Lifting with Sunflowers

Shachar Lovett
Department of Computer Science, University of California San Diego, CA, USA

Raghu Meka
Department of Computer Science, University of California Los Angeles, CA, USA

Ian Mertz
Department of Computer Science, University of Toronto, Canada

Toniann Pitassi
Department of Computer Science, University of Toronto, Canada
Department of Mathematics, Institute for Advanced Study, Princeton, NJ, USA

Jiapeng Zhang
Department of Computer Science, University of Southern California, Los Angeles, CA, USA

Abstract

Query-to-communication lifting theorems translate lower bounds on query complexity to lower bounds for the corresponding communication model. In this paper, we give a simplified proof of deterministic lifting (in both the tree-like and dag-like settings). Our proof uses elementary counting together with a novel connection to the sunflower lemma.

In addition to a simplified proof, our approach opens up a new avenue of attack towards proving lifting theorems with improved *gadget size* – one of the main challenges in the area. Focusing on one of the most widely used gadgets – the index gadget – existing lifting techniques are known to require at least a quadratic gadget size. Our new approach combined with *robust sunflower lemmas* allows us to reduce the gadget size to near linear. We conjecture that it can be further improved to polylogarithmic, similar to the known bounds for the corresponding robust sunflower lemmas.

2012 ACM Subject Classification Mathematics of computing → Combinatorial algorithms; Theory of computation → Communication complexity; Theory of computation → Complexity theory and logic; Theory of computation → Circuit complexity; Theory of computation → Proof complexity

Keywords and phrases Lifting theorems, communication complexity, combinatorics, sunflowers

Digital Object Identifier 10.4230/LIPIcs.ITCS.2022.104

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2020/111/`

Funding *Shachar Lovett*: Research supported by NSF Award DMS-1953928.
Ian Mertz: Research supported by NSERC.
Toniann Pitassi: Research supported by NSF Award CCF-1900460 and NSERC.

Acknowledgements The authors thank Paul Beame for comments.

1 Introduction

A *query-to-communication* lifting theorem is a reductive lower bound technique that translates lower bounds on query complexity (such as decision tree complexity) to lower bounds for the corresponding communication complexity model. For a function $f : \{0,1\}^n \to R$, and a function $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ (called the *gadget*), their composition $f \circ g^n : \mathcal{X}^n \times \mathcal{Y}^n \to R$ is defined by

$$(f \circ g^n)(x,y) := f(g(x_1,y_1),\ldots,g(x_n,y_n)).$$

Here, Alice holds $x \in \mathcal{X}^n$ and Bob holds $y \in \mathcal{Y}^n$. Typically g is the popular *index* gadget $\textsc{Ind}_m : [m] \times \{0,1\}^m \to \{0,1\}$ mapping (x,y) to the x-th bit of y.

There is a substantial body of work proving lifting theorems for a variety of flavors of query-to-communication, including: deterministic [30, 17, 8, 36, 6, 5], nondeterministic [15, 12], randomized [18, 5], degree-to-rank [34, 27, 28, 32], and nonnegative degree to nonnegative rank [4, 22]. In these papers and others, lifting theorems have been applied to simplify and resolve some longstanding open problems, including new separations in communication complexity [16, 17, 18, 6, 5], proof complexity [15, 20, 16, 8, 7, 14] monotone circuit complexity [11], monotone span programs and linear secret sharing schemes [32, 27, 28], and lower bounds on the extension complexity of linear and semi-definite programs [4, 22, 25]. Furthermore within communication complexity most functions of interest – e.g. equality, set-disjointness, inner product, gap-hamming (c.f. [24, 21]) – are lifted functions.

At the heart of these proofs is a *simulation theorem*.[1] A communication protocol for the lifted function can "mimic" a decision tree for the original function by taking $\log m + 1$ steps to calculate each variable queried by the decision tree in turn. For large enough $m = n^{O(1)}$ and for every f the deterministic simulation theorem [30, 17] shows that this simulation goes the other way as well:

$$\mathbf{P}^{cc}(f \circ \textsc{Ind}_m^n) = \mathbf{P}^{dt}(f) \cdot \Theta(\log m)$$

The proof of this theorem has evolved considerably since [30], applying to a wider range of gadgets [36, 6, 5], and with more sharpened results giving somewhat improved parameters and simulation theorems for the more difficult settings of randomized and dag-like lifting. The original proof of [30] used the notion of min-degree for the central invariant used to prove the simulation theorem; later [15] introduced the notion of blockwise min-entropy, which has since been used for a variety of lifting theorems, including randomized [18] and dag-like [11]. Nearly all of these proofs used either intricate combinatorial arguments or tools from Fourier analysis.

