THEORY OF COMPUTING, Volume 16 (13), 2020, pp. 1–30
www.theoryofcomputing.org

Monotone Circuit Lower Bounds from Resolution

Ankit Garg   Mika Göös*   Pritish Kamath†   Dmitry Sokolov‡

Received July 26, 2018; Revised September 26, 2020; Published November 9, 2020

Abstract. For any unsatisfiable CNF formula F that is hard to refute in the *Resolution* proof system, we show that a gadget-composed version of F is hard to refute in any proof system whose lines are computed by efficient communication protocols—or, equivalently, that a monotone function associated with F has large monotone circuit complexity. Our result extends to monotone *real* circuits, which yields new lower bounds for the *Cutting Planes* proof system.

ACM Classification: F.1.3, F.2.3

AMS Classification: 68Q17, 68Q15, 03F20

Key words and phrases: circuit complexity, communication complexity, proof complexity

An extended abstract of this paper appeared in the Proceedings of the 50th Symposium on Theory of Computing (STOC'18) [16].

*Supported by Michael O. Rabin Postdoctoral Fellowship.

†Supported in parts by NSF grants CCF-1650733, CCF-1733808, and IIS-1741137.

‡Supported by the Swedish Research Council grant 2016-00782, Knut and Alice Wallenberg Foundation grants KAW 2016.0066 and KAW 2016.0433.

 DOI: 10.4086/toc.2020.v016a013

1 Appetizer

DAG-*like* communication protocols [46, 40, 52], generalizing the usual notion of *tree-like* communication protocols [35, 31, 41], provide a useful abstraction to study two kinds of objects in complexity theory:

- **Monotone circuits.** Let f be a monotone boolean function. The *monotone circuit complexity* of f can be characterized in the language of DAG-like protocols. Namely, it equals the least size of a DAG-like protocol that solves the *monotone Karchmer–Wigderson (mKW)* search problem for f.
- **Propositional proofs.** Let F be a CNF contradiction (an unsatisfiable CNF formula). Lower bounds for the *Resolution refutation length complexity* of F—or indeed lower bounds for any propositional proof system whose lines are computed by efficient communication protocols—can be proved via DAG-like protocols. Namely, a lower bound is given by the least size of a DAG-like protocol that solves a certain CNF search problem associated with F.

In this paper, we prove a ***query-to-communication lifting theorem*** that escalates lower bounds for a DAG-like query model (essentially Resolution) to lower bounds for DAG-like communication protocols. In particular, this yields a new technique to prove size lower bounds for monotone circuits and several types of proof systems (including Cutting Planes).

The result can be interpreted as a ***converse*** to *monotone feasible interpolation* [10, 33], which is a popular method to prove refutation size lower bounds for proof systems (such as Resolution and Cutting Planes) by reductions to monotone circuit lower bounds. A theorem of this type was conjectured by Beame, Huynh, and Pitassi [5, §6]. We also note that lifting theory for deterministic *tree-like* protocols—with applications to monotone *formula* size, *tree-like* refutation size, and size–space tradeoffs—has been developed in quite some detail [42, 28, 20, 21, 13, 56, 11]. We import techniques from this line of work into the DAG-like setting.

A follow-up article [18] has obtained several concrete applications using our technique: an exponential monotone circuit lower bound for XOR-SAT, and a separation showing that the *Nullstellensatz* proof system can be exponentially more powerful than Cutting Planes.

We formalize our result in Section 3 after we have defined our DAG-like models in Section 2.

2 DAG-like models

We define all computational models as solving *search problems*, defined by a relation $S \subseteq \mathcal{I} \times \mathcal{O}$ for some finite input and output sets $\mathcal{I}$ and $\mathcal{O}$. On input $x \in \mathcal{I}$ the search problem is to find some output in $S(x) := \{o \in \mathcal{O} : (x,o) \in S\}$. We always assume S is *total* so that $S(x) \neq \emptyset$ for all $x \in \mathcal{I}$. We also define $S^{-1}(o) := \{x \in \mathcal{I} : (x,o) \in S\}$. For applications, the two most important examples of search problems, one associated with a monotone function $f\colon \{0,1\}^n \to \{0,1\}$, another with an n-variable CNF contradiction $F = \bigwedge_i D_i$ (where D_i are disjunctions of literals), are as follows.

mKW search problem S_f	$=$	*input:*	a pair $(x,y) \in f^{-1}(1) \times f^{-1}(0)$
		output:	a coordinate $i \in [n]$ such that $x_i > y_i$
CNF search problem S_F	$=$	*input:*	an n-variable truth assignment $z \in \{0,1\}^n$
		output:	clause D of F unsatisfied by z, i. e., $D(z) = 0$

2.1 Abstract DAGs

We work with a *top-down* definition of DAG-like models. A version of the following definition (with a specialized $\mathcal{F}$) was introduced by [46] and subsequently simplified in [40, 52].

Top-down definition. Let $\mathcal{F}$ be a family of functions $\mathcal{I} \to \{0,1\}$. An $\mathcal{F}$-DAG solving $S \subseteq \mathcal{I} \times \mathcal{O}$ is a directed acyclic graph of fan-out ≤ 2 where each node v is associated with a function $f_v \in \mathcal{F}$ (we call $f_v^{-1}(1)$ the *feasible set* for v) satisfying the following conditions.

1. *Root:* There is a distinguished root node r (fan-in 0), and $f_r \equiv 1$ is the constant 1 function.
2. *Non-leaves:* For each non-leaf node v with children u,u', we have $f_v^{-1}(1) \subseteq f_u^{-1}(1) \cup f_{u'}^{-1}(1)$.
3. *Leaves:* Each leaf node v is labeled with an output $o_v \in \mathcal{O}$ such that $f_v^{-1}(1) \subseteq S^{-1}(o_v)$.

The *size* of an $\mathcal{F}$-DAGis its number of nodes. If we specialize S to be a CNF search problem S_F, the above specializes to the familiar definition of refutations in a proof system whose lines are *negations* of functions in $\mathcal{F}$. Here is that dual definition, specialized to $S = S_F$.

Bottom-up definition. Let $\mathcal{G}$ be a family of functions $\{0,1\}^n \to \{0,1\}$. (To match up with the top-down definition, one should take $\mathcal{G} := \{\neg f : f \in \mathcal{F}\}$.) A (semantic) $\mathcal{G}$-*refutation* of an n-variable CNF contradiction F is a directed acyclic graph of fan-out ≤ 2 where each node (or *line*) v is associated with a function $g_v \in \mathcal{G}$ satisfying the following conditions.

1. *Root:* There is a distinguished root node r (fan-in 0), and $g_r \equiv 0$ is the constant 0 function.
2. *Non-leaves:* For each non-leaf node v with children u,u', we have $g_v^{-1}(1) \supseteq g_u^{-1}(1) \cap g_{u'}^{-1}(1)$.
3. *Leaves:* Each leaf node v is labeled with a clause D of F such that $g_v^{-1}(1) \supseteq D^{-1}(1)$.

2.2 Concrete DAGs

We now instantiate the abstract model for the purposes of communication and query complexity.

Rectangle-DAGs (DAG-like protocols). Consider a bipartite input domain $\mathcal{I} := \mathcal{X} \times \mathcal{Y}$ so that Alice holds $x \in \mathcal{X}$, Bob holds $y \in \mathcal{Y}$, and let $\mathcal{F}$ be the set of all indicator functions of *(combinatorial) rectangles over* $\mathcal{X} \times \mathcal{Y}$ (sets of the form $X \times Y$ with $X \subseteq \mathcal{X}$, $Y \subseteq \mathcal{Y}$). Call such $\mathcal{F}$-DAGs simply *rectangle*-DAGs. For a search problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ we define its *rectangle*-DAG *complexity* by

$$\text{rect-DAG}(S) := \text{least } \textit{size} \text{ of a rectangle-DAG that solves } S.$$

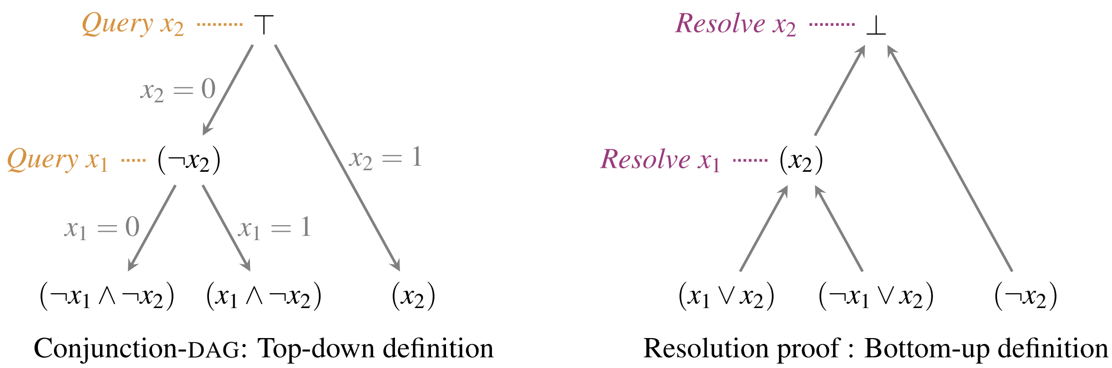


Figure 1: Two equivalent ways to view a Resolution refutation, illustrated in the tree-like case (see [31, §18.2] for more discussion of the tree-like case).

In circuit complexity, a straightforward generalization of the Karchmer–Wigderson depth characterization [32] shows that the monotone circuit complexity of any monotone function f equals rect-DAG(S_f); see [40, 52].

In proof complexity, a useful-to-study semantic proof system is captured by $\mathcal{F}_c$-DAGs *solving CNF search problems* S_F where $\mathcal{F}_c$ is the family of all functions $\mathcal{X} \times \mathcal{Y} \to \{0,1\}$ (where $\mathcal{X} \times \mathcal{Y} = \{0,1\}^n$ corresponds to a bipartition of the n input variables of S_F) that can be computed by tree-like protocols of communication cost c, say for $c = \mathrm{polylog}(n)$. Such a proof system can simulate other systems (such as Resolution and Cutting Planes with bounded coefficients), and hence lower bounds against $\mathcal{F}_c$-DAGs imply lower bounds for other concrete proof systems. Moreover, any $\mathcal{F}_c$-DAG can be simulated by a rectangle-DAG with at most a factor 2^c blow-up in size, and hence we do not lose much generality by studying only rectangle-DAGs.

Conjunction-DAGs (essentially Resolution). Consider the n-bit input domain $\mathcal{I} := \{0,1\}^n$ and let $\mathcal{F}$ be the set of all *conjunctions* of literals over the n input variables. Call such $\mathcal{F}$-DAGs simply *conjunction-DAGs*. We define the *width* of a conjunction-DAG Π as the maximum width of a conjunction associated with a node of Π. For a search problem $S \subseteq \{0,1\}^n \times \mathcal{O}$ we define

$$\begin{aligned} \mathsf{conj\text{-}DAG}(S) &:= \text{ least } \textit{size} \text{ of a conjunction-DAG that solves } S, \\ w(S) &:= \text{ least } \textit{width} \text{ of a conjunction-DAG that solves } S. \end{aligned}$$

In the context of CNF search problems $S = S_F$, conjunction-DAGs are equivalent to Resolution refutations; see also Figure 1. Indeed, conj-DAG(S_F) is just the Resolution refutation length complexity of F, and $w(S_F)$ is the Resolution width complexity of F [8].

The complexity measures introduced so far are related as follows; here S' is *any* two-party version of S obtained by choosing some bipartition $\mathcal{X} \times \mathcal{Y} = \{0,1\}^n$ of the input domain of S.

$$\mathsf{rect\text{-}DAG}(S') \le \mathsf{conj\text{-}DAG}(S) \le n^{O(w(S))}. \tag{2.1}$$

The first inequality holds because each conjunction can be simulated by a rectangle. The second inequality holds since there are at most $n^{O(w)}$ many distinct width-w conjunctions, and we may assume w.l.o.g. that any $f \in \mathcal{F}$ is associated with at most one node in an $\mathcal{F}$-DAG (any incoming edge to a node v can be rewired to the *lowest* node u, in topological order, such that $f_v = f_u$).

3 Our results

Our first theorem is a characterization of the rectangle-DAG complexity for *composed* search problems of the form $S \circ g^n$. Here $S \subseteq \{0,1\}^n \times \mathcal{O}$ is an arbitrary n-bit search problem, and $g\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ is some carefully chosen two-party *gadget* that helps to distribute each input variable of S between the two parties. More precisely, $S \circ g^n \subseteq \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{O}$ is the search problem where Alice holds $x \in \mathcal{X}^n$, Bob holds $y \in \mathcal{Y}^n$, and their goal is to find some $o \in S(z)$ for $z := g^n(x,y) = (g(x_1,y_1),\ldots,g(x_n,y_n))$.

Our concrete choice for a gadget is the usual m-bit *index* function $\text{IND}_m\colon [m] \times \{0,1\}^m \to \{0,1\}$ mapping $(x,y) \mapsto y_x$. For large enough m, we show that the bounds (2.1) are tight.

Theorem 3.1. *Let $m = m(n) := n^{\Delta}$ for a large enough constant Δ. For any $S \subseteq \{0,1\}^n \times \mathcal{O}$,*

$$\mathsf{rect}\text{-DAG}(S \circ \text{IND}_m^n) \;=\; n^{\Theta(w(S))}.$$

We note that the conjunction-DAG width complexity of $S \circ \text{IND}_m^n$ depends on how Alice's gadget inputs $x_i \in [m]$ are encoded as binary variables. For example, we can have $w(S \circ \text{IND}_m^n) = \Theta(w(S))$ when using a "unary" encoding; see Section 8 for a discussion.

