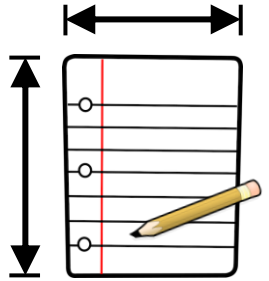
Tutorial: Derandomizing Space-Bounded Computation

Part 1

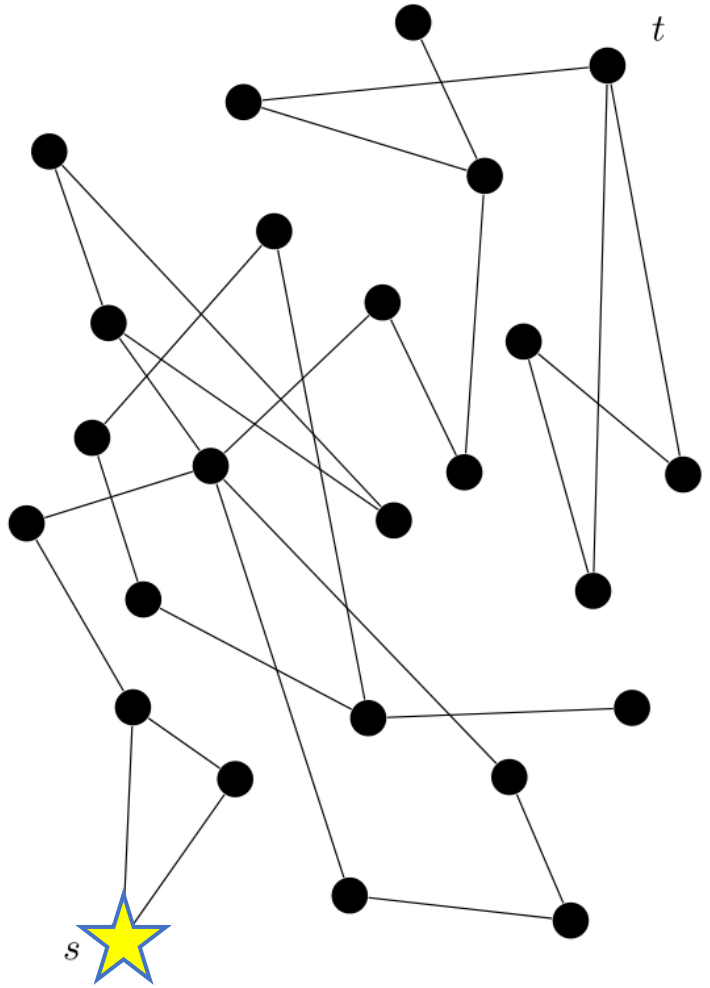
September 1, 2026





William Hoza
The University of Chicago

Motivating example



- **Given:** Undirected graph G and vertices s, t

Theorem [Aleliunas, Karp, Lipton, Lovász, Rackoff 1979]:

∃ randomized algorithm that determines whether there is a path from s to t using $O(\log n)$ bits of space

- **Algorithm**: Take a random walk from s of length $\text{poly}(n)$ and check whether we visit t. (This works!)



No known efficient algorithm



Efficient randomized algorithm



Efficient deterministic algorithm



Derandomization

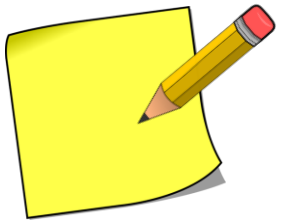


- Is there a deterministic log-space algorithm for undirected s-t connectivity?
  - Yes! [Reingold 2008]
- Can every log-space algorithm be derandomized?
  - "L vs. BPL"