Lifting using the sunflower lemma

One important goal of this paper is to give a readable, self-contained and simplified proof of the deterministic query-to-communication lifting theorem. Our proof uses the same basic setup as in previous arguments, but our proof of the main invariant – showing that any large rectangle can be decomposed into a part that has structure and a part that is pseudo-random – is proven by a direct reduction to the famous sunflower lemma.

The sunflower lemma is one of the most important examples of a structure-versus-randomness theorem in combinatorics. A sunflower with r petals is a collection of r sets such that the intersection of each pair is equal to the intersection of all of them. The sunflower lemma of Erdös and Rado [9] roughly states that any sufficiently large w-uniform set system (of size about w^w) must contain a sunflower. A recent breakthrough result due to Alweiss et al. [1] proves the sunflower lemma with significantly improved parameters, making a huge step towards resolving the longstanding open problem of obtaining optimal parameters. A sequence of followup works [10, 29, 2] extended the technique and sharpened the obtained bounds.

Both the original sunflower lemma as well as Rossman's robust version [33] have played an important role in recent advances in theoretical computer science. Most famously, Razborov proved the first superpolynomial lower bounds for monotone circuits computing the Clique

[1] Here we restrict ourselves to lifting theorems in the setting of Boolean models of query complexity (e.g., decision trees, randomized decision trees). Interestingly *algebraic* lifting theorems which lift polynomial degree to an associated communication measure, exploit duality in order to give nonconstructive proofs of lifting (see e.g. [34, 28, 31])

function, using the sunflower lemma. It has also been a fundamental tool used to obtain a wide variety of other hardness results including: hardness of approximation, matrix multiplication, cryptography, and data structure lower bounds. (See the conference version of [1] for a nice survey of the many applications to Computer Science.)

Additionally, [26] established a connection between sunflowers and randomness extractors, which implicitly connected sunflowers to lifting theorems through the central notion of blockwise min-entropy. In particular they showed that if certain functions are extractors for blockwise min-entropy sources, then one can get improvements on the sunflower lemma. We close the loop by showing the other direction: we use the sunflower lemma to get lifting theorems. As a consequence of these two results together, certain improvements to either lifting theorems or sunflowers directly would imply an improvement in the other. We make this connection explicit in Section 6, while in Section 4 we make an explicit conjecture which would give such an improvement.

Gadget size

The second main goal of this paper is to open up a new avenue of attack towards proving lifting theorems with improved *gadget size* – one of the main challenges in the area. Gadget size is a fundamental parameter in lifting theorems and their applications. We define the gadget size of $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ as $\min(|\mathcal{X}|, |\mathcal{Y}|)$. In most applications, one loses factors that depend polynomially on the gadget size. An ideal lifting theorem - one with constant gadget size - would give a unified way to prove tight lower bounds in several models of computation. For example, the best known size lower bounds for extension complexity as well as monotone circuit size is $2^{\tilde{\Omega}(\sqrt{n})}$ [13, 19, 3]. Improving the gadget size from $\mathrm{poly}(n)$ to $O(1)$ (or even $\mathrm{poly}\log(n)$) would improve the best known lower bounds for extended formulations and monotone circuit size to $2^{\tilde{\Omega}(n)}$. [2]

Despite the tremendous progress in lifting theorems, most generic lifting theorems require gadget sizes that are polynomial in n.[3] Most recently, [6] reduced the gadget size to $n^{2+\epsilon}$ for any $\epsilon > 0$. It has remained an open problem to break through this quadratic barrier.