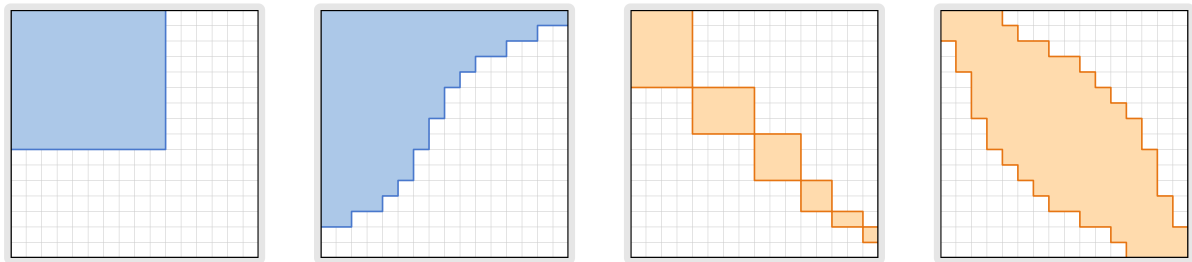
Implications. The primary advantage of such a lifting theorem is that we obtain, in a generic fashion, a large class of hard (explicit) monotone functions and CNF contradictions. Let us outline how to apply our theorem. We can start with any n-variable k-CNF contradiction F of Resolution width w, and conclude from Theorem 3.1 that the composed problem $S' := S_F \circ \text{IND}_m^n$ has rectangle-DAG complexity $n^{\Theta(w)}$. Then we can use reductions (either new or known; see Section 8 for known ones) to translate S' back to a mKW/CNF search problem. The upshot will be the following.

- S' reduces to $S_{f'}$ where f' is some N-bit monotone function with $N := n^{O(k)}$.
- S' reduces to $S_{F'}$ where F' is some $n^{O(1)}$-variable $2k$-CNF contradiction.

A follow-up article [18] has provided concrete applications using a novel reduction framework based on the above template. For example, they consider a monotone function $3\text{XOR-SAT}_n\colon \{0,1\}^N \to \{0,1\}$ over $N := 2n^3$ input bits defined as follows. An input $x \in \{0,1\}^N$ is interpreted as (the indicator vector of) a set of 3XOR constraints over n boolean variables $v_1,\ldots,v_n$ (there are N possible constraints). We define $3\text{XOR-SAT}_n(x) := 1$ iff the set x is *unsatisfiable*, that is, no boolean assignment to the v_i exists that satisfies all constraints in x. They proceed to show that if F is an n-variable "Tseitin" contradiction (which is hard for Resolution [54]), then $S' = S_F \circ \text{IND}_m^n$ reduces to $S_{3\text{XOR-SAT}_{mn}}$. Combining this with Theorem 3.1, one obtains the following.

Corollary 3.2 ([18, Thm. 1])**.** 3XOR-SAT_n *requires monotone circuits of size* $2^{n^{\Omega(1)}}$.

Since 3XOR-SAT_n is in NC^2 [37], this improves on the exponential monotone vs. non-monotone separation due to Tardos [53]; her function is in P and not known to be in NC.



(a) (b) (c) (d)

Figure 2: We show lifting theorems for DAGs whose feasible sets are (a) *rectangles* or (b) *triangles*. It remains open (see Section 10) to prove any lower bounds for explicit mKW/CNF search problems when the feasible sets are (c) *block-diagonal*, which a special case of (d) *intersections of 2 triangles*.

Limitations. A disadvantage, stemming from the large gadget size $m = n^{\Delta}$, is that we get at best (using $w = \Theta(n)$) a monotone circuit lower bound of $\exp(N^{\varepsilon})$ for a small constant $\varepsilon \geq 1/(\Delta+1)$. Such lower bounds fall short of the current best record of $\exp(N^{1/3-o(1)})$ due to Harnik and Raz [25]. We inherit the need for large gadgets from prior work [19, 22]; see Section 4. For this reason (and others), it is an important open problem to develop a lifting theory for gadgets of size $m = O(1)$. In particular, an optimal $2^{\Omega(N)}$ lower bound would follow from an appropriate constant-size-gadget version of Theorem 3.1; see Section 8 for details.

Techniques. We use tools developed in the context of tree-like lifting theorems, specifically from [19, 22]. These tools allow us to relate large rectangles in the input domain of $S \circ \text{IND}^n_m$ with large subcubes in the input domain of S; see Section 4. Given these tools, the proof of Theorem 3.1 is relatively short (two pages). The proof is extremely direct: from any rectangle-DAG of size n^d solving $S \circ \text{IND}^n_m$ we extract a width-$O(d)$ conjunction-DAG solving S.

Classical work on monotone circuit lower bounds has typically focused on specific monotone functions [44, 3, 1, 23, 50] and more generally on studying the power of the underlying proof methods [45, 55, 47, 51, 9, 2]. A notable exception is Jukna's criterion [30], recently applied in [26, 14], which is a general sufficient condition for a monotone function to require large monotone circuit complexity. Our perspective is seemingly even more abstract, as our result is phrased for arbitrary search problems (not just of mKW/CNF type). However, it remains unclear exactly how the power of our methods compare with the classical techniques; for example, can our result be rephrased in the language of Razborov's method of approximations? (An anonymous reviewer thinks this is possible, but not instructive.)

3.1 Extension: Monotone real circuits

Triangle-DAGs. Consider a bipartite input domain $\mathcal{I} := \mathcal{X} \times \mathcal{Y}$ and let $\mathcal{F}$ be the set of all indicator functions of *(combinatorial) triangles over* $\mathcal{X} \times \mathcal{Y}$; here a *triangle* $T \subseteq \mathcal{X} \times \mathcal{Y}$ is a set that can be written as $T = \{(x,y) \in \mathcal{X} \times \mathcal{Y} : a_T(x) < b_T(y)\}$ for some labeling of the rows $a_T : \mathcal{X} \to \mathbb{R}$ and columns $b_T : \mathcal{Y} \to \mathbb{R}$ by real numbers; see Figure 2b. In particular, every rectangle is a triangle. Call such $\mathcal{F}$-DAGs simply *triangle*-DAGs. For a search problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ we define

$$\text{tri-DAG}(S) \;:=\; \text{least } \textit{size} \text{ of a triangle-DAG that solves } S.$$

Hrubeš and Pudlák [27] showed recently that the *monotone real circuit complexity* of an f equals $\text{tri-DAG}(S_f)$. Monotone real circuits [24, 38] generalize monotone circuits by allowing the wires to carry arbitrary real numbers and the binary gates to compute arbitrary monotone functions $\mathbb{R} \times \mathbb{R} \to \mathbb{R}$. The original motivation to study such circuits, and what interests us here, is that lower bounds for monotone real circuits imply lower bounds for the *Cutting Planes* proof system [12]. In our language, semantic Cutting Planes refutations are equivalent to $\mathcal{L}$-DAGs solving CNF search problems, where $\mathcal{L}$ is the family of linear threshold functions (each $f \in \mathcal{L}$ is defined by some $(n+1)$-tuple $a \in \mathbb{R}^{n+1}$ so that $f(x) = 1$ iff $\sum_{i \in [n]} a_i x_i > a_{n+1}$).

Our second theorem states that Theorem 3.1 holds more generally with rectangle-DAGs replaced with triangle-DAGs. The proof is however more involved than the proof for Theorem 3.1.

Theorem 3.3. *Let $m = m(n) := n^{\Delta}$ for a large enough constant Δ. For any $S \subseteq \{0,1\}^n \times \mathcal{O}$,*

$$\text{tri-DAG}(S \circ \text{IND}_m^n) \;=\; n^{\Theta(w(S))}.$$

A pithy corollary is that if we start with any k-CNF contradiction F that is hard for Resolution and compose F with a gadget (as described in Section 8), the formula becomes hard for Cutting Planes. In particular, the composed formula can itself be written as a $2k$-CNF.

Corollary 3.4. *For any unsatisfiable k-CNF F on n variables, there is a related unsatisfiable $2k$-CNF F' on $n^{O(1)}$ variables, such that any Cutting Planes refutation for F' has length at least $n^{\Omega(w(S_F))}$.*

The follow-up article [18] observed a near-immediate corollary: the Nullstellensatz proof system (over any field) can be exponentially more powerful than Cutting Planes.

Corollary 3.5 ([18, §4.2])**.** *There exists an n-variable, $n^{O(1)}$-clause CNF contradiction F that can be refuted by Nullstellensatz (over any field) in degree $O(\log n)$, but that requires Cutting Planes refutations of length $2^{n^{\Omega(1)}}$.*

Previously, only few examples of hard contradictions were known for Cutting Planes, all proved via feasible interpolation [38, 24, 26, 14]. A widely-asked question has been to improve this state-of-the-art by developing alternative lower bound methods; see the surveys [6, §4] and [49, §5]. In particular, Jukna [31, Research Problem 19.17] asked to find a more intuitive "combinatorial" proof method "explicitly showing what properties of [contradictions] force long derivations." While our method does implicitly use feasible interpolation for Cutting Planes, at least it does afford a simple combinatorial intuition: the hardness is simply borrowed from the realm of Resolution (where we understand very well what makes formulas hard).

4 Subcubes from rectangles

In this section, as preparation, we recall some technical notions from [19, 22] concerning the index gadget $g := \textsc{Ind}_m$. Specifically, writing $G := g^n \colon [m]^n \times \{0,1\}^{mn} \to \{0,1\}^n$ for n copies of g, we explain how large rectangles in the domain of G are related with large subcubes in the codomain of G. In what follows, we will always assume that $m \geq n^{\Delta}$ for a sufficiently large constant Δ.

4.1 Structured rectangles

For a partial assignment $\rho \in \{0,1,*\}^n$ we let $\operatorname{free}\rho := \rho^{-1}(*)$ denote its *free* coordinates, and $\operatorname{fix}\rho := [n] \smallsetminus \operatorname{free}\rho$ denote its *fixed* coordinates. The number of fixed coordinates $|\operatorname{fix}\rho|$ is the *width* of ρ. Width-d partial assignments are naturally in 1-to-1 correspondence with width-d conjunctions: for any ρ we define $C_\rho \colon \{0,1\}^n \to \{0,1\}$ as the width-$|\operatorname{fix}\rho|$ conjunction that accepts an $x \in \{0,1\}^n$ iff x is consistent with ρ. Thus $C_\rho^{-1}(1) = \{x \in \{0,1\}^n : x_i = \rho_i \text{ for all } i \in \operatorname{fix}\rho\}$ is a subcube. We say that $R \subseteq [m]^n \times \{0,1\}^{mn}$ is ρ*-like* if the image of R under G is precisely the subcube of n-bit strings consistent with ρ, that is,

$$R \text{ is } \rho\text{-}like \quad \Longleftrightarrow \quad G(R) \;=\; C_\rho^{-1}(1).$$

For a random variable $\boldsymbol{x}$ we let $\mathbf{H}_\infty(\boldsymbol{x}) := \min_x \log(1/\mathbf{Pr}[\boldsymbol{x} = x])$ denote the usual *min-entropy* of $\boldsymbol{x}$. When $\boldsymbol{x} \in [m]^J$ for some index set J, we write $\boldsymbol{x}_I \in [m]^I$ for the marginal distribution of $\boldsymbol{x}$ on a subset $I \subseteq J$ of coordinates. For a set X we use the boldface $\boldsymbol{X}$ to denote a random variable uniformly distributed over X.

Definition 4.1 ([19])**.** A random variable $\boldsymbol{x} \in [m]^J$ is δ*-dense* if for every nonempty $I \subseteq J$, $\boldsymbol{x}_I$ has *min-entropy rate* $\geq \delta$, that is, $\mathbf{H}_\infty(\boldsymbol{x}_I) \geq \delta \cdot |I| \log m$.

Definition 4.2 ([17, 22])**.** A rectangle $R := X \times Y \subseteq [m]^n \times \{0,1\}^{mn}$ is ρ*-structured* if

1. $\boldsymbol{X}_{\operatorname{fix}\rho}$ is fixed, and every $z \in G(R)$ is consistent with ρ, that is, $G(R) \subseteq C_\rho^{-1}(1)$;
2. $\boldsymbol{X}_{\operatorname{free}\rho}$ is 0.9*-dense*;
3. Y is large enough: $\mathbf{H}_\infty(\boldsymbol{Y}) \geq mn - n^3$.

Lemma 4.3 ([17, 22])**.** *For $m \geq n^{\Delta}$, every ρ-structured rectangle is ρ-like.*

In this article we need a slight strengthening of Lemma 4.3: for a ρ-structured R, there is a *single row* of R that is already ρ-like. The proof of the following lemma is defered to Section 9.

Lemma 4.4. *Let $X \times Y$ be ρ-structured. For $m \geq n^{\Delta}$, there exists $x \in X$ such that $\{x\} \times Y$ is ρ-like.*

We remark that the *only* reason why our proofs require $m \geq n^{\Delta}$ is due to Lemma 4.4.

