoplus_{i \in S} D_i \right] - \frac{1}{2} \right| \leq \varepsilon$$

Forbes-Kelley pseudorandom restriction

- A distribution D over $\{0,1\}^n$ is ε-biased if it fools parities:

$$S \neq \varnothing \implies \left| \mathbb{E}\left[\bigoplus_{i \in S} D_i \right] - \frac{1}{2} \right| \leq \varepsilon$$

- Let D, D' be independent small-bias strings

Forbes-Kelley pseudorandom restriction

- A distribution D over $\{0,1\}^n$ is ε-biased if it fools parities:

$$S \neq \varnothing \implies \left| \mathbb{E}\left[\bigoplus_{i \in S} D_i \right] - \frac{1}{2} \right| \leq \varepsilon$$

- Let D, D' be independent small-bias strings
- Let $X = \mathrm{Res}(D, D')$ (seed length $\widetilde{O}(\log n)$)

Forbes-Kelley pseudorandom restriction

- A distribution D over $\{0,1\}^n$ is ε-biased if it fools parities:

$$S \neq \varnothing \implies \left| \mathbb{E}\left[\bigoplus_{i \in S} D_i \right] - \frac{1}{2} \right| \leq \varepsilon$$

- Let D, D' be independent small-bias strings
- Let $X = \mathrm{Res}(D, D')$ (seed length $\widetilde{O}(\log n)$)
- **Theorem** [Forbes, Kelley '18]: For any $O(1)$-width ROBP f,

$$\mathop{\mathbb{E}}_{X,U}[f|_X(U)] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

Forbes-Kelley pseudorandom restriction

- A distribution D over $\{0,1\}^n$ is ε-biased if it fools parities:

$$S \neq \varnothing \implies \left| \mathbb{E}\left[\bigoplus_{i \in S} D_i \right] - \frac{1}{2} \right| \leq \varepsilon$$

- Let D, D' be independent small-bias strings
- Let $X = \mathrm{Res}(D, D')$ (seed length $\widetilde{O}(\log n)$)
- **Theorem** [Forbes, Kelley '18]: For any $O(1)$-width ROBP f,

$$\mathop{\mathbb{E}}_{X,U}[f|_X(U)] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

- In words, X preserves expectation of f

Forbes-Kelley pseudorandom restriction

- A distribution D over $\{0,1\}^n$ is ε-biased if it fools parities:

$$S \neq \varnothing \implies \left| \mathbb{E}\left[\bigoplus_{i \in S} D_i\right] - \frac{1}{2} \right| \leq \varepsilon$$

- Let D, D' be independent small-bias strings
- Let $X = \mathrm{Res}(D, D')$ (seed length $\widetilde{O}(\log n)$)
- **Theorem** [Forbes, Kelley '18]: For any $O(1)$-width ROBP f,

$$\mathop{\mathbb{E}}_{X,U}[f|_X(U)] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

- In words, X preserves expectation of f
- (Proof involves clever Fourier analysis, building on [RSV13, HLV18, CHRT18])

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP
- So we can apply another pseudorandom restriction

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP
- So we can apply another pseudorandom restriction
- Let $X^{\circ t}$ denote composition of t independent copies of X

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP
- So we can apply another pseudorandom restriction
- Let $X^{\circ t}$ denote composition of t independent copies of X
- Let $t = O(\log n)$

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP
- So we can apply another pseudorandom restriction
- Let $X^{\circ t}$ denote composition of t independent copies of X
- Let $t = O(\log n)$
- With high probability, $X^{\circ t} \in \{0, 1\}^n \quad$ (no $\star$)

Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP
- So we can apply another pseudorandom restriction
- Let $X^{\circ t}$ denote composition of t independent copies of X
- Let $t = O(\log n)$
- With high probability, $X^{\circ t} \in \{0,1\}^n$ (no $\star$)
- Expectation preserved at every step, so total error is low:

$$\mathop{\mathbb{E}}_{X^{\circ t}}[f(X^{\circ t})] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

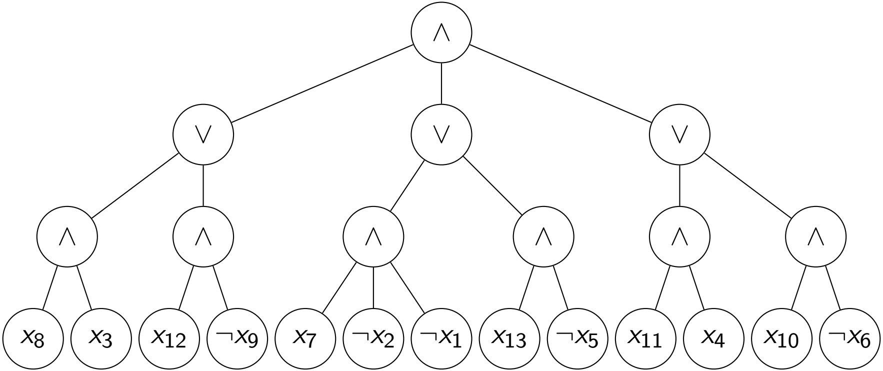
Forbes-Kelley pseudorandom generator

- So [FK18] can assign values to half the inputs using $\widetilde{O}(\log n)$ truly random bits
- After restricting, $f|_X$ is another ROBP
- So we can apply another pseudorandom restriction
- Let $X^{\circ t}$ denote composition of t independent copies of X
- Let $t = O(\log n)$
- With high probability, $X^{\circ t} \in \{0,1\}^n$ (no $\star$)
- Expectation preserved at every step, so total error is low:

$$\mathop{\mathbb{E}}_{X^{\circ t}}[f(X^{\circ t})] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

