INTRODUCTION TO MATRIX RIGIDITY

PART 2

Sasha Golovnev

December 14, 2020

OUTLINE

- Reminder of Part 1

OUTLINE

- Reminder of Part 1
    - Explicit Constructions

OUTLINE

- Reminder of Part 1
    - Explicit Constructions
    - Semi-explicit Constructions

OUTLINE

- Reminder of Part 1
    - Explicit Constructions
    - Semi-explicit Constructions

- Limitations

OUTLINE

- Reminder of Part 1
    - Explicit Constructions
    - Semi-explicit Constructions

- Limitations

- Applications

REMINDER OF PART 1

RIGIDITY. DEFINITION

Definition

$$\mathcal{R}_A^{\mathbb{F}}(r) := \min_{\mathrm{rank}(A+C)\le r} \|C\|_0 \ .$$

Rigidity. Definition

Definition

$$\mathcal{R}_A^{\mathbb{F}}(r) := \min_{\mathrm{rank}(A+C)\leq r} \|C\|_0 \ .$$

Let $\mathbb{F}$ be a field, $A \in \mathbb{F}^{n\times n}$ be a matrix, and $0 \leq r \leq n$. The rigidity of A over $\mathbb{F}$, denoted by $\mathcal{R}_A^{\mathbb{F}}(\underset{\sim}{r})$, is the Hamming distance between A and the set of matrices of rank at most r.

Non-rigid = Sparse + Low-Rank

Rigid $\neq$ Sparse $+$ Low-Rank

EXAMPLE [MID05]

$$\left(\frac{n}{2R}\right)^2 \text{ matrices}$$

$$M_n = \begin{pmatrix} I_{2r} & \cdots & I_{2r} \\ \vdots & \ddots & \vdots \\ I_{2r} & \cdots & I_{2r} \end{pmatrix} .$$

$$R \cdot \frac{n^2}{4R^2} = \Omega\left(\frac{n^2}{R}\right)$$

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r}\right) .$$

EXPLICIT BOUNDS ON RIGIDITY

- What we need for circuit lower bounds:

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = n^{1+\delta} \text{ for } r = \Omega(n)\,.$$

Explicit Bounds on Rigidity

- What we need for circuit lower bounds:

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = n^{1+\delta} \text{ for } r = \Omega(n)\,.$$

- (Even $\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \underline{\omega(n)}$ for $r = \Omega(n)$ would give new circuit lower bounds).

EXPLICIT BOUNDS ON RIGIDITY

- What we need for circuit lower bounds:

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = n^{1+\delta} \text{ for } r = \Omega(n)\,.$$

- (Even $\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \omega(n)$ for $r = \Omega(n)$ would give new circuit lower bounds).
- A simple explicit matrix of rigidity

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r}\right)$$

when $R = \Theta(n)$,

$= \Omega(n)$

EXPLICIT BOUNDS ON RIGIDITY

- What we need for circuit lower bounds:

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = n^{1+\delta} \text{ for } r = \Omega(n)\,.$$

- ✓ (Even $\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \omega(n)$ for $r = \Omega(n)$ would give new circuit lower bounds). Series-Parallel
- A simple explicit matrix of rigidity

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r}\right)\,.$$

- The best known explicit bound:

$$\mathcal{R}^{\mathbb{F}}_{M_n}(r) = \Omega\left(\frac{n^2}{r} \cdot \log\frac{n}{r}\right)\,.$$

$R = \Theta(n)$

$\log \frac{n}{R} = \Theta(1)$

$E = \mathrm{Time}[2^{O(n)}]$

E^{NP}

Semi-explicit Bounds on Rigidity

construction	rigidity	run-time

SEMI-EXPLICIT BOUNDS ON RIGIDITY

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	$\mathrm{poly}(n)$

SEMI-EXPLICIT BOUNDS ON RIGIDITY

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \underline{\frac{n^2}{r} \cdot \log \frac{n}{r}}$	$\mathrm{poly}(n)$
brute force	$\mathcal{R}(\varepsilon n) \geq \underline{n^2}$	$\underline{2^{n^2}}$

Semi-explicit Bounds on Rigidity

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	$\text{poly}(n)$
brute force	$\mathcal{R}(\varepsilon n) \geq n^2$	2^{n^2}
sparse	$\mathcal{R}(\varepsilon n) \geq n^{1+\delta}$	$\underline{2^{n^{1+\delta} \log n}}$

Semi-explicit Bounds on Rigidity

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	$\mathrm{poly}(n)$
brute force	$\mathcal{R}(\varepsilon n) \geq n^2$	2^{n^2}
sparse	$\mathcal{R}(\varepsilon n) \geq n^{1+\delta}$	$2^{n^{1+\delta} \log n}$
sub-exponential	$\mathcal{R}(\underline{n^{0.5-\varepsilon})} \geq \frac{n^2}{\log n}$	$\underline{2^{n^{1-\varepsilon}}}$

Semi-explicit Bounds on Rigidity

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	$\mathrm{poly}(n)$
brute force	$\mathcal{R}(\varepsilon n) \geq n^2$	2^{n^2}
sparse	$\mathcal{R}(\varepsilon n) \geq n^{1+\delta}$	$2^{n^{1+\delta} \log n}$
sub-exponential	$\mathcal{R}(n^{0.5-\varepsilon}) \geq \frac{n^2}{\log n}$	$2^{n^{1-\varepsilon}}$
Vanderm. alg. ind	$\mathcal{R}(\sqrt{n}) \geq \delta n^2$	NA

SEMI-EXPLICIT BOUNDS ON RIGIDITY

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	$\text{poly}(n)$
brute force	$\mathcal{R}(\varepsilon n) \geq n^2$	2^{n^2}
sparse	$\mathcal{R}(\varepsilon n) \geq n^{1+\delta}$	$2^{n^{1+\delta} \log n}$
sub-exponential	$\mathcal{R}(n^{0.5-\varepsilon}) \geq \frac{n^2}{\log n}$	$2^{n^{1-\varepsilon}}$
Vanderm. alg. ind	$\mathcal{R}(\sqrt{n}) \geq \delta n^2$	NA
$\sqrt{p_i}$	$\mathcal{R}(\varepsilon n) \geq \delta n^2$	NA

Ben Lee's talk

SEMI-EXPLICIT BOUNDS ON RIGIDITY

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	$\mathrm{poly}(n)$
brute force	$\mathcal{R}(\varepsilon n) \geq n^2$	2^{n^2}
sparse	$\mathcal{R}(\varepsilon n) \geq n^{1+\delta}$	$2^{n^{1+\delta} \log n}$
sub-exponential	$\mathcal{R}(n^{0.5-\varepsilon}) \geq \frac{n^2}{\log n}$	$2^{n^{1-\varepsilon}}$
Vanderm. alg. ind	$\mathcal{R}(\sqrt{n}) \geq \delta n^2$	NA
$\sqrt{p_i}$	$\mathcal{R}(\varepsilon n) \geq \delta n^2$	NA
Hankel	$\mathcal{R}(r) \geq \underline{\frac{n^3}{r^2 \log n}}$	$\underline{2^n}$

