Simple Optimal Hitting Sets for Small-Success **RL**

William M. Hoza[1] David Zuckerman[2]

The University of Texas at Austin

September 24, 2018
Dagstuhl Seminar 18391

[1] Supported by the NSF GRFP under Grant DGE-1610403 and by a Harrington Fellowship from UT Austin
[2] Supported by NSF Grant CCF-1526952, NSF Grant CCF-1705028, and a Simons Investigator Award (#409864)

Randomized log-space complexity classes

- Let L be a language

Randomized log-space complexity classes

- Let L be a language
- $L \in \mathbf{BPL}$ if there is a randomized log-space algorithm A that always halts such that

$$\begin{aligned} x \in L &\implies \Pr[A(x) \text{ accepts}] \geq 2/3 \\ x \notin L &\implies \Pr[A(x) \text{ accepts}] \leq 1/3. \end{aligned}$$

Randomized log-space complexity classes

- Let L be a language
- $L \in \mathbf{BPL}$ if there is a randomized log-space algorithm A that always halts such that

$$\begin{aligned} x \in L &\implies \Pr[A(x) \text{ accepts}] \geq 2/3 \\ x \notin L &\implies \Pr[A(x) \text{ accepts}] \leq 1/3. \end{aligned}$$

- $L \in \mathbf{RL}$ if there is a randomized log-space algorithm A that always halts such that

$$\begin{aligned} x \in L &\implies \Pr[A(x) \text{ accepts}] \geq 1/2 \\ x \notin L &\implies \Pr[A(x) \text{ accepts}] = 0. \end{aligned}$$

The power of randomness for small-space algorithms

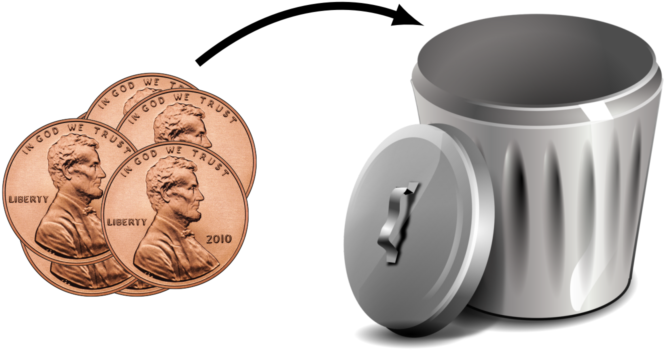
- $\mathbf{L} \subseteq \mathbf{RL} \subseteq \mathbf{BPL}$

The power of randomness for small-space algorithms

- $\mathbf{L} \subseteq \mathbf{RL} \subseteq \mathbf{BPL}$
- **Conjecture**: $\mathbf{L} = \mathbf{RL} = \mathbf{BPL}$

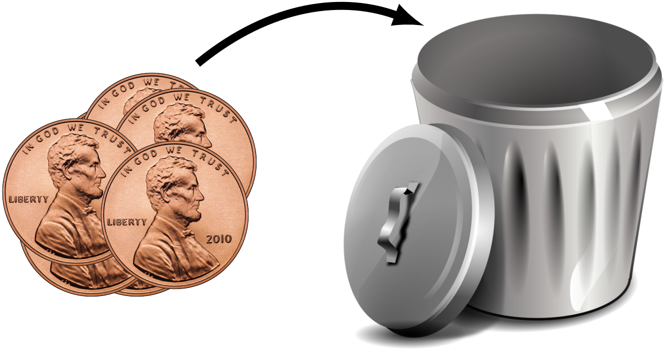
The power of randomness for small-space algorithms

- $\mathbf{L} \subseteq \mathbf{RL} \subseteq \mathbf{BPL}$
- **Conjecture**: $\mathbf{L} = \mathbf{RL} = \mathbf{BPL}$

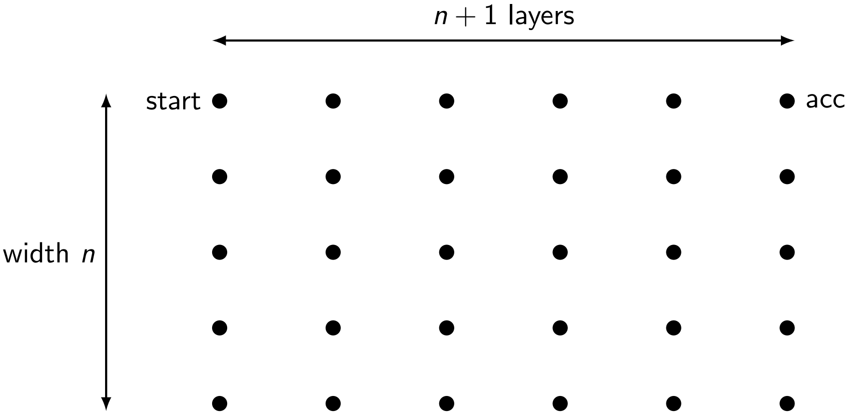


The power of randomness for small-space algorithms

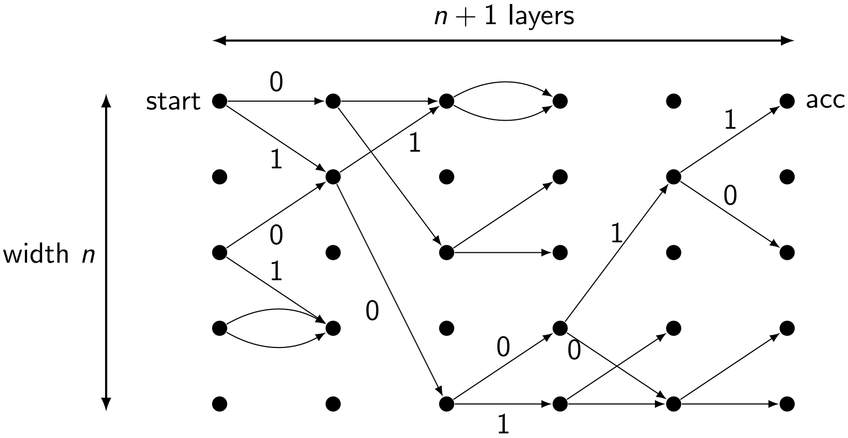
- $\mathbf{L} \subseteq \mathbf{RL} \subseteq \mathbf{BPL}$
- **Conjecture**: $\mathbf{L} = \mathbf{RL} = \mathbf{BPL}$