The complexity class BPL



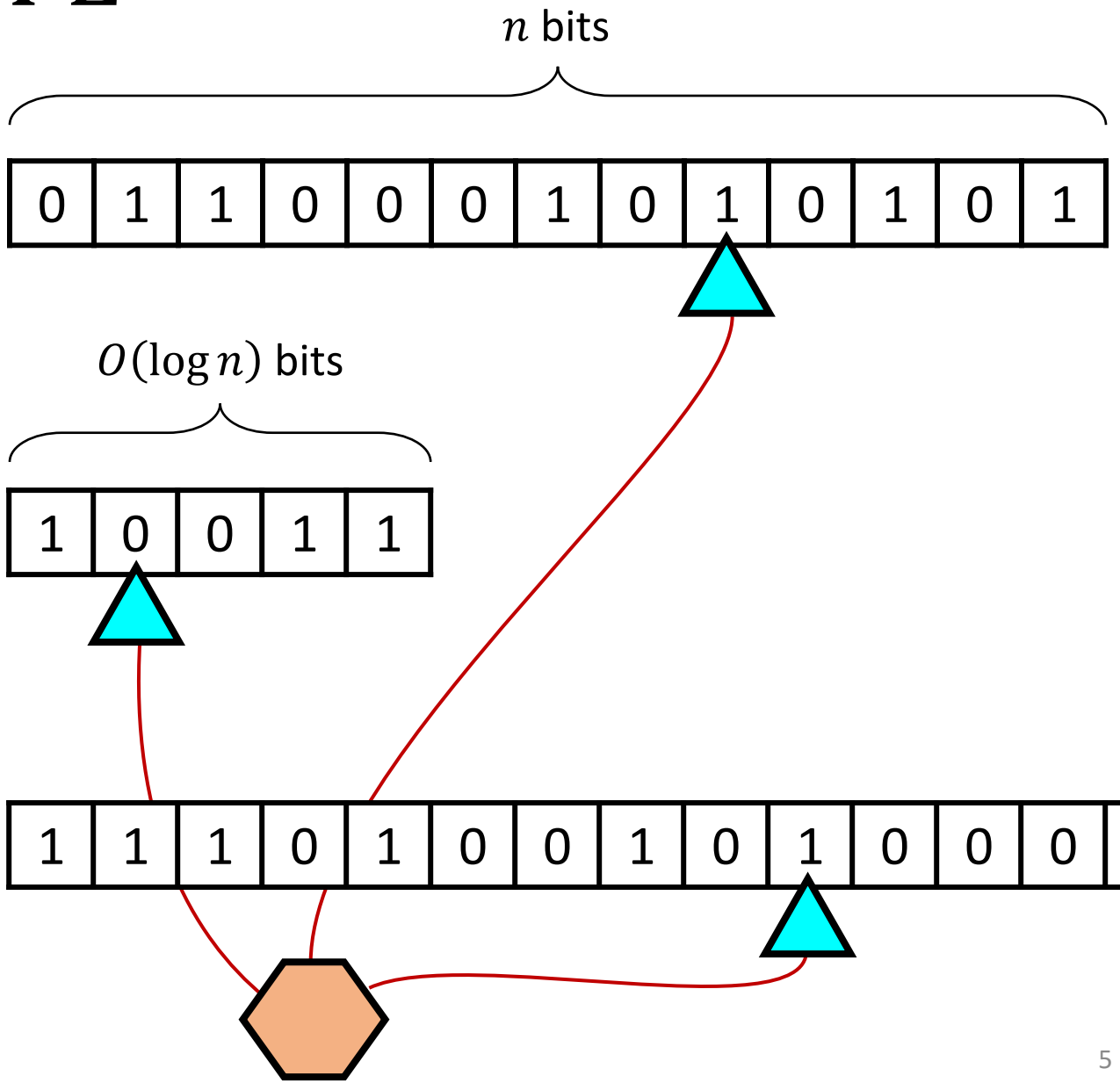
Read-only input tape



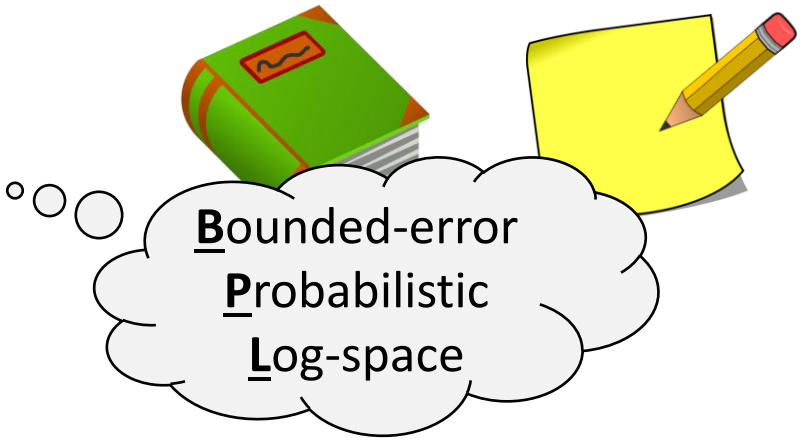
Read-write work tape



Read-once random tape



The complexity class BPL



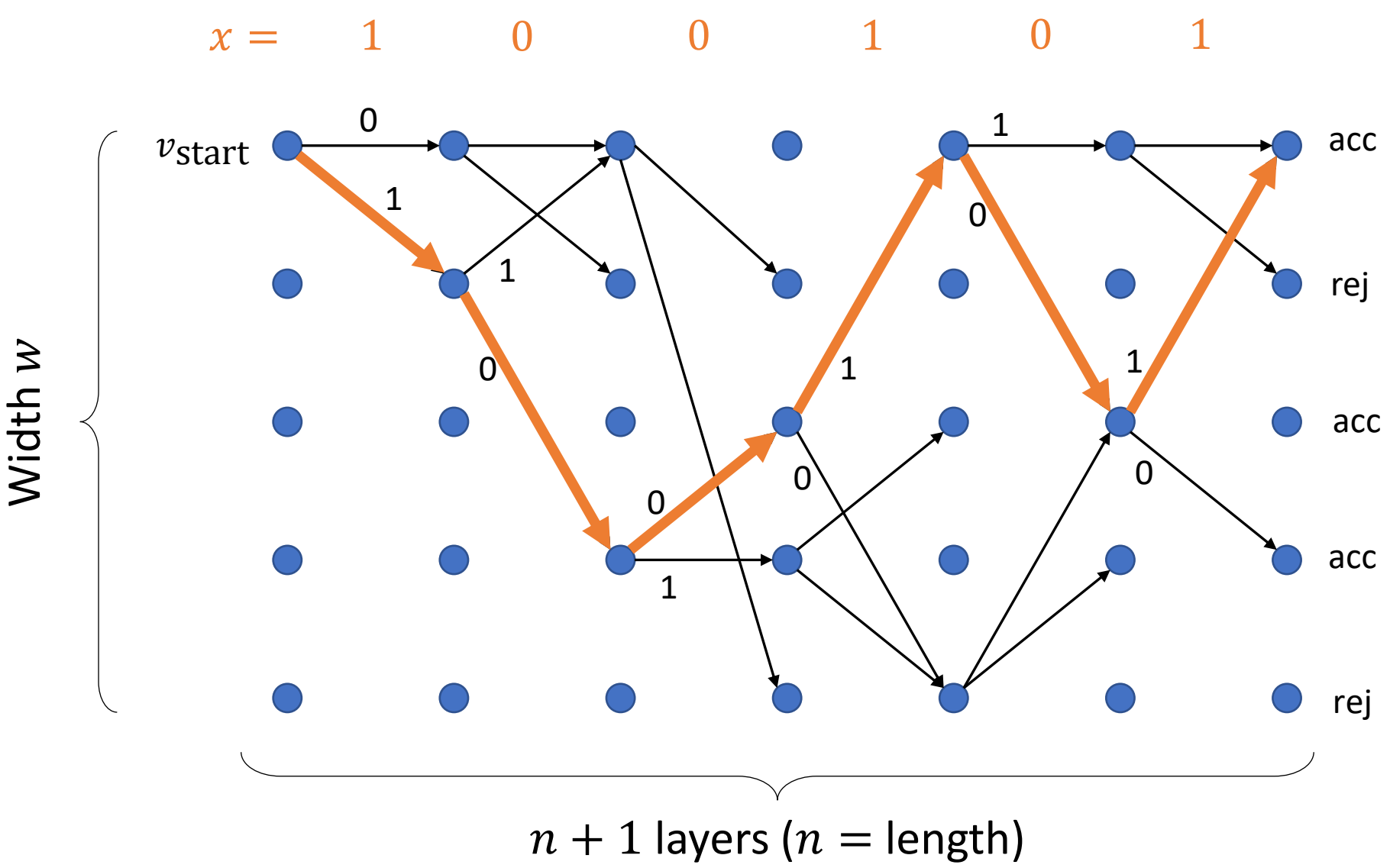


- Let $f: \{0,1\}^* \to \{0,1\}$
- **Definition:** $f \in \mathrm{BPL}$ if there exists a Turing machine M such that:
  - There is a read-only tape storing the n-bit input
  - There is a read/write work tape of size $O(\log n)$
  - There is a read-once tape filled with random bits
  - For every $x \in \{0,1\}^*$, we have $\Pr[M(x) = f(x)] \geq 2/3$
  - M halts for every input and every setting of the random tape

Can replace $2/3$ with 0.51 or 0.99