Semi-explicit Bounds on Rigidity

construction	rigidity	run-time
explicit	$\mathcal{R}(r) \geq \frac{n^2}{r} \cdot \log \frac{n}{r}$	poly(n)
brute force	$\mathcal{R}(\varepsilon n) \geq n^2$	2^{n^2}
sparse	$\mathcal{R}(\varepsilon n) \geq n^{1+\delta}$	$2^{n^{1+\delta}\log n}$
sub-exponential	$\mathcal{R}(n^{0.5-\varepsilon}) \geq \frac{n^2}{\log n}$	$2^{n^{1-\varepsilon}}$
Vanderm. alg. ind	$\mathcal{R}(\sqrt{n}) \geq \delta n^2$	NA
$\sqrt{p_i}$	$\mathcal{R}(\varepsilon n) \geq \delta n^2$	NA
Hankel	$\mathcal{R}(r) \geq \frac{n^3}{r^2 \log n}$	2^n
PCP Amey's talk	$\mathcal{R}(\underline{2^{\log n / \log\log n}}) \geq \underline{\delta n^2}$	$\underline{\mathsf{P}^{\mathsf{NP}}}$

LIMITATIONS

RIGIDITY CANDIDATES

Conjecture [Lokam 2009]

Many candidate matrices are conjectured to have rigidity as high as in Valiant's question. Examples include:

Rigidity Candidates

Walsh-Hadamard matrix	?
Generalized Hadamard matrices	?
Fourier transform matrices	?
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	?
Good linear codes	?
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

UNTOUCHED MINOR LOWER BOUND

- Untouched minor

UNTOUCHED MINOR LOWER BOUND

- Untouched minor
- Step 1: $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$ changes in $n \times n$ matrix leave an $r \times r$ submatrix untouched

UNTOUCHED MINOR LOWER BOUND

- Untouched minor
- Step 1: $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$ changes in $n \times n$ matrix leave an $r \times r$ submatrix untouched
- Step 2: Take a matrix where each $r \times r$ submatrix is full-rank

UNTOUCHED MINOR LOWER BOUND

- Untouched minor
- Step 1: $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$ changes in $n \times n$ matrix leave an $r \times r$ submatrix untouched
- Step 2: Take a matrix where each $r \times r$ submatrix is full-rank
- After $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$ changes, the rank is $\geq r$

LIMITATIONS OF UNTOUCHED MINOR

- This method can give bounds of $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$

LIMITATIONS OF UNTOUCHED MINOR

- This method can give bounds of $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$
- Limitation 1:
  There is a set of $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$ elements of a matrix that touches every $r \times r$ submatrix [Lok00]

LIMITATIONS OF UNTOUCHED MINOR

- This method can give bounds of $\underline{O(\frac{n^2}{r} \cdot \log \frac{n}{r})}$
- Limitation 1:
  There is a set of $O(\frac{n^2}{r} \cdot \log \frac{n}{r})$ elements of a matrix that touches every $r \times r$ submatrix [Lok00]
- Limitation 2:
  There is a matrix where all submatrices have full rank, yet it is not rigid [Val75]

Rigidity Candidates

Walsh-Hadamard matrix	?
Generalized Hadamard matrices	?
Fourier transform matrices	?
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	?
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

LINEAR CODES

- A linear code C is a k-dimensional subspace of $\mathbb{F}^n$

LINEAR CODES

- A linear code C is a k-dimensional subspace of $\mathbb{F}^n$
- The distance of C is

$$d(C) = \mathbf{min}\left(\|w\|_0 : w \in C, w \neq \mathbf{0}\right)$$

LINEAR CODES

- A linear code C is a k-dimensional subspace of $\mathbb{F}^n$
- The distance of C is

$$d(C) = \min\left(\|w\|_0 : w \in C, w \neq \mathbf{0}\right)$$

- A generator matrix $G \in \mathbb{F}^{n \times k}$ is a matrix whose columns form a basis of C

Explicit Linear Codes

Proposition

For any finite field $\mathbb{F}$, there exists an explicit family of linear error correcting codes over $\mathbb{F}$ of dimension $\underline{k = n/4}$ and minimum distance $\underline{d = \delta n}$ for a constant $\delta > 0$.

EXPLICIT LINEAR CODES

Proposition

For any finite field $\mathbb{F}$, there exists an explicit family of linear error correcting codes over $\mathbb{F}$ of dimension $k = n/4$ and minimum distance $d = \delta n$ for a constant $\delta > 0$.

Such codes are called good.

RIGIDITY OF CODES

- [Fri93], [PR94], [SSS97]: every generator matrix G of a good code has rigidity

$$\mathcal{R}_G^{\mathbb{F}}(r) \geq \Omega\left(\frac{n^2}{r} \cdot \log \frac{n}{r}\right) .$$

RIGIDITY OF CODES

- [Fri93], [PR94], [SSS97]: every generator matrix G of a good code has rigidity

$$\mathcal{R}_G^{\mathbb{F}}(r) \geq \Omega\left(\frac{n^2}{r} \cdot \log \frac{n}{r}\right) .$$

- Every good code has a generator matrix G

$$\mathcal{R}_G^{\mathbb{F}}(\varepsilon n) \geq \Omega(n^2) .$$

RIGIDITY OF CODES

- [Fri93], [PR94], [SSS97]: every generator matrix G of a good code has rigidity

$$\mathcal{R}_G^{\mathbb{F}}(r) \geq \Omega\left(\frac{n^2}{r} \cdot \log \frac{n}{r}\right) .$$

- Every good code has a generator matrix G

$$\mathcal{R}_G^{\mathbb{F}}(\varepsilon n) \geq \Omega(n^2) .$$

- [Dvi16] Some good codes have a generator matrix G

$$\mathcal{R}_G^{\mathbb{F}}(r) \leq O\left(\frac{n^2}{r} \cdot \log \frac{n}{r}\right) .$$

RIGIDITY OF CODES

- [Fri93], [PR94], [SSS97]: every generator matrix G of a good code has rigidity

$$\mathcal{R}_G^{\mathbb{F}}(r) \geq \Omega\left(\frac{n^2}{r} \cdot \log \frac{n}{r}\right) .$$

- Every good code has a generator matrix G

$$\mathcal{R}_G^{\mathbb{F}}(\varepsilon n) \geq \Omega(n^2) .$$

- [Dvi16] Some good codes have a generator matrix G

$$\mathcal{R}_G^{\mathbb{F}}(r) \leq O\left(\frac{n^2}{r} \cdot \log \frac{n}{r}\right) .$$

- Thus, we cannot improve the known explicit bound for all generator matrices of good codes

Rigidity Candidates

Walsh-Hadamard matrix	?
Generalized Hadamard matrices	?
Fourier transform matrices	?
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

HADAMARD MATRIX

$$
\begin{aligned}
H_2 &= \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \\
H_N &= \begin{pmatrix} H_{N/2} & H_{N/2} \\ H_{N/2} & -H_{N/2} \end{pmatrix} \text{ for } N = 2^n > 2 \,.
\end{aligned}
$$

KNOWN BOUNDS FOR HADAMARD

rigidity	reference
$\frac{n^2}{r^4 \log^2 r}$	Pudlák and Savickỳ, 88
$\frac{n^2}{r^3 \log r}$	Razborov, 88
$\frac{n^2}{r^2}$	Alon, 90
$\frac{n^2}{r^2}$	Lokam, 95
$\frac{n^2}{256r}$	Kashin and Razborov, 98
$\frac{n^2}{4r}$	Midrijānis, 05; de Wolf, 06

KNOWN BOUNDS FOR HADAMARD

rigidity	reference
$\frac{n^2}{r^4 \log^2 r}$	Pudlák and Savickỳ, 88
$\frac{n^2}{r^3 \log r}$	Razborov, 88
$\frac{n^2}{r^2}$	Alon, 90
$\frac{n^2}{r^2}$	Lokam, 95
$\frac{n^2}{256r}$	Kashin and Razborov, 98
$\frac{n^2}{4r}$	Midrijānis, 05; de Wolf, 06

[AW17]: H is not rigid for any $r = O(n)$.