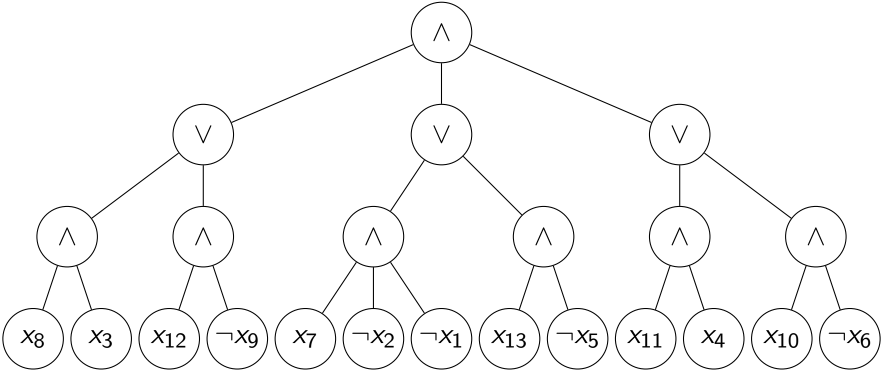
- Total cost: $\widetilde{O}(\log^2 n)$ truly random bits

Improved PRGs via simplification [GMRTV12]



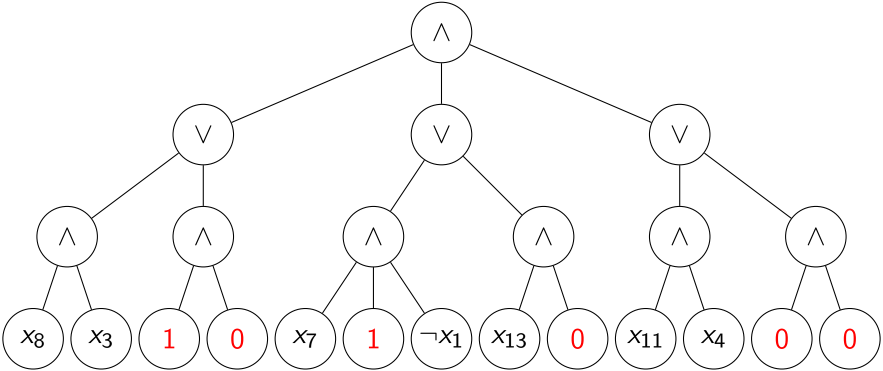
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$



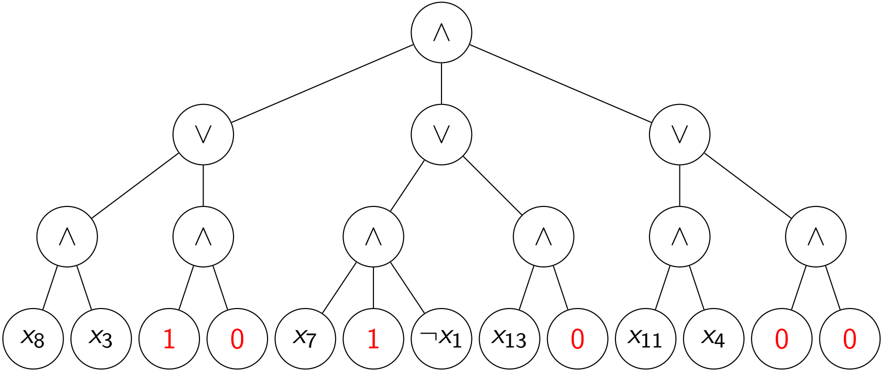
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$



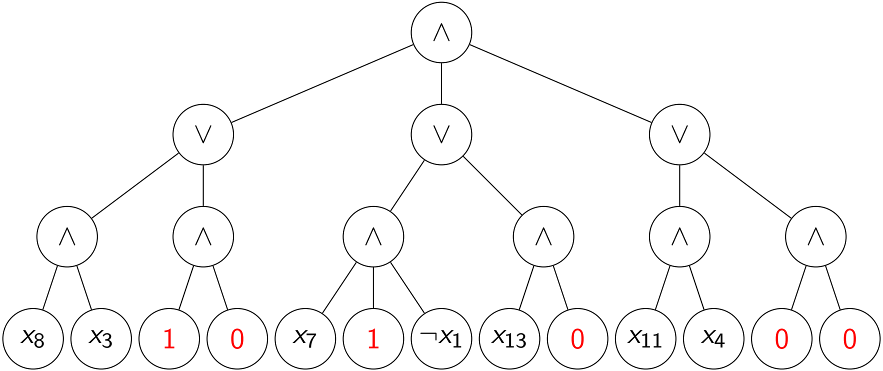
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation



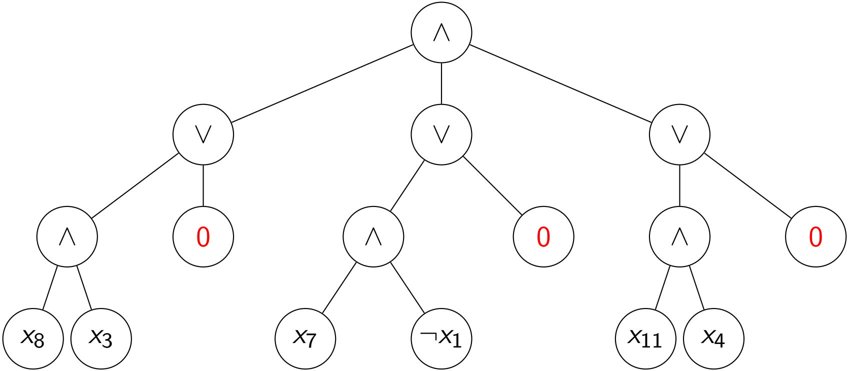
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation
- Design X so that $X^{\circ t}$ also simplifies formula, for $t \ll \log n$



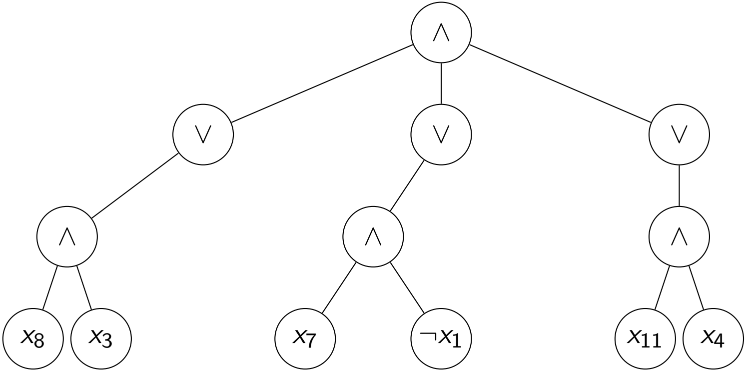
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation
- Design X so that $X^{\circ t}$ also simplifies formula, for $t \ll \log n$



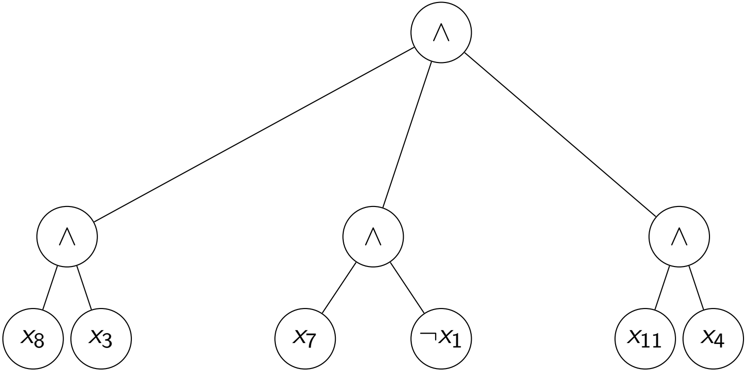
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation
- Design X so that $X^{\circ t}$ also simplifies formula, for $t \ll \log n$



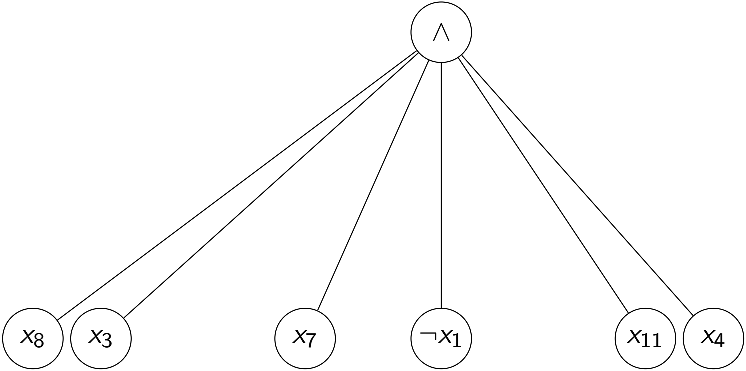
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation
- Design X so that $X^{\circ t}$ also simplifies formula, for $t \ll \log n$