One of our main contributions is to cross the quadratic barrier firmly; our simplified proof immediately gives us a gadget of size $n^{1+\epsilon}$ for any $\epsilon > 0$. Our approach does not seem to have the same bottleneck as previous approaches and presents a way forward for obtaining lifting theorems for polylogarithmic gadget sizes (similar to the improvements made for the sunflower lemma in [1]; see Section 4). Furthermore, by inspecting the parameters of the argument, we can prove a "sliding" lifting theorem which allows us to make a tradeoff between the strength of our lower bound and the size of the gadget, down to a gadget of size $O(n \log n)$.

Dag-like lifting and other improvements

A further strength of our approach is that it can be adapted straightforwardly to prove a lifting theorem for *dag-like* communication protocols. Note that previous approaches such as those of [30], [17] do not extend to such protocols. Such a lifting theorem was first proven in [11], whose central lemma was built on the randomized lifting theorem of [18]. Our main

[2] Typically, a gadget of size q with n variables can lead to a lower bound of $2^{\Omega(n)}$ but on a combinatorial problem of size $N = nq$. So for instance, previous black-box lifting theorems would, in the best-case scenario, lead to a $2^{\tilde{\Omega}(N^{1/3})}$ lower bound on extension complexity for graph problems on N vertices [19]. Our new lifting theorem with a near-linear gadget size could lead to a $2^{\tilde{\Omega}(N^{1/2})}$ lower bound; independently, this lower bound was proven by [3].

[3] Some notable exceptions for models of communication with better gadget size are [34, 35, 16, 27].

contribution is a substantially simpler proof of their main lemma, which as in our tree-like lifting theorem, follows from a direct application of the sunflower lemma. Consequently, our dag-like lifting theorem also improves on the gadget size, from polynomial to near-linear size. We note that (almost) all of our results extend straightforwardly to the real communication setting as well.[4]

Our proof also immediately extends to give a new proof (with even tighter parameters) of [14] who prove deterministic lifting with the gadget size bounded by a polynomial in the query complexity of the outer function. This applies to situations such as fixed-parameter complexity, where the query complexity is modest, allowing us to lift problems whose query complexity and gadget size are comparable. Again our approach does not seem to suffer from a bottleneck, and improvements to this theorem would yield, e.g., stronger lower bounds on the automatizability of Cutting Planes [14].

Organization for the rest of the paper

After setting up the preliminaries in Section 2, in Section 3 we give an overview of our proof of the basic lifting theorem, as well as some ideas behind the extensions in the rest of the paper. In Section 4 we discuss a conjecture related to sunflowers which would make direct progress towards proving lifting with sublinear sized gadgets. In Section 5 we present our main contribution: a simplified proof of lifting via the sunflower lemma. Then for the remainder of the paper we investigate various extensions of this basic lifting theorem. In Section 6 we show that the gadget size m can be improved. Specifically in Subsection 6.1 we show that the basic lifting theorem can be done with $m = n^{1+\epsilon}$, and by sacrificing in the strength of the lifting theorem we can even push it down to $O(n \log n)$. In Subsection 6.2 we give a lifting theorem that scales with the decision tree complexity of the underlying function, instead of the number of variables n. We also briefly discuss the modifications needed to extend our results to the real communication setting in Subsection 6.3. For these extensions we make extensive reference to the basic lifting theorem in order to highlight how the proofs differ, and where necessary how our results fit into the context of their original proofs.

We refer readers to the full version of our paper for results on lifting dag-like query complexity to dag-like communication complexity.

2 Preliminaries

We will use n to denote the length of the input and $N \leq n$ to denote an arbitrary number less than n.[5] We also use m to denote an external parameter, and for this preliminaries section we will use $\mathcal{U}$ to denote an arbitrary set. We will mostly focus on two types of universes, $\mathcal{U}^N$ and $(\mathcal{U}^m)^N$. In the case of $\mathcal{U}^N$ we often refer to $i \in [N]$ as being a *coordinate*, while in the case of $(\mathcal{U}^m)^N$ we often refer to $i \in [N]$ as being a *block*. We will be primarily using terminology from previous lifting papers and computational complexity; for a connection to the language more commonly used in sunflower papers and combinatorics, see Appendix A in the full version of our paper.

[4] In most query-to-communication settings it is relatively simple to extend results for communication complexity to the real communication setting [23]; we refer readers to, e.g., [8, 11] for examples of these techniques and applications of lifting to real communication complexity.