Josh's talk

Rigidity Candidates

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	?
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

Rigidity Candidates

[DE17]

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	?
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

Rigidity Candidates

[DL19]

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	?
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

Rigidity Candidates

[DL19]

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	✗
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	?
Incidence matrices of projective planes	?
Cayley graphs	?

Rigidity Candidates

[DL19]

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	✗
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	✗
Incidence matrices of projective planes	?
Cayley graphs	?

Rigidity Candidates

[DL19]

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	✗
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	✗
Incidence matrices of projective planes	✗
Cayley graphs	?

Rigidity Candidates

[DL19]

Walsh-Hadamard matrix	✗
Generalized Hadamard matrices	?
Fourier transform matrices	✗
Vandermonde matrices	?
Cauchy matrices	?
Super regular matrices	✗
Good linear codes	✗
Hankel matrices	✗
Incidence matrices of projective planes	✗
Cayley graphs	✗

Josh's talk

APPLICATIONS

APPLICATIONS OF RIGIDITY

- Communication Complexity
- Circuit Complexity
- Data Structures
- Error Correcting Codes

Rigidity and Communication Complexity

Theorem (Raz89)

If $M \in \mathbb{F}_2^{n\times n}$ *has rigidity*

$$\mathcal{R}_M^{\mathbb{F}_2}(r) \geq \frac{n^2}{2^{\log r^{o(1)}}} \text{ for } \underline{r \geq 2^{\log\log n^{\omega(1)}}}$$

then $M \notin$ **PH**$^{\text{cc}}$.

RIGIDITY AND COMMUNICATION COMPLEXITY

Theorem (Raz89)

If $M \in \mathbb{F}_2^{n\times n}$ has rigidity

$$\mathcal{R}_M^{\mathbb{F}_2}(r) \geq \frac{n^2}{2^{\log r^{o(1)}}} \text{ for } r \geq 2^{\log\log n^{\omega(1)}}$$

then $M \notin \mathsf{PH}^{\mathsf{cc}}$.

Theorem (AC19, BHPT20)

$\mathsf{E}^{\mathsf{NP}} \not\subseteq \mathsf{PH}^{\mathsf{cc}}$.

$\mathrm{Time}[2^{2^{(\log\log n)^2}}]^{NP} \notin PH^{cc}$

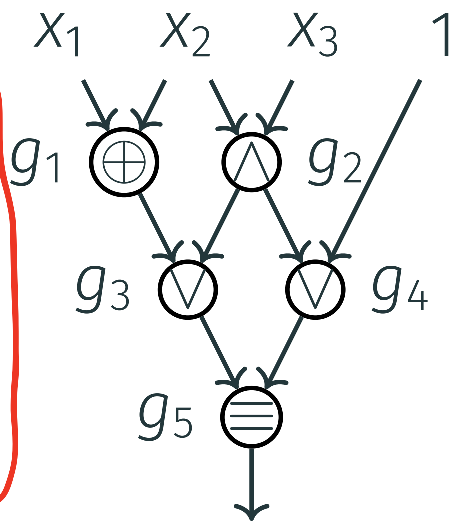
CIRCUITS AND RIGIDITY

BOOLEAN CIRCUITS

$$f\colon \{0,1\}^n \to \{0,1\}^n$$

$$\begin{aligned} g_1 &= x_1 \oplus x_2 \\ g_2 &= x_2 \wedge x_3 \\ g_3 &= g_1 \vee g_2 \\ g_4 &= g_2 \vee 1 \\ g_5 &= g_3 \equiv g_4 \end{aligned}$$

circuit



Inputs:

$x_1, \ldots, x_n, 0, 1$

Gates:

binary functions

Fan-out:

unbounded

EXPONENTIAL BOUNDS

Lower Bound [Sha1949]

Counting shows that almost all functions of n variables have circuit size at least

$$2^n \, .$$

EXPONENTIAL BOUNDS

Lower Bound [Sha1949]

Counting shows that almost all functions of n variables have circuit size at least

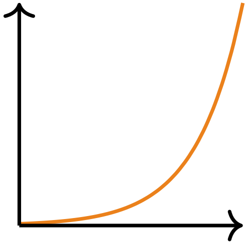
$$2^n .$$

Upper Bound [Lup1958]

Every function can be computed by a circuit of size

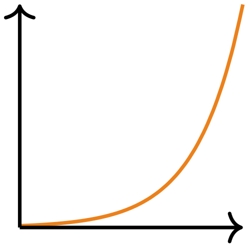
$$2^n .$$

EXPLICIT BOUNDS



Most functions have exponential circuit complexity

EXPLICIT BOUNDS

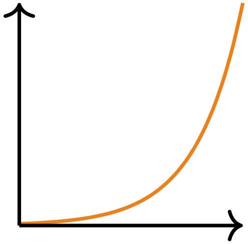


Most functions have exponential circuit complexity

$P \neq NP$

We want to prove super-polynomial lower bounds

EXPLICIT BOUNDS

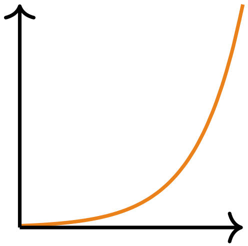


Most functions have exponential circuit complexity

$P \neq NP$

We want to prove super-polynomial lower bounds (for a function from **NP**)

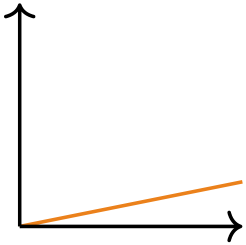
Explicit Bounds



Most functions have exponential circuit complexity

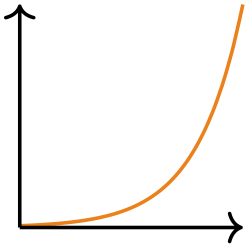
$\mathrm{P} \neq \mathrm{NP}$

We want to prove superpolynomial lower bounds (for a function from **NP**)



We can prove only $\approx 3n$ lower bounds

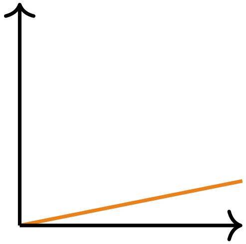
Explicit Bounds



Most functions have exponential circuit complexity

$P \neq NP$

We want to prove superpolynomial lower bounds
(for a function from **NP**)



We can prove only $\approx 3n$ lower bounds
(even for a function from $\mathbf{E}^{\mathbf{NP}}$)

SUPER-LINEAR CIRCUIT LOWER BOUNDS?

- Two n-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]

SUPER-LINEAR CIRCUIT LOWER BOUNDS?

- Two n-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length n can be computed by a circuit of size $O(n \log n)$

SUPER-LINEAR CIRCUIT LOWER BOUNDS?