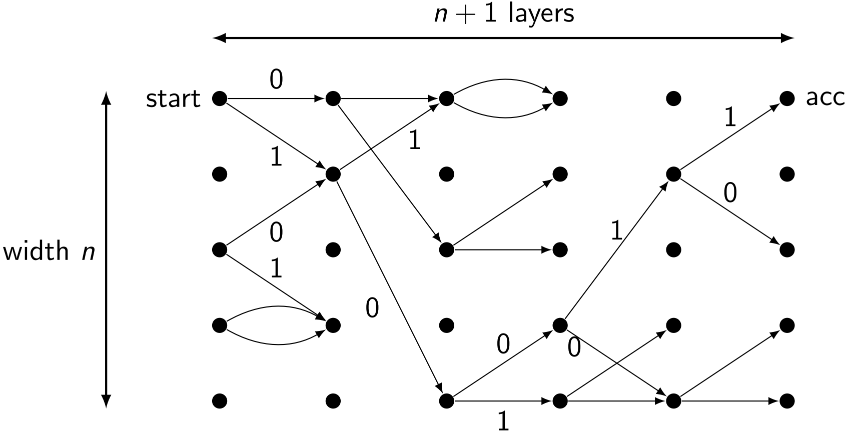
Read-once branching programs



Read-once branching programs

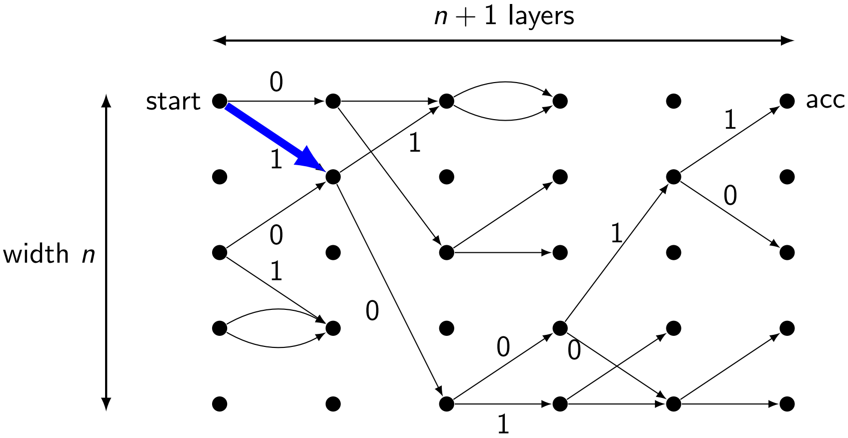


Read-once branching programs



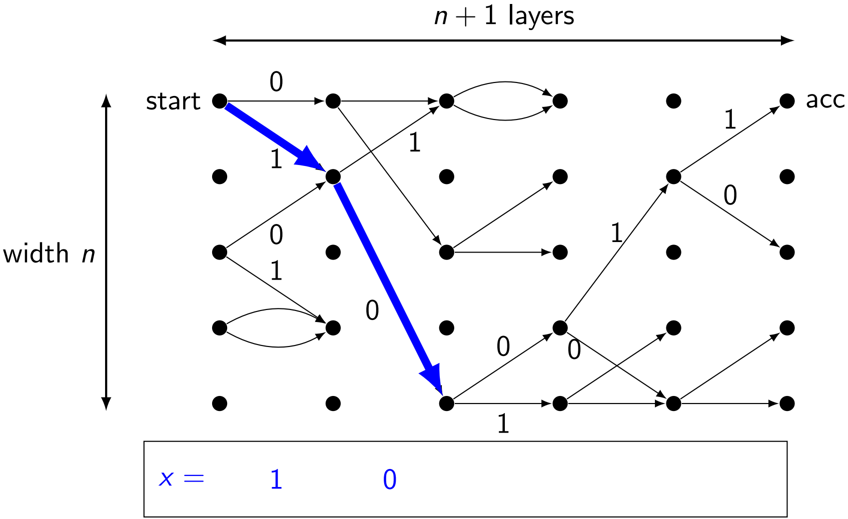
$x =$

Read-once branching programs

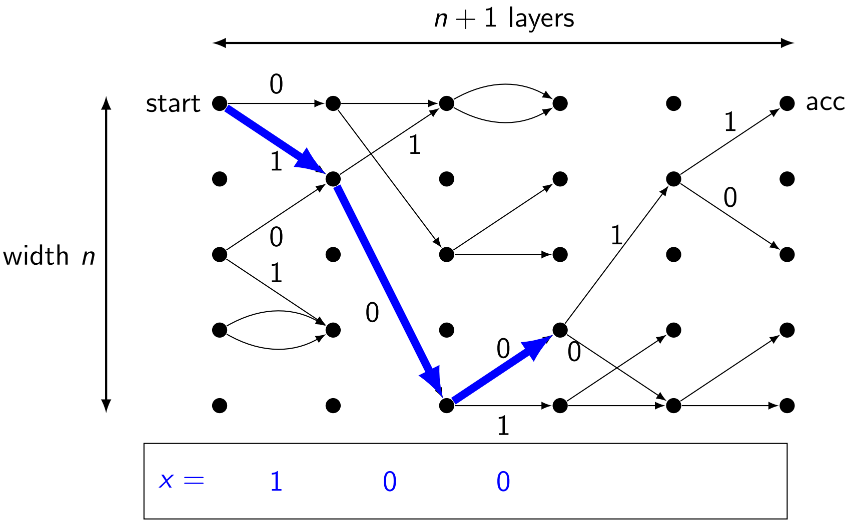


$x =$ 1

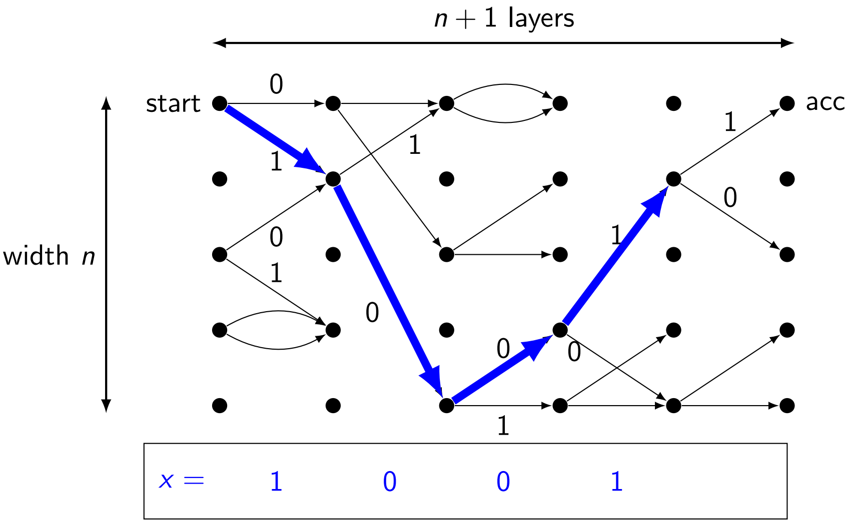
Read-once branching programs



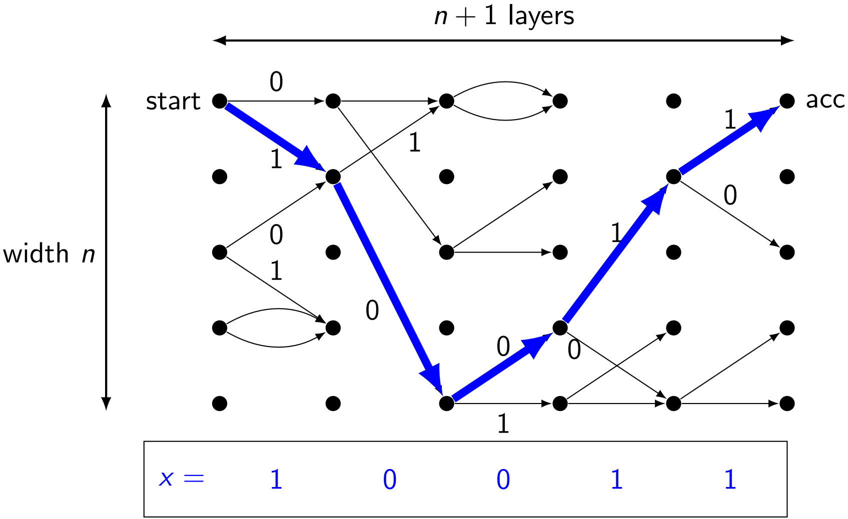
Read-once branching programs



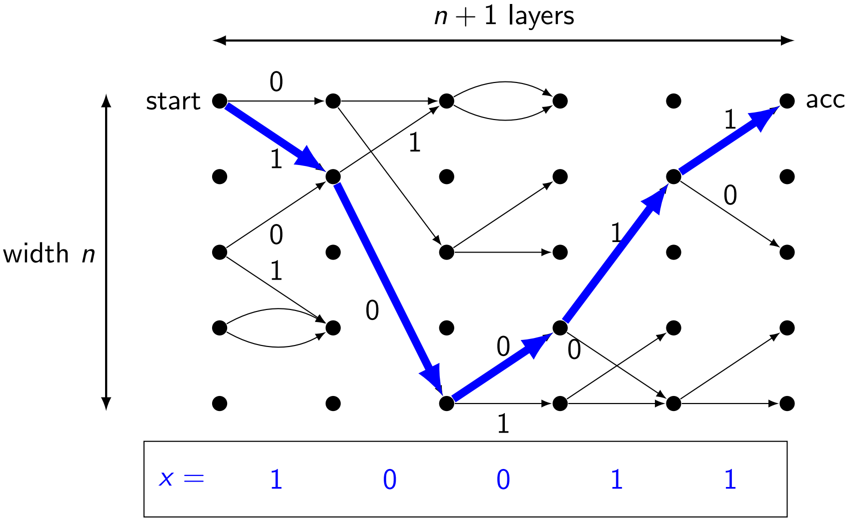
Read-once branching programs



Read-once branching programs

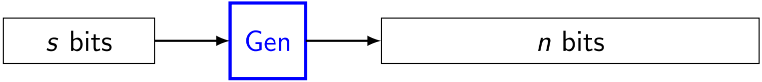


Read-once branching programs

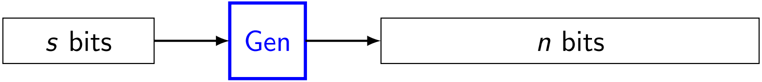


- Computes function $f : \{0,1\}^n \to \{0,1\}$

Fooling / Hitting ROBPs



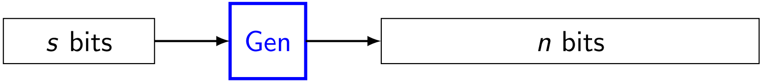
Fooling / Hitting ROBPs



Pseudorandom generator: For every width-n ROBP,

$$\left|\Pr_x[f(x) = 1] - \Pr_z[f(\mathrm{Gen}(z)) = 1]\right| \leq \varepsilon$$

Fooling / Hitting ROBPs



Pseudorandom generator: For every width-n ROBP,

$$|\Pr_x[f(x) = 1] - \Pr_z[f(\mathrm{Gen}(z)) = 1]| \leq \varepsilon$$

Suitable for derandomizing **BPL**

Fooling / Hitting ROBPs

s bits $\longrightarrow$ Gen $\longrightarrow$ n bits

Pseudorandom generator: For every width-n ROBP,

$$\left|\Pr_x[f(x) = 1] - \Pr_z[f(\mathrm{Gen}(z)) = 1]\right| \leq \varepsilon$$

Suitable for derandomizing **BPL**

Hitting set generator: For every width-n ROBP,

$$\Pr_x[f(x) = 1] \geq \varepsilon \implies \exists z, f(\mathrm{Gen}(z)) = 1$$

Fooling / Hitting ROBPs

s bits $\longrightarrow$ Gen $\longrightarrow$ n bits

Pseudorandom generator: For every width-n ROBP, $$\left\lvert \Pr_x[f(x) = 1] - \Pr_z[f(\mathrm{Gen}(z)) = 1]\right\rvert \le \varepsilon$$	Suitable for derandomizing **BPL**
Hitting set generator: For every width-n ROBP, $$\Pr_x[f(x) = 1] \ge \varepsilon \implies \exists z, f(\mathrm{Gen}(z)) = 1$$	Suitable for derandomizing **RL**

Prior generators and main result

- Nonconstructive: PRG with seed length $O(\log n + \log(1/\varepsilon))$

Prior generators and main result

- Nonconstructive: PRG with seed length $O(\log n + \log(1/\varepsilon))$
- Babai, Nisan, Szegedy 1989: PRG with seed length

$$2^{O(\sqrt{\log n})} \cdot \log(1/\varepsilon)$$

Prior generators and main result

- Nonconstructive: PRG with seed length $O(\log n + \log(1/\varepsilon))$
- Babai, Nisan, Szegedy 1989: PRG with seed length

$$2^{O(\sqrt{\log n})} \cdot \log(1/\varepsilon)$$

- Nisan 1990: PRG with seed length

$$O(\log^2 n + \log(1/\varepsilon) \log n)$$

Prior generators and main result

- Nonconstructive: PRG with seed length $O(\log n + \log(1/\varepsilon))$
- Babai, Nisan, Szegedy 1989: PRG with seed length

$$2^{O(\sqrt{\log n})} \cdot \log(1/\varepsilon)$$

- Nisan 1990: PRG with seed length

$$O(\log^2 n + \log(1/\varepsilon) \log n)$$

- Braverman, Cohen, Garg 2018: HSG with seed length

$$\widetilde{O}(\log^2 n + \log(1/\varepsilon))$$

Prior generators and main result

- Nonconstructive: PRG with seed length $O(\log n + \log(1/\varepsilon))$
- Babai, Nisan, Szegedy 1989: PRG with seed length

$$2^{O(\sqrt{\log n})} \cdot \log(1/\varepsilon)$$

- Nisan 1990: PRG with seed length

$$O(\log^2 n + \log(1/\varepsilon) \log n)$$

- Braverman, Cohen, Garg 2018: HSG with seed length

$$\widetilde{O}(\log^2 n + \log(1/\varepsilon))$$

- **This work**: HSG with seed length

$$O(\log^2 n + \log(1/\varepsilon))$$

Comparison with [BCG '18]

- Our construction and analysis are simple

Comparison with [BCG '18]

- Our construction and analysis are simple
- Braverman, Cohen, Garg '18:

Comparison with [BCG '18]

- Our construction and analysis are simple
- Braverman, Cohen, Garg '18:

This work

Hitting Set Generator

Suitable for **RL**

Comparison with [BCG '18]

- Our construction and analysis are simple
- Braverman, Cohen, Garg '18:

Nisan '90		This work
Pseudorandom Generator	$\Longrightarrow$	Hitting Set Generator
Suitable for **BPL**		Suitable for **RL**

Comparison with [BCG '18]

- Our construction and analysis are simple
- Braverman, Cohen, Garg '18:

Nisan '90		BCG '18		This work
Pseudorandom Generator	$\Longrightarrow$	"Pseudorandom Pseudodistribution"	$\Longrightarrow$	Hitting Set Generator
Suitable for **BPL**		Suitable for **BPL**		Suitable for **RL**

Structural lemma for ROBPs

- Let f be a width-n, length-n ROBP

Structural lemma for ROBPs

- Let f be a width-n, length-n ROBP
- Assume $\Pr[\text{accept}] = \varepsilon \ll 1/n^3$