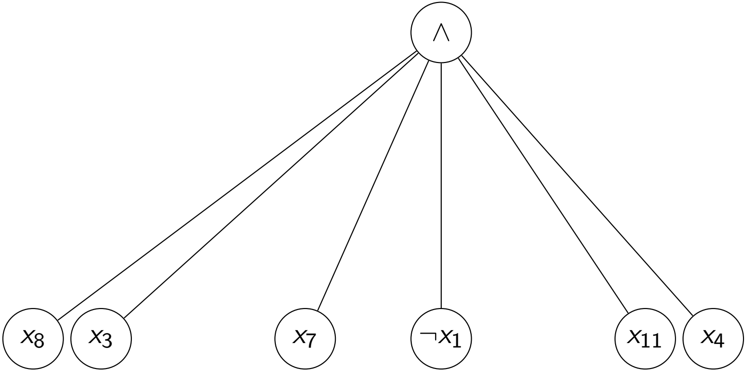
Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation
- Design X so that $X^{\circ t}$ also simplifies formula, for $t \ll \log n$



Improved PRGs via simplification [GMRTV12]

- Step 1: Apply pseudorandom restriction $X \in \{0, 1, \star\}^n$
- Design X to preserve expectation
- Design X so that $X^{\circ t}$ also simplifies formula, for $t \ll \log n$



- Step 2: Fool restricted formula, taking advantage of simplicity

Our pseudorandom restriction

- Assume by recursion: PRG for depth d with seed length $\widetilde{O}(\log n)$

Our pseudorandom restriction

- Assume by recursion: PRG for depth d with seed length $\widetilde{O}(\log n)$
- Let's sample $X \in \{0, 1, \star\}^n$ for depth $d + 1$

Our pseudorandom restriction

- Assume by recursion: PRG for depth d with seed length $\widetilde{O}(\log n)$
- Let's sample $X \in \{0, 1, \star\}^n$ for depth $d + 1$

1. Recursively sample $G_d, G'_d \in \{0, 1\}^n$

Our pseudorandom restriction

- Assume by recursion: PRG for depth d with seed length $\widetilde{O}(\log n)$
- Let's sample $X \in \{0, 1, \star\}^n$ for depth $d + 1$

1. Recursively sample $G_d, G'_d \in \{0, 1\}^n$
2. Sample $D, D' \in \{0, 1\}^n$ with small bias

Our pseudorandom restriction

- Assume by recursion: PRG for depth d with seed length $\widetilde{O}(\log n)$
- Let's sample $X \in \{0, 1, \star\}^n$ for depth $d + 1$

1. Recursively sample $G_d, G'_d \in \{0, 1\}^n$
2. Sample $D, D' \in \{0, 1\}^n$ with small bias
3. $X = \mathrm{Res}(G_d \oplus D, G'_d \oplus D')$

Preserving expectation

- **Claim**: For any depth-$(d+1)$ read-once $\mathbf{AC}^0$ formula f,

$$\mathop{\mathbb{E}}_{X,U}[f|_X(U)] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

Preserving expectation

- **Claim**: For any depth-$(d+1)$ read-once $\mathbf{AC}^0$ formula f,

$$\mathop{\mathbb{E}}_{X,U}[f|_X(U)] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

- **Proof**: Read-once $\mathbf{AC}^0$ can be simulated by constant-width ROBPs [CSV15]

Preserving expectation

- **Claim**: For any depth-$(d+1)$ read-once $\mathbf{AC}^0$ formula f,

$$\mathop{\mathbb{E}}_{X,U}[f|_X(U)] \approx \mathop{\mathbb{E}}_{U}[f(U)]$$

- **Proof**: Read-once $\mathbf{AC}^0$ can be simulated by constant-width ROBPs [CSV15]
- So we can simply apply Forbes-Kelley result:

$$X = \mathrm{Res}(G_d \oplus D, G'_d \oplus D')$$

Simplification

- $\Delta(f) \stackrel{\text{def}}{=}$ maximum fan-in of any gate other than root

Simplification

- $\Delta(f) \stackrel{\text{def}}{=}$ maximum fan-in of any gate other than root
- **Main Lemma**: With high probability over $X^{\circ t}$,

$$\Delta(f|_{X^{\circ t}}) \leq \text{polylog } n,$$

where $t = O((\log \log n)^2)$

Simplification

- $\Delta(f) \stackrel{\text{def}}{=}$ maximum fan-in of any gate other than root
- **Main Lemma**: With high probability over $X^{\circ t}$,

$$\Delta(f|_{X^{\circ t}}) \leq \text{polylog}\, n,$$

  where $t = O((\log\log n)^2)$
- Actually we only prove this statement "up to sandwiching"

Simplification under truly random restrictions

- Let f be a read-once $\mathbf{AC}^0$ formula

Simplification under truly random restrictions

- Let f be a read-once $\mathbf{AC}^0$ formula
- Let $R = \mathrm{Res}(U, U')$ (truly random restriction)

Simplification under truly random restrictions

- Let f be a read-once $\mathbf{AC}^0$ formula
- Let $R = \mathrm{Res}(U, U')$ (truly random restriction)
- Chen, Steinke, Vadhan '15 $\implies$ W.h.p. over $R^{\circ t}$,

$$\Delta(f|_{R^{\circ t}}) \leq \text{polylog}\, n$$

Simplification under truly random restrictions

- Let f be a read-once $\mathbf{AC}^0$ formula
- Let $R = \text{Res}(U, U')$ (truly random restriction)
- Chen, Steinke, Vadhan '15 $\implies$ W.h.p. over $R^{\circ t}$,

$$\Delta(f|_{R^{\circ t}}) \leq \text{polylog}\, n$$

- (In fact the simplification they show is more severe)

