Monotone Circuit Lower Bounds from Resolution

Ankit Garg  MSR
Mika Göös  Harvard
Pritish Kamath  MIT
Dmitry Sokolov  KTH
Background:
Query-to-communication lifting
(topic of my PhD thesis)
Query vs. Communication

Decision trees

\[ f(z) \]

\[ z_1? \]
\[ z_5? \]
\[ z_2? \]

\[ z_7? \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ 1 \]

\[ 1 \]
\[ 0 \]

\[ 1 \]
\[ 0 \]

Communication protocols

\[ F(x,y) \]

\[ a_1(x)? \]
\[ b_1(y)? \]
\[ a_2(x)? \]

\[ b_2(y)? \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ 1 \]

\[ 1 \]
\[ 0 \]

\[ 1 \]
\[ 0 \]
Composed functions $f \circ g^n$

Examples:
- Set-disjointness: $\text{OR} \circ \text{AND}^n$
- Inner-product: $\text{XOR} \circ \text{AND}^n$
- Equality: $\text{AND} \circ \neg\text{XOR}^n$
Composed functions $f \circ g^n$

\[
\begin{array}{cccccc}
z_1 & z_2 & z_3 & z_4 & z_5 & \text{Compose with } g^n \\
\end{array}
\]

In general: $g: \{0,1\}^m \times \{0,1\}^m \rightarrow \{0,1\}$ is a small gadget

- Alice holds $x \in (\{0,1\}^m)^n$
- Bob holds $y \in (\{0,1\}^m)^n$
Composed functions $f \circ g^n$

Composed with $g^n$

Lifting Theorem Template:

$$M^{cc}(f \circ g^n) \approx M^{dt}(f)$$
Composed functions $f \circ g^n$

<table>
<thead>
<tr>
<th>$M$</th>
<th>Query</th>
<th>Communication</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>deterministic</td>
<td>deterministic</td>
<td>[RM99, GPW15, dRNV16, HHL16]</td>
</tr>
<tr>
<td>BPP</td>
<td>randomised</td>
<td>randomised</td>
<td>[GPW17, AGJ+17]</td>
</tr>
<tr>
<td>NP</td>
<td>nondeterministic</td>
<td>nondeterministic</td>
<td>[GLM+15, G15]</td>
</tr>
<tr>
<td></td>
<td>many poly degree</td>
<td>rank</td>
<td>[SZ09, She11, RS10, RPRC16]</td>
</tr>
<tr>
<td></td>
<td>many conical</td>
<td>nonnegative rank</td>
<td>[GLM+15, KMR17]</td>
</tr>
<tr>
<td></td>
<td>junta deg.</td>
<td>rectangle overlay</td>
<td>[GKPW17]</td>
</tr>
<tr>
<td>P$^\text{NP}$</td>
<td>decision list</td>
<td>LP complexity</td>
<td>[CLRS16, KMR17]</td>
</tr>
<tr>
<td></td>
<td>sum-of-squares</td>
<td>SDP complexity</td>
<td>[LRS15]</td>
</tr>
</tbody>
</table>

Lifting Theorem Template:

$$M^\text{cc}(f \circ g^n) \approx M^\text{dt}(f)$$
Example: Classical vs. Quantum
[ABK16, ABB+16, GPW17]

\[
\text{BPP}^{\text{dt}}(f) \geq \text{BQP}^{\text{dt}}(f)^{2.5}
\]

\[
\downarrow
\]

\[
\text{BPP}^{\text{cc}}(f \circ g^n) \geq \text{BQP}^{\text{cc}}(f \circ g^n)^{2.5}
\]
More lifting applications

1. Monotone circuit complexity
2. Lower bounds in proof complexity
3. Multiparty set-disjointness
4. Communication vs. partition numbers
5. Clique vs. independent set
6. Alon–Saks–Seymour in graph theory
7. LP and SDP extension complexity
8. Learning theory (sign rank)
9. Approximate Nash equilibria
More lifting applications