Structural lemma for ROBPs

- Let f be a width-n, length-n ROBP
- Assume $\Pr[\text{accept}] = \varepsilon \ll 1/n^3$
- **Lemma**: There is a vertex u so that

$$\Pr[\text{reach } u] \geq \frac{1}{2n^3} \quad \text{and} \quad \Pr[\text{accept} \mid \text{reach } u] \geq \varepsilon \boldsymbol{n}.$$

Proof of lemma ($\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[\text{acc} \mid u] \geq \varepsilon n$)

- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$

Proof of lemma $(\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[\text{acc} \mid u] \geq \varepsilon n)$

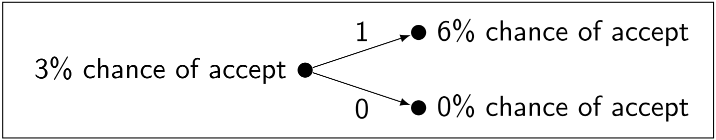
- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$
- Claim: Every accepting path passes through a milestone

Proof of lemma $(\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[\text{acc} \mid u] \geq \varepsilon n)$

- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$
- Claim: Every accepting path passes through a milestone
  - Proof: Probability of acceptance at most doubles in each step

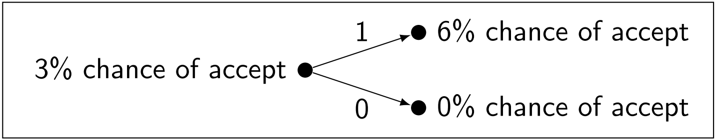
Proof of lemma $(\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[\text{acc} \mid u] \geq \varepsilon n)$

- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$
- Claim: Every accepting path passes through a milestone
  - Proof: Probability of acceptance at most doubles in each step



Proof of lemma $(\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[\text{acc} \mid u] \geq \varepsilon n)$

- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$
- Claim: Every accepting path passes through a milestone
  - Proof: Probability of acceptance at most doubles in each step



- $\varepsilon = \Pr[\text{accept}] \leq \sum_{u \text{ milestone}} \Pr[\text{reach } u \text{ and accept}]$

Proof of lemma $(\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[acc \mid u] \geq \varepsilon n)$

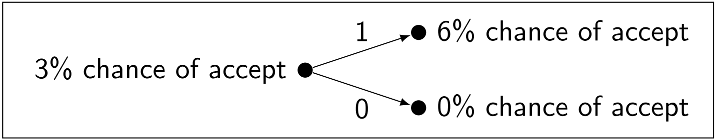
- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$
- Claim: Every accepting path passes through a milestone
  - Proof: Probability of acceptance at most doubles in each step

$$
\begin{tikzcd}
 & \bullet \text{ 6\% chance of accept} \\
\text{3\% chance of accept } \bullet \arrow[ur, "1"] \arrow[dr, "0"'] & \\
 & \bullet \text{ 0\% chance of accept}
\end{tikzcd}
$$

- $$\varepsilon = \Pr[\text{accept}] \leq \sum_{u \text{ milestone}} \Pr[\text{reach } u \text{ and accept}]$$
$$\leq \sum_{u \text{ milestone}} \Pr[\text{reach } u] \cdot 2\varepsilon n$$

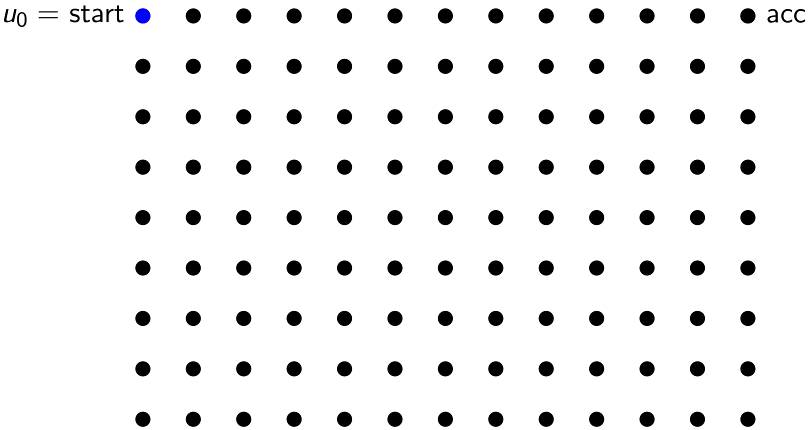
Proof of lemma ($\exists u, \Pr[u] \geq \frac{1}{2n^3} \wedge \Pr[\mathrm{acc} \mid u] \geq \varepsilon n$)

- Say u is a milestone if $\Pr[\text{accept} \mid \text{reach } u] \in [\varepsilon n, 2\varepsilon n]$
- Claim: Every accepting path passes through a milestone
  - Proof: Probability of acceptance at most doubles in each step



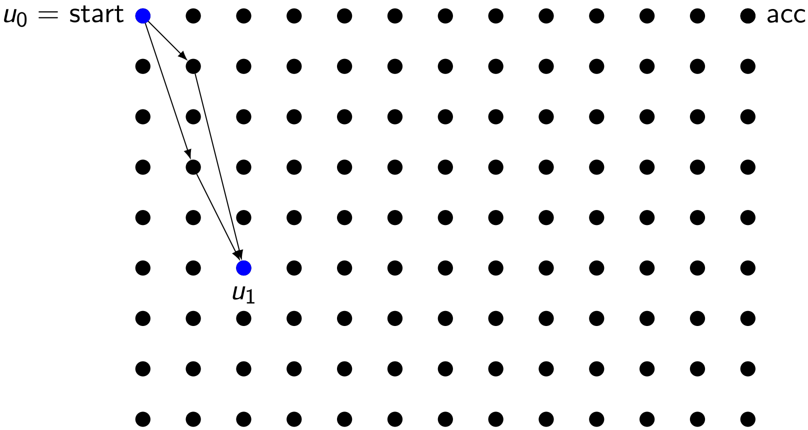
- $\varepsilon = \Pr[\text{accept}] \leq \displaystyle\sum_{u \text{ milestone}} \Pr[\text{reach } u \text{ and accept}]$
  $\leq \displaystyle\sum_{u \text{ milestone}} \Pr[\text{reach } u] \cdot 2\varepsilon n$
- $\#$ milestones $\leq n^2$, so for some milestone u, $\Pr[\text{reach } u] \geq \frac{1}{2n^3}$ $\square$

Iterating the structural lemma



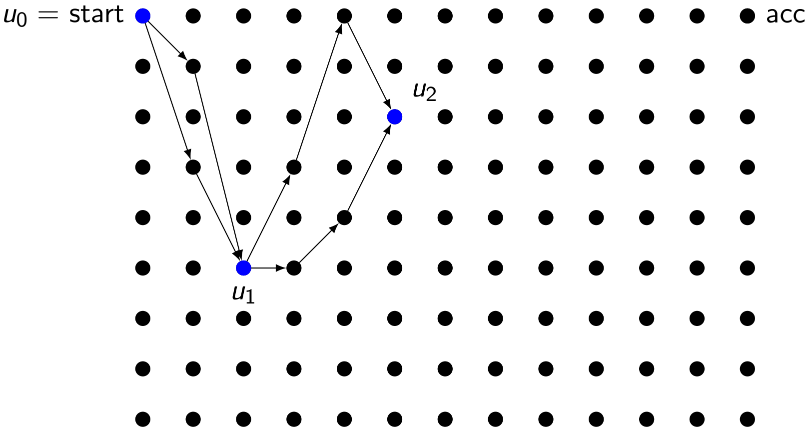
$\Pr[\text{accept}] = \varepsilon$

Iterating the structural lemma



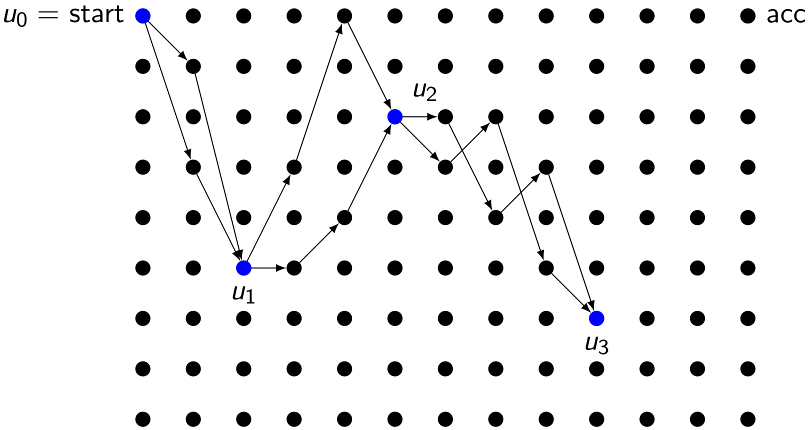
$\Pr[\text{accept}] = \varepsilon$ $\qquad n\varepsilon$

Iterating the structural lemma



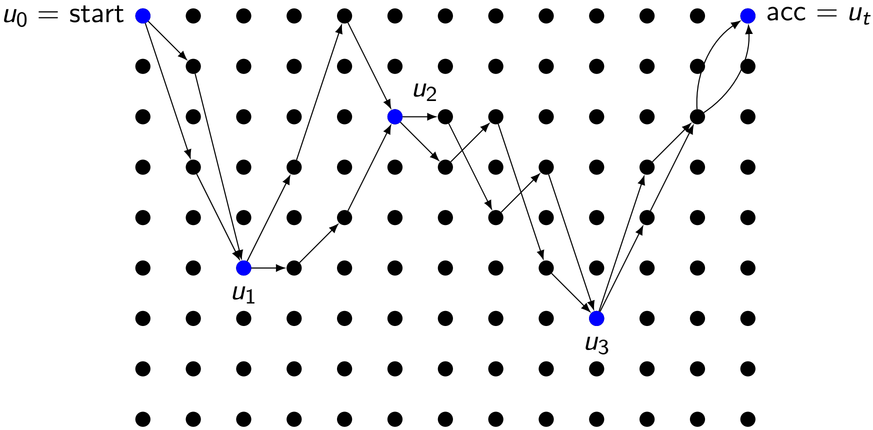
$\Pr[\text{accept}] = \varepsilon$ $\quad n\varepsilon \quad n^2\varepsilon$

Iterating the structural lemma



$\Pr[\text{accept}] = \varepsilon$ $\quad n\varepsilon \quad n^2\varepsilon \quad n^3\varepsilon$

Iterating the structural lemma



$\Pr[\text{accept}] = \varepsilon$ $\quad n\varepsilon \quad n^2\varepsilon \quad n^3\varepsilon \quad n^t\varepsilon = 1$

Idea of our HSG

- Use Nisan's generator for each individual hop $u_i \to u_{i+1}$

Idea of our HSG

- Use Nisan's generator for each individual hop $u_i \to u_{i+1}$
- Use a "hitter" to recycle the seed of Nisan's generator from one hop to the next

Hitters (equivalent to dispersers)

- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density$(E) \geq \theta$

Hitters (equivalent to dispersers)

- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density(E) $\geq \theta$
- **Theorem** (BGG '93): Algorithm that outputs some $z \in E$ with probability $1 - \delta$

Hitters (equivalent to dispersers)

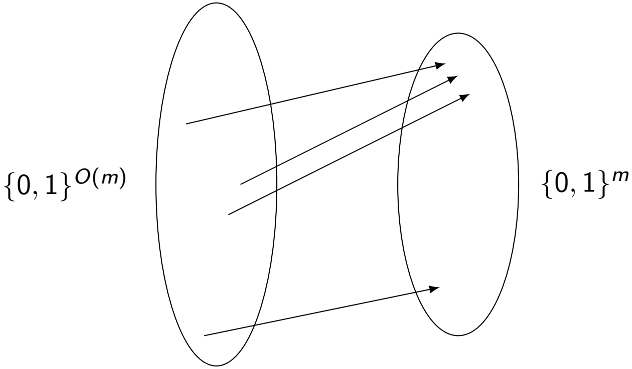
- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density(E) $\geq \theta$
- **Theorem** (BGG '93): Algorithm that outputs some $z \in E$ with probability $1 - \delta$
  - # queries: $O(\theta^{-1} \cdot \log(1/\delta))$

Hitters (equivalent to dispersers)

- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density$(E) \geq \theta$
- **Theorem** (BGG '93): Algorithm that outputs some $z \in E$ with probability $1 - \delta$
  - # queries: $O(\theta^{-1} \cdot \log(1/\delta))$
  - # random bits: $O(m + \log(1/\delta))$

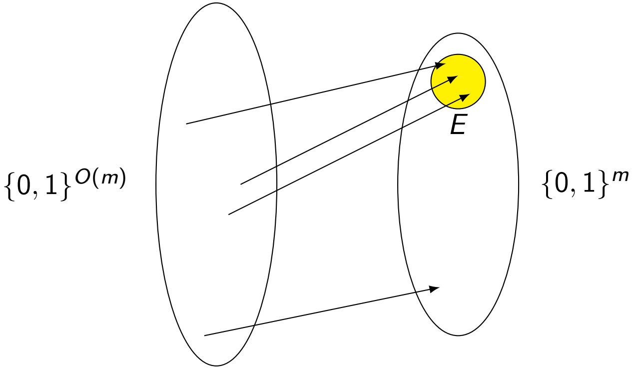
Hitters (equivalent to dispersers)

- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density$(E) \geq \theta$
- **Theorem** (BGG '93): Algorithm that outputs some $z \in E$ with probability $1 - \delta$
  - # queries: $O(\theta^{-1} \cdot \log(1/\delta))$
  - # random bits: $O(m + \log(1/\delta))$



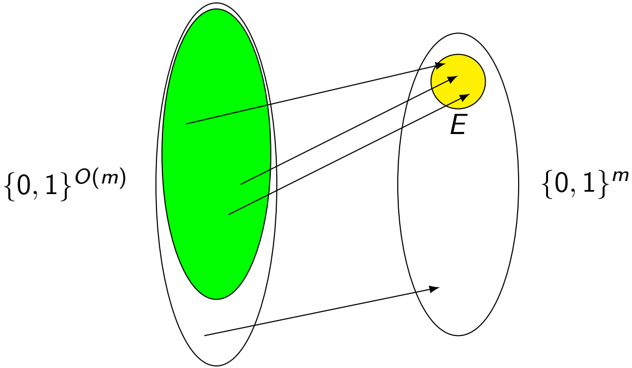
Hitters (equivalent to dispersers)

- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density$(E) \geq \theta$
- **Theorem** (BGG '93): Algorithm that outputs some $z \in E$ with probability $1-\delta$
  - # queries: $O(\theta^{-1} \cdot \log(1/\delta))$
  - # random bits: $O(m + \log(1/\delta))$

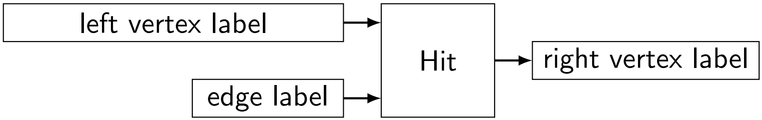


Hitters (equivalent to dispersers)

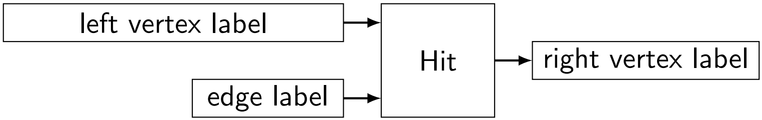
- Assume query access to unknown $E \subseteq \{0,1\}^m$ with density$(E) \geq \theta$
- **Theorem** (BGG '93): Algorithm that outputs some $z \in E$ with probability $1 - \delta$
  - # queries: $O(\theta^{-1} \cdot \log(1/\delta))$
  - # random bits: $O(m + \log(1/\delta))$



Hitter as a function



Hitter as a function



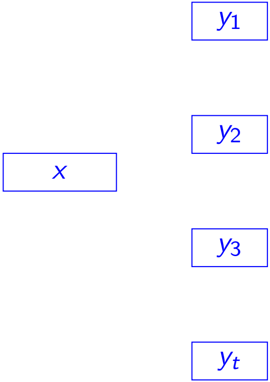
- For any E with density$(E) \geq \theta$,