- Two n-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length n can be computed by a circuit of size $O(n \log n)$
- Shifts, Permutations

SUPER-LINEAR CIRCUIT LOWER BOUNDS?

- Two n-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length n can be computed by a circuit of size $O(n \log n)$
- Shifts, Permutations
- NP-hard problems

SUPER-LINEAR CIRCUIT LOWER BOUNDS?

- Two n-bit integers can be multiplied by a circuit of size $O(n \log n)$ [SS71,F07,HH19]
- Discrete Fourier Transform of a sequence of length n can be computed by a circuit of size $O(n \log n)$
- Shifts, Permutations
- NP-hard problems
- $\ldots$

What We Can Prove

WHAT WE CAN PROVE

- Depth 2: CNF/DNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.

WHAT WE CAN PROVE

- Depth 2: CNF/DNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.
- Constant depth d. Lower bounds $2^{n^{1/(d-1)}}$.

WHAT WE CAN PROVE

- Depth 2: CNF/DNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.
- Constant depth d. Lower bounds $2^{n^{1/(d-1)}}$.
- Depth $1.9 \log n$. Know functions that cannot be computed.

WHAT WE CAN PROVE

- Depth 2: CNF/DNF. Even $\oplus_n$ requires circuits of size $\Omega(2^n)$.
- Constant depth d. Lower bounds $2^{n^{1/(d-1)}}$.
- Depth $1.9 \log n$. Know functions that cannot be computed.
- Depth $2 \log n$. Nothing better than $\approx 3n$.

PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $10n$ *against circuits of depth* $10 \log n$.

PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $10n$ *against circuits of depth* $10 \log n$.

More generally, a lower bound of $\omega(n)$ *against circuits of depth* $O(\log n)$.

PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $10n$ *against circuits of depth* $10 \log n$.

More generally, a lower bound of $\omega(n)$ *against circuits of depth* $O(\log n)$.

Valiant [Val77] gives us an amazing tool to study such circuits.

ANOTHER PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $\omega(n)$ *against linear circuits of depth* $O(\log n)$.

ANOTHER PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $\omega(n)$ *against linear circuits of depth* $O(\log n)$.

- Incomparable to the previous problem (bounds against non-linear circuits):

ANOTHER PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $\omega(n)$ *against linear circuits of depth* $O(\log n)$.

- Incomparable to the previous problem (bounds against non-linear circuits):
- Weaker computational model

ANOTHER PROBLEM ON THE FRONTIER

Problem

Prove a lower bound of $\omega(n)$ against linear circuits of depth $O(\log n)$.

- Incomparable to the previous problem (bounds against non-linear circuits):
- Weaker computational model
- But fewer problems to prove lower bounds for.

RIGIDITY IMPLIES CIRCUIT LOWER BOUNDS

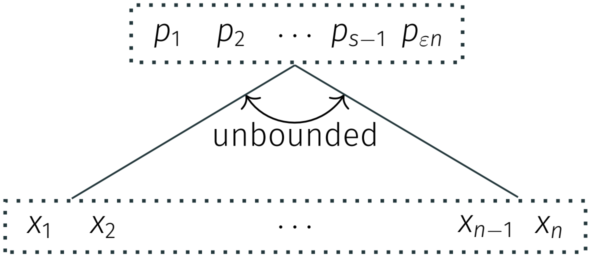
Theorem (Val77)

If $\mathcal{R}_A^{\mathbb{F}}(\varepsilon n) > n^{1+\delta}$ *for constant* $\varepsilon, \delta > 0$, *then any* $O(\log n)$*-depth linear circuit computing* $x \to Ax$ *must be of size* $\omega(n)$.

RIGIDITY IMPLIES CIRCUIT LOWER BOUNDS

Theorem (Val77)

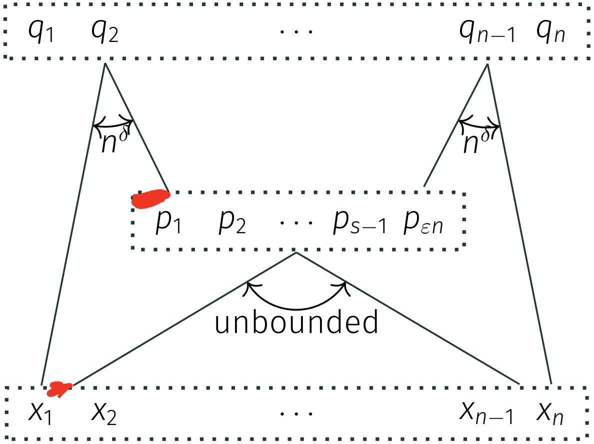
If $\mathcal{R}^{\mathbb{F}}_{A}(\varepsilon n) > n^{1+\delta}$ for constant $\varepsilon, \delta > 0$, then any $O(\log n)$-depth linear circuit computing $x \to Ax$ must be of size $\omega(n)$.



RIGIDITY IMPLIES CIRCUIT LOWER BOUNDS

Theorem (Val77)

If $\mathcal{R}_A^{\mathbb{F}}(\varepsilon n) > n^{1+\delta}$ *for constant* $\varepsilon, \delta > 0$, *then any* $O(\log n)$*-depth linear circuit computing* $x \to Ax$ *must be of size* $\omega(n)$.



Rigidity for rank $n/100$ and sparsity $n^{1.01}$ implies super-linear log-depth circuit lower bounds

DEPTH REDUCTIONS

- The proof reduces the depth of a circuit from $O(\log n)$ to 2 (and the latter is equivalent to rigidity)

DEPTH REDUCTIONS

- The proof reduces the depth of a circuit from $O(\log n)$ to 2 (and the latter is equivalent to rigidity)
- The proof is graph-theoretic, and graph-theoretic proofs cannot go beyond $O(\log n)$ depth [Sch82, Sch83, Kla94]

DEPTH REDUCTIONS

- The proof reduces the depth of a circuit from $O(\log n)$ to 2 (and the latter is equivalent to rigidity)
- The proof is graph-theoretic, and graph-theoretic proofs cannot go beyond $O(\log n)$ depth [Sch82, Sch83, Kla94]
- A non-graph-theoretic proof [GKW21] works for unbounded-depth circuits, but alas only for $\underline{\text{size} < 4n}$

Unbounded-depth and Rigidity

Theorem (GKW21)

Let $\mathbb{F}$ *be a field, and* $A \in \mathbb{F}^{n\times n}$ *be a family of matrices for* $n \in \mathbb{N}$.

If $\mathcal{R}_A^{\mathbb{F}}(\underline{\varepsilon n}) > \boxed{16n}$*, then any linear circuit computing* $x \to Ax$ *must be of size* $\geq \boxed{\underline{4\varepsilon n}}$.