1. Monotone circuit complexity
2. Lower bounds in proof complexity
3. Multiparty set-disjointness
4. Communication vs. partition numbers
5. Clique vs. independent set
6. Alon–Saks–Seymour in graph theory
7. LP and SDP extension complexity
8. Learning theory (sign rank)
9. Approximate Nash equilibria
This work:
Monotone Circuit Lower Bounds from Resolution
Monotone circuit

Resolution refutation

Mon. feasible interpolation [BPR97, Kra97] ⇒ This work

Dag-like protocol
Resolution refutation
Dag-like query model
Monotone feasible interpolation [BPR97, Kra97]

Monotone circuit

Resolution refutation
Monotone feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]

Monotone circuit

Resolution refutation

This work

Mon. feasible interpolation
[BPR97, Kra97]
Monotone circuit
Dag-like protocol

Resolution refutation
Dag-like query model

Mon. feasible interpolation
[BPR97, Kra97]

⇒
This work
Search problems

▶ Monotone circuits

**mKW search problem** for monotone \( f : \{0, 1\}^n \rightarrow \{0, 1\} \)

- **input:** \((x, y) \in f^{-1}(1) \times f^{-1}(0)\)
- **output:** coordinate \(i \in [n]\) with \(x_i = 1, y_i = 0\)

▶ Proof systems

**CNF search problem** for unsatisfiable \( F = \bigwedge_i D_i \)

- **input:** truth assignment \( z \in \{0, 1\}^n \)
- **output:** clause \( D_i \) such that \( D_i(z) = 0 \)
Resolution refutation

Each dag node $v$ is labeled with a disjunction $D_v : \{0, 1\}^n \rightarrow \{0, 1\}$

- **root $r$:** $D_r \equiv 0$ (constant 0)
- **node $v$ with children $u, u'$:**
  \[
  D_v^{-1}(1) \supseteq D_u^{-1}(1) \cap D_{u'}^{-1}(1)
  \]
- **leaf $v$:** $D_v$ is an axiom
Top-down definition

Each dag node $v$ is labeled with a conjunction $C_v: \{0, 1\}^n \to \{0, 1\}$

- **root $r$:** $C_r \equiv 1$ (constant 1)

- **node $v$ with children $u, u'$:**
  $$C_v^{-1}(1) \subseteq C_u^{-1}(1) \cup C_{u'}^{-1}(1)$$

- **leaf $v$:** Labeled with solution to **CNF search problem** valid for all $C_v^{-1}(1)$
Dag models

Top-down definition

Each dag node \( v \) is labeled with a conjunction \( C_v : \{0, 1\}^n \rightarrow \{0, 1\} \)

- **root** \( r \): \( C_r \equiv 1 \) (constant 1)
- **node** \( v \) **with children** \( u, u' \):
  \[
  C_v^{-1}(1) \subseteq C_u^{-1}(1) \cup C_{u'}^{-1}(1)
  \]
  feasible set
- **leaf** \( v \): Labeled with solution to **CNF search problem** valid for all \( C_v^{-1}(1) \)
**Monotone circuits**

Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be monotone

Each dag node $v$ is labeled with a rectangle $R_v \subseteq f^{-1}(1) \times f^{-1}(0)$

- **root $r$:** $R_r = f^{-1}(1) \times f^{-1}(0)$
- **node $v$ with children $u, u'$:** $R_v \subseteq R_u \cup R_{u'}$
- **leaf $v$:** labeled with solution to mKW search problem valid for all $R_v$
Abstract $\mathcal{F}$-dags

Let $S \subseteq \mathcal{I} \times \mathcal{O}$ be a search problem

Each dag node $v$ is labeled with an $f_v: \mathcal{I} \to \{0, 1\}$ from family $\mathcal{F}$

- **root $r$:** $f_r \equiv 1$ (constant 1)