4.2 Rectangle partition scheme

We claim that, given any rectangle $R := X \times Y \subseteq [m]^n \times \{0,1\}^{mn}$, we can partition most of $X \times Y$ into ρ-structured subrectangles with $|\operatorname{fix}\rho|$ bounded in terms of the size of $X \times Y$. Indeed, we describe a simple 2-round partitioning scheme from [22] below; see also Figure 3. In the 1st round of the algorithm,

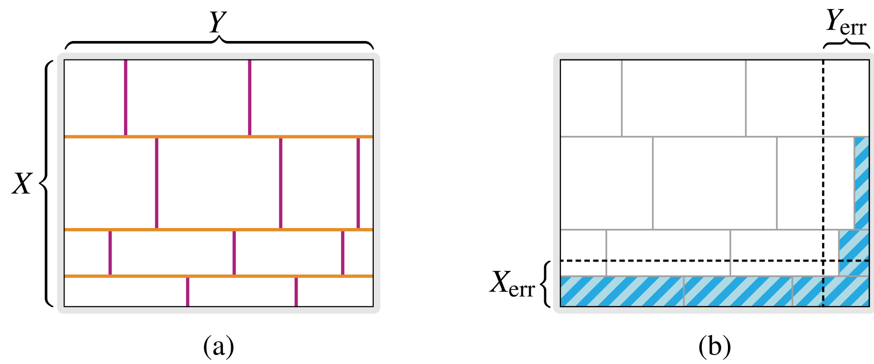


Figure 3: (a) Rectangle Scheme partitions $R = X \times Y$ first along rows, then along columns. (b) Lemma 4.5 illustrated: most subrectangles are ρ-structured for low-width ρ, except some error parts (highlighted in figure) that are contained in few error rows/columns X_{err}, Y_{err}.

Rectangle Scheme

Input: $R = X \times Y \subseteq [m]^n \times \{0,1\}^{mn}$.
Output: A partition of R into subrectangles.

1: **1st round:** Iterate the following for $i = 1, 2, \ldots$, until X becomes empty:
   (i) Let $I_i \subseteq [n]$ be a *maximal* subset (possibly $I_i = \emptyset$) such that $\boldsymbol{X}_{I_i}$ has min-entropy rate < 0.95, and let $\alpha_i \in [m]^{I_i}$ be an outcome witnessing this: $\mathbf{Pr}[\boldsymbol{X}_{I_i} = \alpha_i] > m^{-0.95|I_i|}$
   (ii) Define $X^i := \{x \in X : x_{I_i} = \alpha_i\}$
   (iii) Update $X \leftarrow X \smallsetminus X^i$

2: **2nd round:** For each part X^i and $\gamma \in \{0,1\}^{I_i}$, define $Y^{i,\gamma} := \{y \in Y : g^{I_i}(\alpha_i, y_{I_i}) = \gamma\}$

3: **return** $\left\{R^{i,\gamma} := X^i \times Y^{i,\gamma} : Y^{i,\gamma} \neq \emptyset\right\}$

we partition the rows as $X = \bigsqcup_i X^i$ where each X^i will be fixed on some blocks $I_i \subseteq [n]$ and 0.95-dense on the remaining blocks $[n] \smallsetminus I_i$. In the 2nd round, each $X^i \times Y$ is further partitioned along columns so as to fix the outputs of the gadgets on coordinates I_i.

All the properties of Rectangle Scheme that we will subsequently need are formalized below; see also Figure 3. For terminology, given a subset $A' \subseteq A$ we define its *density* (inside A) as $|A'|/|A|$. The proof of the following lemma is postponed to Section 7.

Lemma 4.5 (Rectangle Lemma)**.** Fix any parameter $k \leq n \log n$. Given a rectangle $R \subseteq [m]^n \times \{0,1\}^{mn}$, let $R = \bigsqcup_i R^i$ be the output of Rectangle Scheme. Then there exist "error" sets $X_{\text{err}} \subseteq [m]^n$ and $Y_{\text{err}} \subseteq \{0,1\}^{mn}$, both of density $\leq 2^{-k}$, such that for each i, one of the following holds:

- ***Structured case:*** R^i is ρ^i-structured for some ρ^i of width at most $O(k/\log n)$.
- ***Error case:*** R^i is covered by error rows/columns, i. e., $R^i \subseteq X_{\text{err}} \times \{0,1\}^{mn} \cup [m]^n \times Y_{\text{err}}$.

Finally, a ***query alignment*** property holds: for every $x \in [m]^n \smallsetminus X_{\text{err}}$, there exists a subset $I_x \subseteq [n]$ with $|I_x| \leq O(k/\log n)$ such that every "structured" R^i intersecting $\{x\} \times \{0,1\}^{mn}$ has fix $\rho^i \subseteq I_x$.

5 Lifting for rectangle-DAGs

In this section we prove the nontrivial direction of Theorem 3.1: Let Π be a rectangle-DAG solving $S \circ G$ of size n^d for some d. Our goal is to show that $w(S) \leq O(d)$.

5.1 Game semantics for DAGs

For convenience (and fun), we use the language of two-player competitive games, introduced in [39, 4], which provide an alternative way of thinking about conjunction-DAGs solving $S \subseteq \{0,1\}^n \times \mathcal{O}$. The game involves two competing players, *Explorer* and *Adversary*, and proceeds in rounds. The state of the game in each round is modeled as a partial assignment $\rho \in \{0,1,*\}^n$. At the start of the game, $\rho := *^n$. In each round, Explorer makes one of two moves:

– *Query a variable:* Explorer specifies an $i \in \operatorname{free}\rho$, and Adversary responds with a bit $b \in \{0,1\}$. The state ρ is updated by $\rho_i \leftarrow b$.
– *Forget a variable:* Explorer specifies an $i \in \operatorname{fix}\rho$, and the state is updated by $\rho_i \leftarrow *$.

An important detail is that Adversary is allowed to choose $b \in \{0,1\}$ afresh even if the i-th variable was queried and subsequently forgotten during past play. The game ends when a solution to S can be inferred from ρ, that is, when $C_\rho^{-1}(1) \subseteq S^{-1}(o)$ for some $o \in \mathcal{O}$.

Explorer's goal is to end the game while keeping the width of the game state ρ as small as possible. Indeed, Atserias and Dalmau [4] prove that $w(S)$ is characterized (up to an additive ± 1) as the least w such that the Explorer has a strategy for ending the game that keeps the width of the game state at most w throughout the game. (A similar characterization exists for DAG *size* [39].) Hence our goal becomes to describe an Explorer-strategy for S such that the width of the game state never exceeds $O(d)$ regardless of how the Adversary plays.

5.2 Simplified proof

To explain the basic idea, we first give a simplified version of the proof: We assume that all rectangles R involved in Π—call them the *original* rectangles—can be partitioned *errorlessly* into ρ-structured subrectangles for ρ of width $O(d)$. That is, invoking Rectangle Scheme for each original R, we assume that

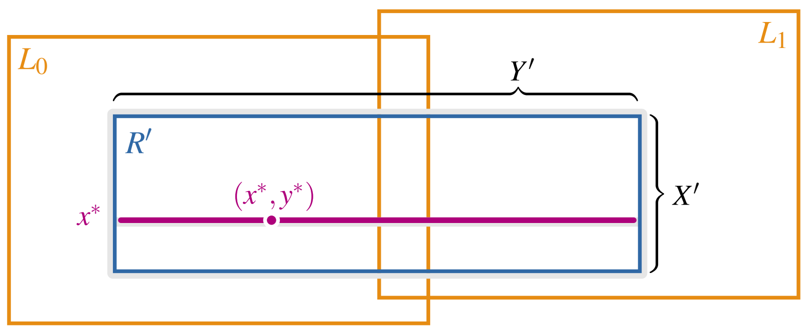
($*$) *Assumption:* All subrectangles in the partition $R = \bigsqcup_i R^i$ output by Rectangle Scheme satisfy the "structured" case of Lemma 4.5 for $k := 2d \log n$.

In Section 5.3 we remove this assumption by explaining how the proof can be modified to work in the presence of some error rows/columns.

Overview. We extract a width-$O(d)$ Explorer-strategy for S by walking down the rectangle-DAG Π, starting at the root. For each original rectangle R that is reached in the walk, we maintain a ρ-structured subrectangle $R' \subseteq R$ chosen from the partition of R. Note that ρ will have width $O(d)$ by our choice of k. The intention is that ρ 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 Π to one of its children? (3) Why are we done once we reach a leaf?

(1) Root case. At start, the root of Π is associated with the original rectangle $R = [m]^n \times \{0,1\}^{mn}$ comprising the whole domain. The partition of R computed by Rectangle Scheme is trivial: it contains a single part, the $*^n$-structured R itself. Hence we simply maintain the $*^n$-structured $R \subseteq R$, which meets the starting condition for the game.

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



Since $R' =: X' \times Y'$ is $\rho_{R'}$-structured, we have from Lemma 4.4 that there exists some $x^* \in X'$ such that $\{x^*\} \times Y'$ is $\rho_{R'}$-like. Let the two original rectangles associated with the children of v be L_0 and L_1. Let $\bigsqcup_i L_b^i$ be the partition of L_b output by Rectangle Scheme. By query alignment in Lemma 4.5, there is some $I_b^* \subseteq [n]$, $|I_b^*| \leq O(d)$, such that all L_b^i that intersect the x^*-th row are ρ^i-structured with $\operatorname{fix} \rho^i \subseteq I_b^*$. As Explorer, we now query the input variables in coordinates $J := (I_0^* \cup I_1^*) \smallsetminus \operatorname{fix} \rho_{R'}$ (in any order) obtaining some response string $z_J \in \{0,1\}^J$ from the Adversary. As a result, the state of the game becomes the extension of $\rho_{R'}$ by z_J, call it ρ^*, which has width $|\operatorname{fix} \rho^*| = |\operatorname{fix} \rho_{R'} \cup J| \leq O(d)$.

Note that there is some $y^* \in Y'$ (and hence $(x^*, y^*) \in R' \subseteq L_0 \cup L_1$) such that $G(x^*, y^*)$ is consistent with ρ^*; indeed, the whole row $\{x^*\} \times Y'$ is $\rho_{R'}$-like and ρ^* extends $\rho_{R'}$. Suppose $(x^*, y^*) \in L_0$; the case of L_1 is analogous. In the partition of L_0, let L' be the unique part such that $(x^*, y^*) \in L'$. Note that L' is $\rho_{L'}$-like for some $\rho_{L'}$ that is consistent with $G(x^*, y^*)$ and $\operatorname{fix} \rho_{L'} \subseteq I_0^*$ (by query alignment). Hence ρ^* extends $\rho_{L'}$. As Explorer, we now forget all queried variables in ρ^* except those queried in $\rho_{L'}$.

We have recovered our invariant: the game state is $\rho_{L'}$ and we maintain a $\rho_{L'}$-structured subrectangle L' of an original rectangle L_0. Moreover, the width of the game state remained $O(d)$.

(3) Leaf case. Suppose the game state is ρ and we are maintaining an associated ρ-structured subrectangle $R' \subseteq R$ corresponding to a *leaf* node. The leaf node is labeled with some solution $o \in \mathcal{O}$ satisfying $R' \subseteq (S \circ G)^{-1}(o)$, that is, $G(R') \subseteq S^{-1}(o)$. But $G(R') = C_\rho^{-1}(1)$ by Lemma 4.3 so that $C_\rho^{-1}(1) \subseteq S^{-1}(o)$. Therefore the game ends. This concludes the (simplified) proof.

5.3 Accounting for error

Next, we explain how to get rid of Assumption (∗) by accounting for the rows and columns that are classified as error in Lemma 4.5 for $k := 2d\log n$. The partitioning of rectangles in Π is done more carefully. We sort all original rectangles in *reverse topological order* $R_1, R_2, \ldots, R_{n^d}$ 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 perform the following process on the rectangles in this order.

Initialize cumulative error sets $X^*_{\mathrm{err}} = Y^*_{\mathrm{err}} := \emptyset$*. Iterate for* $i = 1, 2, \ldots, n^d$ rounds:

1. Remove from R_i the rows/columns X^*_{err}, Y^*_{err}. That is, update
$$R_i \;\leftarrow\; R_i \smallsetminus \big(X^*_{\mathrm{err}} \times \{0,1\}^{mn} \cup [m]^n \times Y^*_{\mathrm{err}}\big).$$
2. Apply the Rectangle Scheme for R_i. Output all resulting subrectangles that satisfy the "structured" case of Lemma 4.5 for $k := 2d\log n$. (All non-structured subrectangles are omitted). Call the resulting error rows/columns X_{err} and Y_{err}.
3. Update $X^*_{\mathrm{err}} \leftarrow X^*_{\mathrm{err}} \cup X_{\mathrm{err}}$ and $Y^*_{\mathrm{err}} \leftarrow Y^*_{\mathrm{err}} \cup Y_{\mathrm{err}}$.

In words, an original rectangle R_i is processed only after all of its descendants are partitioned. Each descendant may contribute some error rows/columns, accumulated into sets X^*_{err}, Y^*_{err}, which are deleted from R_i before it is partitioned. The partitioning of R_i will in turn contribute its error rows/columns to its ancestors.

We may now repeat the proof of Section 5.2 verbatim *using only the structured subrectangles output by the above process*. That is, we still maintain the same invariant: when the game state is ρ, we maintain a ρ-structured R' (output by the above process) of an original R. We highlight only the key points below.