Rigidity for rank $0.99n$ and sparsity $16n$ implies circuit lower bound of $\underline{3.9n}$

unbounded depth

General Boolean $\{0,1\}^n \to \{0,1\}$

(non-linear gates)

$\wedge \ \vee \ \neg$ $5n-o(n)$

All binary $\wedge \ \vee \ \neg$ $\oplus \ \equiv$ $3.01n$

$\{0,1\}^n \to \{0,1\}^n$

lower bounds $6n-O(n)$

$4.01n$

Linear Boolean ckts

$\{0,1\}^n \to \{0,1\}^n$

$3n-o(n)$

$\{0,1\}^n \to \{0,1\}^{\log n}$

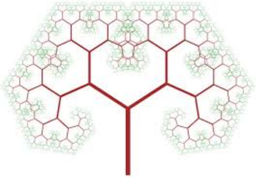
$2n-o(n)$

DATA STRUCTURES AND RIGIDITY

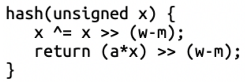
Siva's talk

DATA STRUCTURES

Stack, Queue, List, Heap



Search Trees



```
hash(unsigned x) {
    x ^= x >> (w-m);
    return (a*x) >> (w-m);
}
```

Hash Tables

STATIC DATA STRUCTURES. EXAMPLES

- Graph Distances: Preprocess a road network in order to efficiently compute distances between cities
  (Google Maps)

STATIC DATA STRUCTURES. EXAMPLES

- Graph Distances: Preprocess a road network in order to efficiently compute distances between cities
  (Google Maps)
- Nearest Neighbors: Preprocess a set of points in order to efficiently find closest point to a query point
  (Netflix recommendations)

STATIC DATA STRUCTURES. EXAMPLES

- **Graph Distances**: Preprocess a road network in order to efficiently compute distances between cities
  (Google Maps)
- **Nearest Neighbors**: Preprocess a set of points in order to efficiently find closest point to a query point
  (Netflix recommendations)
- **Range Counting**: Preprocess a set of points in order to efficiently compute the number of points in a given rectangle
  (Amazon market size estimation)

STATIC DATA STRUCTURES

1	0	1	0	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	1	1
0	0	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0	1	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	0	1	0	1	1
0	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	1
1	1	1	0	0	1	0	0	0	1	0	0	1	1	0	1	0	1	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	0	1
0	1	0	1	1	1	0	1	1	1	1	0	0	1	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	1	1	0	0	1	0	1	1	0
1	0	1	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	0	1	1	0	1	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	0	0	1	0	0
0	1	1	0	1	0	1	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	0	0	0	0
1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	1	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	0	1	1	1	0	1	0	0	1	0	1	1	1	0	1	0	1	0

Preprocessing

STATIC DATA STRUCTURES

Queries

1	0	1	0	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	1	1
0	0	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0	1	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	0	1	0	1	1
0	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	1
1	1	1	0	0	1	0	0	0	1	0	0	1	1	0	1	0	1	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	0	1
0	1	0	1	1	1	0	1	1	1	1	0	0	1	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	1	1	0	0	1	0	1	1	0
1	0	1	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	0	1	1	0	1	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	0	0	1	0	0
0	1	1	0	1	0	1	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	0	0	0	0
1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	1	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	0	1	1	1	0	1	0	0	1	0	1	1	1	0	1	0	1	0

Preprocessing

STATIC DATA STRUCTURES

Queries

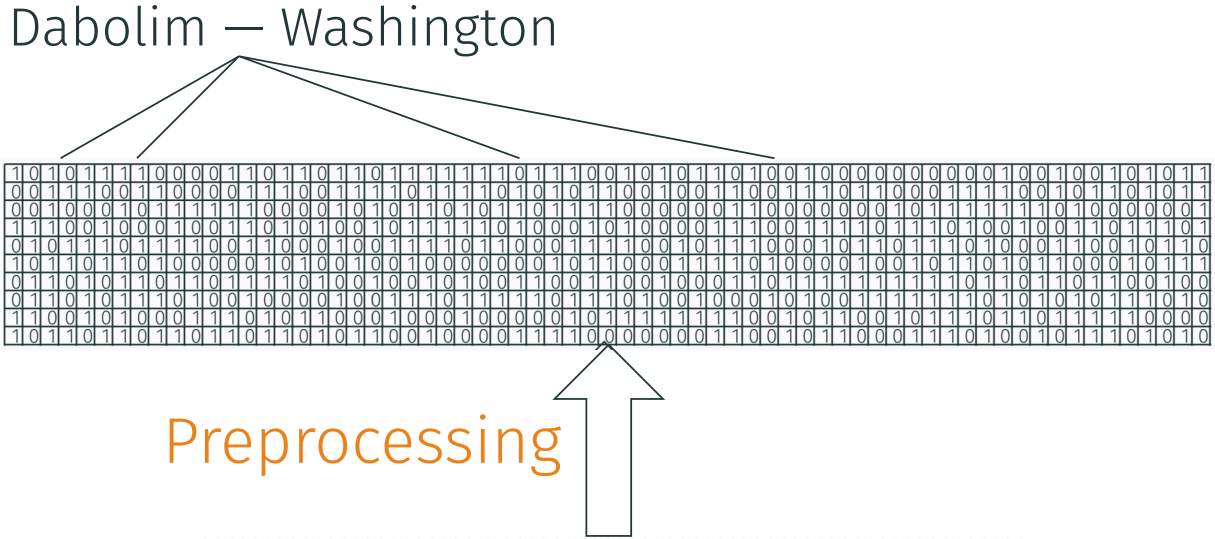
Dabolim — Washington

1	0	1	0	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	1	1
0	0	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0	1	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	0	1	0	1	1
0	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	1
1	1	1	0	0	1	0	0	0	1	0	0	1	1	0	1	0	1	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	0	1
0	1	0	1	1	1	0	1	1	1	1	0	0	1	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	1	1	0	0	1	0	1	1	0
1	0	1	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	0	1	1	0	1	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	0	0	1	0	0
0	1	1	0	1	0	1	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	0	0	0	0
1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	1	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	0	1	1	1	0	1	0	0	1	0	1	1	1	0	1	0	1	0

Preprocessing

STATIC DATA STRUCTURES

Queries



Dabolim — Washington

1	0	1	0	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	1	1
0	0	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0	1	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	0	1	0	1	1
0	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	1
1	1	1	0	0	1	0	0	0	1	0	0	1	1	0	1	0	1	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	0	1
0	1	0	1	1	1	0	1	1	1	1	0	0	1	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	1	1	0	0	1	0	1	1	0
1	0	1	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	0	1	1	0	1	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	0	0	1	0	0
0	1	1	0	1	0	1	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	0	0	0	0
1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	1	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	0	1	1	1	0	1	0	0	1	0	1	1	1	0	1	0	1	0