$$\Pr_x[\exists y, \mathsf{Hit}(x, y) \in E] \geq 1 - \delta$$

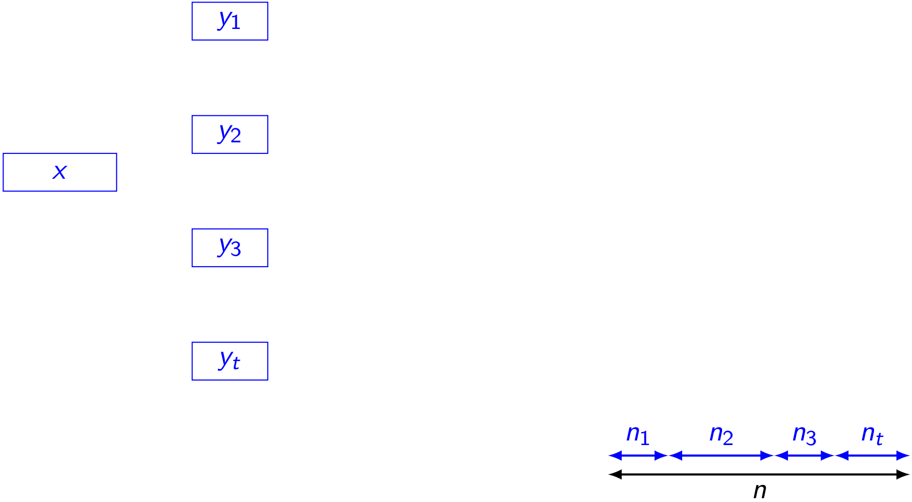
Our HSG

$\boxed{x}$

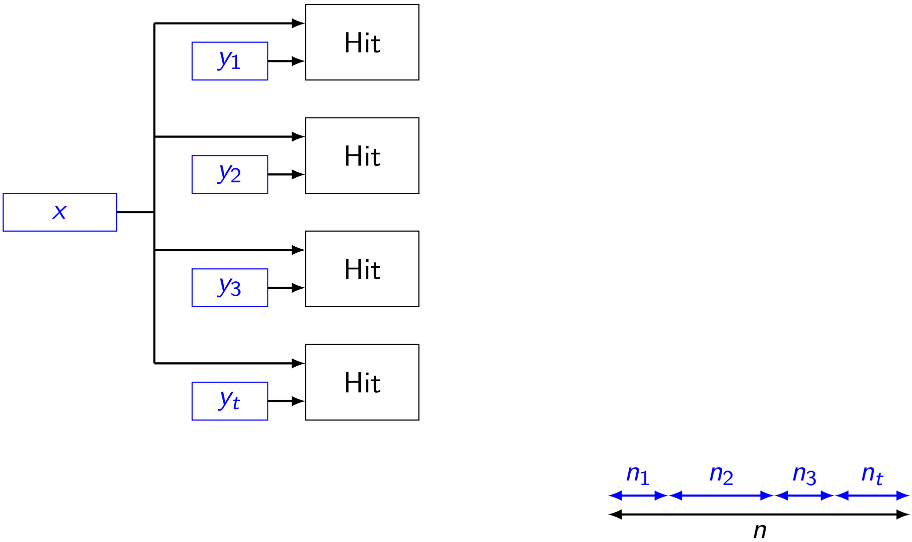
Our HSG



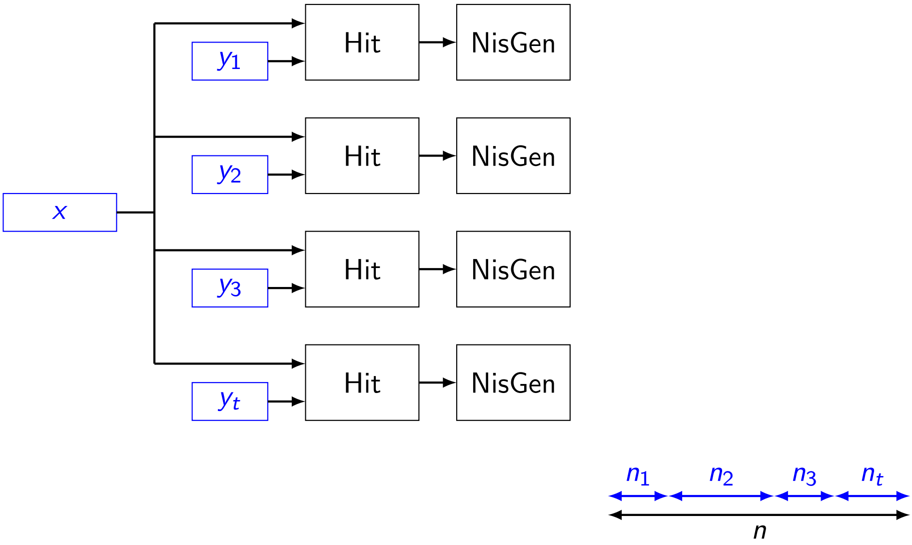
Our HSG



Our HSG

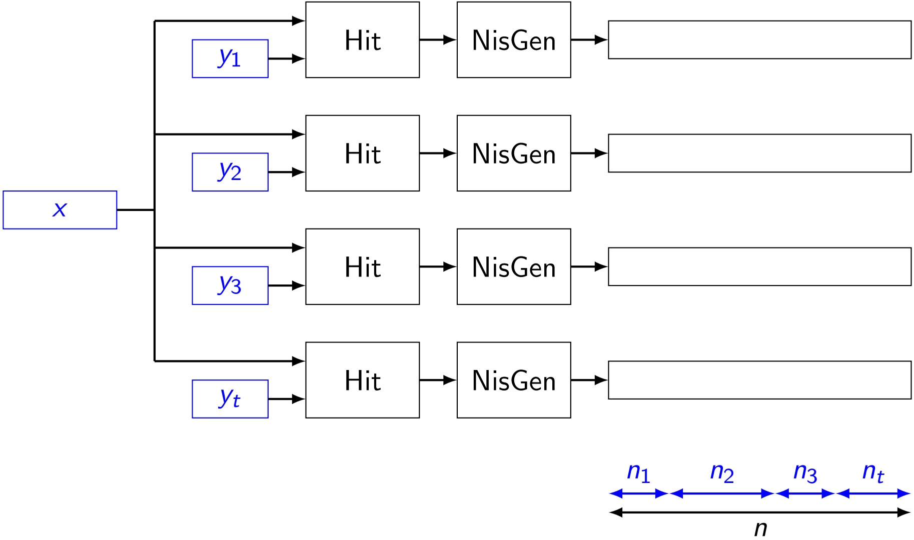


Our HSG



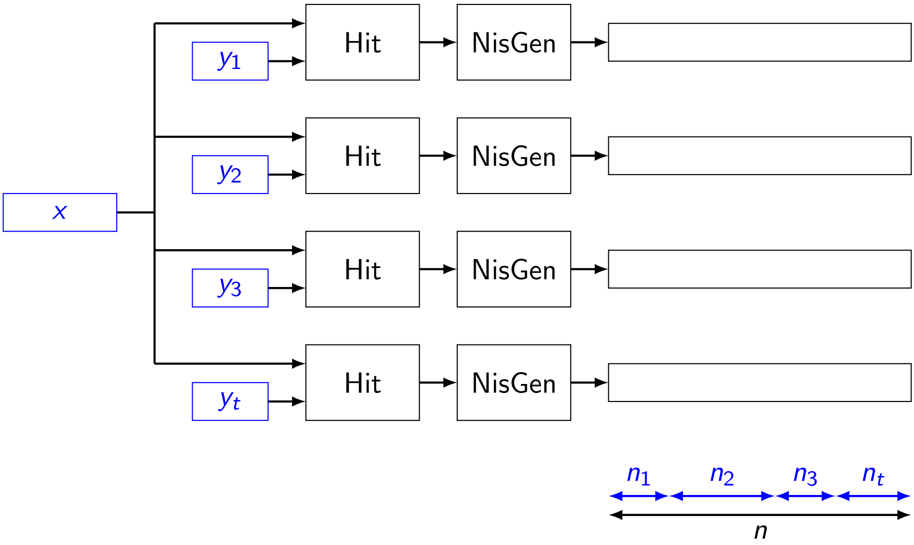
x
y_1
y_2
y_3
y_t
Hit
Hit
Hit
Hit
NisGen
NisGen
NisGen
NisGen
n_1
n_2
n_3
n_t
n

Our HSG

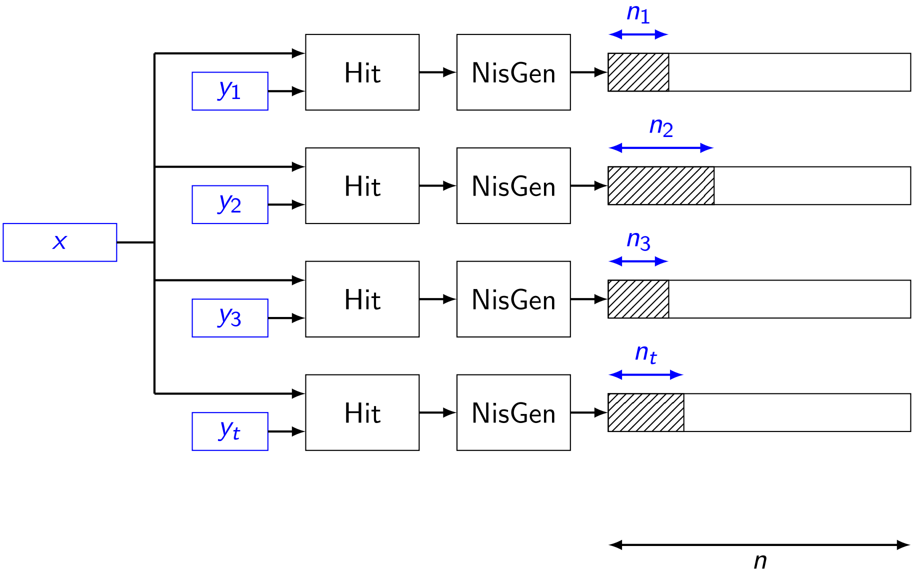


y_1
y_2
y_3
y_t
x
Hit
NisGen
Hit
NisGen
Hit
NisGen
Hit
NisGen
n_1
n_2
n_3
n_t
n

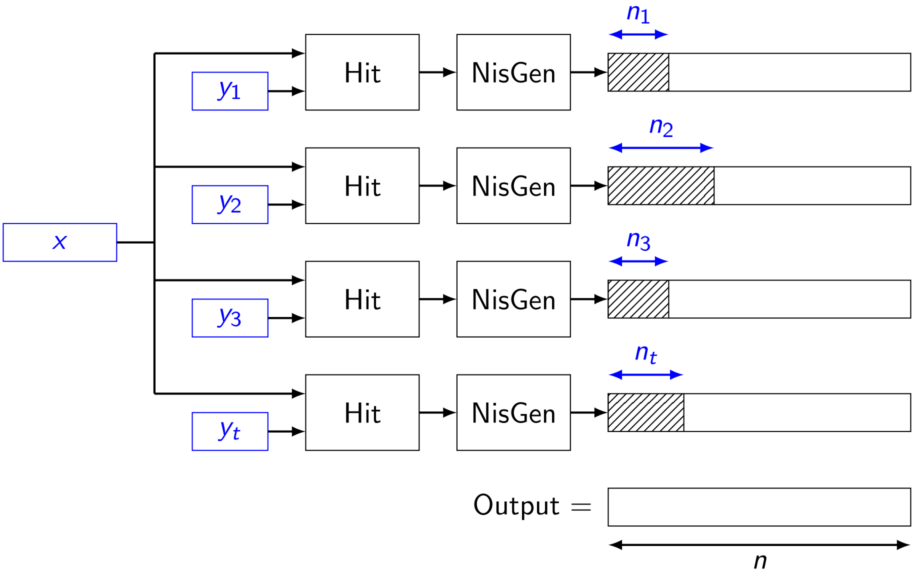
Our HSG


x
y_1
Hit
NisGen
y_2
Hit
NisGen
y_3
Hit
NisGen
y_t
Hit
NisGen
n_1
n_2
n_3
n_t
n

Our HSG


x
y_1
y_2
y_3
y_t
Hit
Hit
Hit
Hit
NisGen
NisGen
NisGen
NisGen
n_1
n_2
n_3
n_t
n

Our HSG



Our HSG in symbols

- For numbers $n_1, \ldots, n_t$ with $n_1 + \cdots + n_t = n$:

$$\boxed{\begin{aligned}&\mathrm{Gen}(x, y_1, \ldots, y_t, n_1, \ldots, n_t) = \\ &\quad \mathrm{NisGen}(\mathrm{Hit}(x, y_1))|_{n_1} \circ \cdots \circ \mathrm{NisGen}(\mathrm{Hit}(x, y_t))|_{n_t} \in \{0,1\}^n\end{aligned}}$$

Our HSG in symbols

- For numbers $n_1, \ldots, n_t$ with $n_1 + \cdots + n_t = n$:

$$\boxed{\begin{aligned}&\mathrm{Gen}(x, y_1, \ldots, y_t, n_1, \ldots, n_t) = \\ &\quad \mathrm{NisGen}(\mathrm{Hit}(x, y_1))|_{n_1} \circ \cdots \circ \mathrm{NisGen}(\mathrm{Hit}(x, y_t))|_{n_t} \in \{0,1\}^n\end{aligned}}$$

- Here $\circ$ = concatenation, $|_r$ = first r bits

Our HSG in symbols

- For numbers $n_1, \ldots, n_t$ with $n_1 + \cdots + n_t = n$:

$$\boxed{\begin{aligned}&\mathrm{Gen}(x, y_1, \ldots, y_t, n_1, \ldots, n_t) = \\ &\quad \mathrm{NisGen}(\mathrm{Hit}(x, y_1))|_{n_1} \circ \cdots \circ \mathrm{NisGen}(\mathrm{Hit}(x, y_t))|_{n_t} \in \{0,1\}^n\end{aligned}}$$

- Here $\circ$ = concatenation, $|_r$ = first r bits
- $|x| = O(\log^2 n)$, $|y_i| = O(\log n)$, $t = \frac{\log(1/\varepsilon)}{\log n}$

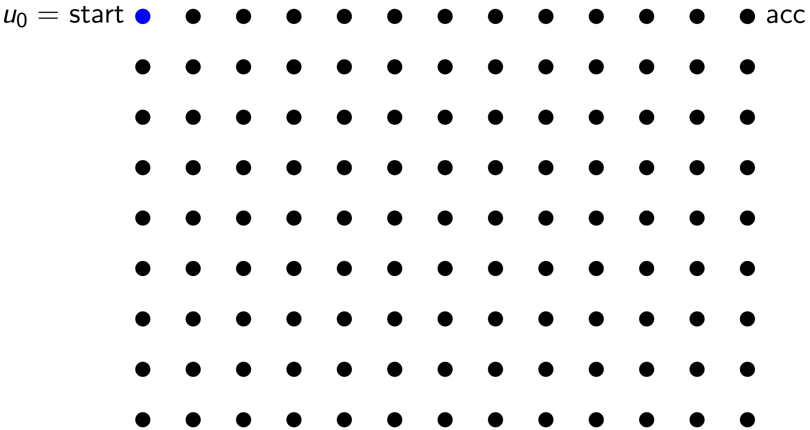
Our HSG in symbols

- For numbers $n_1, \ldots, n_t$ with $n_1 + \cdots + n_t = n$:

$$\boxed{\begin{aligned}&\mathrm{Gen}(x, y_1, \ldots, y_t, n_1, \ldots, n_t) = \\ &\quad \mathrm{NisGen}(\mathrm{Hit}(x, y_1))|_{n_1} \circ \cdots \circ \mathrm{NisGen}(\mathrm{Hit}(x, y_t))|_{n_t} \in \{0,1\}^n\end{aligned}}$$