[5] Later in the paper we will often be dealing with some subset of the input variables, and so N will generically refer to the number of variables we currently care about.

Basic notation

For a set $S \subseteq \mathcal{U}$ we write $\bar{S} := \mathcal{U} \setminus S$. For a set $\mathcal{U}$ and a set $I \subseteq [N]$ we say a string x is in $\mathcal{U}^I$ if each value in x is an element of $\mathcal{U}$ indexed by a unique element of I. For a string $x \in \mathcal{U}^N$ and $I \subseteq [N]$ we define $x[I] \in \mathcal{U}^I$ to be the values of x at the locations in I, and for a string $y \in (\mathcal{U}^m)^N$ and $I \subseteq [N]$, $\alpha \in [m]^I$ we define $y[I, \alpha] \in \mathcal{U}^I$ to be the values of y at the locations α_i for each $i \in I$. For a set $X \subseteq \mathcal{U}^N$ we define $X_I \subseteq \mathcal{U}^I$ to be the set that is the projection of X onto coordinates I, and for a set $Y \subseteq (\mathcal{U}^m)^N$ we define $Y_I \subseteq (\mathcal{U}^m)^I$ likewise. For a set system $\mathcal{F}$ of subsets of $\mathcal{U}$ and a set $S \subseteq \mathcal{U}$, we define $\mathcal{F}_{\bar{S}} := \{\gamma \setminus S : \gamma \in \mathcal{F}, S \subseteq \gamma\}$.

▶ **Definition 1.** *Let $\gamma \subseteq [mN]$. Treating each element in γ as being a pair (i, a) where $i \in [N]$ and $a \in [m]$, we say γ is over $(\mathcal{U}^m)^N$, meaning that for $s \in (\mathcal{U}^m)^N$ and each $(i, a) \in \gamma$ there is a corresponding element $s[i, a]$ from $\mathcal{U}$. We sometimes say (i, a) is a pointer. We say a set system $\mathcal{F}$ is over $(\mathcal{U}^m)^N$ if all sets in $\mathcal{F}$ are over $(\mathcal{U}^m)^N$.*

For γ over $(\mathcal{U}^m)^N$, γ is a block-respecting subset of $[mN]$ if γ contains at most one element per block, or in other words if $i \neq i'$ for all distinct $(i, a), (i', a') \in \gamma$. We can represent such γ by a pair (I, α), where $I \subseteq [N]$ and $\alpha \in [m]^I$; here γ chooses one element (indicated by α_i) from each block $i \in I$. A set system $\mathcal{F}$ over $(\mathcal{U}^m)^N$ is block-respecting if all elements $\gamma \in \mathcal{F}$ are block-respecting.

We say that a set $\rho \in \{0, 1, *\}^N$ is a *restriction*, or sometimes a *partial assignment*. We denote by $\text{free}(\rho) \subseteq [N]$ the variables assigned a star, and define $\text{fix}(\rho) := [N] \setminus \text{free}(\rho)$. If we have two restrictions ρ, ρ' such that $\text{fix}(\rho) \cap \text{fix}(\rho') = \emptyset$, then we define $\rho \cup \rho'$ to be the restriction which assigns $\text{fix}(\rho)$ to $\rho[\text{fix}(\rho)]$ and $\text{fix}(\rho')$ to $\rho'[\text{fix}(\rho')]$, with all other coordinates being assigned $*$.

In general in this paper we will use bold letters to denote random variables. For a set S we denote by $\boldsymbol{S} \in S$ the random variable that is uniform over S. For a block-respecting set system $\mathcal{F}$ over $(\mathcal{U}^m)^N$ and $I \subseteq [N]$, we denote by $\boldsymbol{\mathcal{F}}_I$ the marginal distribution over $\mathcal{F}_I$, where we remove all sets $\gamma \in \mathcal{F}$ which do not contain elements in all blocks I.