(1) Root case. The cumulative error at the end of the process is tiny: X^*_{err}, Y^*_{err} have density at most $n^d \cdot n^{-2d} \le 1\%$ by a union bound over all rounds. In particular, the root rectangle R_{n^d} (with errors removed) still has density 98% inside $[m]^n \times \{0,1\}^{mn}$, and so the partition output by Rectangle Scheme is trivial, containing only the $*^n$-structured R_{n^d} itself. This meets the starting condition for the game.

(2) Internal step. By construction, the cumulative error sets *shrink* when we take a step from a node to one of its children. This means that our error handling does not interfere with the internal step: each structured subrectangle R' of an original rectangle R is wholly covered by the structured subrectangles of the children of R.

(3) Leaf case. This case is unchanged.

6 Lifting for triangle-DAGs

In this section we prove the nontrivial direction of Theorem 3.3. Let Π be a triangle-DAG solving $S \circ G$ of size n^d for some d. Our goal is to show that $w(S) \le O(d)$.

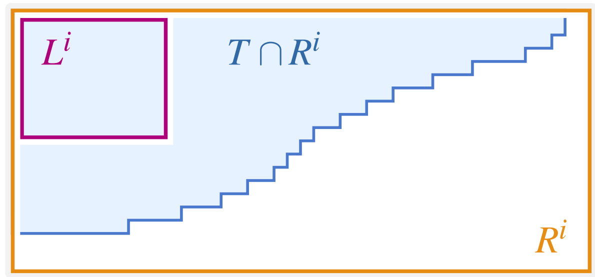


Figure 4: Structured case of Lemma 6.1: The subtriangle $T \cap R^i$ is sandwiched between two ρ^i-structured rectangles L^i and R^i.

The proof is conceptually the same as for rectangle-DAGs. The only difference is that we need to replace Rectangle Scheme (and the associated Lemma 4.5) with an algorithm that partitions a given triangle $T \subseteq [m]^n \times \{0,1\}^{mn}$ into subtriangles that behave like conjunctions.

6.1 Triangle partition scheme

We introduce a triangle partitioning algorithm, Triangle Scheme. Its precise definition is postponed to Section 7.2. For now, we only need its high-level description. On input a triangle T, Triangle Scheme outputs a disjoint cover $\bigsqcup_i R^i \supseteq T$ where R^i are rectangles. This induces a partition of T into subtriangles $T \cap R^i$. Each (non-error) rectangle R^i is ρ^i-structured (for low-width ρ^i) and is associated with a ρ^i-structured "inner" subrectangle $L^i \subseteq R^i$ satisfying $L^i \subseteq T \cap R^i \subseteq R^i$; see Figure 4. Hence $T \cap R^i$ is ρ^i-like, as it is sandwiched between two ρ^i-like rectangles.

More formally, all the properties of Triangle Scheme that we will subsequently need are formalized below (note the similarity with Lemma 4.5); see Section 7.4 for the proof.

Lemma 6.1 (Triangle Lemma)**.** *Fix any parameter $k \leq n \log n$. Given a triangle $T \subseteq [m]^n \times \{0,1\}^{mn}$, let $\bigsqcup_i R^i \supseteq T$ be the output of Triangle Scheme. Then there exist "error" sets $X_{\mathrm{err}} \subseteq [m]^n$ and $Y_{\mathrm{err}} \subseteq \{0,1\}^{mn}$, both of density $\leq 2^{-k}$, such that for each i, one of the following holds.*

- ***Structured case:*** *R^i is ρ^i-structured for some ρ^i of width at most $O(k/\log n)$. Moreover, there exists an "inner" rectangle $L^i \subseteq T \cap R^i$ such that L^i is also ρ^i-structured.*
- ***Error case:*** *R^i is covered by error rows/columns, i. e., $R^i \subseteq X_{\mathrm{err}} \times \{0,1\}^{mn} \cup [m]^n \times Y_{\mathrm{err}}$.*

*Finally, a **query alignment** property holds: for every $x \in [m]^n \smallsetminus X_{\mathrm{err}}$, there exists a subset $I_x \subseteq [n]$ with $|I_x| \leq O(k/\log n)$ such that every "structured" R^i intersecting $\{x\} \times \{0,1\}^{mn}$ has* $\operatorname{fix} \rho^i \subseteq I_x$.

6.2 Simplified proof

As in the rectangle case, we give a simplified proof assuming no errors. That is, invoking Triangle Scheme for each triangle T involved in Π, we assume that

(†) *Assumption:* All rectangles in the cover $\bigsqcup_i R^i \supseteq T$ output by Triangle Scheme satisfy the "structured" case of Lemma 6.1 for $k := 2d \log n$.

The argument for getting rid of the assumption (†) is the same as in the rectangle case, and hence we omit that step—one only needs to observe that removing cumulative error rows/columns from a triangle still leaves us with a triangle.

Overview. As before, we extract a width-$O(d)$ Explorer-strategy for S by walking down the triangle-DAG Π, starting at the root. For each triangle T of Π that is reached in the walk, we maintain a ρ-structured inner rectangle $L \subseteq T$. Here ρ (of width $O(d)$ by the choice of k) will record the current state of the game. There are the three steps (1)–(3) to address, of which (1) and (3) remain exactly the same as in the rectangle case. So we only explain step (2), which requires us to replace the use of Lemma 4.5 with the new Lemma 6.1.

(2) Internal step. Supposing the game has reached state ρ_L and we are maintaining some ρ_L-structured inner rectangle $L \subseteq T$ associated with an internal node v, we want to move to some $\rho_{\widetilde{L}}$-structured inner rectangle $\widetilde{L} \subseteq \widetilde{T}$ associated with a child of v. Moreover, we must keep the width of the game state at most $O(d)$ during this move.

Since $L =: X' \times Y'$ is ρ_L-structured, we have from Lemma 4.4 that there exists some $x^* \in X'$ such that $\{x^*\} \times Y'$ is ρ_L-like. Let the two triangles associated with the children of v be T_0 and T_1, so that $L \subseteq T_0 \cup T_1$.

Let $\bigsqcup_i R_b^i$ be the rectangle cover of T_b output by Triangle Scheme. By query alignment in Lemma 6.1, there is some $I_b^* \subseteq [n]$, $|I_b^*| \le O(d)$, such that all R_b^i that intersect the x^*-th row are ρ^i-structured with $\operatorname{fix}\rho^i \subseteq I_b^*$. As Explorer, we now query the input variables in coordinates $J := (I_0^* \cup I_1^*) \smallsetminus \operatorname{fix}\rho_L$ (in any order) obtaining some response string $z_J \in \{0,1\}^J$ from the Adversary. As a result, the state of the game becomes the extension of ρ_L by z_J, call it ρ^*, which has width $|\operatorname{fix}\rho^*| = |\operatorname{fix}\rho_L \cup J| \le O(d)$.

Note that there is some $y^* \in Y'$ (and hence $(x^*, y^*) \in L \subseteq T_0 \cup T_1$) such that $G(x^*, y^*)$ is consistent with ρ^*; indeed, the whole row $\{x^*\} \times Y'$ is ρ_L-like and ρ^* extends ρ_L. Suppose $(x^*, y^*) \in T_0$; the case of T_1 is analogous. In the rectangle covering of T_0, let R be the unique part such that $(x^*, y^*) \in R$. Note that R is ρ_R-like for some ρ_R that is consistent with $G(x^*, y^*)$ and $\operatorname{fix}\rho_R \subseteq I_0^*$ (by query alignment). Hence ρ^* extends ρ_R. As Explorer, we now forget all queried variables in ρ^* except those queried in ρ_R. Also we move to the inner rectangle $\widetilde{L} \subseteq R$ promised by Lemma 6.1 that satisfies $\widetilde{L} \subseteq T_0$ and is $\rho_{\widetilde{L}} = \rho_R$ structured.

We have recovered our invariant: the game state is $\rho_{\widetilde{L}}$ and we maintain a $\rho_{\widetilde{L}}$-structured subrectangle $\widetilde{L}$ of a triangle T_0. Moreover, the width of the game state remained $O(d)$.

7 Partitioning rectangles and triangles

In this section, we prove Lemma 4.5, define Triangle Scheme, and prove Lemma 6.1. We use repeatedly the following simple fact about min-entropy.

Fact 7.1. *Let $\boldsymbol{X}$ be a random variable and E an event. Then* $\mathbf{H}_\infty(\boldsymbol{X} \mid E) \ge \mathbf{H}_\infty(\boldsymbol{X}) - \log 1/\mathbf{Pr}[E]$.

7.1 Proof of Rectangle Lemma

The proof is more-or-less implicit in [19, 22]. We start by recording a key property of the 1st round of Rectangle Scheme.

Claim 7.2. *Each part X^i obtained in 1st round of Rectangle Scheme satisfies the following conditions.*

- Blockwise-density: $\boldsymbol{X}^i_{[n]\smallsetminus I_i}$ *is* 0.95*-dense.*
- Relative size: $|X^{\geqslant i}| \leq m^{n-0.05|I_i|}$ *where* $X^{\geqslant i} := \bigcup_{j\geq i} X^j$.

Proof. By definition, $\boldsymbol{X}^i = (\boldsymbol{X}^{\geqslant i} \,|\, \boldsymbol{X}^{\geqslant i}_{I_i} = \alpha_i)$. Suppose for contradiction that $\boldsymbol{X}^i_{[n]\smallsetminus I_i}$ is not 0.95-dense. Then there is some nonempty subset $K \subseteq [n] \smallsetminus I_i$ and an outcome $\beta \in [m]^K$ violating the min-entropy condition, namely $\mathbf{Pr}[\boldsymbol{X}^i_K = \beta] > m^{-0.95|K|}$. But this contradicts the maximality of I_i since the larger set $I_i \cup K$ now violates the min-entropy condition for $\boldsymbol{X}^{\geqslant i}$:

$$\mathbf{Pr}[\boldsymbol{X}^{\geqslant i}_{I_i\cup K} = \alpha_i\beta] \;=\; \mathbf{Pr}[\boldsymbol{X}^{\geqslant i}_{I_i} = \alpha_i]\cdot \mathbf{Pr}[\boldsymbol{X}^i_K = \beta] \;>\; m^{-0.95|I_i|}\cdot m^{-0.95|K|} \;=\; m^{-0.95(|I_i\cup K|)}\,.$$

This shows the first property. For the second property, apply Theorem 7.1 for $\boldsymbol{X}^i = (\boldsymbol{X}^{\geqslant i} \,|\, \boldsymbol{X}^{\geqslant i}_{I_i} = \alpha_i)$ to find that $\mathbf{H}_\infty(\boldsymbol{X}^i) \geq \mathbf{H}_\infty(\boldsymbol{X}^{\geqslant i}) - 0.95|I_i|\log m$. On the other hand, since $\boldsymbol{X}^i$ is fixed on I_i, we have $\mathbf{H}_\infty(\boldsymbol{X}^i) \leq (n-|I_i|)\log m$. Combining these two inequalities we get $\mathbf{H}_\infty(\boldsymbol{X}^{\geqslant i}) \leq (n-0.05|I_i|)\log m$, which yields the second property. ☐

Proof of Lemma 4.5. *Identifying Y_{err}, X_{err}.* We define $Y_{\text{err}} := \bigcup_{i,\gamma} Y^{i,\gamma}$ subject to $|Y^{i,\gamma}| < 2^{mn-n^2}$. To bound the size of Y_{err}, we claim that there are at most $(4m)^n$ possible choices of i,γ. Indeed, each X^i is associated with a unique pair $(I_i \subseteq [n], \alpha_i \in [m]^{I_i})$, and there are at most 2^n choices of I_i and at most m^n choices of corresponding α_i. Also, for each X^i, there are at most 2^n possible assignments to $\gamma \in \{0,1\}^{I_i}$. For each i,γ, we add at most 2^{mn-n^2} columns to Y_{err}. Thus, Y_{err} has density at most $(4m)^n\cdot 2^{-n^2} < 2^{-k}$ inside $\{0,1\}^{mn}$.

We define $X_{\text{err}} := \bigsqcup_i X^i$ subject to $|I_i| > 20k/\log m$. Let i be the least index with $|I_i| > 20k/\log m$ so that $X_{\text{err}} \subseteq X^{\geqslant i}$. By Claim 7.2, $|X^{\geqslant i}| \leq m^{n-0.05|I_i|} < m^n\cdot 2^{-k}$ since $|I_i| > 20k/\log m$. In other words, $X^{\geqslant i}$, and hence X_{err}, has density at most 2^{-k} inside $[m]^n$.

Structured vs. error. Let $R^{i,\gamma} := X^i \times Y^{i,\gamma}$, where X_i is associated with (I_i,α_i), be a rectangle *not* contained in the error rows/columns. By definition of X_{err}, Y_{err}, this means $|Y^{i,\gamma}| \geq 2^{mn-n^2}$ (so that $\mathbf{H}_\infty(\boldsymbol{Y}^{i,\gamma}) \geq mn-n^2$) and $|I_i| \leq 20k/\log m$. We have from Claim 7.2 that $\boldsymbol{X}^i_{[n]\smallsetminus I_i}$ is 0.95-dense. Hence, $R^{i,\gamma}$ is ρ^i-structured where ρ^i equals γ on I_i and consists of stars otherwise.

Query alignment. For each $x \in [m]^n \smallsetminus X_{\text{err}}$, we define $I_x = I_i$ where X^i is the unique part that contains x. It follows that any ρ-structured rectangle that intersects the x-th row is of the form $X^i \times Y^{i,\gamma}$ and hence has fix $\rho = I_i$. Since $X^i \not\subseteq X_{\text{err}}$, we have $|I_i| \leq O(k/\log n)$.