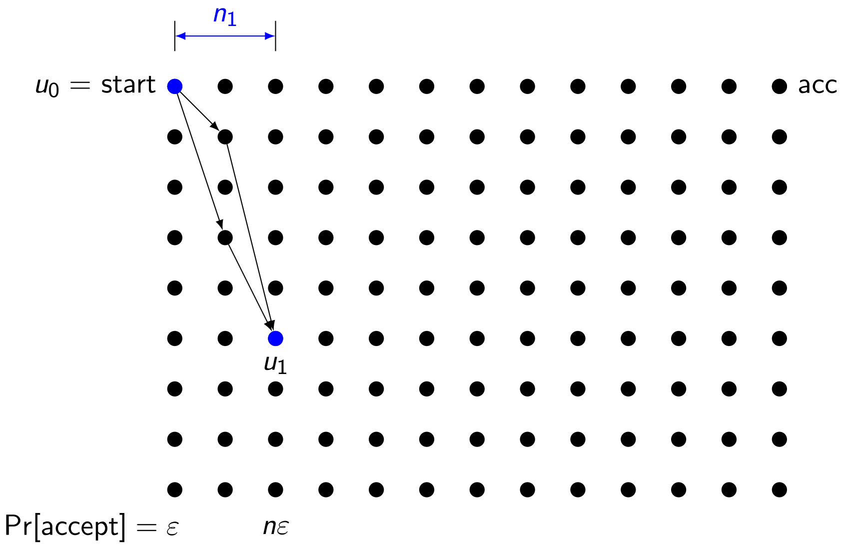
- Here $\circ$ = concatenation, $|_r$ = first r bits
- $|x| = O(\log^2 n)$, $|y_i| = O(\log n)$, $t = \frac{\log(1/\varepsilon)}{\log n}$
- So seed length $= O(\log^2 n + \log(1/\varepsilon))$

Proof of correctness of our HSG

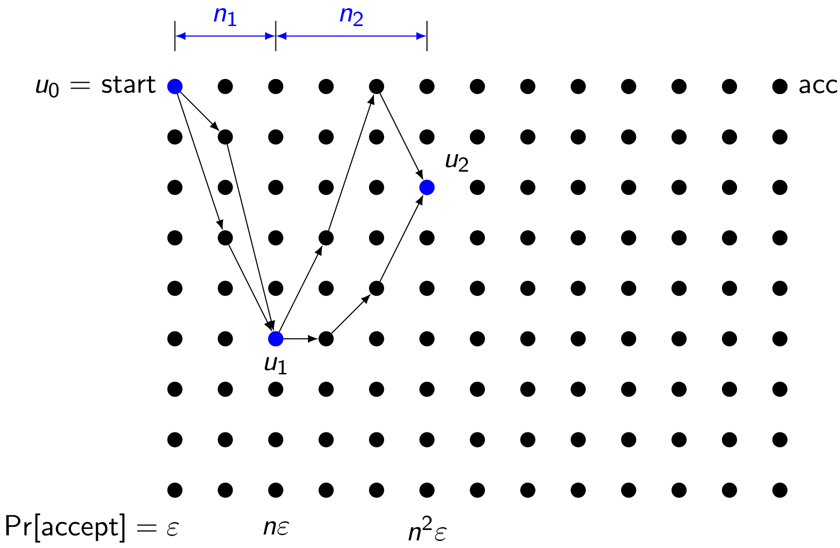


$\Pr[\text{accept}] = \varepsilon$

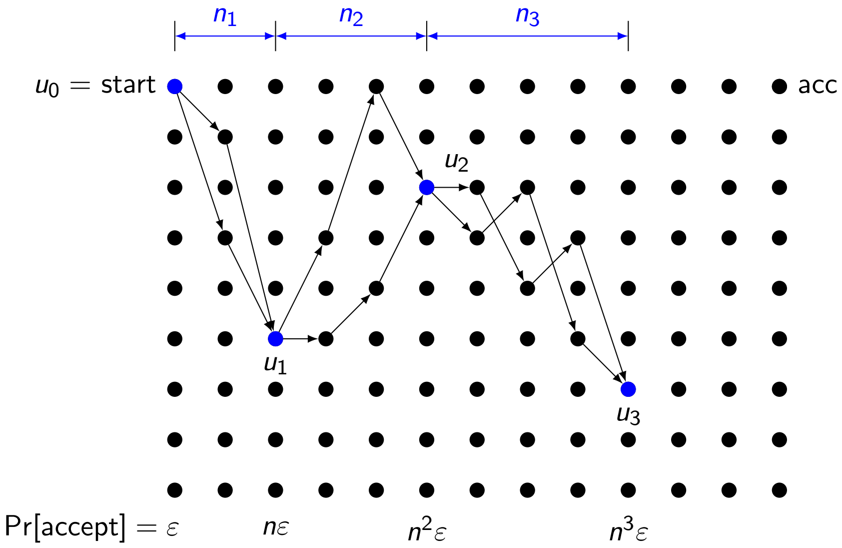
Proof of correctness of our HSG



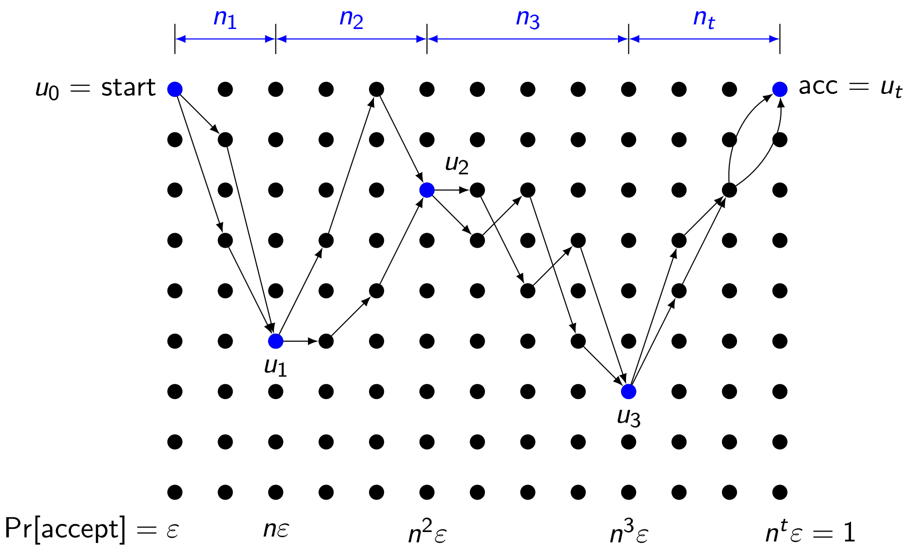
Proof of correctness of our HSG



Proof of correctness of our HSG



Proof of correctness of our HSG



Proof of correctness of our HSG (continued)

- Define $E_i \subseteq \{0,1\}^m$ by

$$E_i = \{z \mid \text{start at } u_{i-1}, \text{ read NisGen}(z) \implies \text{ reach } u_i\}$$

Proof of correctness of our HSG (continued)

- Define $E_i \subseteq \{0,1\}^m$ by

$$E_i = \{z \mid \text{start at } u_{i-1}, \text{ read NisGen}(z) \implies \text{ reach } u_i\}$$

- $\Pr[\text{reach } u_i \mid \text{reach } u_{i-1}] \geq \frac{1}{2n^3} \implies \text{density}(E_i) > \frac{1}{4n^3}$

Proof of correctness of our HSG (continued)

- Define $E_i \subseteq \{0,1\}^m$ by

$$E_i = \{z \mid \text{start at } u_{i-1},\ \text{read NisGen}(z) \implies \text{reach } u_i\}$$

- $\Pr[\text{reach } u_i \mid \text{reach } u_{i-1}] \geq \frac{1}{2n^3} \implies \text{density}(E_i) > \frac{1}{4n^3}$
- Hitter property: $\Pr_x[\exists y, \text{Hit}(x,y) \in E_i] > 1 - \frac{1}{t}$

Proof of correctness of our HSG (continued)

- Define $E_i \subseteq \{0,1\}^m$ by

$$E_i = \{z \mid \text{start at } u_{i-1},\ \text{read NisGen}(z) \implies \text{reach } u_i\}$$

- $\Pr[\text{reach } u_i \mid \text{reach } u_{i-1}] \geq \frac{1}{2n^3} \implies \text{density}(E_i) > \frac{1}{4n^3}$
- Hitter property: $\Pr_x[\exists y, \text{Hit}(x,y) \in E_i] > 1 - \frac{1}{t}$
- Union bound: There is one x so that for all i,

$$\exists y_i, \text{Hit}(x, y_i) \in E_i.$$

Proof of correctness of our HSG (continued)

- Define $E_i \subseteq \{0,1\}^m$ by

$$E_i = \{z \mid \text{start at } u_{i-1},\ \text{read NisGen}(z) \implies \text{reach } u_i\}$$

- $\Pr[\text{reach } u_i \mid \text{reach } u_{i-1}] \geq \frac{1}{2n^3} \implies \text{density}(E_i) > \frac{1}{4n^3}$
- Hitter property: $\Pr_x[\exists y, \text{Hit}(x,y) \in E_i] > 1 - \frac{1}{t}$
- Union bound: There is one x so that for all i,

$$\exists y_i, \text{Hit}(x, y_i) \in E_i.$$

- $f(\text{Gen}(x, y_1, \ldots, y_t, n_1, \ldots, n_t)) = 1$ $\square$

Application: Derandomizing small-success **RL**

- Suppose language L can be decided by a randomized log-space algorithm A that always halts with

$$\begin{aligned} x \in L &\implies \Pr[A(x) \text{ accepts}] \geq \varepsilon = \varepsilon(n) \\ x \notin L &\implies \Pr[A(x) \text{ accepts}] = 0. \end{aligned}$$

Application: Derandomizing small-success **RL**

- Suppose language L can be decided by a randomized log-space algorithm A that always halts with

$$x \in L \implies \Pr[A(x) \text{ accepts}] \geq \varepsilon = \varepsilon(n)$$
$$x \notin L \implies \Pr[A(x) \text{ accepts}] = 0.$$

- $\varepsilon = \frac{1}{2} \implies L \in \mathbf{RL}$.