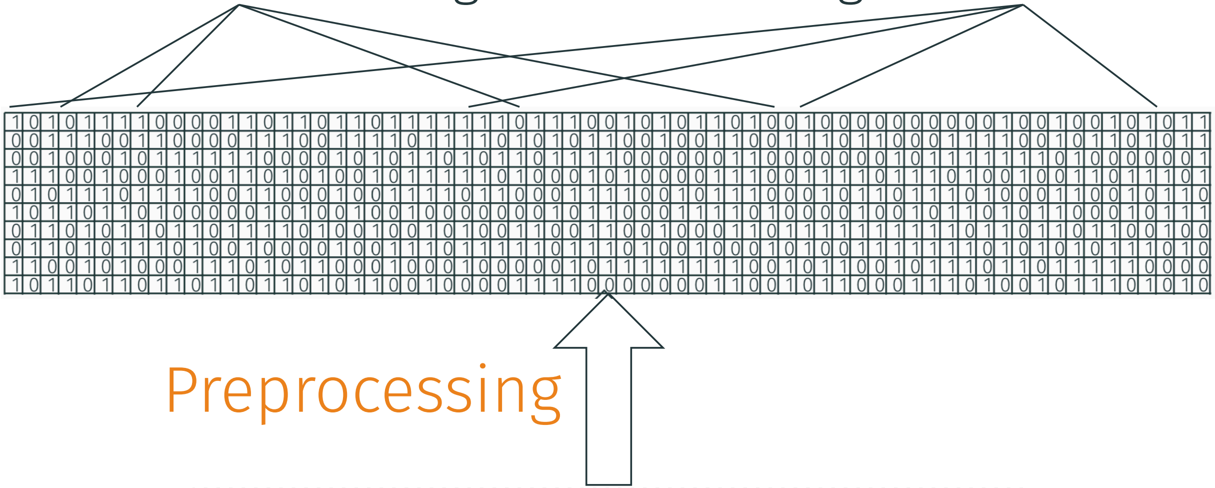
Preprocessing

STATIC DATA STRUCTURES

Queries

Dabolim — Washington

Bangalore — New York



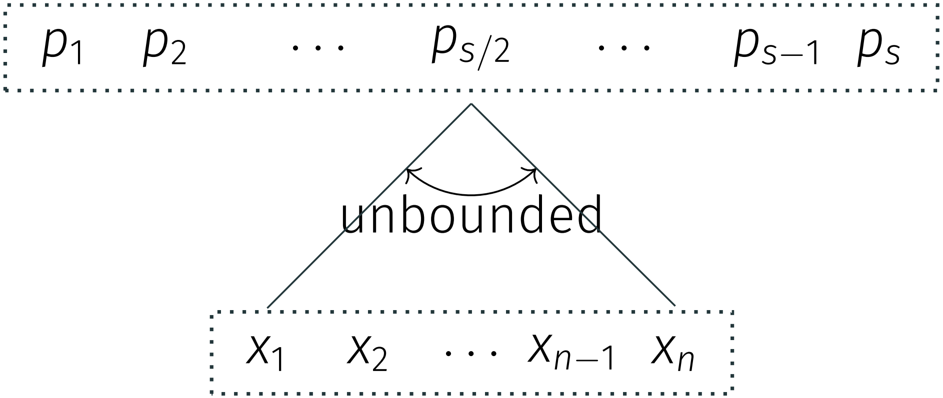
1	0	1	0	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	1	1
0	0	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0	1	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	0	1	0	1	1
0	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	1
1	1	1	0	0	1	0	0	0	1	0	0	1	1	0	1	0	1	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	0	1
0	1	0	1	1	1	0	1	1	1	1	0	0	1	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	1	1	0	0	1	0	1	1	0
1	0	1	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	0	1	1	0	1	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	0	0	1	0	0
0	1	1	0	1	0	1	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	0	0	0	0
1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	1	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	0	1	1	1	0	1	0	0	1	0	1	1	1	0	1	0	1	0