7.2 Definition of Triangle Scheme

In the description of Triangle Scheme, we denote projections of a set $S \subseteq [m]^n \times \{0,1\}^{mn}$ by

$$\begin{aligned} X^S &:= \{x \in [m]^n : \exists y \in \{0,1\}^{mn} \text{ such that } (x,y)\in S\}\,,\\ Y^S &:= \{y \in \{0,1\}^{mn} : \exists x \in [m]^n \text{ such that } (x,y)\in S\}\,. \end{aligned}$$

Triangle Scheme

Input: Triangle $T \subseteq [m]^n \times \{0,1\}^{mn}$ with labeling functions (a_T, b_T)
Output: A disjoint rectangle cover $\bigsqcup_i R^i \supseteq T$

$Y_{\text{err}} \leftarrow$ Column Cleanup on T
Initialize $\mathcal{R}^0_{\text{alive}} := \{[m]^n \times (\{0,1\}^{mn} \smallsetminus Y_{\text{err}})\}$; $\mathcal{R}^r_{\text{alive}} := \emptyset$ for all $r \geq 1$; $\mathcal{R}_{\text{final}} := \emptyset$
loop for $r = 0, 1, 2, \ldots,$ rounds until $\mathcal{R}^r_{\text{alive}}$ is empty:
  for all $R \in \mathcal{R}^r_{\text{alive}}$ **do**
    $\bigsqcup_i R^i \leftarrow$ Rectangle Scheme on R relative to free coordinates
    for all parts R^i **do**
      if $|X^{T \cap R^i}| \geq |X^{R^i}|/2$ **then**
        Add R^i to $\mathcal{R}_{\text{final}}$
      else
        $R^{i,\text{top}} :=$ top half of R^i according to a_T (in particular $T \cap R^i \subseteq R^{i,\text{top}}$)
        Add $R^{i,\text{top}}$ to $\mathcal{R}^{r+1}_{\text{alive}}$ subject to $T \cap R^{i,\text{top}} \neq \emptyset$
return $\mathcal{R}_{\text{final}} \cup \{[m]^n \times Y_{\text{err}}\}$

Overview. Triangle Scheme computes a disjoint rectangle cover $\bigsqcup_i R^i$ of T. Starting with a trivial cover of the whole communication domain by a single part, the algorithm progressively *refines* this cover over several rounds as guided by the input triangle T. As outlined in Section 6.1, the goal is to end up with ρ-structured rectangles R^i that contain a large enough portion of T so that we may sandwich $L^i \subseteq T \cap R^i \subseteq R^i$ where L^i is a ρ-structured "inner" rectangle.

The main idea is as follows. The algorithm maintains a pool of *alive* rectangles. In a single round, for each alive rectangle R, we first invoke Rectangle Scheme in order to restore ρ-structuredness for the resulting subrectangles R^i. Then for each R^i we check if the subtriangle $T \cap R^i$ occupies at least half the rows of R^i. If *yes*, we add it to the *final* pool, which will eventually form the output of the algorithm. If *no*, we discard the "lower" half of R^i as determined by the labeling a_T, that is, the half that does not intersect T. The "top" half (containing $T \cap R^i$) will enter the alive pool for next round.

Column Cleanup. An important detail is the subroutine Column Cleanup, run at the start of Triangle Scheme, which computes a small set of columns that will eventually be declared as Y_{err}. By discarding the columns Y_{err}, we ensure that whatever subrectangle R^i is returned by Rectangle Scheme, the rows of $T \cap R^i$ will satisfy an *empty-or-heavy dichotomy*: for every $x \in X^{R^i}$, the x-th row of $T \cap R^i$ is either empty, or "heavy", that is, of size at least 2^{mn-n^2}. For intuition, an extreme bad example we want to avoid is a triangle T that is just a single column; such T would be completely declared as "error" by Column Cleanup. Having many heavy rows helps towards satisfying the 3rd item in Theorem 4.2 of ρ-stucturedness, and hence in finding the inner rectangle L^i. This property of Column Cleanup is formalized in Claim 7.3 below.

Column Clean-up

Input: Triangle $T \subseteq [m]^n \times \{0,1\}^{mn}$ with labeling functions (a_T, b_T)
Output: Error columns $Y_{\text{err}} \subseteq \{0,1\}^{mn}$

$Y_{\text{err}} \leftarrow \emptyset$
For $I \subseteq [n]$, $\alpha \in [m]^I$, $\gamma \in \{0,1\}^I$, define $Y_{I,\alpha,\gamma} := \left\{y \in \{0,1\}^{mn} : g^I(\alpha, y_I) = \gamma\right\}$
while there exists I, α, γ, x such that $0 < |T \cap (\{x\} \times (Y_{I,\alpha,\gamma} \smallsetminus Y_{\text{err}}))| < 2^{mn-n^2}$ **do**
    $Y_{\text{err}} \leftarrow Y_{\text{err}} \cup Y^{T \cap (\{x\} \times Y_{I,\alpha,\gamma})}$
return Y_{err}

Free coordinates. Another detail to explain is the underlined phrase *relative to free coordinates*. For each alive rectangle R we tacitly associate a subset of *free coordinates* $J_R \subseteq [n]$ and *fixed coordinates* $[n] \smallsetminus J_R$. At start, the single alive rectangle has $J_R := [n]$, and whenever we invoke Rectangle Scheme for a rectangle R *relative to free coordinates*, the understanding is that in line (i) of Rectangle Scheme, the choice of I_i is made among subsets of J_R alone. The resulting subrectangle $R^i = X^i \times Y^i$, obtained by fixing the coordinates I_i in X^i, will have its free coordinates $J_{R^i} := J_R \smallsetminus I_i$. (Restricting a rectangle to its top half on line 10 does not modify the free coordinates.)

7.3 Properties of Triangle Scheme

Claim 7.3. *For a triangle $T \subseteq [m]^n \times \{0,1\}^{mn}$, let Y_{err} be the output of Column Cleanup. Then Y_{err} has the following properties.*

- Empty-or-heavy: *For every triple $(I \subseteq [n], \alpha \in [m]^I, \gamma \in \{0,1\}^I)$, and every $x \in [m]^n$, it holds that $T \cap (\{x\} \times (Y_{I,\alpha,\gamma} \smallsetminus Y_{\text{err}}))$ is either empty or has size at least 2^{mn-n^2}.*
- Size bound: $|Y_{\text{err}}| \leq 2^{mn-\Omega(n^2)}$.

Proof. The first property is immediate by definition of Column Cleanup. For the second property, in each while-iteration, at most 2^{mn-n^2} columns get added to Y_{err}. Moreover, there are no more than $2^n \cdot m^n \cdot 2^n \cdot m^n = (2m)^{2n}$ choices of $I \subseteq [n]$, $\alpha \in [m]^I$, $\gamma \in \{0,1\}^I$ and $x \in [m]^n$, and the loop executes at most once for each choice of I, α, γ, x. Thus, $|Y_{\text{err}}| \leq (2m)^{2n} \cdot 2^{mn-n^2} \leq 2^{mn-\Omega(n^2)}$. □

Next, we list some key invariants that hold for Triangle Scheme.

Lemma 7.4. *For every $r \geq 0$, there exists a partition $\mathcal{X}^r := \left\{X^i\right\}_i$ of $[m]^n$ satisfying the following.*

(P1) *For every $R \in \mathcal{R}^r_{\text{alive}}$ we have $X^R \in \mathcal{X}^r$.*
(P2) *Each $X^i \in \mathcal{X}^r$ is labeled by a pair $(I_i \subseteq [n], \alpha_i \in [m]^{I_i})$ such that $\boldsymbol{X}^i_{I_i} = \alpha_i$ is fixed.*
(P3) *The partition $\mathcal{X}^{r+1}$ is a refinement of $\mathcal{X}^r$. The labels respect this: if $X^j \in \mathcal{X}^{r+1}$ is a subset of $X^i \in \mathcal{X}^r$, then $I_j \supseteq I_i$ and α_j agrees with α_i on coordinates I_i.*

Moreover, let $\mathcal{X} := \mathcal{X}^{r^*}$ be the final partition assuming Triangle Scheme completes in r^* rounds.

(P4) *For every $R \in \mathcal{R}_{\text{final}}$ the row set X^R is a union of parts of $\mathcal{X}$. If $X^i \in \mathcal{X}$, labeled (I_i, α_i), is such that $X^R \supseteq X^i$, then the fixed coordinates of R are a subset of I_i.*

(P5) *For every $r \geq 0$, $\mathcal{X}^r$ and $\mathcal{X}$ agree on a fraction $\geq 1 - 2^{-r}$ of rows, that is, there is a subset of "final" parts $\mathcal{X}^r_{\text{final}} \subseteq \mathcal{X}^r$ such that $\bigcup \mathcal{X}^r_{\text{final}}$ has density $\geq 1 - 2^{-r}$ inside $[m]^n$, and $\mathcal{X}^r_{\text{final}} \subseteq \mathcal{X}$.*

Proof. Let us define the row partitions $\mathcal{X}^r$. The partition $\mathcal{X}^1$ contains only a single part, $[m]^n$, labeled by $I_1 := \emptyset$. Supposing $\mathcal{X}^r$ has been defined, the next partition $\mathcal{X}^{r+1}$ is obtained by refining each old part $X^i \in \mathcal{X}^r$. Consider one such old part $X^i \in \mathcal{X}^r$ with label (I_i, α_i). If there is *no* rectangle $R \in \mathcal{R}^r_{\text{alive}}$ with $X^R = X^i$ then we need not partition X^i any further; we simply include X^i in $\mathcal{X}^{r+1}$ as a whole. Otherwise, let $R \in \mathcal{R}^r_{\text{alive}}$ be any rectangle such that $X^R = X^i$; we emphasize that there can be many such choices for R, but the upcoming refinement of X^i will not depend on that choice. The r-th round of the algorithm first computes $R = \bigsqcup_i R^i$ using Rectangle Scheme, and then each R^i might be horizontally split in half. We interpret this as a refinement of X^i according to the 1st round of Rectangle Scheme on R (which only depends on $X^R = X^i$), with each part adding more fixed coordinates to the label (I_i, α_i). Letting $X^i = \bigsqcup_j X^{i,j}$ denote the resulting row partition, we then split each $X^{i,j}$ into two halves $X^{i,j,\text{top}}$ and $X^{i,j,\text{bot}}$. This completes the definition of $\mathcal{X}^{r+1}$.

The properties (P1)–(P5) are straightforward to verify. For (P5), we only note that when the algorithm horizontally splits a rectangle (inducing $X^{i,j} = X^{i,j,\text{top}} \cup X^{i,j,\text{bot}}$), the bottom halves are discarded, and never again touched in future rounds. That is, $X^{i,j,\text{bot}} \in \mathcal{X}^{r'}$ for all $r' > r$. This cuts the number of "alive" rows $\bigcup_{R \in \mathcal{R}^r_{\text{alive}}} X^R$ in half each round. □

Lemma 7.5 (Error rows)**.** *Let $\mathcal{X} = \{X^i\}_i$ be the final row partition in Lemma 7.4. Fix any parameter $k < n \log n$. There is a density-2^{-k} subset $X_{\text{err}} \subseteq [m]^n$ (which is a union of parts of $\mathcal{X}$) such that for any part $X^i \not\subseteq X_{\text{err}}$, we have $|I_i| \leq O(k/\log n)$.*

Proof. Our strategy is as follows (cf. [22, Lemma 7]). For $x \in [m]^n$, let $i(x)$ be the unique index such that $x \in X^{i(x)} \in \mathcal{X}$; recall that $X^{i(x)}$ is labeled by some $(I_{i(x)}, \alpha_{i(x)})$. We will study a uniform random $\boldsymbol{x} \sim [m]^n$ and show that the distribution of the number of fixed coordinates $|I_{i(\boldsymbol{x})}|$ has an exponentially decaying tail. This allows us to define X_{err} as the set of outcomes of $\boldsymbol{x}$ for which $|I_{i(\boldsymbol{x})}|$ is exceptionally large. More quantitatively, it suffices to show for a large constant C,

$$\mathbf{Pr}\big[\,|I_{i(\boldsymbol{x})}| > C \cdot k/\log n\,\big] \;\leq\; 2^{-k}. \tag{7.1}$$

Recall that $\mathcal{X}$ and $\mathcal{X}^\ell$, where $\ell := k+1$, agree on all but a fraction $2^{-k}/2$ of rows by (P5). Hence by a union bound, it suffices to show a version of (7.1) truncated at level ℓ:

$$\mathbf{Pr}\big[\,|I_{i'(\boldsymbol{x})}| > C' \cdot \ell/\log n\,\big] \;\leq\; 2^{-\ell} \quad (= 2^{-k}/2), \tag{7.2}$$

where $i'(x)$ is defined as the unique index with $x \in X^{i'(x)} \in \mathcal{X}^\ell$.

Partitions as a tree. The sequence $\mathcal{X}^0, \ldots, \mathcal{X}^\ell$, of row partitions can be visualized as a depth-ℓ tree where the nodes at depth r corresponds to parts of $\mathcal{X}^r$, and there is an edge from $X \in \mathcal{X}^r$ to $X' \in \mathcal{X}^{r+1}$ iff $X' \subseteq X$. A way to generate a uniform random $\boldsymbol{x} \sim [m]^n$ is to take a random walk down this tree, starting at the root:

- At a non-leaf node $X \in \mathcal{X}^r$ we take a tree edge (X, X') with probability $|X'|/|X|$.
- Once at a leaf node $X \in \mathcal{X}^\ell$, we output a uniformly random $\boldsymbol{x} \sim X$.