Simplification under truly random restrictions

- Let f be a read-once $\mathbf{AC}^0$ formula
- Let $R = \text{Res}(U, U')$ (truly random restriction)
- Chen, Steinke, Vadhan '15 $\implies$ W.h.p. over $R^{\circ t}$,

$$\Delta(f|_{R^{\circ t}}) \leq \text{polylog } n$$

- (In fact the simplification they show is more severe)
- Again, these statements are true "up to sandwiching." Proof uses Fourier analysis

Derandomizing simplification

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula

Derandomizing simplification

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0, 1\}$

Derandomizing simplification

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0,1\}$
- Computational problem: Given $y, z \in \{0,1\}^n$, decide whether

$$f|_{\mathrm{Res}(y,z)} \equiv b$$

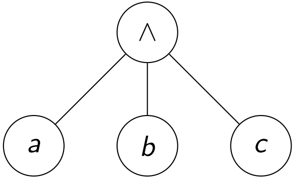
Derandomizing simplification

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0,1\}$
- Computational problem: Given $y, z \in \{0,1\}^n$, decide whether

$$f|_{\mathsf{Res}(y,z)} \equiv b$$

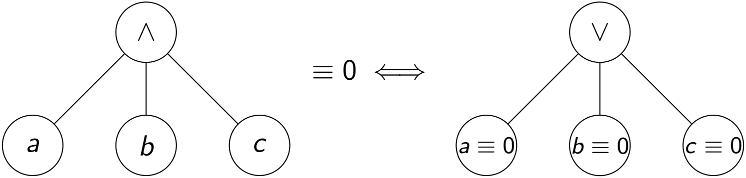
- **Lemma**: Can be decided in depth-d read-once $\mathbf{AC}^0$

Deciding whether $f|_{\text{Res}(y,z)} \equiv b$

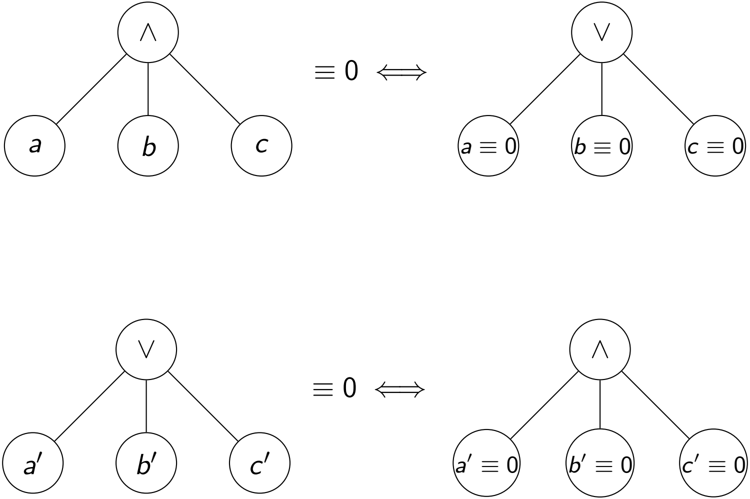


$\equiv 0 \iff$

Deciding whether $f|_{\mathrm{Res}(y,z)} \equiv b$



Deciding whether $f|_{\mathrm{Res}(y,z)} \equiv b$



Deciding whether $f|_{\mathrm{Res}(y,z)} \equiv b$ (continued)

- At bottom, we get one additional layer:

$$(\mathrm{Res}(y,z)_i \equiv b) \iff (y_i = 0 \wedge z_i = b)$$
$$(\neg\,\mathrm{Res}(y,z)_i \equiv b) \iff (y_i = 0 \wedge z_i = 1 - b)$$

Collapse under pseudorandom restrictions

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0, 1\}$

Collapse under pseudorandom restrictions

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0,1\}$
- $X = \text{Res}(G_d \oplus D, G'_d \oplus D')$

Collapse under pseudorandom restrictions

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0, 1\}$
- $X = \mathrm{Res}(G_d \oplus D, G'_d \oplus D')$
- G_d, G'_d fool depth d, so

$$\Pr_X[f|_X \equiv b] \approx \Pr_R[f|_R \equiv b]$$

Collapse under pseudorandom restrictions

- Let f be a depth-$(d-1)$ read-once $\mathbf{AC}^0$ formula
- Let $b \in \{0,1\}$
- $X = \mathrm{Res}(G_d \oplus D, G'_d \oplus D')$
- G_d, G'_d fool depth d, so

$$\Pr_X[f|_X \equiv b] \approx \Pr_R[f|_R \equiv b]$$

- Hybrid argument:

$$\Pr_{X^{\circ t}}[f|_{X^{\circ t}} \equiv b] \approx \Pr_{R^{\circ t}}[f|_{R^{\circ t}} \equiv b]$$