- **node $v$ with children $u, u'$:**
  $f_v^{-1}(1) \subseteq f_u^{-1}(1) \cup f_{u'}^{-1}(1)$

- **leaf $v$:** labeled with solution to $S$ valid for all $f_v^{-1}(1)$
## Summary

<table>
<thead>
<tr>
<th>Model</th>
<th>Family $\mathcal{F}$</th>
<th>Problem $S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract $\mathcal{F}$-dags</td>
<td>$\mathcal{F}$</td>
<td>any $S$</td>
</tr>
<tr>
<td>Resolution</td>
<td>conjunctions</td>
<td>CNF search</td>
</tr>
<tr>
<td>Monotone circuit</td>
<td>rectangles</td>
<td>mKW search</td>
</tr>
</tbody>
</table>
Our result

Setup

- $S \subseteq \{0, 1\}^n \times \mathcal{O}$ any query search problem
- $w(S)$ is the least width of conjunction-dag that solves $S$ (aka Resolution width)
- $g : [m] \times \{0, 1\}^m \rightarrow \{0, 1\}$ where $m = n^{O(1)}$ is two-party index function: $g(x, y) = y_x$
- $S \circ g^n$ is composed search problem
Our result

Setup

- $S \subseteq \{0, 1\}^n \times \mathcal{O}$ any query search problem
- $w(S)$ is the least width of conjunction-dag that solves $S$ (aka Resolution width)
- $g : [m] \times \{0, 1\}^m \rightarrow \{0, 1\}$ where $m = n^{O(1)}$
  - is two-party index function: $g(x, y) = y_x$
- $S \circ g^n$ is composed search problem

Result

Rectangle-dag complexity of $S \circ g^n$ is

$$n^{\Theta(w(S))}$$
Our result

**Bonus**

- **Triangle-dags** $\equiv$ **Monotone real circuits**
  
  \[ \text{[HC99, Pud97, HP17]} \]

- **LTF-dags** $\equiv$ **Cutting Planes refutations**

---

Rectangle

Triangle
Our result

**Bonus**

- Triangle-dags $\equiv$ Monotone real circuits
  
  \[\text{[HC99, Pud97, HP17]}\]

- LTF-dags $\equiv$ Cutting Planes refutations

**Result**

Triangle-dag complexity of $S \circ g^n$ is

$\mathcal{N}^{\Theta(w(S))}$
Our result

Upshot

1. Start with $n$-variable $k$-CNF $F$ of Resolution width $w$

2. **Apply result:** $S_F \circ g^n$ has triangle-dag complexity $n^{\Theta(w)}$

3. Interpret $S_F \circ g^n$ as mKW/CNF search problem:

   **mKW:** monotone function $f : \{0, 1\}^{n^{O(k)}} \rightarrow \{0, 1\}$
   with monotone circuit complexity $n^{\Theta(w)}$

   **CNF:** $n^{O(1)}$-variable $(k + O(1))$-CNF formula
   with Cutting Planes complexity $n^{\Theta(w)}$

*Previously: Clique [Pud97], random CNF [HP17, FPPR17]*
Our result

Upshot

Jukna’s 2012 textbook (Research Problem 19.17)

“It would be nice to have a lower bounds argument for cutting plane proofs *explicitly* showing what properties of contradictions do force long derivations.”

**mKW:** monotone function \( f : \{0, 1\}^{n^{O(k)}} \rightarrow \{0, 1\} \)
with monotone circuit complexity \( n^{\Theta(w)} \)

**CNF:** \( n^{O(1)} \)-variable \( (k + O(1)) \)-CNF formula
with Cutting Planes complexity \( n^{\Theta(w)} \)

*Previously:* Clique [Pud97], random CNF [HP17, FPPR17]
Tools from Prior Work

[GLMWZ15, GPW17]
Rectangles $\leftrightarrow$ conjunctions

Large rectangle $R \subseteq [m]^n \times \{0, 1\}^{mn}$ in the domain of $S \circ g^n$ can be partitioned into subrectangles

$$R = \bigcup_i R^i$$

such that $g^n(R^i) =$ large subcube in the domain of $S$

$R$ of density $2^{-d} \iff$ codimension-$d$ subcubes
Rectangles $\leftrightarrow$ conjunctions