Potential function. We define a nonnegative potential function on the nodes of the tree. For each part $X \in \mathcal{X}^r$, labeled $(I \subseteq [n], \alpha \in \{0,1\}^I)$, we define

$$D(X) \;:=\; (n - |I|)\log m - \log|X| \;\geq\; 0.$$

How does the potential change as we take a step starting at node $X \in \mathcal{X}^r$ labeled (J, α)? If X has one child, the value of D remains unchanged. Otherwise, we move to a child of X in two substeps.

- *Substep 1:* Recall that we partition $X = \bigsqcup_i X^i$ according to the 1st round of Rectangle Scheme relative to free coordinates. That is, X^i is further restricted on $I_i \subseteq [n] \smallsetminus J$ to some value $\alpha_i \in [m]^{I_i}$. For a child X^i labeled $(J \sqcup I_i, \alpha \sqcup \alpha_i)$ the potential change is

$$\begin{aligned} D(X^i) - D(X) &= (n - |J \cup I_i|)\log m - \log|X^i| - (n - |J|)\log m + \log|X| \\ &= \log|X| - \log|X^i| - |I_i|\log m \\ &= \log(|X|/|X^{\geqslant i}|) - \log(|X^i|/|X^{\geqslant i}|) - |I_i|\log m \\ &= \log(|X|/|X^{\geqslant i}|) - \log \mathbf{Pr}[\boldsymbol{X}^{\geqslant i}_{I_i} = \alpha_i] - |I_i|\log m \\ &\leq \log(|X|/|X^{\geqslant i}|) + 0.95|I_i|\log m - |I_i|\log m \\ &= \delta(i) - 0.05|I_i|\log m. \qquad\qquad (\text{where } \delta(i) := \log(|X|/|X^{\geqslant i}|)) \end{aligned}$$

- *Substep 2:* Each X^i gets split into two halves, $X^{i,\mathrm{top}}$ and $X^{i,\mathrm{bot}}$. Moving to either child makes the potential increase by exactly 1 bit.

In summary, when we take a step to a random child in our random walk, the overall change in potential is itself a random variable, which is at most

$$\boldsymbol{\delta} - 0.05|\boldsymbol{I}|\log m + 1, \tag{7.3}$$

where $(\boldsymbol{I}, \cdot)$ is the label of the random child, and $\boldsymbol{\delta} := \delta(\boldsymbol{i})$ is the random variable generated by choosing $\boldsymbol{i}$ with $\mathbf{Pr}[\boldsymbol{i} = i] = |X^i|/|X|$. Summing (7.3) over ℓ many rounds, we see that ℓ steps of the random walk takes us to a node $X^{\boldsymbol{j}} \in \mathcal{X}^\ell$ with random index $\boldsymbol{j}$, which is labeled $(I_{\boldsymbol{j}}, \alpha_{\boldsymbol{j}})$, and which satisfies $D(X^{\boldsymbol{j}}) \leq \sum_{r\in[\ell]}(\boldsymbol{\delta}_r + 1) - 0.05|I_{\boldsymbol{j}}|\log m$ where $\boldsymbol{\delta}_r$ is the "$\boldsymbol{\delta}$" variable corresponding to the r-th step. Since the potential is nonnegative, we get that

$$|I_{\boldsymbol{j}}| \;\leq\; \frac{20}{\log m} \cdot \sum_{r\in[\ell]} (\boldsymbol{\delta}_r + 1). \tag{7.4}$$

Bounding this quantity is awkward since, in general, the variables $\boldsymbol{\delta}_r$ are not mutually independent. However, a standard trick to overcome this is to define mutually independent and identically distributed random variables $\boldsymbol{d}_r$ and couple them with $\boldsymbol{\delta}_r$ so that $\boldsymbol{\delta}_r \leq \boldsymbol{d}_r$ with probability 1.

- *Definition of $\boldsymbol{d}_r$:* Sample a uniform real $\boldsymbol{p}_r \in [0,1)$ and define $\boldsymbol{d}_r := \log(1/(1-\boldsymbol{p}_r))$ and couple with $\boldsymbol{\delta}_r$ such that $\boldsymbol{\delta}_r = \delta(\boldsymbol{i})$ where $\boldsymbol{i}$ is such that $\boldsymbol{p}_r$ falls in the $\boldsymbol{i}$-th interval, assuming we have partitioned $[0,1)$ into half-open intervals with lengths $|X^i|/|X|$ (where $X^1, X^2, \ldots$ are the sets from Substep 1) in the natural left-to-right order. Thus, $\boldsymbol{\delta}_r$ is correctly distributed and $\boldsymbol{\delta}_r \leq \boldsymbol{d}_r$ holds with probability 1.

Note that $\mathbf{E}[2^{\boldsymbol{d}_r/2}] = \int_0^1 1/(1-p)^{1/2}\mathrm{d}p = 2$. For a large enough constant $C > 0$, we calculate

$$\begin{aligned}\mathbf{Pr}\big[\textstyle\sum_{r\in[\ell]}\boldsymbol{d}_r > C\ell\big] &= \mathbf{Pr}[2^{\sum_{r\in[\ell]}(\boldsymbol{d}_r/2)} > 2^{C\ell/2}]\\ &\leq \mathbf{E}[2^{\sum_{r\in[\ell]}(\boldsymbol{d}_r/2)}]/2^{C\ell/2}\\ &= \big(\textstyle\prod_{r\in[\ell]}\mathbf{E}[2^{\boldsymbol{d}_r/2}]\big)/2^{C\ell/2}\\ &= 2^{\ell}\cdot 2^{-C\ell/2}\\ &\leq 2^{-C\ell/3}.\end{aligned}$$

Plugging this estimate in (7.4) (using $\boldsymbol{\delta}_r \leq \boldsymbol{d}_r$) we get that $\mathbf{Pr}[|I_{\boldsymbol{j}}| > C'\cdot\ell/\log n] < 2^{-\ell}$ for a sufficiently large C'. This proves (7.2) and concludes the proof of the lemma. □

7.4 Proof of Triangle Lemma

Identifying Y_{err}, X_{err}. The column error set Y_{err} is already defined by Triangle Scheme. Note that only one rectangle, $[m]^n \times Y_{\mathrm{err}}$, is covered by the error columns. Claim 7.3 ensures that Y_{err} has density at most $2^{-\Omega(n^2)} < 2^{-k}$. The row error set X^{err} is defined by Lemma 7.5 (for the given k).

Structured vs. error. Let $\bigsqcup_i R^i$ be the output of Triangle Scheme, and consider an $R^i = X^i \times Y^i$ which is not covered by error rows/columns; in particular $R^i \in \mathcal{R}_{\mathrm{final}}$. Let $I_i \subseteq [n]$ denote the fixed coordinates of R^i such that $\boldsymbol{X}^i_{I_i} = \alpha_i$ for some $\alpha_i \in \{0,1\}^{I_i}$. From Claim 7.2 we have that $\boldsymbol{X}^i_{[n]\smallsetminus I_i}$ is 0.95-dense. From (P4) and Lemma 7.5 we have $|I_i| \leq O(k/\log n)$. Moreover, we observe that $Y^i = Y_{I_i,\alpha_i,\gamma_i} \smallsetminus Y_{\mathrm{err}}$ for some $\gamma_i \in \{0,1\}^{I_i}$ (notation from Column Cleanup) since Rectangle Scheme, and hence Triangle Scheme by extension, only partitions columns by fixing individual gadget outputs. We have $|Y_{I_i,\alpha_i,\gamma_i}| \geq 2^{mn-n}$ by definition, and so $|Y^i| \geq 2^{mn-2n}$ is large enough. Hence we conclude that R^i is ρ^i-structured for ρ^i that equals γ_i on I_i and consists of stars otherwise.

Next, we locate the associated inner rectangle $L^i \subseteq R^i$. All final rectangles returned by Triangle Scheme are such that $|X^{(T\cap R^i)}| \geq |X^i|/2$. That is, every top row in $R^{i,\mathrm{top}}$ has a nonempty intersection with T. Hence the empty-vs-heavy property of Claim 7.3 says that for all $x \in X^{i,\mathrm{top}}$, we have $|T \cap (\{x\}\times Y^i)| \geq 2^{mn-n^2}$. Moreover, note that $\boldsymbol{X}^{i,\mathrm{top}}$ is 0.9-dense on its free coordinates $[n] \smallsetminus I_i$ (we lose at most 1 bit of min-entropy compared to $\boldsymbol{X}^i$ by Theorem 7.1). We can now define $L^i := X^{i,\mathrm{top}} \times Y' \subseteq T \cap R^i$ where Y' is the set of the first (according to b_T) 2^{mn-n^2} columns of Y^i; see Figure 4. This L^i meets all the conditions for being ρ^i-structured.

Query alignment. For $x \in [m]^n \smallsetminus X_{\mathrm{err}}$, we define (I_x, α_x) as the label of the unique part $i(x)$ such that $x \in X^{i(x)} \in \mathcal{X}$. By Lemma 7.5, $|I_x| \leq O(k/\log n)$. Every ρ-structured rectangle $R^j := X^j \times Y^j$ with $X^j \supseteq X^{i(x)}$ is, by (P4), such that $\operatorname{fix}\rho \subseteq I_x$.

8 Translating between mKW and CNF

In this section, for exposition, we recall some known reductions between mKW and CNF search problems (as outlined in Section 3). These reductions are *generic* in that they are not adapted to the special properties of the search problem $S \subseteq \{0,1\}^n \times \mathcal{O}$ one starts with. For concrete applications to natural problems, one often needs more fine-grained reductions; for example, as mentioned in Section 3, the follow-up article [18] has introduced a more specific framework.

In an effort to add some new perspective to the old reductions expounded here, we continue to use the somewhat abstract search problem–centric "top-down" language. We encourage the readers who prefer the CNF-centric "bottom-up" language to refer to the original cited papers.

Certificates. The key property of an n-variable search problem $S \subseteq \{0,1\}^n \times \mathcal{O}$ that facilitates an efficient reduction to a mKW/CNF search problem is having a low *certificate* (a.k.a. nondeterministic) complexity. A *certificate for* $(x,o) \in S$ is a partial assignment $\rho \in \{0,1,*\}^n$ such that x is consistent with ρ and o is a valid output for every input consistent with ρ; in short, $x \in C_\rho^{-1}(1) \subseteq S^{-1}(o)$. A *certificate for* x is a certificate for $(x,o) \in S$ for some $o \in S(x)$. The *certificate complexity of* x is the least width of a certificate for x. The *certificate complexity of* S is the maximum over all $x \in \{0,1\}^n$ of the certificate complexity of x.

For any search problem S one can associate a "certification" search problem S_{cert}: on input x to S, return a certificate for x in S. Algorithmically speaking, such an S_{cert} is clearly at least as hard as S: if we solve S_{cert} by finding a certificate for $(x,o) \in S$, we can solve S by returning o.

CNF search $\Leftrightarrow$ low certificate complexity. For any k-CNF contradiction F, the associated CNF search problem S_F has certificate complexity at most k. Conversely [36], for any total search problem $S \subseteq \{0,1\}^n \times \mathcal{O}$, we can construct a k-CNF contradiction F, where k is the certificate complexity of S, such that S_F is a type of certification problem for S (and hence at least as hard as S). Namely, we can pick a collection $\mathcal{C}$ of width-k certificates, one for each $x \in \{0,1\}^n$. The k-CNF formula F is then defined as $\bigwedge_{\rho \in \mathcal{C}} \neg C_\rho$.

Gadget composition. For the purposes of query complexity, there are two ways to represent the first argument $x \in [m]$ to the index function $\text{IND}_m \colon [m] \times \{0,1\}^m \to \{0,1\}$ as a binary string. The simplest is to write x as a $\log m$-bit string. Under this convention, IND_m has certificate complexity $\log m + 1$. If $S \subseteq \{0,1\}^n \times \mathcal{O}$ has certificate complexity k, the composed problem $S \circ \text{IND}_m^n$ has certificate complexity $k(\log m + 1)$ (by composing certificates). This means that if we start with a k-CNF contradiction F, we may reduce $S_F \circ \text{IND}_m^n$ to solving $S_{F'}$ where F' is a $k(\log m + 1)$-CNF contradiction over $O(mn)$ variables.

A better representation [5, 13], which does not blow up the certificate complexity (or CNF width), is to write x as an m-bit string of Hamming weight 1 (the index of the unique 1-entry encodes $x \in [m]$). Under this convention, $\text{IND}_m \colon \{0,1\}^m \times \{0,1\}^m \to \{0,1\}$ becomes a *partial* function of certificate complexity 2. Hence, if S has certificate complexity k, the *partial* composed problem $S' := S \circ \text{IND}_m^n$ has certificate complexity $2k$.