Application: Derandomizing small-success **RL**

- Suppose language L can be decided by a randomized log-space algorithm A that always halts with

$$x \in L \implies \Pr[A(x) \text{ accepts}] \geq \varepsilon = \varepsilon(n)$$
$$x \notin L \implies \Pr[A(x) \text{ accepts}] = 0.$$

- $\varepsilon = \frac{1}{2} \implies L \in \mathbf{RL}$. Saks, Zhou '95: $\mathbf{RL} \subseteq \mathbf{DSPACE}(\log^{3/2} n)$

Application: Derandomizing small-success **RL**

- Suppose language L can be decided by a randomized log-space algorithm A that always halts with

$$x \in L \implies \Pr[A(x) \text{ accepts}] \geq \varepsilon = \varepsilon(n)$$
$$x \notin L \implies \Pr[A(x) \text{ accepts}] = 0.$$

- $\varepsilon = \frac{1}{2} \implies L \in \mathbf{RL}$. Saks, Zhou '95: $\mathbf{RL} \subseteq \mathbf{DSPACE}(\log^{3/2} n)$
- In general, Saks and Zhou showed

$$L \in \mathbf{DSPACE}(\log^{3/2} n + \sqrt{\log n} \log(1/\varepsilon))$$

Application: Derandomizing small-success **RL**

- Suppose language L can be decided by a randomized log-space algorithm A that always halts with

$$x \in L \implies \Pr[A(x) \text{ accepts}] \geq \varepsilon = \varepsilon(n)$$
$$x \notin L \implies \Pr[A(x) \text{ accepts}] = 0.$$

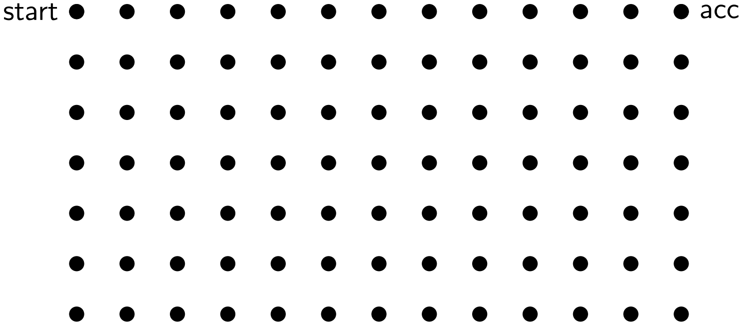
- $\varepsilon = \frac{1}{2} \implies L \in \mathbf{RL}$. Saks, Zhou '95: $\mathbf{RL} \subseteq \mathbf{DSPACE}(\log^{3/2} n)$
- In general, Saks and Zhou showed

$$L \in \mathbf{DSPACE}(\log^{3/2} n + \sqrt{\log n} \log(1/\varepsilon))$$

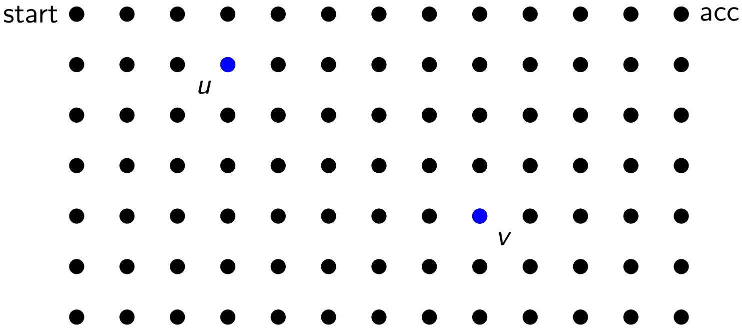
- **Theorem**:

$$L \in \mathbf{DSPACE}(\log^{3/2} n + \log n \log\log(1/\varepsilon))$$

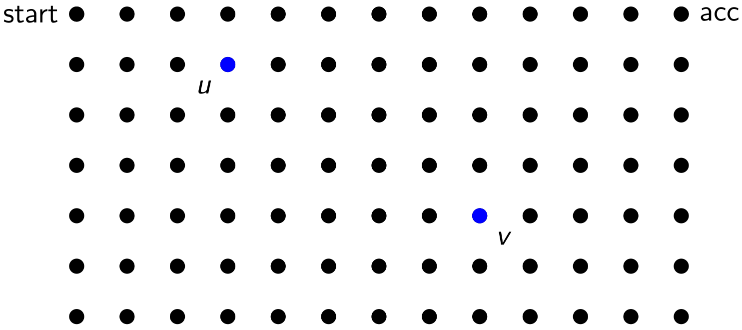
Derandomization algorithm for small-success **RL**



Derandomization algorithm for small-success **RL**



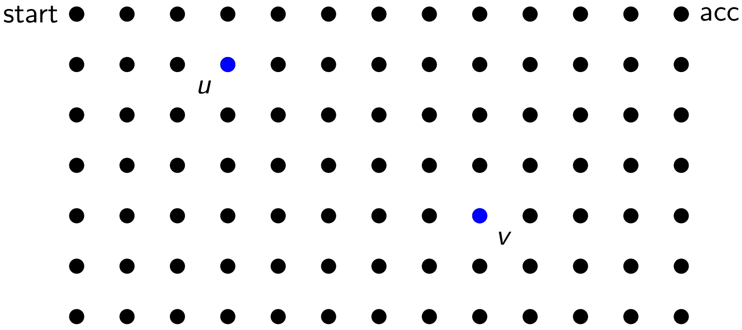
Derandomization algorithm for small-success **RL**



- Saks, Zhou '95: Can distinguish in $O(\log^{3/2} n)$ space between

$$\Pr[\text{reach } v \mid \text{reach } u] = 0 \quad \text{vs.} \quad \Pr[\text{reach } v \mid \text{reach } u] \geq \frac{1}{2n^3}$$

Derandomization algorithm for small-success **RL**

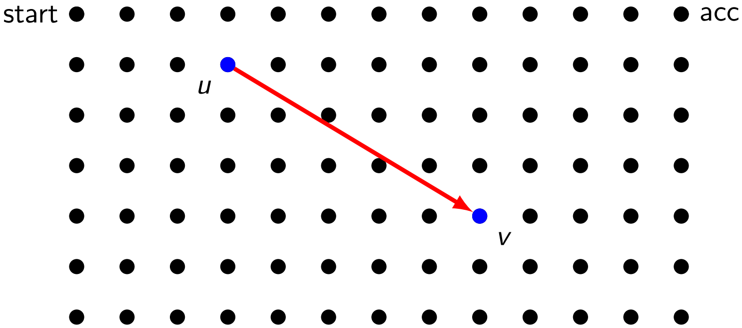


- Saks, Zhou '95: Can distinguish in $O(\log^{3/2} n)$ space between

$$\Pr[\text{reach } v \mid \text{reach } u] = 0 \quad \text{vs.} \quad \Pr[\text{reach } v \mid \text{reach } u] \geq \frac{1}{2n^3}$$

- In second case, add red edge (u, v)

Derandomization algorithm for small-success **RL**

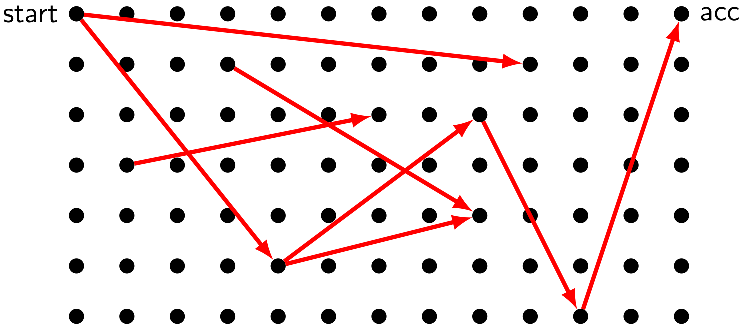


- Saks, Zhou '95: Can distinguish in $O(\log^{3/2} n)$ space between

$$\Pr[\text{reach } v \mid \text{reach } u] = 0 \quad \text{vs.} \quad \Pr[\text{reach } v \mid \text{reach } u] \geq \frac{1}{2n^3}$$

- In second case, add red edge (u, v)

Derandomization algorithm for small-success **RL**



- Saks, Zhou '95: Can distinguish in $O(\log^{3/2} n)$ space between

$$\Pr[\text{reach } v \mid \text{reach } u] = 0 \quad \text{vs.} \quad \Pr[\text{reach } v \mid \text{reach } u] \geq \frac{1}{2n^3}$$

- In second case, add red edge (u, v)

Derandomization algorithm for small-success **RL** (2)

- Use Savitch's algorithm to check for path of length $t = \frac{\log(1/\varepsilon)}{\log n}$ from start to acc using red edges

Derandomization algorithm for small-success **RL** (2)

- Use Savitch's algorithm to check for path of length $t = \frac{\log(1/\varepsilon)}{\log n}$ from start to acc using red edges
- If $x \in L$, such a path exists by structural lemma

Derandomization algorithm for small-success **RL** (2)

- Use Savitch's algorithm to check for path of length $t = \frac{\log(1/\varepsilon)}{\log n}$ from start to acc using red edges
- If $x \in L$, such a path exists by structural lemma
- If $x \notin L$, no path exists $\square$

Restricted case: Derandomizing low-randomness **RL**

- How many random bits can be derandomized in $O(\log n)$ space?

Restricted case: Derandomizing low-randomness **RL**

- How many random bits can be derandomized in $O(\log n)$ space?
- $(\log n)$-space algorithm that uses r random bits $\implies$ ROBP with width n and length r

Restricted case: Derandomizing low-randomness **RL**

- How many random bits can be derandomized in $O(\log n)$ space?
- $(\log n)$-space algorithm that uses r random bits $\implies$ ROBP with width n and length r
- Ajtai, Komlós, Szemerédi '87: HSG with seed length $O(\log n)$ for

$$r \leq O\left(\frac{\log^2 n}{\log \log n}\right), \qquad \varepsilon = \frac{1}{\mathrm{poly}(n)}$$

Restricted case: Derandomizing low-randomness **RL**

- How many random bits can be derandomized in $O(\log n)$ space?
- $(\log n)$-space algorithm that uses r random bits $\implies$ ROBP with width n and length r
- Ajtai, Komlós, Szemerédi '87: HSG with seed length $O(\log n)$ for

$$r \leq O\left(\frac{\log^2 n}{\log\log n}\right), \qquad \varepsilon = \frac{1}{\mathrm{poly}(n)}$$

- Nisan, Zuckerman '93: PRG with seed length $O(\log n)$ for

$$r \leq \mathrm{polylog}\, n, \qquad \varepsilon = \frac{1}{2^{\log^{0.99} n}}$$

Restricted case: Derandomizing low-randomness **RL**

- How many random bits can be derandomized in $O(\log n)$ space?
- $(\log n)$-space algorithm that uses r random bits $\implies$ ROBP with width n and length r
- Ajtai, Komlós, Szemerédi '87: HSG with seed length $O(\log n)$ for

$$r \leq O\left(\frac{\log^2 n}{\log\log n}\right), \qquad \varepsilon = \frac{1}{\mathrm{poly}(n)}$$

- Nisan, Zuckerman '93: PRG with seed length $O(\log n)$ for

$$r \leq \mathrm{polylog}\, n, \qquad \varepsilon = \frac{1}{2^{\log^{0.99} n}}$$

- **Theorem**: HSG with seed length $O(\log(n/\varepsilon))$ for $r \leq \mathrm{polylog}\, n$

Optimal HSG for $r \leq \operatorname{polylog} n$

- The generator: Same as main construction but with the Nisan-Zuckerman PRG in place of Nisan's PRG

Optimal HSG for $r \leq \operatorname{polylog} n$

- The generator: Same as main construction but with the Nisan-Zuckerman PRG in place of Nisan's PRG
- Analysis difficulty: Vertex u from structural lemma merely satisfies

$$\Pr[\text{reach } u] \geq \frac{1}{2nr^2}.$$

Optimal HSG for $r \leq \text{polylog}\, n$

- The generator: Same as main construction but with the Nisan-Zuckerman PRG in place of Nisan's PRG
- Analysis difficulty: Vertex u from structural lemma merely satisfies

$$\Pr[\text{reach } u] \geq \frac{1}{2nr^2}.$$

- Nizan-Zuckerman PRG has too much error

Optimal HSG for $r \leq \text{polylog}\, n$

- The generator: Same as main construction but with the Nisan-Zuckerman PRG in place of Nisan's PRG
- Analysis difficulty: Vertex u from structural lemma merely satisfies

$$\Pr[\text{reach } u] \geq \frac{1}{2nr^2}.$$

- Nizan-Zuckerman PRG has too much error
- Solution: Better structural lemma!