Large rectangle $R \subseteq [m]^n \times \{0, 1\}^{mn}$ in the domain of $S \circ g^n$ can be partitioned into subrectangles

$$R = \bigcup_i R^i$$

such that $g^n(R^i)$ = large subcube in the domain of $S$

$R$ of density $2^{-d}$ $\iff$ codimension-$d$ subcubes
Game-Theoretic Characterisation of Resolution Width

[Pud00, AD08]
Let $S \subseteq \{0, 1\}^n \times \mathcal{O}$ be a search problem

- Game state is $\rho \in \{0, 1, \ast\}^n$, initially $\rho = \ast^n$
- In each round Explorer makes a move

  **Query:** Explorer chooses $i \in [n]
  \quad$ Adversary responds $b \in \{0, 1\}
  \quad$ Update $\rho_i \leftarrow b$

  **Forget:** Explorer chooses $i \in [n]
  \quad$ Update $\rho_i \leftarrow \ast$

- Game ends when solution to $S$ can be deduced for $\rho$
Explorer vs Adversary

Let $S \subseteq \{0, 1\}^n \times \mathcal{O}$ be a search problem

- Game state is $\rho \in \{0, 1, *\}^n$, initially $\rho = *^n$
- In each round Explorer makes a move
  - **Query:** Explorer chooses $i \in [n]$  
    Adversary responds $b \in \{0, 1\}$  
    Update $\rho_i \leftarrow b$
  - **Forget:** Explorer chooses $i \in [n]$  
    Update $\rho_i \leftarrow *$

- Game ends when solution to $S$ can be deduced for $\rho$

$$w(S) = \text{least } w \text{ such that Explorer has a strategy that maintains } \rho \text{ of width } \leq w$$
Proof outline:

Given size-$2^d$ rectangle-dag for $S \circ g^n$
extract width-$d$ Explorer-strategy for $S$
Proof outline

1. For each node $v$ of rectangle-dag, partition $R_v = \bigcup_i R_v^i$ where each subrectangle is $\rho$-like for $|\rho| \leq d$

   $g^n(R_v^i) = \text{strings consistent with } \rho$
Proof outline

1. For each node $v$ of rectangle-dag, partition $R_v = \bigcup_i R^i_v$ where each subrectangle is $\rho$-like for $|\rho| \leq d$

$$g^n(R^i_v) = \text{strings consistent with } \rho$$

2. Extract width-$d$ Explorer-strategy by walking down the rectangle-dag, starting at root

Invariant

At node $v$: Game state $\rho$, maintain $\rho$-like $R' \subseteq R_v$
1. Root

\[ R_{\text{root}} = \text{domain of } g^n \]

which is \( *^n \)-like

**Invariant**

At node \( v \): Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
2. Internal node

Crux!

**Invariant**

At node $v$: Game state $\rho$, maintain $\rho$-like $R' \subseteq R_v$
\[
\rho = \begin{array}{cccccccccccc}
0 & 1 & 1 & 0 & * & * & * & * & * & * & * & *
\end{array}
\]

**Invariant**

At node \(v\): Game state \(\rho\), maintain \(\rho\)-like \(R' \subseteq R_v\)
\[ \rho = 0 \ 1 \ 1 \ 0 \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \]

**Invariant**

At node \( v \): Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[
\rho = 0110 \ast \ast \ast \ast \ast \ast \ast \ast \ast \ast \\
\]

**Invariant**

**At node \(v\):** Game state \(\rho\), maintain \(\rho\)-like \(R' \subseteq R_v\)
\[ \rho = 0 \ 1 \ 1 \ 0 \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \ \ast \]