Bridging the gap from $d-1$ to $d+1$

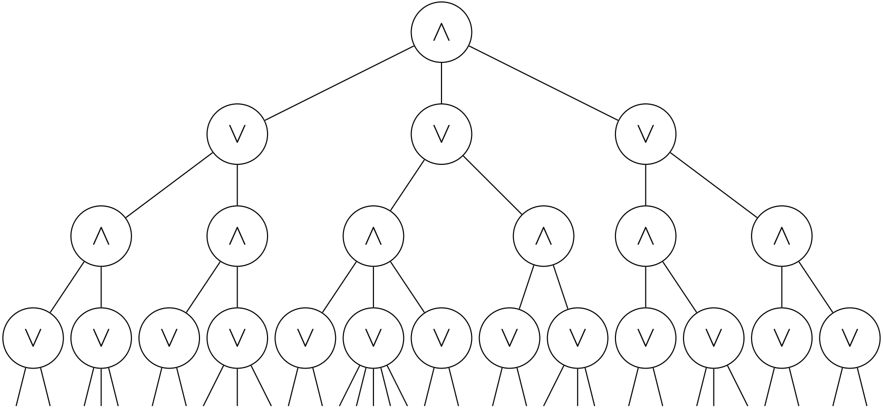
- So far: Depth-$(d-1)$ formulas collapse with about the right probability

Bridging the gap from $d-1$ to $d+1$

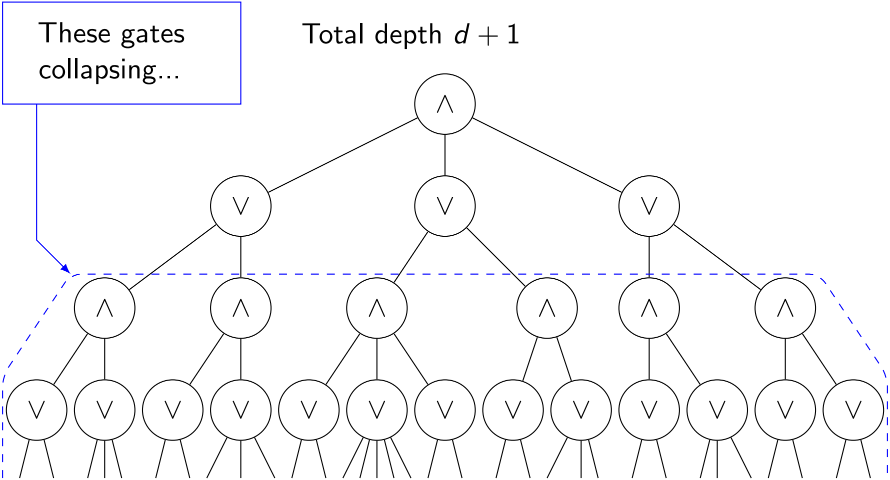
- So far: Depth-$(d-1)$ formulas collapse with about the right probability
- We were supposed to show that depth-$(d+1)$ formulas simplify w.r.t. Δ w.h.p.

Idea of proof that $\Delta \mapsto \text{polylog}\, n$

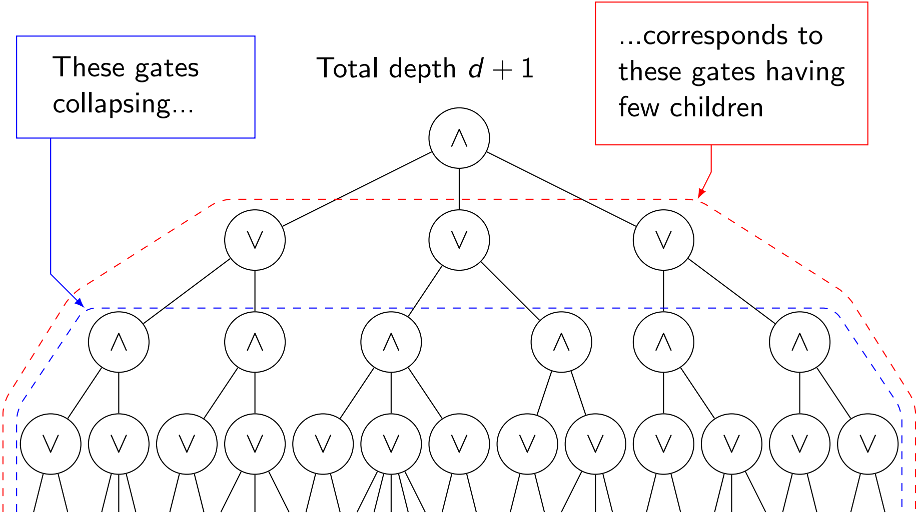
Total depth $d + 1$



Idea of proof that $\Delta \mapsto$ polylog n



Idea of proof that $\Delta \mapsto$ polylog n

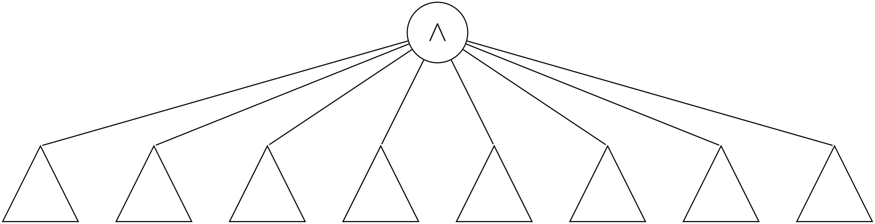


$\Delta = \text{polylog}\, n$

- To recap, after $t = O((\log\log n)^2)$ restrictions, $\Delta = \text{polylog}\, n$