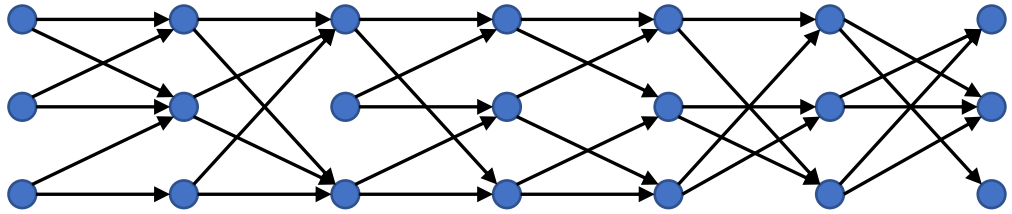
Exercise: Show that M halts in polynomial time

Read-once branching programs (ROBPs)



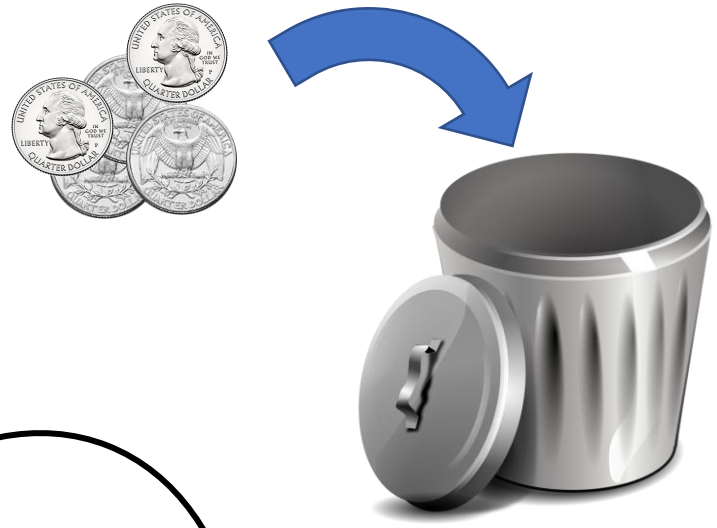
Computes
$f: \{0,1\}^n \to \{0,1\}$

ROBPs and BPL



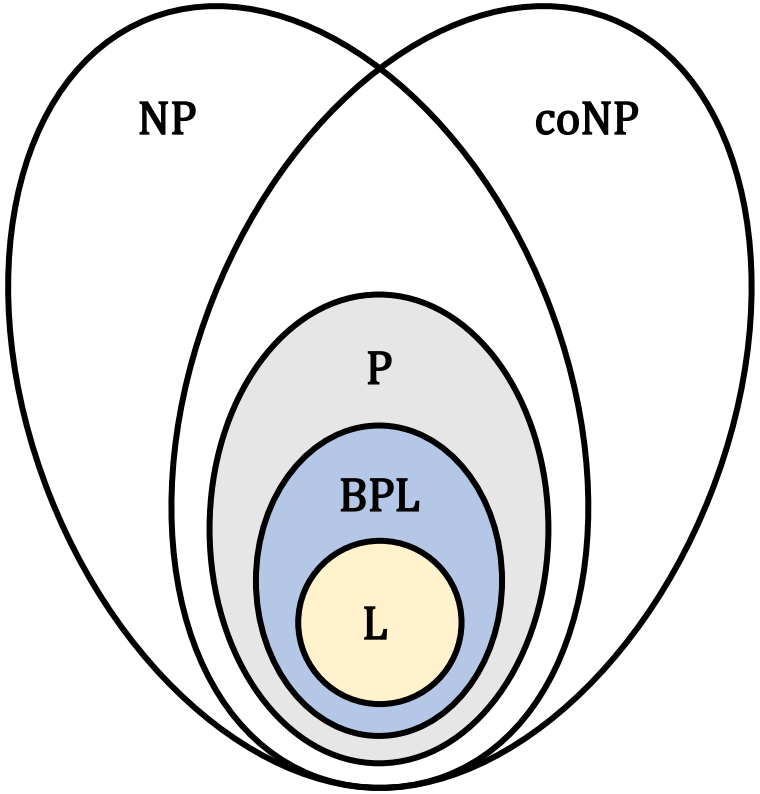
- **Problem:** Given an ROBP f, estimate $\mathbb{E}[f]$ to within ± 0.1
  - **Claim:** This problem is "BPL-complete." Proof sketch:
  - **Problem is "in BPL" because:** Can estimate $\mathbb{E}[f]$ by computing $f(x)$ for a few random $x \in \{0,1\}^n$ and outputting the average
  - **Problem is "BPL-hard" because:** Can construct an ROBP where each vertex represents a TM configuration and each edge represents a coin toss