Moreover, the partial problem S' can be extended into a *total* problem S_{tot} without making it any easier to solve for rectangle/triangle-DAGs, while still allowing for a $O(\log m)$-depth decision tree to find a 1-entry in a given x. Indeed, we introduce new variables/certificates in order to say that an input (x,y) to S' is trivially solved with output $\perp \notin \mathcal{O}$, if $x_i = 0^m$ for some $i \in [n]$. Specifically, Alice will receive new input bits $x' \in (\{0,1\}^m)^n$ (in addition to the original $x \in (\{0,1\}^m)^n$) with the convention that $x_{i,1} := 0$ and $x_{i,m+1} := 1$. We say that an Alice input xx' is *good* if whenever the string $x'_i \in \{0,1\}^m$ contains the substring 01 starting at position j, then $x_{i,j} = 1$. Note that since each x'_i starts with a 0 and ends with a 1, the substring 01 must appear somewhere in x'_i. Thus when xx' is good, each x_i will have Hamming weight at least 1. If xx' is *not* good (meaning some x'_i contains a substring 01 but the corresponding bit of x_i is 0), there is a width-3 certificate witnessing this. Our total search problem $S_{\text{tot}} \subseteq \{0,1\}^{2mn} \times \{0,1\}^{mn} \times (\mathcal{O} \cup \{\perp\})$ is defined by all these width-3 certificates (for output $\perp$) together with all the original certificates of S'. To see that S_{tot} is at least as hard as S' for rectangle/triangle-DAGs, we note that for any input (x,y) to S', Alice can compute a unique x' so that xx' is *good*. Now any output $o \in S_{\text{tot}}(xx',y)$ is also such that $o \in S'(x,y)$. Finally, we note that for every $i \in [n]$, there is a $(\log m+1)$-query decision tree that either finds some $j \in [m]$ with $x_{i,j} = 1$ or finds a certificate that xx' is not good; namely, the decision tree performs a binary search on x'_i for an occurrence of the substring 01. (This decision tree is useful when finding upper bounds for S_{tot}, such as for Theorem 3.4.)

In summary, we can reduce (in the context of our DAG-like models) $S_F \circ \text{IND}_m^n$ to solving $S_{F'}$ where F' is a $2k$-CNF contradiction over $2mn$ variables.

mKW problems. A rectangle $R \subseteq \mathcal{X} \times \mathcal{Y}$ is *monochromatic* for a search problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ if $R \subseteq S^{-1}(o)$ for some $o \in \mathcal{O}$. The nondeterministic communication complexity of S is the logarithm of the least number of monochromatic rectangles that cover the whole input domain $\mathcal{X} \times \mathcal{Y}$. If S has nondeterministic communication complexity $\log N$, then by a standard reduction (e. g., [15, Lemma 2.3]) S reduces to S_f for some monotone $f\colon \{0,1\}^N \to \{0,1\}$.

Consider a composed search problem $S_F \circ g^n$ obtained from a k-CNF contradiction with ℓ clauses. Its nondeterministic communication complexity is at most $\log \ell + k \cdot (\log m + 1)$; intuitively, it takes $\log \ell$ bits to specify an unsatisfied clause C, and $\log m + 1$ bits to verify the output of a single gadget, and there are k gadgets relevant to C. In summary, $S_F \circ g^n$ reduces to S_f for some monotone $f : \{0,1\}^N \to \{0,1\}$ on $N = \ell \cdot (2m)^k$ variables.

Suppose for a moment that a version of Theorem 3.1, proving a $2^{\Omega(w)}$ lower bound, held for a gadget of constant size $m = O(1)$. Then we could lift any of the known CNF contradictions with parameters $k = O(1)$, $\ell = O(n)$, $w = \Omega(n)$, to obtain an explicit monotone function on $N = \Theta(n)$ variables, with essentially maximal monotone circuit complexity $2^{\Omega(N)}$. This gives some motivation to further develop lifting tools for small gadgets.

9 Proof of Lemma 4.4

To prove Lemma 4.4, we recall two claims from [22] (which were used to prove Lemma 4.3). We need the first claim in a slightly strengthened form. First, define $\chi(z) := (-1)^{\sum_i z_i}$.

Claim 9.1 (Strengthening [22, Lemma 8])**.** *For any ρ-structured $X \times Y$ with* free $\rho =: J \subseteq [n]$,

$$\forall I \subseteq J, I \neq \emptyset : \qquad \mathbf{E}_{\boldsymbol{X}} \left| \mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(\boldsymbol{X}_I, \boldsymbol{Y}_I))] \right| \leq 2^{-5|I|\log n}.$$

Proof. Fix any $I \subseteq J$, $I \neq \emptyset$. Define subsets

$$X^+ := \left\{ x \in X \,:\, \mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(x_I, \boldsymbol{Y}_I))] > 0 \right\} \quad \text{and} \quad X^- := \left\{ x \in X \,:\, \mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(x_I, \boldsymbol{Y}_I))] < 0 \right\}$$

so that

$$\mathbf{E}_{\boldsymbol{X}} \left| \mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(x_I, \boldsymbol{Y}_I))] \right| = \frac{|X^+|}{|X|} \cdot \mathbf{E}_{\boldsymbol{X}^+} \mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(\boldsymbol{X}_I^+, \boldsymbol{Y}_I))] + \frac{|X^-|}{|X|} \cdot \mathbf{E}_{\boldsymbol{X}^-} \mathbf{E}_{\boldsymbol{Y}}[-\chi(g^I(\boldsymbol{X}_I^-, \boldsymbol{Y}_I))]\,.$$

It suffices to show that each of the two terms is at most $0.5 \cdot 2^{-5|I|\log n}$. Let us focus only on the first term (a similar argument takes care of the second term). If $|X^+| \leq 0.5 \cdot 2^{-5|I|\log n} \cdot |X|$, then we are already done, so assume the contrary so that $\mathbf{H}_\infty(\boldsymbol{X}_I^+) \geq \mathbf{H}_\infty(\boldsymbol{X}_I) - 5|I|\log n - 1 \geq 0.8|I|\log m$; here recall that $\mathbf{H}_\infty(\boldsymbol{X}_I) \geq 0.9|I|\log m$ and we may assume $m \geq n^{60}$. To complete the proof, we rely on a calculation from [22, Lem. 8]. There, the following is proved for constant 0.9 in place of 0.8, but this is inconsequential, as one can always increase the exponent in $m = n^\Delta$ if necessary.

Calculation from [22, Lem. 8, Eq. 4]: If $\mathbf{H}_\infty(\boldsymbol{X}_I^+) \geq 0.8|I|\log m$ and $\mathbf{H}_\infty(\boldsymbol{Y}) \geq mn - n^3$ then

$$|\mathbf{E}_{\boldsymbol{X}^+}\mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(\boldsymbol{X}_I^+, \boldsymbol{Y}_I))]| \leq 0.5 \cdot 2^{-5|I|\log n}. \qquad \square$$

Claim 9.2 ([22, Lem. 9])**.** *If a random variable $\boldsymbol{z}_J$ over $\{0,1\}^J$ satisfies*[1] $|\mathbf{E}[\chi(\boldsymbol{z}_I)]| \leq 2^{-3|I|\log n}$ *for every nonempty $I \subseteq J$, then $\boldsymbol{z}_J$ has full support over $\{0,1\}^J$.* $\square$

Proof of Lemma 4.4. Say that $x \in X$ is *good* if $|\mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(x_I, \boldsymbol{Y}_I))]| \leq 2^{-3|I|\log n}$ for all $\emptyset \neq I \subseteq J$. By applying Markov's inequality to Theorem 9.1, we have for a uniform random $\boldsymbol{x} \sim X$ and any $\emptyset \neq I \subseteq J$ that

$$\mathbf{Pr}_{\boldsymbol{x}\sim X}\left[\left|\mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(\boldsymbol{x}_I, \boldsymbol{Y}_I))]\right| > 2^{-3|I|\log n}\right] \leq 2^{-2|I|\log n}.$$

Taking a union bound over all $\emptyset \neq I \subseteq J$, we get

$$\begin{aligned}
\mathbf{Pr}_{\boldsymbol{x}\sim X}[\boldsymbol{x} \text{ is not } \textit{good}] &\leq \textstyle\sum_{\emptyset\neq I\subseteq J} \mathbf{Pr}_{\boldsymbol{x}\sim X}\left[|\mathbf{E}_{\boldsymbol{Y}}[\chi(g^I(\boldsymbol{x}_I, \boldsymbol{Y}_I))]| > 2^{-3|I|\log n}\right] \\
&\leq \textstyle\sum_{\emptyset\neq I\subseteq J} 2^{-2|I|\log n} = \sum_{d=1}^{|J|}\binom{|J|}{d} \cdot 2^{-2d\log n} \\
&\leq \textstyle\sum_{d=1}^{|J|} 2^{-d\log n} \leq 2/n.
\end{aligned}$$

Hence most $x \in X$ are *good*. Finally, observe that for any *good* x, the random variable $\boldsymbol{z}_J$ defined as $g^J(x, \boldsymbol{y})$ for a random $\boldsymbol{y} \sim Y$, satisfies the Fourier condition in Theorem 9.2. Therefore, such a $\boldsymbol{z}_J$ has full support over $\{0,1\}^J$, which means that $\{x\} \times Y$ is ρ-like. $\square$

10 Open problems

If the long line of work on *tree-like* lifting theory is of any indication, there should be much to explore also in the DAG-*like* setting. We propose a few concrete directions.

Can our methods be extended to prove lower bounds for DAGs whose feasible sets are *intersections of k triangles* for $k \geq 2$? See Figure 2. This would imply lower bounds for proofs systems such as width-k Resolution over Cutting Planes [34] and Resolution over linear equations [43, 29].

Question 10.1. Prove a lifting theorem for $\mathcal{F}$-DAGs where $\mathcal{F} := \{\text{intersections of } k \text{ triangles}\}$.

One of the most important open problems (e. g., [49, §5]) regarding semi-algebraic proof systems that manipulate low-degree polynomials—where $\mathcal{F}$ is, say, degree-d polynomial threshold functions—is to prove lower bounds on their DAG-*like* refutation length (*tree-like* lower bounds are known [7, 20]).

[1] In [22, §4.6], the claim is proved for the condition $|\mathbf{E}[\chi(z_I)]| \leq 2^{-5|I|\log n}$. However, the proof still works with the weaker condition $2^{-3|I|\log n}$, as we only require that $\boldsymbol{z}_J$ has full support instead of being pointwise-close to uniform.

Since degree-d polynomials can be efficiently evaluated by $(d+1)$-party number-on-forehead (NOF) protocols, one might hope to prove a DAG-like NOF lifting theorem. However, we currently lack a good understanding of NOF lifting even in the tree-like case. We believe the first necessary step should be to settle the following (a two-party analogue of which was proved in [19]).

Question 10.2. Prove a *nondeterministic* lifting theorem for NOF protocols.

The proof of Theorem 3.1, which extracts a width-$O(d)$ conjunction-DAG from a size-n^d rectangle-DAG, has the additional property of preserving the DAG *depth* (up to an $O(d)$ factor). This raises the question of whether one could investigate size–depth tradeoffs for monotone circuits via lifting.

Question 10.3. Does there exist, for any $d \geq 1$, an $f\colon \{0,1\}^n \to \{0,1\}$ computable with monotone circuits of size n^d such that any subexponential-size monotone circuit computing f has depth $n^{\Omega(d)}$?

Razborov [48] has recently obtained related results for Resolution, but the parameters in his construction seem not to be good enough for a direct application of Theorem 3.1.

Acknowledgements

We thank Jakob Nordström for extensive feedback on an early draft of this paper. We also thank Toniann Pitassi, Thomas Watson, and anonymous STOC and ToC reviewers for comments.

References

[1] NOGA ALON AND RAVI B. BOPPANA: The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987. [doi:10.1007/BF02579196] 6

[2] KAZUYUKI AMANO AND AKIRA MARUOKA: The potential of the approximation method. *SIAM J. Comput.*, 33(2):433–447, 2004. [doi:10.1137/S009753970138445X] 6

[3] ALEXANDER E. ANDREEV: On a method for obtaining lower bounds for the complexity of individual monotone functions. *Dokl. Math.*, 282(5):1033–1037, 1985. Link at Math-Net.ru. 6

[4] ALBERT ATSERIAS AND VÍCTOR DALMAU: A combinatorial characterization of resolution width. *J. Comput. System Sci.*, 74(3):323–334, 2008. [doi:10.1016/j.jcss.2007.06.025] 10

[5] PAUL BEAME, TRINH HUYNH, AND TONIANN PITASSI: Hardness amplification in proof complexity. In *Proc. 42nd STOC*, pp. 87–96. ACM Press, 2010. [doi:10.1145/1806689.1806703] 2, 21

[6] PAUL BEAME AND TONIANN PITASSI: Propositional proof complexity: Past, present, and future. In *Current Trends in Theoretical Computer Science: Entering the 21st Century*, pp. 42–70. World Scientific, 2001. [doi:10.1142/9789812810403_0001, ECCC:TR98-067] 7

[7] PAUL BEAME, TONIANN PITASSI, AND NATHAN SEGERLIND: Lower bounds for Lovász–Schrijver systems and beyond follow from multiparty communication complexity. *SIAM J. Comput.*, 37(3):845–869, 2007. [doi:10.1137/060654645] 23

[8] ELI BEN-SASSON AND AVI WIGDERSON: Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, 2001. [doi:10.1145/375827.375835] 4

[9] CHRISTER BERG AND STAFFAN ULFBERG: Symmetric approximation arguments for monotone lower bounds without sunflowers. *Comput. Complexity*, 8(1):1–20, 1999. [doi:10.1007/s000370050017] 6

[10] MARIA LUISA BONET, TONIANN PITASSI, AND RAN RAZ: Lower bounds for cutting planes proofs with small coefficients. *J. Symbolic Logic*, 62(3):708–728, 1997. [doi:10.2307/2275569] 2