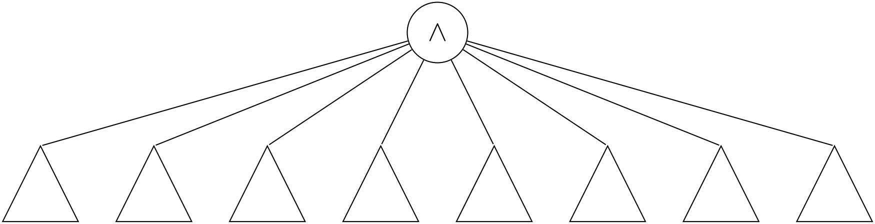
$\Delta = \text{polylog } n$

- To recap, after $t = O((\log \log n)^2)$ restrictions, $\Delta = \text{polylog } n$



$\Delta = \text{polylog } n$

- To recap, after $t = O((\log\log n)^2)$ restrictions, $\Delta = \text{polylog } n$
- Total cost so far: $\widetilde{O}(\log n)$ truly random bits



Final step: MRT PRG

- **Theorem** (Meka, Reingold, Tal '19): There is an explicit PRG with seed length $\widetilde{O}(\log(n/\varepsilon))$ that fools functions of the form

$$f = \bigoplus_{i=1}^{m} f_i,$$

Final step: MRT PRG

- **Theorem** (Meka, Reingold, Tal '19): There is an explicit PRG with seed length $\widetilde{O}(\log(n/\varepsilon))$ that fools functions of the form

$$f = \bigoplus_{i=1}^{m} f_i,$$

where $f_1, \ldots, f_m$ are on disjoint variables and f_i can be computed by an ROBP with width $O(1)$, length polylog n

Final step: MRT PRG

- **Theorem** (Meka, Reingold, Tal '19): There is an explicit PRG with seed length $\widetilde{O}(\log(n/\varepsilon))$ that fools functions of the form

$$f = \bigoplus_{i=1}^{m} f_i,$$

where $f_1, \ldots, f_m$ are on disjoint variables and f_i can be computed by an ROBP with width $O(1)$, length polylog n

- (Proof uses GMRTV approach, building on [GY14, CHRT18, Vio09])

Final step: MRT PRG

- **Theorem** (Meka, Reingold, Tal '19): There is an explicit PRG with seed length $\widetilde{O}(\log(n/\varepsilon))$ that fools functions of the form

$$f = \bigoplus_{i=1}^{m} f_i,$$

where $f_1, \dots, f_m$ are on disjoint variables and f_i can be computed by an ROBP with width $O(1)$, length $\operatorname{polylog} n$

- (Proof uses GMRTV approach, building on [GY14, CHRT18, Vio09])
- In our case,

$$f = \bigwedge_{i=1}^{m} f_i$$

Final step: MRT PRG

- **Theorem** (Meka, Reingold, Tal '19): There is an explicit PRG with seed length $\widetilde{O}(\log(n/\varepsilon))$ that fools functions of the form

$$f = \bigoplus_{i=1}^{m} f_i,$$

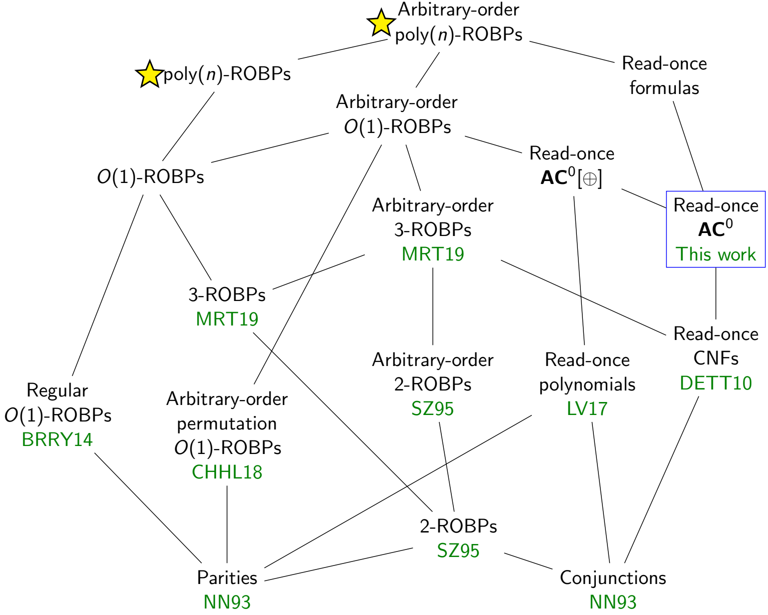
where $f_1, \ldots, f_m$ are on disjoint variables and f_i can be computed by an ROBP with width $O(1)$, length $\operatorname{polylog} n$