L vs. BPL



- $\mathrm{L} \subseteq \mathrm{BPL}$

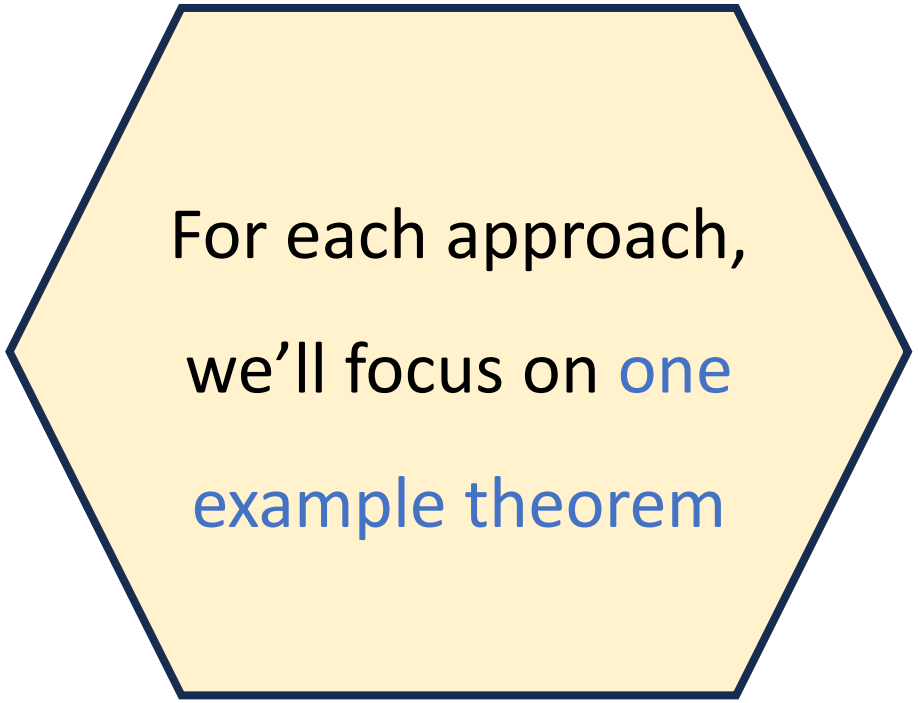
Conjecture: $\mathrm{L} = \mathrm{BPL}$



- "Randomness is not necessary for space-efficient computation"

Outline

- Today:
  - The INW Approach
  - The Nisan Approach
- Tomorrow:
  - The Laplacian Approach
  - The Restrictions Approach