**Invariant**

**At node** \( v \): Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[ \rho = 0110**??***\]

**Invariant**

**At node \( v \):** Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[ \rho = 0 \quad 1 \quad 1 \quad 0 \quad * \quad * \quad * \quad 1 \quad * \quad * \quad ? \quad * \quad * \]

**Invariant**

**At node \( v \):** Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[ \rho = 0 \; 1 \; 1 \; 0 \; * \; * \; * \; * \; ? \; * \; 0 \; * \]

**Invariant**

At node \( v \): Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[ \rho = 0110**10?0^* \]

**Invariant**

**At node \( v \):** Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[ \rho = \begin{array}{cccccccc}
0 & 1 & 1 & 0 & * & * & * & 1 & 0 & 1 & 0 & * \\
\end{array} \]

**Invariant**

**At node \( v \):** Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
\[ \rho = * * * * * * * * 1 0 1 0 * \]

**Invariant**

**At node \( v \):** Game state \( \rho \), maintain \( \rho \)-like \( R' \subseteq R_v \)
3. Leaf

Leaf labeled with $o \in \mathcal{O}$
also valid for $\rho$

*Game ends!*

**Invariant**

| At node $v$:  | Game state $\rho$, maintain $\rho$-like $R' \subseteq R_v$ |
3.2 Simplified proof

To explain the basic idea, we first give a simplified version of the proof: We assume that all rectangles $R$ involved in $P$—call them the original rectangles—can be partitioned 

\begin{itemize}
  \item \textit{Assumption:} All subrectangles $R'$ in the resulting partition $P = \bigcup_i R'$ satisfy the “structured” case of Lemma 5 for $k = 2\log n$.
\end{itemize}

In Section 3.3 we remove this assumption by explaining how the proof can be modified to work in the presence of some error rows/columns.

\textbf{Overview.} We extract a width-$O(d)$ $\text{Explorer-strategy}$ for $S$ by walking down the rectangle-dag $\Pi$, starting at the root. For each original rectangle $R$ that is reached in the walk, we maintain a $\rho$-structured subrectangle $R' \subseteq R$ chosen from the 2-round partition of $R$. Note that $\rho$ will have width $O(d)$ by our choice of $k$. The intention is that $\rho$ will record the current state of the game. There are three issues to address: (1) Why is the starting condition of the game met? (2) How do we take a step from a node of $\Pi$ to one of its children? (3) Why are we done once we reach a leaf?

(1) Root case. At start, the root of $\Pi$ is associated with the original rectangle $R = [m]^n \times [n,1]^m$ comprising the whole domain. The 2-round partition of $R$ is trivial: it contains a single leaf, the $s^0$-structured $R$ itself. Hence we simply maintain the $s^0$-structured $R \subseteq R$, which meets the starting condition for the game.

(2) Internal step. This is the core of the argument: Suppose the game has reached state $\rho_R$ and we are maintaining some $\rho_R$-structured subrectangle $R' \subseteq R$ associated with an internal node $v$, we want to move to some $\rho_{R'}$-structured subrectangle $R' \subseteq L$ associated with a child of $v$. Moreover, we must keep the width of the game state at most $O(d)$ during this move.

Let the two original rectangles associated with the children of $v$ be $L$ and $L'$. Let $R' \subseteq L \times L'$ be at least one and $L \times L'$ has density $\geq 1/2$ inside $R$. Note that $R'$ satisfies the “structured” case of Lemma 5. Let $X' \times Y' = R' \cap L$ be the unique index such that $X' \times Y'$ is $\rho_{R'}$-structured. By Lemma 4 there exists some $s^k \times X'$ such that $s^k \times Y'$ is $\rho_{R'}$-like. Let the partition of $L$ according to the 2-round scheme be $L = [L_0, L_1] \times [Y', Y]$. Let $s^0$ be the unique index such that $s^0 \times X'$ is $\rho_{R'}$-structured. Recall that $X'$ is associated with some set of blocks $I' \subseteq [n]$ such that all parts of the form $X' \times i$ are $\rho_{R'}$-structured with $\fix R' = I'$. In particular, we have $|I'| \leq O(d)$.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig1}
\caption{Example of a 2-round partitioning scheme.}
\end{figure}