Preprocessing

Static Data Structures. Definition

$$x_1 \quad x_2 \quad \cdots \quad x_{n-1} \quad x_n$$

STATIC DATA STRUCTURES. DEFINITION

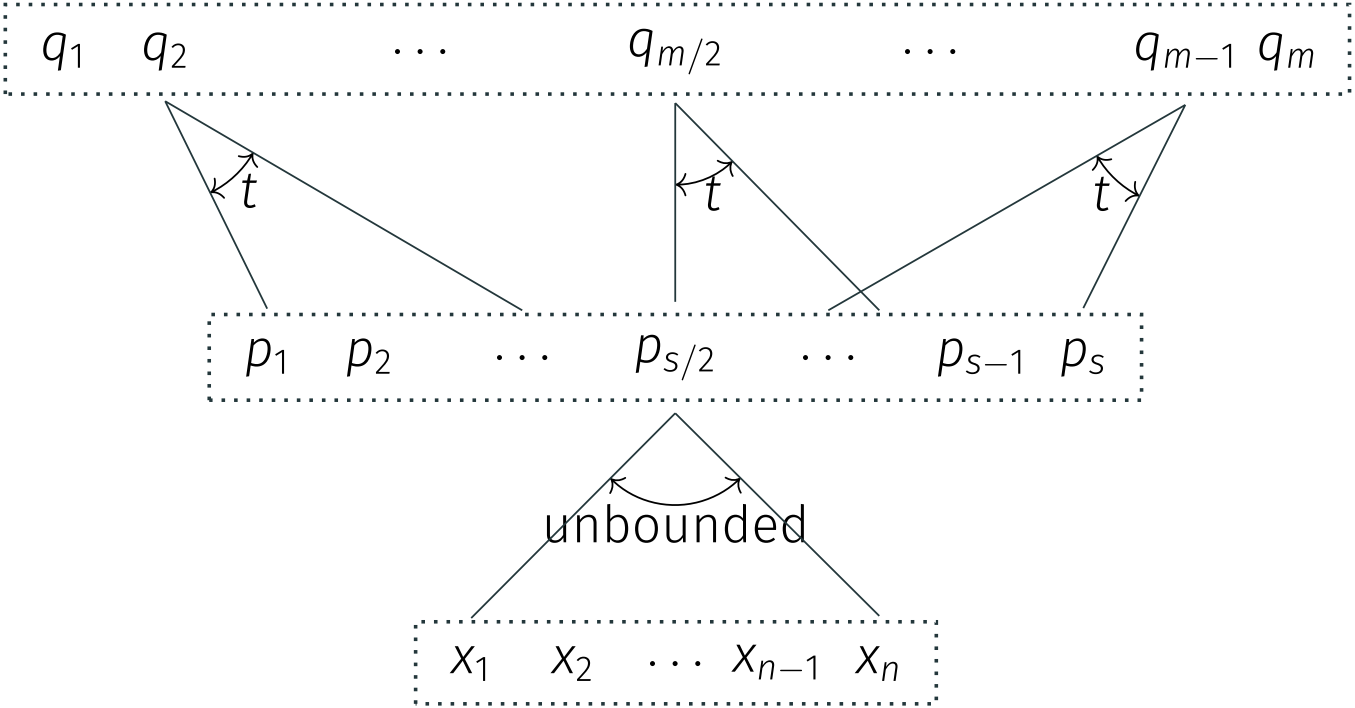


$p_1 \quad p_2 \quad \cdots \quad p_{s/2} \quad \cdots \quad p_{s-1} \quad p_s$

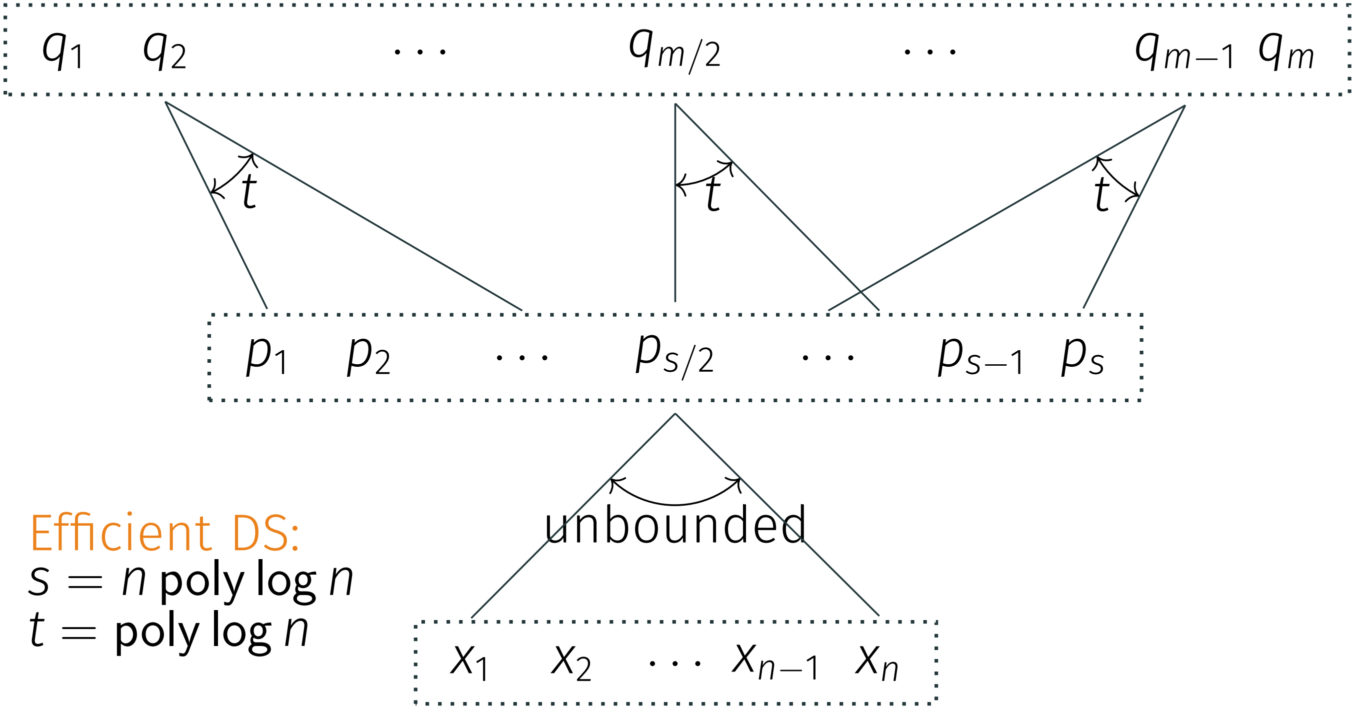
unbounded

$x_1 \quad x_2 \quad \cdots \quad x_{n-1} \quad x_n$

STATIC DATA STRUCTURES. DEFINITION



STATIC DATA STRUCTURES. DEFINITION



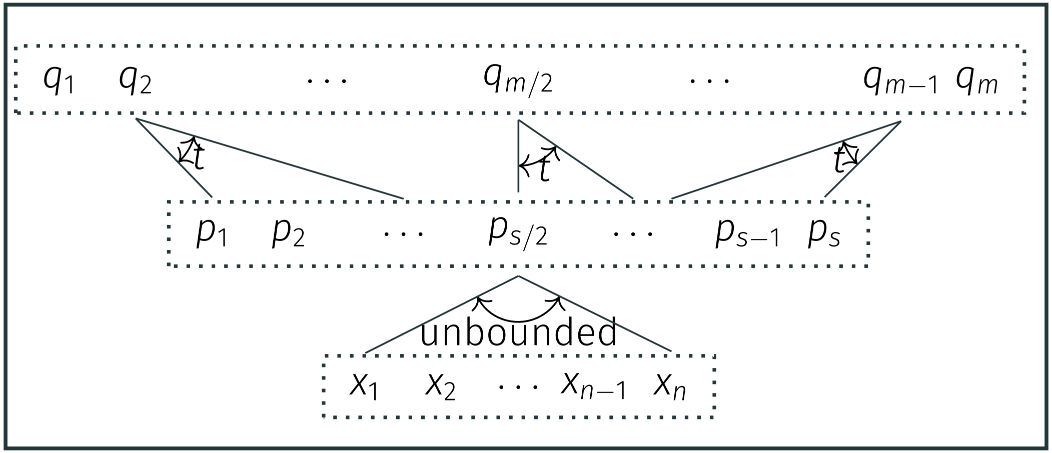
Efficient DS:

$s = n \operatorname{poly} \log n$

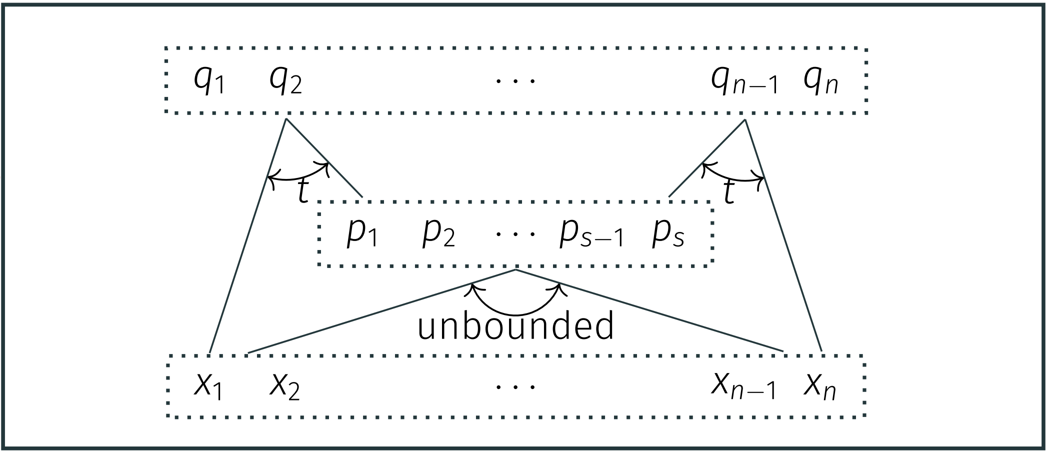
$t = \operatorname{poly} \log n$

COMPARISON

DS



Circuits



LINEAR CIRCUITS

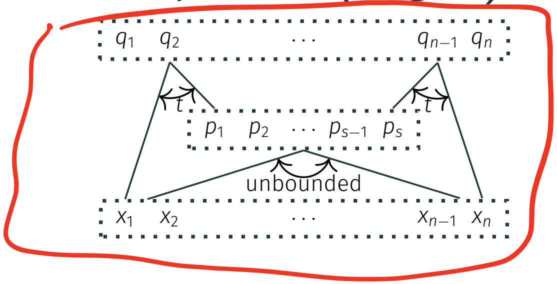
- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$

LINEAR CIRCUITS

- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,

LINEAR CIRCUITS

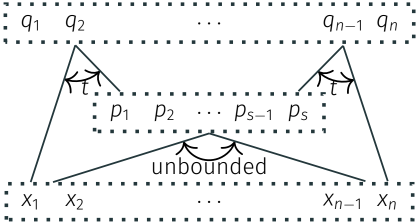
- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,



LINEAR CIRCUITS

- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,

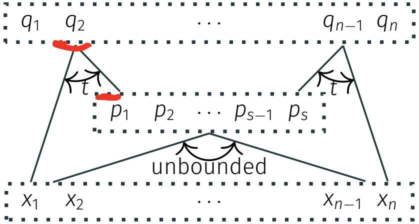
$$M = A + C \cdot D$$



LINEAR CIRCUITS

- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m\times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,