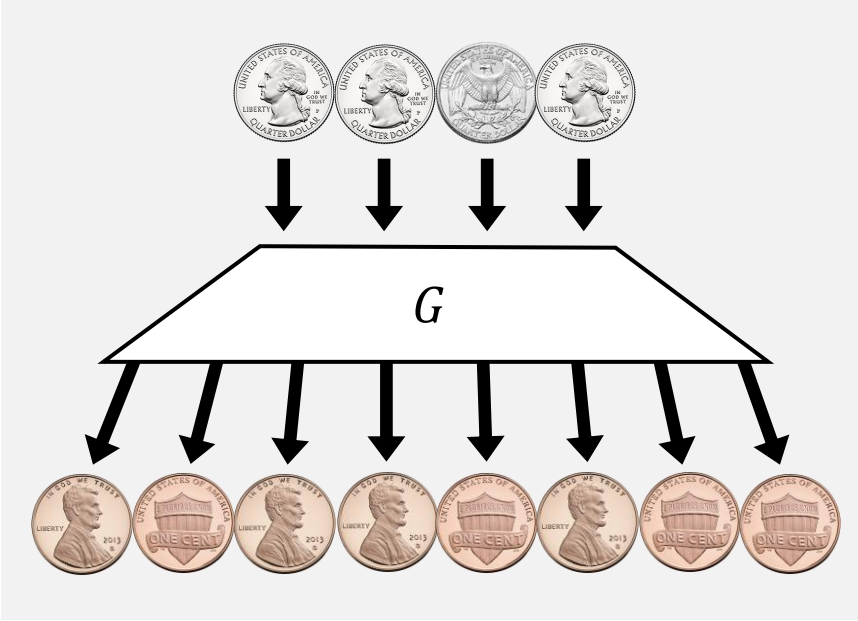
The INW Approach

The INW Pseudorandom Generator

Corollary:
$\mathrm{BPL} \subseteq \mathrm{L}^2$

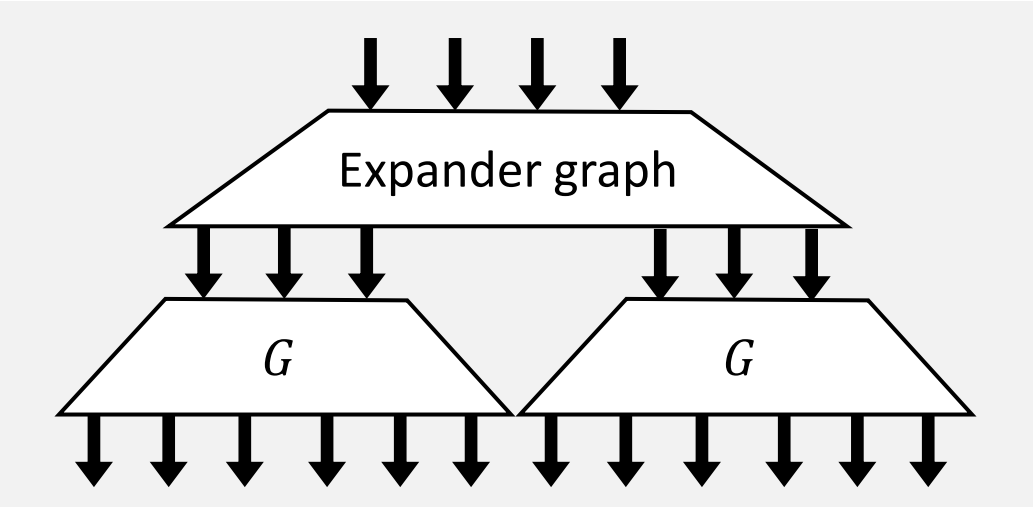
Theorem: There exists an explicit PRG that fools width-n length-n ROBPs with error 0.1 and seed length $O(\log^2 n)$

- We'll present a proof by **I**mpagliazzo, **N**isan, and **W**igderson (STOC 1994)
- Key ingredient: Expander graphs



INW construction

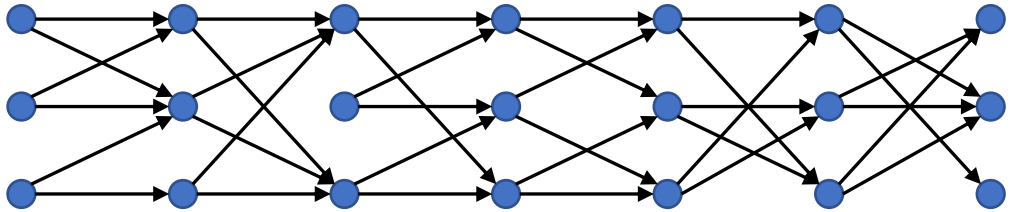
- By induction, assume we already have a PRG $G: \{0,1\}^s \to \{0,1\}^{n/2}$
- Let H be a degree-D expander graph on the vertex set $\{0,1\}^s$
- **New generator:** $G'(x,y) = \big(G(x), G(H[x,y])\big)$
  - $H[x,y]$ denotes y-th neighbor of x
- **Analysis:** *[On whiteboard]*



Open problem: Better PRGs

Explicit PRG with seed length $O(\log n)$ would imply $\mathrm{L} = \mathrm{BPL}$

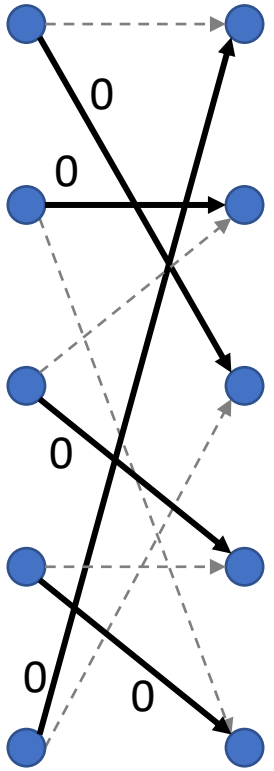
- **Fact:** There exists a non-explicit PRG that fools width-n length-n ROBPs with seed length $O(\log n)$
- **Open problem:** Design an explicit PRG that fools constant-width length-n ROBPs with seed length $o(\log^2 n)$



The INW Approach:

Advanced Topics

Regular ROBPs / Permutation ROBPs



Model	PRG seed length	References
Constant-width ROBPs	$O(\log^2 n)$	[Nisan 1990] [INW 1994]
Constant-width "regular" ROBPs	$\tilde{O}(\log n)$	[BRRY 2010]
Constant-width "permutation" ROBPs	$O(\log n)$	[KNP 2011] [De 2011] [Steinke 2012] [CDG 2026]
"Unbounded-width" permutation ROBPs	$\tilde{O}(\log n)$	[AKMPSV 2020] [HPV 2021]

- In each case, the INW generator does the job (with appropriate λ)
- Analyses are all about tracking and bounding the error accumulation