\begin{itemize}
  \item As Explorer, we now query the input bits in coordinates $j := I' \setminus \fix R'$ (in any order) obtaining some response string $s_j \subseteq [0,1]^n$ from the Adversary. As a result, the state of the game becomes the extension of $\rho_{R'}$ by $s_j$, call it $\rho''$, which has width $\|\fix R''\| = \|\fix R'' \cup I'\| \leq O(d)$.
\end{itemize}

3.5 Accounting for error

Next, we explain how to get rid of the assumption $(\ast)$ by accounting for the rows and columns that are classified as error in Lemma 5 for $k := 2\log n$. The partitioning of $\mathcal{R}$'s rectangles is done more carefully: We sort all original rectangles in reverse topological order $R_1, R_2, \ldots, R_n$ from leaves to root, that is, if $R_i$ is a descendant of $R_j$ then $R_i$ comes before $R_j$ in the order. Then we process the rectangles in this order:

**Initialization**
- Cumulative error-
  \begin{align*}
  X_{i=0}^n = Y_{i=0}^n \quad \text{and} \quad Y_{i=0}^n = \emptyset.
  \end{align*}

**Iterate**

- For $i = 1, 2, \ldots, n$ rounds:
  \begin{enumerate}
    \item Remove from $R_i$ the rows/columns $X_{i-1}^n, Y_{i-1}^n$.
    \item Suppose $X_{i-1}^n \prec Y_{i-1}^n$ is processed only after all of its descendants are partitioned. Each descendant may contribute some error rows/columns, accumulated into sets $X_{i-1}^n, Y_{i-1}^n$, which are deleted from $R_i$ before it is partitioned. The partitioning of $R_i$ will in turn contribute to its error rows/columns into its ancestors.
  \end{enumerate}

We may now repeat the proof of Section 3.2 using only the structured subrectangles output by the above process. We highlight two key properties that allow the proof to go through verbatim:

- First, the average width at the end of the process is tiny: $X_{i=0}^n, Y_{i=0}^n$ have density at most $\rho_{X_{i=0}^n, X_{i=0}^n} \leq 1/4$ by a union bound over all rounds. In particular, the root rectangle $R_0$ (with errors removed) still has density $\geq 1/2$ inside $[m]^n \times [0,1]^m$ and so it is $s^0$-structured. This allows us to meet the starting condition for the game.
- Second, by construction, the cumulative error sets grow as we walk from leaves towards the root. This means that our error handling does not interfere with the internal steps: each structured subrectangle $R' \subseteq R$ is covered by the structured subrectangles of $R$'s children.
Open problems
Open problems

Q1. Lifting for dags over intersections-of-k-triangles
   (Resolution over Cutting Planes)

- Rectangle
- Triangle
- Block-diagonal
- Intersection of 2 triangles
Open problems

Q1. Lifting for dags over \textit{intersections-of-k-triangles}
    (Resolution over Cutting Planes)

Q2. Lifting for \textit{nondeterministic} NOF protocols
    (Towards dag-like LBs for semi-algebraic proof systems)

Q3. Superlinear depth for small monotone circuits?
    (Razborov’16: “A New Kind of Tradeoff”)

Open problems

**Q1.** Lifting for dags over *intersections-of-k-triangles* (Resolution over Cutting Planes)

**Q2.** Lifting for *nondeterministic* NOF protocols (Towards dag-like LBs for semi-algebraic proof systems)

**Q3.** Superlinear depth for small monotone circuits? (Razborov’16: “A New Kind of Tradeoff”)

Cheers!