[11] ARKADEV CHATTOPADHYAY, MICHAL KOUCKÝ, BRUNO LOFF, AND SAGNIK MUKHOPADHYAY: Simulation theorems via pseudo-random properties. *Comput. Complexity*, 28:617–659, 2019. [doi:10.1007/s00037-019-00190-7, arXiv:1704.06807] 2

[12] WILLIAM COOK, COLLETTE COULLARD, AND GYÖRGY TURÁN: On the complexity of cutting-plane proofs. *Discr. Appl. Math.*, 18(1):25–38, 1987. [doi:10.1016/0166-218X(87)90039-4] 7

[13] SUSANNA F. DE REZENDE, JAKOB NORDSTRÖM, AND MARC VINYALS: How limited interaction hinders real communication (and what it means for proof and circuit complexity). In *Proc. 57th FOCS*, pp. 295–304. IEEE Comp. Soc., 2016. [doi:10.1109/FOCS.2016.40] 2, 21

[14] NOAH FLEMING, DENIS PANKRATOV, TONIANN PITASSI, AND ROBERT ROBERE: Random $\Theta(\log n)$-CNFs are hard for cutting planes. In *Proc. 58th FOCS*, pp. 109–120. IEEE Comp. Soc., 2017. Update: ECCC TR17-045. [doi:10.1109/FOCS.2017.19] 6, 7

[15] ANNA GÁL: A characterization of span program size and improved lower bounds for monotone span programs. *Comput. Complexity*, 10(4):277–296, 2001. [doi:10.1007/s000370100001] 22

[16] ANKIT GARG, MIKA GÖÖS, PRITISH KAMATH, AND DMITRY SOKOLOV: Monotone circuit lower bounds from resolution. In *Proc. 50th STOC*, pp. 902–911. ACM Press, 2018. [doi:10.1145/3188745.3188838] 1

[17] MIKA GÖÖS, PRITISH KAMATH, TONIANN PITASSI, AND THOMAS WATSON: Query-to-communication lifting for P^{NP}. *Comput. Complexity*, 28(1):113–144, 2019. Preliminary version in CCC'17. [doi:10.1007/s00037-018-0175-5] 8

[18] MIKA GÖÖS, PRITISH KAMATH, ROBERT ROBERE, AND DMITRY SOKOLOV: Adventures in monotone complexity and TFNP. In *Proc. 10th Innovations in Theoret. Comp. Sci. (ITCS'19)*, pp. 38:1–38:19. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019. [doi:10.4230/LIPIcs.ITCS.2019.38] 2, 5, 7, 21

[19] MIKA GÖÖS, SHACHAR LOVETT, RAGHU MEKA, THOMAS WATSON, AND DAVID ZUCKERMAN: Rectangles are nonnegative juntas. *SIAM J. Comput.*, 45(5):1835–1869, 2016. [doi:10.1137/15M103145X] 6, 8, 15, 24

[20] MIKA GÖÖS AND TONIANN PITASSI: Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018. Preliminary version in STOC'14. [doi:10.1137/16M1082007] 2, 23

[21] MIKA GÖÖS, TONIANN PITASSI, AND THOMAS WATSON: Deterministic communication vs. partition number. *SIAM J. Comput.*, 47(6):2435–2450, 2018. Preliminary version in FOCS'15. [doi:10.1137/16M1059369] 2

[22] MIKA GÖÖS, TONIANN PITASSI, AND THOMAS WATSON: Query-to-communication lifting for BPP. *SIAM J. Comput.*, 49(4):132–143, 2020. Preliminary version in FOCS'17. [doi:10.1137/17M115339X] 6, 8, 15, 18, 22, 23

[23] ARMIN HAKEN: Counting bottlenecks to show monotone $\mathrm{P} \neq \mathrm{NP}$. In *Proc. 36th FOCS*, pp. 36–40. IEEE Comp. Soc., 1995. [doi:10.1109/SFCS.1995.492460] 6

[24] ARMIN HAKEN AND STEPHEN A. COOK: An exponential lower bound for the size of monotone real circuits. *J. Comput. System Sci.*, 58(2):326–335, 1999. [doi:10.1006/jcss.1998.1617] 7

[25] DANNY HARNIK AND RAN RAZ: Higher lower bounds on monotone size. In *Proc. 32nd STOC*, pp. 378–387. ACM Press, 2000. [doi:10.1145/335305.335349] 6

[26] PAVEL HRUBEŠ AND PAVEL PUDLÁK: Random formulas, monotone circuits, and interpolation. In *Proc. 58th FOCS*, pp. 121–131. IEEE Comp. Soc., 2017. [doi:10.1109/FOCS.2017.20] 6, 7

[27] PAVEL HRUBEŠ AND PAVEL PUDLÁK: A note on monotone real circuits. *Information Processing Letters*, 131:15–19, 2018. [doi:10.1016/j.ipl.2017.11.002, ECCC:TR17-048] 7

[28] TRINH HUYNH AND JAKOB NORDSTRÖM: On the virtue of succinct proofs: Amplifying communication complexity hardness to time–space trade-offs in proof complexity. In *Proc. 44th STOC*, pp. 233–248. ACM Press, 2012. [doi:10.1145/2213977.2214000] 2

[29] DMITRY ITSYKSON AND DMITRY SOKOLOV: Lower bounds for splittings by linear combinations. In *Proc. 39th Math. Found. Comp. Sci. (MFCS'14)*, pp. 372–383. Springer, 2014. [doi:10.1007/978-3-662-44465-8_32] 23

[30] STASYS JUKNA: Finite limits and monotone computations: The lower bounds criterion. In *Proc. 12th IEEE Conf. on Comput. Complexity (CCC'97)*, pp. 302–313. IEEE Comp. Soc., 1997. [doi:10.1109/CCC.1997.612325] 6

[31] STASYS JUKNA: *Boolean Function Complexity: Advances and Frontiers*. Volume 27 of *Algorithms and Combinatorics*. Springer, 2012. [doi:10.1007/978-3-642-24508-4] 2, 4, 7

[32] MAURICIO KARCHMER AND AVI WIGDERSON: Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discr. Math.*, 3(2):255–265, 1990. Preliminary version in STOC'88. [doi:10.1137/0403021] 4

[33] JAN KRAJÍČEK: Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. Symbolic Logic*, 62(2):457–486, 1997. [doi:10.2307/2275541] 2

[34] JAN KRAJÍČEK: Discretely ordered modules as a first-order extension of the cutting planes proof system. *J. Symbolic Logic*, 63(4):1582–1596, 1998. [doi:10.2307/2586668] 23

[35] EYAL KUSHILEVITZ AND NOAM NISAN: *Communication Complexity*. Cambridge Univ. Press, 1997. [doi:10.1017/CBO9780511574948] 2

[36] LÁSZLÓ LOVÁSZ, MONI NAOR, ILAN NEWMAN, AND AVI WIGDERSON: Search problems in the decision tree model. *SIAM J. Discr. Math.*, 8(1):119–132, 1995. [doi:10.1137/S0895480192233867] 21

[37] KETAN MULMULEY: A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987. [doi:10.1007/BF02579205] 5

[38] PAVEL PUDLÁK: Lower bounds for resolution and cutting plane proofs and monotone computations. *J. Symbolic Logic*, 62(3):981–998, 1997. [doi:10.2307/2275583] 7

[39] PAVEL PUDLÁK: Proofs as games. *Amer. Math. Monthly*, 107(6):541–550, 2000. [doi:10.2307/2589349] 10

[40] PAVEL PUDLÁK: On extracting computations from propositional proofs (a survey). In *Proc. 30th Found. Software Techn. Theoret. Comp. Sci. (FSTTCS'10)*, pp. 30–41. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010. [doi:10.4230/LIPIcs.FSTTCS.2010.30] 2, 3, 4

[41] ANUP RAO AND AMIR YEHUDAYOFF: *Communication Complexity and Applications*. Cambridge Univ. Press, 2020. [doi:10.1017/9781108671644] 2

[42] RAN RAZ AND PIERRE MCKENZIE: Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999. [doi:10.1007/s004930050062] 2

[43] RAN RAZ AND IDDO TZAMERET: Resolution over linear equations and multilinear proofs. *Ann. Pure Appl. Logic*, 155(3):194–224, 2008. [doi:10.1016/j.apal.2008.04.001] 23

[44] ALEXANDER A. RAZBOROV: Lower bounds on the monotone complexity of some Boolean functions. *Dokl. Math.*, 31(4):354–357, 1985. Link at Math-Net.ru. 6

[45] ALEXANDER A. RAZBOROV: On the method of approximations. In *Proc. 21st STOC*, pp. 167–176. ACM Press, 1989. [doi:10.1145/73007.73023] 6

[46] ALEXANDER A. RAZBOROV: Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya Math.*, (1):205–227, 1995. Link at Math-Net.ru. 2, 3

[47] ALEXANDER A. RAZBOROV: On small size approximation models. In *The Mathematics of Paul Erdös I*, pp. 385–392. Springer, 1997. [doi:10.1007/978-3-642-60408-9_28] 6

[48] ALEXANDER A. RAZBOROV: A new kind of tradeoffs in propositional proof complexity. *J. ACM*, 63(2):16:1–16:14, 2016. [doi:10.1145/2858790] 24

[49] ALEXANDER A. RAZBOROV: Proof complexity and beyond. *SIGACT News*, 47(2):66–86, 2016. [doi:10.1145/2951860.2951875] 7, 23

[50] BENJAMIN ROSSMAN: The monotone complexity of k-clique on random graphs. *SIAM J. Comput.*, 43(1):256–279, 2014. [doi:10.1137/110839059] 6

[51] JANOS SIMON AND SHI-CHUN TSAI: On the bottleneck counting argument. *Theoret. Comput. Sci.*, 237(1–2):429–437, 2000. Preliminary version in CCC'97. [doi:10.1016/S0304-3975(99)00321-7] 6

[52] DMITRY SOKOLOV: Dag-like communication and its applications. In *Proc. 12th Comp. Sci. Symp. in Russia (CSR'17)*, pp. 294–307. Springer, 2017. [doi:10.1007/978-3-319-58747-9_26] 2, 3, 4

[53] ÉVA TARDOS: The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988. [doi:10.1007/BF02122563] 5

[54] ALASDAIR URQUHART: Hard examples for resolution. *J. ACM*, 34(1):209–219, 1987. [doi:10.1145/7531.8928] 5

[55] AVI WIGDERSON: The fusion method for lower bounds in circuit complexity. In *Combinatorics, Paul Erdős is Eighty*, pp. 453–468. János Bolyai Math. Soc., Budapest, 1993. 6

[56] XIAODI WU, PENGHUI YAO, AND HENRY YUEN: Raz–McKenzie simulation with the inner product gadget. *Electron. Colloq. Comput. Complexity*, TR17-010, 2017. [ECCC] 2

AUTHORS

Ankit Garg
Researcher
Microsoft Research India
garga@microsoft.com
https://ankit-garg-6.github.io/

Mika Göös
Assistant professor
EPFL
mika.goos@epfl.ch
https://theory.epfl.ch/mika/

Pritish Kamath
Postdoctoral scholar
Toyota Technological Institute at Chicago
Chicago, IL, USA
pritish@alum.mit.edu
https://pritishkamath.github.io/

Dmitry Sokolov
Postdoctoral researcher
KTH Royal Institute of Technology
sokolovd@kth.se
https://logic.pdmi.ras.ru/~sokolov/

ABOUT THE AUTHORS

ANKIT GARG is a researcher at Microsoft Research India. Previously, he obtained his Ph. D. in computer science from Princeton University under the supervision of Mark Braverman, and then spent two years as a postdoc at Microsoft Research New England. Even before that, he got his undergraduate degree in computer science from the Indian Insitute of Technology Delhi. His research interests lie in algebraic complexity, optimization, and communication complexity. He grew up in Bathinda (a small city in Punjab, India) and like most Indians, loves playing and watching cricket.

MIKA GÖÖS is an assistant professor at EPFL in the Theory Group. Previously, he was a postdoc at Stanford Theory, the Institute for Advanced Study, and the ToC group at Harvard. He completed his Ph. D. at the University of Toronto under the watchful eye of Toniann Pitassi. He obtained his B. S. from Aalto University, and his M. S. from the University of Oxford. In his spare time, Mika enjoys rock climbing, hoping to climb a 7C-grade boulder one day.

PRITISH KAMATH is a postdoctoral scholar at the Toyota Technological Institute at Chicago. He completed his Ph. D. at MIT, co-advised by Madhu Sudan and Ronitt Rubinfeld. Prior to that, he completed a B. Tech. in Computer Science at IIT Bombay in 2012 after which he was a Research Fellow at Microsoft Research India until 2013, where he was mentored by Neeraj Kayal. He has broad interests in complexity theory and has worked in areas touching upon communication complexity, information theory, Boolean and algebraic circuit complexity and proof complexity. Most recently he has also become interested in foundational aspects of machine learning. He likes to juggle multiple things in life; sometimes on a bicycle.

DMITRY SOKOLOV is a postdoc at Lund University and the University of Copenhagen. He got his Ph. D. in mathematics from the St. Petersburg Department of Steklov Mathematical Institute of the Russian Academy of Sciences under the supervision of Edward Hirsch and Dmitry Itsykson, and then spent two years as a postdoc at KTH Royal Institute of Technology. He defended his master's thesis in computer science at St. Petersburg Academic University. His research interests include proof complexity, communication complexity, and circuit complexity. He likes to play table tennis.