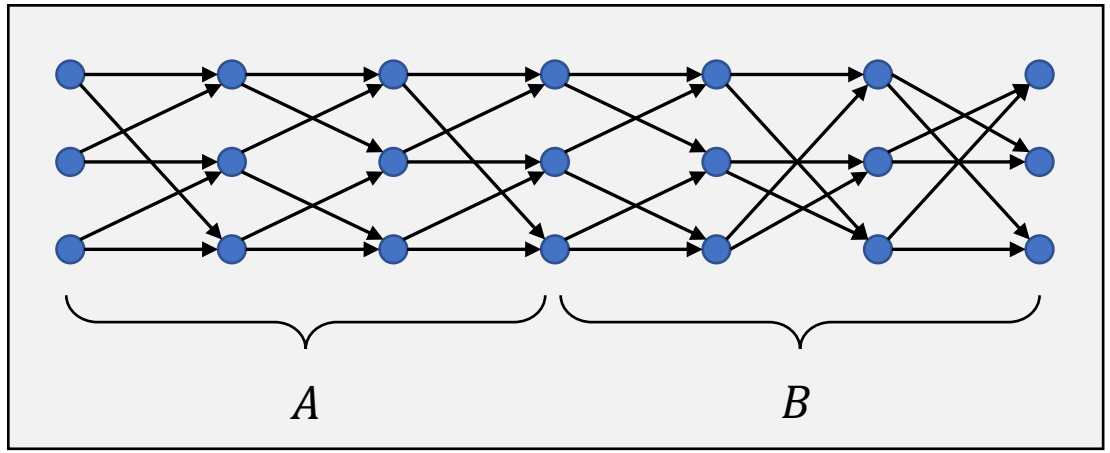
Singular-value approximation

[Ahmadinejad, Peebles, Pyne, Sidford, Vadhan 2023]

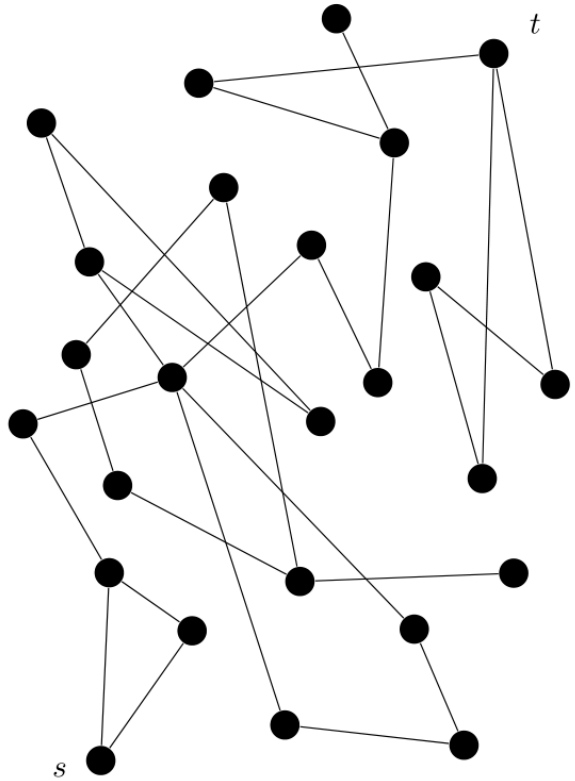
- Let $W, \widetilde{W}$ be doubly stochastic matrices
- **Definition:** $\widetilde{W} \approx_{\varepsilon}^{\mathrm{sv}} W$ if, for all vectors x, y, we have

$$\left|x^{\top}\left(\widetilde{W} - W\right)y\right| \leq \frac{\varepsilon}{4} \cdot \left(\|x\|_2^2 - \|W^{\top}x\|_2^2 + \|y\|_2^2 - \|Wy\|_2^2\right)$$

- **Lemma:** For permutation programs, each INW step induces $\widetilde{W}$ such that $\widetilde{W} \approx_{\lambda}^{\mathrm{sv}} AB$



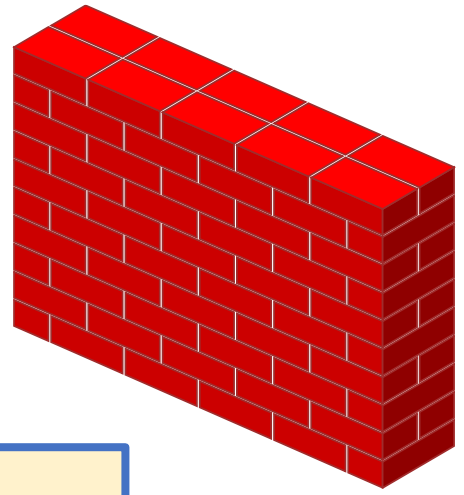
Undirected s-t connectivity



Theorem: There is a deterministic algorithm that solves the undirected s-t connectivity problem using $O(\log n)$ bits of space

- Original proof [Reingold, STOC 2005 / JACM 2008] uses "zig-zag product"
- Alternative proof [Rozenman and Vadhan, RANDOM 2005] uses "derandomized square" / INW generator

Limitations of INW

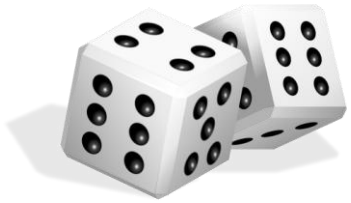


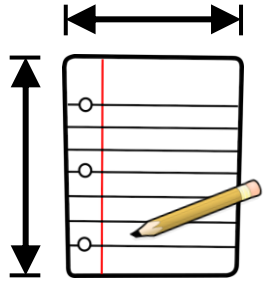
Theorem [BV 2010] [HPV 2024]: There exist expander graphs $H_1, \ldots, H_{\log n}$ such that $\lambda(H_i) \leq 1/n^{\Omega(1)}$, but the corresponding INW generator does not fool width-3 ROBPs

- **More generally:** If an INW generator fools width-3 ROBPs and the correctness proof only uses $\lambda(H_i)$, then the seed length is $\Omega(\log^2 n)$