- (Proof uses GMRTV approach, building on [GY14, CHRT18, Vio09])
- In our case,

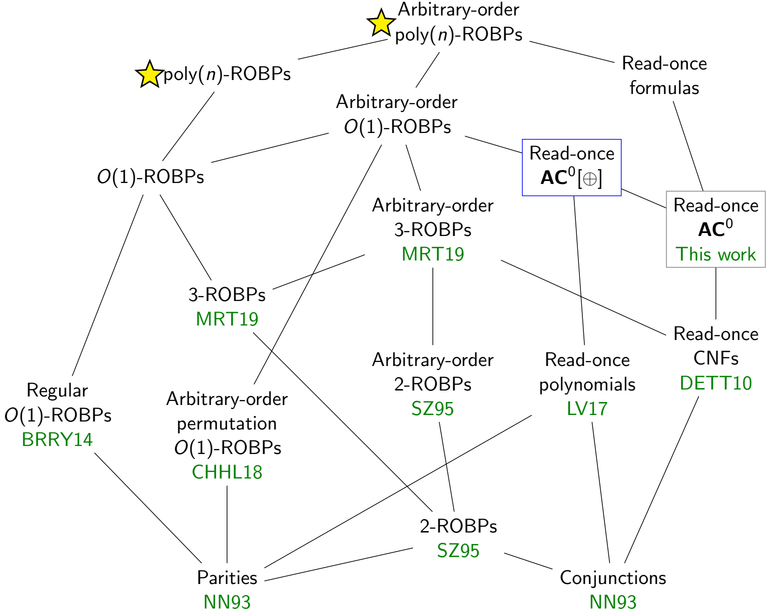
$$f = \bigwedge_{i=1}^{m} f_i = \sum_{S \subseteq [m]} \frac{(-1)^{|S|}}{2^m} \prod_{i \in S} (-1)^{f_i}$$

Directions for further research

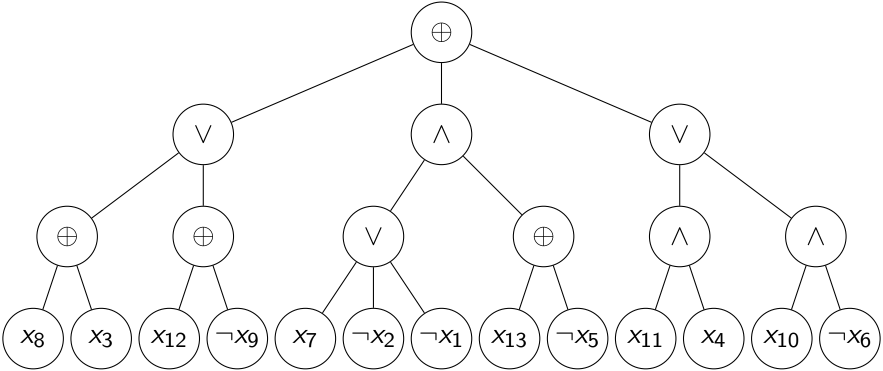
Directions for further research



Directions for further research



Read-once $\mathbf{AC}^0[\oplus]$



Fooling read-once $\mathbf{AC}^0[\oplus]$

- Natural next step toward derandomizing **BPL**

Fooling read-once $\mathbf{AC}^0[\oplus]$

- Natural next step toward derandomizing **BPL**
- Best prior PRG: seed length $\widetilde{O}(\log^2 n)$ [FK '18]

Fooling read-once $\mathbf{AC}^0[\oplus]$

- Natural next step toward derandomizing **BPL**
- Best prior PRG: seed length $\widetilde{O}(\log^2 n)$ [FK '18]
- **Theorem**: Our PRG fools read-once $\mathbf{AC}^0[\oplus]$ with seed length

$$\widetilde{O}(t + \log(n/\varepsilon))$$

where $t = \#$ parity gates

Fooling read-once $\mathbf{AC}^0[\oplus]$

- Natural next step toward derandomizing **BPL**
- Best prior PRG: seed length $\widetilde{O}(\log^2 n)$ [FK '18]
- **Theorem**: Our PRG fools read-once $\mathbf{AC}^0[\oplus]$ with seed length

$$\widetilde{O}(t + \log(n/\varepsilon))$$

  where $t = \#$ parity gates
- Fool read-once $\mathbf{AC}^0[\oplus]$ with seed length $\widetilde{O}(\log(n/\varepsilon))$?

Fooling read-once $\mathbf{AC}^0[\oplus]$

- Natural next step toward derandomizing **BPL**
- Best prior PRG: seed length $\widetilde{O}(\log^2 n)$ [FK '18]
- **Theorem**: Our PRG fools read-once $\mathbf{AC}^0[\oplus]$ with seed length

$$\widetilde{O}(t + \log(n/\varepsilon))$$

  where $t = \#$ parity gates
- Fool read-once $\mathbf{AC}^0[\oplus]$ with seed length $\widetilde{O}(\log(n/\varepsilon))$?

- Thanks! Questions?