Better structural lemma

- Let f be a length-r ROBP of any width

Better structural lemma

- Let f be a length-r ROBP of any width
- Assume $\Pr[\text{accept}] = \varepsilon \ll 1/r^2$

Better structural lemma

- Let f be a length-r ROBP of any width
- Assume $\Pr[\text{accept}] = \varepsilon \ll 1/r^2$
- **Lemma**: There is a subset U of some layer so that

$$\Pr[\text{reach } U] \geq \frac{1}{2r^2} \quad \text{and} \quad \forall u \in U, \ \Pr[\text{accept} \mid \text{reach } u] \geq \varepsilon r.$$

Better structural lemma

- Let f be a length-r ROBP of any width
- Assume $\Pr[\text{accept}] = \varepsilon \ll 1/r^2$
- **Lemma**: There is a subset U of some layer so that

$$\Pr[\text{reach } U] \geq \frac{1}{2r^2} \quad \text{and} \quad \forall u \in U, \; \Pr[\text{accept} \mid \text{reach } u] \geq \varepsilon r.$$

- **Proof**: Similar to the proof of the original structural lemma

Better structural lemma

- Let f be a length-r ROBP of any width
- Assume $\Pr[\text{accept}] = \varepsilon \ll 1/r^2$
- **Lemma**: There is a subset U of some layer so that

$$\Pr[\text{reach } U] \geq \frac{1}{2r^2} \quad \text{and} \quad \forall u \in U, \ \Pr[\text{accept} \mid \text{reach } u] \geq \varepsilon r.$$

- **Proof**: Similar to the proof of the original structural lemma
- (Error of NZ generator) $\ll \frac{1}{2r^2} = \frac{1}{\text{polylog } n}$ $\square$

Application: Randomness vs. nondeterminism

- $\mathbf{RL} \subseteq \mathbf{NL}$

Application: Randomness vs. nondeterminism

- $\mathbf{RL} \subseteq \mathbf{NL}$
- **Theorem**: For any $r = r(n)$

$$(\mathbf{RL} \text{ with } r \text{ coins}) \subseteq$$

Application: Randomness vs. nondeterminism

- $\mathbf{RL} \subseteq \mathbf{NL}$
- **Theorem**: For any $r = r(n)$ and any constant c,

$$(\mathbf{RL}\text{ with } r \text{ coins}) \subseteq \left(\mathbf{NL}\text{ with } \frac{r}{\log^c n} \text{ nondeterministic bits}\right)$$

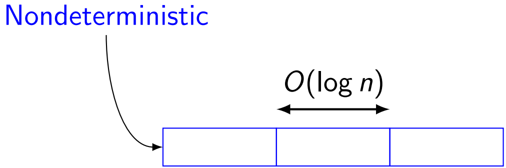
Application: Randomness vs. nondeterminism

- **RL** $\subseteq$ **NL**
- **Theorem**: For any $r = r(n)$ and any constant c,

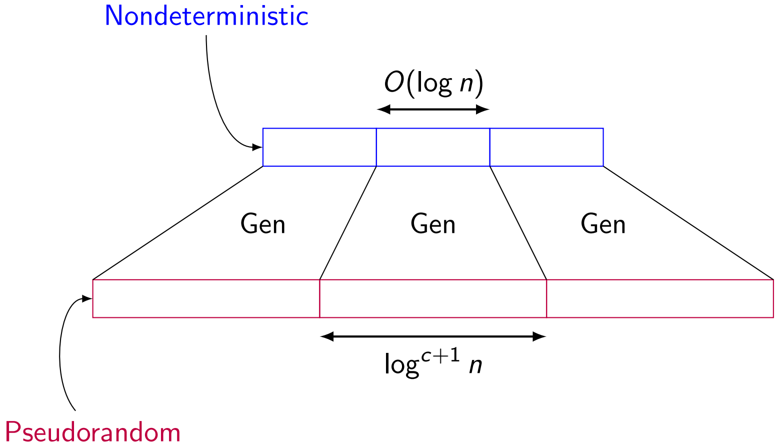
$$(\textbf{RL} \text{ with } r \text{ coins}) \subseteq \left(\textbf{NL} \text{ with } \frac{r}{\log^c n} \text{ nondeterministic bits}\right)$$

$\geq$

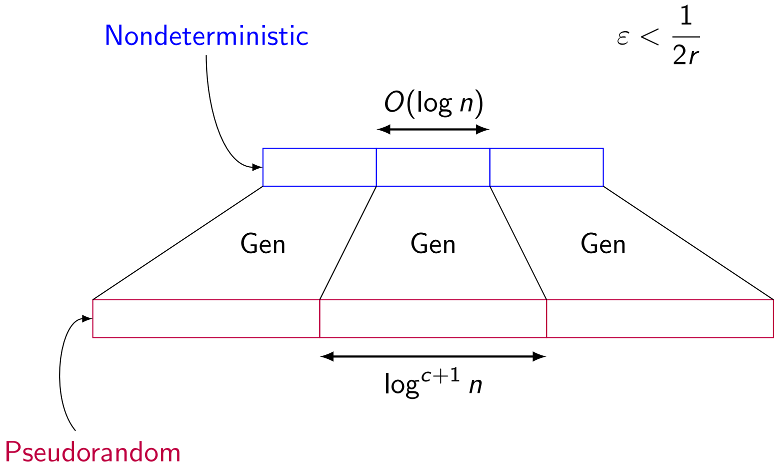
Simulating r coins with $r/\log^c n$ nondeterministic bits



Simulating r coins with $r/\log^c n$ nondeterministic bits



Simulating r coins with $r/\log^c n$ nondeterministic bits



Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$

Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$
- Let L be the layer reached after $\log^{c+1} n$ steps

Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$
- Let L be the layer reached after $\log^{c+1} n$ steps
- Define $U = \{u \in L : \Pr[\text{accept} \mid \text{reach } u] \geq \alpha - \varepsilon\}$

Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$
- Let L be the layer reached after $\log^{c+1} n$ steps
- Define $U = \{u \in L : \Pr[\text{accept} \mid \text{reach } u] \geq \alpha - \varepsilon\}$
- Then $\alpha = \Pr[\text{accept}]$

$$= \sum_{u \in U} \Pr[u] \cdot \Pr[\text{acc} \mid u] + \sum_{u \in L \setminus U} \Pr[u] \cdot \Pr[\text{acc} \mid u]$$

Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$
- Let L be the layer reached after $\log^{c+1} n$ steps
- Define $U = \{u \in L : \Pr[\text{accept} \mid \text{reach } u] \geq \alpha - \varepsilon\}$
- Then $\alpha = \Pr[\text{accept}]$

$$= \sum_{u \in U} \Pr[u] \cdot \Pr[\text{acc} \mid u] + \sum_{u \in L \setminus U} \Pr[u] \cdot \Pr[\text{acc} \mid u]$$

$$\leq \Pr[U] + (\alpha - \varepsilon)$$

Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$
- Let L be the layer reached after $\log^{c+1} n$ steps
- Define $U = \{u \in L : \Pr[\text{accept} \mid \text{reach } u] \geq \alpha - \varepsilon\}$
- Then $\alpha = \Pr[\text{accept}]$

$$
\begin{aligned}
&= \sum_{u \in U} \Pr[u] \cdot \Pr[\text{acc} \mid u] + \sum_{u \in L \setminus U} \Pr[u] \cdot \Pr[\text{acc} \mid u] \\
&\leq \Pr[U] + (\alpha - \varepsilon)
\end{aligned}
$$

$$\Pr[U] \geq \varepsilon.$$

Simulating r coins with $r/\log^c n$ nondeterministic bits (2)

- Proof that this works: Suppose $\Pr[\text{accept}] = \alpha$
- Let L be the layer reached after $\log^{c+1} n$ steps
- Define $U = \{u \in L : \Pr[\text{accept} \mid \text{reach } u] \geq \alpha - \varepsilon\}$
- Then $\alpha = \Pr[\text{accept}]$

$$= \sum_{u \in U} \Pr[u] \cdot \Pr[\text{acc} \mid u] + \sum_{u \in L \setminus U} \Pr[u] \cdot \Pr[\text{acc} \mid u]$$

$$\leq \Pr[U] + (\alpha - \varepsilon)$$

$$\Pr[U] \geq \varepsilon.$$

- So some seed x leads to U. Induct $\square$

General theorem: Reduction to $1/\operatorname{poly}$ error case

- Assume efficient PRG for ROBPs with seed length m and error $\frac{1}{r^2}$

General theorem: Reduction to $1/\operatorname{poly}$ error case

- Assume efficient PRG for ROBPs with seed length m and error $\frac{1}{r^2}$
- **Theorem**: For every $\varepsilon > 0$, there's an efficient HSG for ROBPs with seed length

$$O(m + \log(nr/\varepsilon))$$

The case polylog $n \ll r \ll n$

- **Theorem**: HSG for width-n, length-r ROBPs with seed length

$$O\left(\frac{\log(nr)\log r}{\max\{1, \log\log n - \log\log r\}} + \log(1/\varepsilon)\right)$$

- **Proof**: Plug in PRG of [Armoni '98]

Open questions

- **Conjecture**: For any $r = r(n)$, for any constant c,

$$(\textbf{BPL} \text{ with } r \text{ coins}) = \left(\textbf{BPL} \text{ with } \frac{r}{\log^c n} \text{ coins}\right)$$

Open questions

- **Conjecture**: For any $r = r(n)$, for any constant c,

$$(\textbf{BPL} \text{ with } r \text{ coins}) = \left(\textbf{BPL} \text{ with } \frac{r}{\log^c n} \text{ coins}\right)$$

  - True for $r \leq 2^{\log^{0.99} n}$ by Nisan-Zuckerman

Open questions

- **Conjecture**: For any $r = r(n)$, for any constant c,

$$(\textbf{BPL with } r \text{ coins}) = \left(\textbf{BPL with } \frac{r}{\log^c n} \text{ coins}\right)$$

  - True for $r \le 2^{\log^{0.99} n}$ by Nisan-Zuckerman

- ACR '96: Explicit HSG for circuits $\implies$ $\mathbf{P} = \mathbf{BPP}$. Similar theorem for **BPL**?

Open questions

- **Conjecture**: For any $r = r(n)$, for any constant c,

$$(\textbf{BPL} \text{ with } r \text{ coins}) = \left(\textbf{BPL} \text{ with } \frac{r}{\log^c n} \text{ coins}\right)$$

  - True for $r \leq 2^{\log^{0.99} n}$ by Nisan-Zuckerman

- ACR '96: Explicit HSG for circuits $\implies$ $\mathbf{P} = \mathbf{BPP}$. Similar theorem for **BPL**?

- Thanks! Questions?