The Nisan Approach

$\mathrm{BPL} \subseteq \mathrm{SC}$



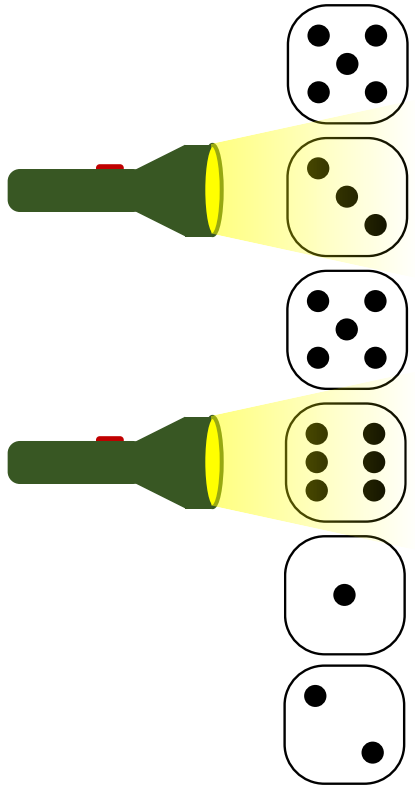




Theorem [Nisan 1994]: For every $f \in \mathrm{BPL}$, there exists a deterministic algorithm that computes f using $O(\log^2 n)$ bits of space and polynomial time

- Key ingredient: Pairwise uniform hash functions

Pairwise uniform hash functions



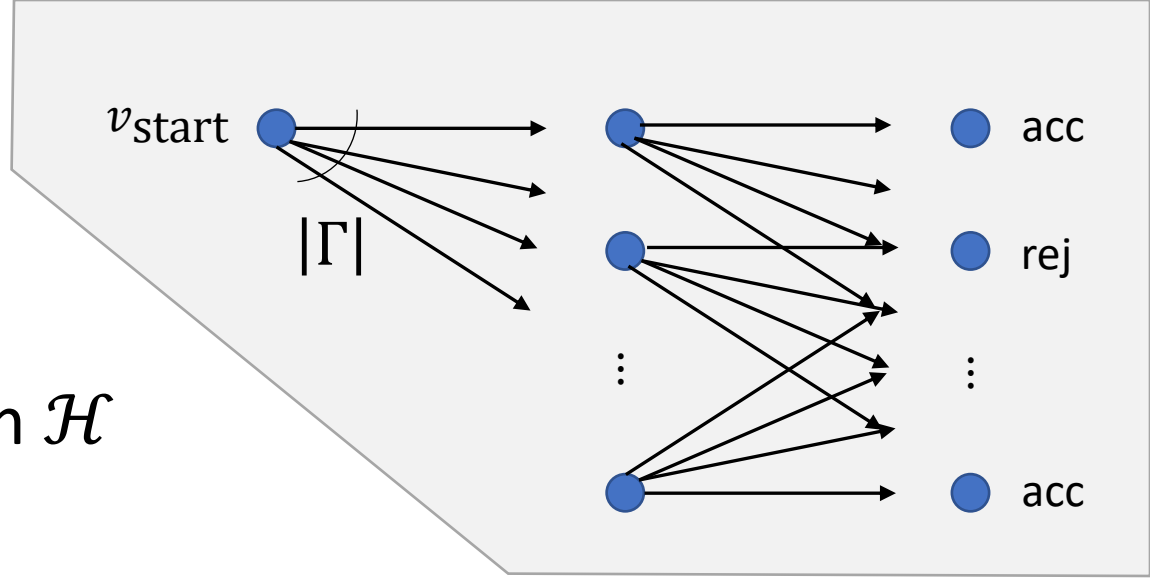
- Let $\mathcal{H}$ be a family of hash functions $h: \Gamma \to \Gamma$
- **Definition:** $\mathcal{H}$ is pairwise uniform if, for all

  $x_1, x_2, y_1, y_2 \in \Gamma$ such that $x_1 \neq x_2$, we have

$$\Pr_{h \sim \mathcal{H}}[h(x_1) = y_1 \text{ and } h(x_2) = y_2] = \frac{1}{|\Gamma|^2}$$

- Explicit constructions known with $|\mathcal{H}| \leq \text{poly}(|\Gamma|)$

Good hash functions

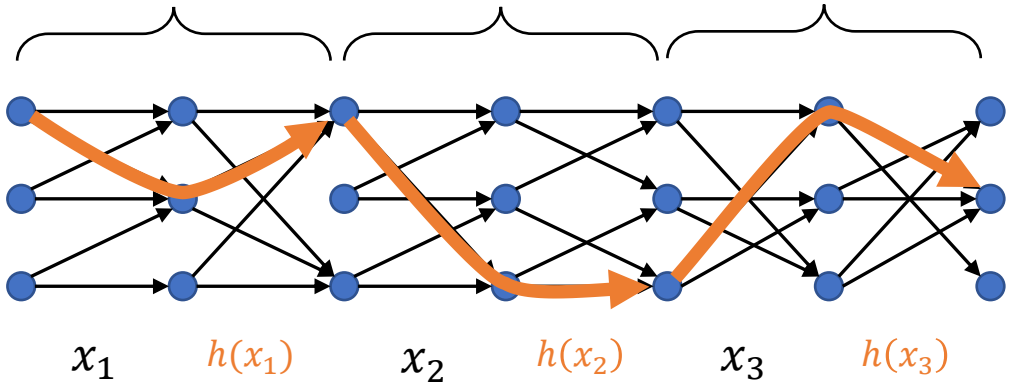


- Sample $h: \Gamma \to \Gamma$ from a pairwise uniform $\mathcal{H}$
- Define $G_h: \Gamma \to \Gamma^2$ by $G_h(x) = \big(x, h(x)\big)$
- Let $f: \Gamma^2 \to \{0,1\}$ be a width-w length-2 ROBP over the alphabet Γ
- Say h is "good" if G_h fools f with error ε

Lemma: The fraction of h that are "bad" is at most $\frac{w^3}{\varepsilon^2|\Gamma|}$.