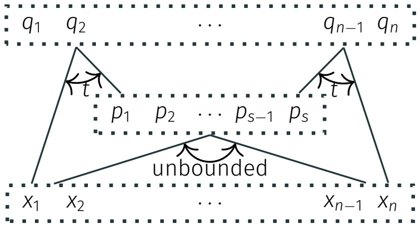
$$\overset{m \times n}{M} = \underbrace{\overset{m \times n}{A}}_{\text{outputs on inputs}} + \underline{\overset{m \times \varepsilon n}{C}} \cdot \overset{\varepsilon n \times n}{D}$$



LINEAR CIRCUITS

- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m\times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,

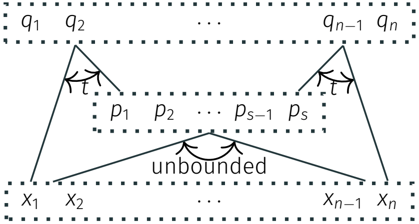
$$\overset{m \times n}{M} = \underbrace{\overset{m \times n}{A} + \overset{m \times \varepsilon n}{C}}_{\text{sparse}} \cdot \overset{\varepsilon n \times n}{D}$$



LINEAR CIRCUITS

- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m\times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,

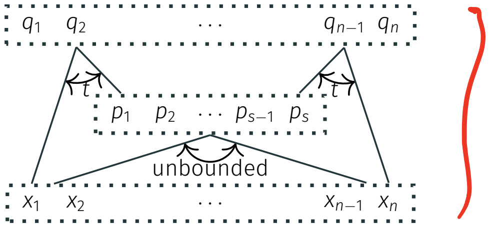
$$\overset{m \times n}{M} = \underbrace{\overset{m \times n}{A} + \overset{m \times \varepsilon n}{C}}_{\text{sparse}} \cdot \overset{\varepsilon n \times n}{D} = \underset{\text{sparse}}{A} + \underset{\text{low-rank}}{B}$$



LINEAR CIRCUITS

- A linear circuit computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$
- For a circuit of size $O(n)$ and depth $O(\log n)$,

$$\overset{m \times n}{M} = \underbrace{\overset{m \times n}{A} + \overset{m \times \varepsilon n}{C}}_{\text{sparse}} \cdot \overset{\varepsilon n \times n}{D} = \underset{\text{sparse}}{A} + \underset{\text{low-rank}}{B}$$



- $M \in \mathbb{F}^{m \times n}$ is $(\varepsilon n, t)$-rigid iff

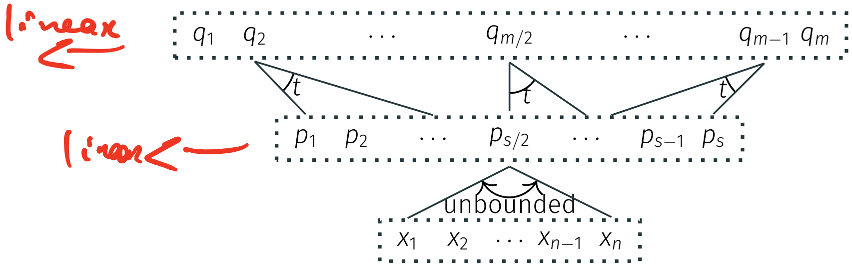
$$M \neq \underset{t\text{-sparse}}{A} + \underset{\mathbf{rk} \leq \varepsilon n}{B}$$

LINEAR DATA STRUCTURES

- A linear DS computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$

LINEAR DATA STRUCTURES

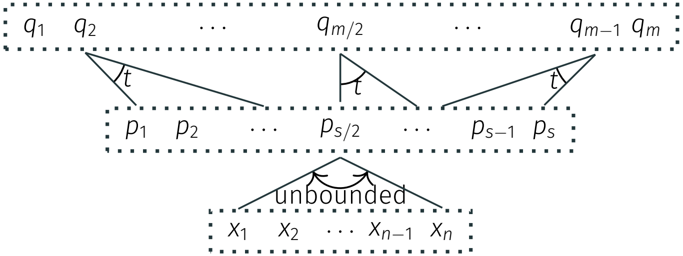
- A linear DS computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$



LINEAR DATA STRUCTURES

- A linear DS computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$

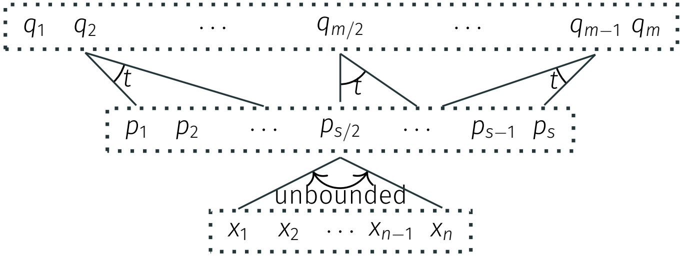
$$M = A \cdot B$$



LINEAR DATA STRUCTURES

- A linear DS computes Mx for input $x \in \mathbb{F}^n$ where $\boxed{M \in \mathbb{F}^{m\times n}}$

$$\overset{m \times n}{M} = \overset{\underline{m \times s}}{A} \cdot \overset{\underline{s \times n}}{B}$$

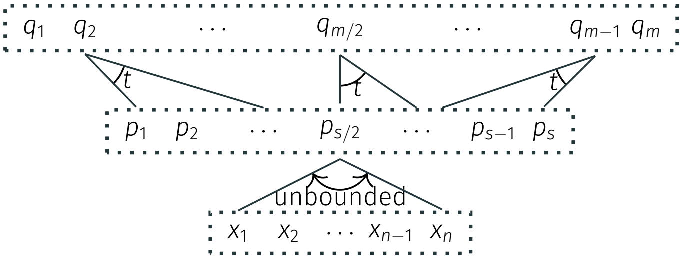


LINEAR DATA STRUCTURES

$m = n^2, n^{10}, n^{100}$

- A linear DS computes Mx for input $x \in \mathbb{F}^n$ where $M \in \mathbb{F}^{m \times n}$

$$\overset{m \times n}{M} = \underset{t\text{-sparse}}{\overset{m \times s}{A}} \cdot \underset{\text{small}}{\overset{s \times n}{B}}$$



$\underline{\underline{M \in \mathbb{F}^{n^{100} \times n}}}$

$\underline{\underline{B \in \mathbb{F}^{n \log n \times n}}}$

COMPARISON

Small circuit / Non-rigid

$$\overset{m \times n}{M} = \underset{t\text{-sparse}}{\overset{m \times n}{A}} + \underset{\mathbf{rk} \leq \varepsilon n}{\overset{m \times n}{B}}$$

COMPARISON

Small circuit / Non-rigid

$$\overset{m \times n}{M} = \underset{t\text{-sparse}}{\overset{m \times n}{A}} + \underset{\mathbf{rk} \le \varepsilon n}{\overset{m \times n}{B}}$$

R, S

Siva's talk

Efficient Data Structure

$$\overset{m \times n}{M} = \underset{t\text{-sparse}}{\overset{m \times s}{A}} \cdot \underset{\text{small}}{\overset{s \times n}{B}}$$

S, t

I'm looking for prospective PhD students who are interested in theory.

`alex.golovnev@gmail.com`

Thank you for your attention!