[Proof on whiteboard]

$\mathrm{BPL} \subseteq \mathrm{SC}$: Proof sketch



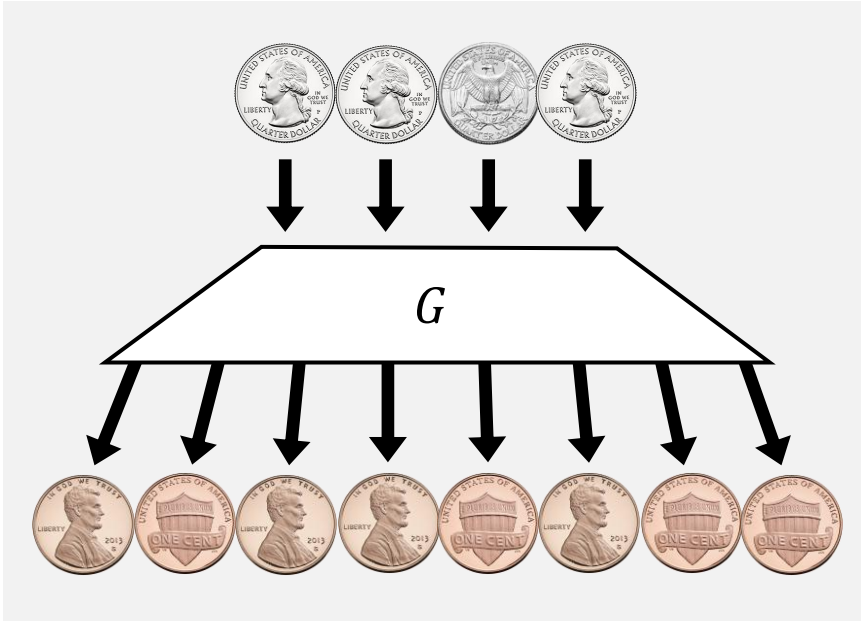
- **Given:** Width-n length-n ROBP f over binary alphabet
- **Goal:** Estimate $\mathbb{E}[f]$ to within ± 0.1

1. Artificially increase alphabet size to $\mathrm{poly}(n)$
2. Find $h \in \mathcal{H}$ that is "good" for all length-two subprograms
3. Use h to compress two layers down to one
4. Repeat steps 2 & 3 until length $= 1$

Exhaustive search (polynomial time)

$O(\log^2 n)$ bits of space to store hash functions

Comparison: INW vs. Nisan



- Can use Nisan's approach to construct PRG
  - Use seed to pick hash functions

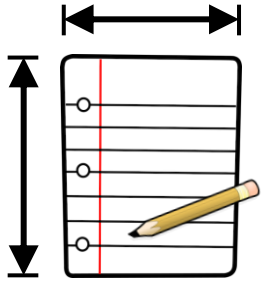
    $h_1, \ldots, h_{\log n} : \Gamma \to \Gamma$ and to pick $x \in \Gamma$

- Seed length is not better than INW
- But the specific structure of Nisan's PRG is sometimes useful

The Nisan Approach:

Advanced Topics

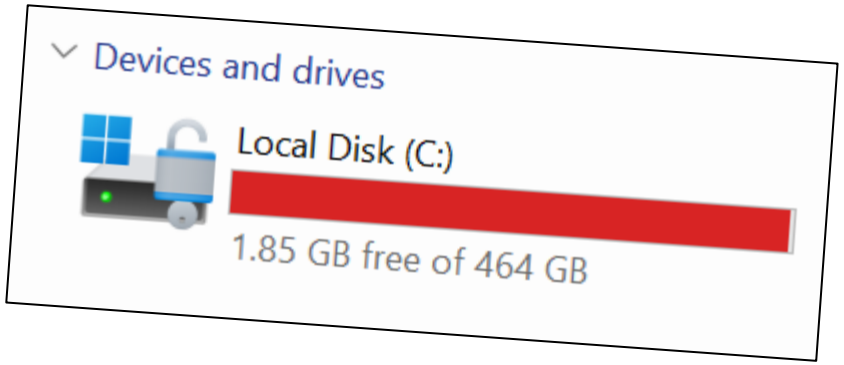
$\mathrm{BPL} \subseteq \mathrm{L}^{3/2}$



Theorem [Saks, Zhou 1999]: For every $f \in \mathrm{BPL}$, there exists a deterministic algorithm that computes f using $O\left(\log^{3/2} n\right)$ bits of space

- Proof based on reusing Nisan's hash functions many times, with randomized rounding to break stochastic dependencies
- Current state of the art is only slightly better

Catalytic derandomization



Theorem [Pyne 2024]: For every $f \in \mathrm{BPL}$, there exists a deterministic algorithm that computes f using $O(\log n)$ bits of ordinary space and $O(\log^2 n)$ bits of catalytic space

- Proof idea: Use initial data as hash functions
- Either they are "good," or else they can be compressed

Summary

- Today:
  - The INW Approach ✔
  - The Nisan Approach ✔
- Tomorrow:
  - The Laplacian Approach
  - The Restrictions Approach