▶ **Definition 2.** *Let S be a set. For a random variable $\boldsymbol{s} \in S$ we define its* min-entropy *by $\mathbf{H}_\infty(\boldsymbol{s}) := \min_s \log(1/\mathbf{Pr}[\boldsymbol{s} = s])$. We also define the* deficiency *of $\boldsymbol{s}$ by $\mathbf{D}_\infty(\boldsymbol{s}) := \log|S| - \mathbf{H}_\infty(\boldsymbol{s}) \geq 0$.*

▶ **Definition 3.** *Let $\mathcal{F}$ be a block-respecting set system over $(\mathcal{U}^m)^N$. We define the* blockwise min-entropy *of $\mathcal{F}$ by $\min_{\emptyset \neq I \subseteq [N]} \frac{1}{|I|}\mathbf{H}_\infty(\mathcal{F}_\mathbf{I})$, or in other words the least (normalized) marginal min-entropy over all subsets I of the coordinates $[N]$.*

Search problems

A *search problem* is a relation $f \subseteq \mathcal{Z} \times \mathcal{O}$ such that for every $z \in \mathcal{Z}$ there exists some $o \in \mathcal{O}$ such that $(z, o) \in f$. Let $f(z) \neq \emptyset$ denote the set of all $o \in \mathcal{O}$ such that $(z, o) \in f$. Likewise a *bipartite search problem* is a relation $F \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ such that $F(x, y) \neq \emptyset$, where $F(x, y)$ is defined analogously to $f(z)$. We say that f is on $\mathcal{Z}$ and F is on $\mathcal{X} \times \mathcal{Y}$.

▶ **Definition 4.** *Let $m \in \mathbb{N}$. The index gadget, denoted $\textsc{Ind}_m$, is a Boolean function which takes two inputs $x \in [m]$ and $y \in \{0, 1\}^m$, and outputs $y[x]$. We will often have multiple separate instances of the index gadget; we use the notation $\textsc{Ind}_m^N$ to refer to the function which takes two inputs $x \in [m]^N$ and $y \in (\{0, 1\}^m)^N$ and outputs the Boolean string $(y[i, x_i])_{i \in [N]}$. For a search problem f with $\mathcal{Z} = \{0, 1\}^n$, the lifted search problem $f \circ \textsc{Ind}_m^n$ is a bipartite search problem defined by $\mathcal{X} := [m]^n$, $\mathcal{Y} := (\{0, 1\}^m)^n$, and $f \circ \textsc{Ind}_m^n(x, y) = \{o \in \mathcal{O} : o \in f(\textsc{Ind}_m^n(x, y))\}$. Following our existing convention, for a set of variables $J \subseteq [n]$ we write $\textsc{Ind}_m^J(x, y)$ to refer to the function $\textsc{Ind}_m^{|J|}((x_i)_{i \in J}, (y_i)_{i \in J})$.*

Intuitively, each $x \in \mathcal{X}$ can be viewed as a block-respecting subset over the universe $[mn]$ where n elements are chosen, one from each block of size m. For each $i \in [n]$, to determine the value of the variable z_i in the original problem f, we restrict ourselves to the i-th block of y and take the bit indexed by the i-th coordinate of x.

Consider a search problem $f \subseteq \{0,1\}^n \times \mathcal{O}$. A *decision tree* T is a binary tree such that each non-leaf node v is labeled with an input variable z_i, and each leaf v is labeled with a solution $o_v \in \mathcal{O}$. The tree T solves f if, for any input $z \in \{0,1\}^n$, the unique root-to-leaf path, generated by walking left at node v if the variable z_i that v is labeled with is 0 (and right otherwise), terminates at a leaf u with $o_u \in f(z)$. We define

$\mathbf{P}^{dt}(f) \coloneqq$ least *depth* of a decision tree solving f

Consider a bipartite search problem F. A *communication protocol* Π is a binary tree where now each non-leaf node v is labeled with a binary function g_v which takes its input either from $\mathcal{X}$ or $\mathcal{Y}$. This is informally viewed as two players Alice and Bob jointly computing a function, where Alice receives $x \in \mathcal{X}$ and Bob receives $y \in \mathcal{Y}$, and where at each node in the protocol either Alice or Bob computes $g_v(x)$ or $g_v(y)$, respectively, and "speaks" as to which child to go to, depending on whose turn it is. The protocol Π solves F if, for any input $(x,y) \in \mathcal{X} \times \mathcal{Y}$, the unique root-to-leaf path, generated by walking left at node v if $g_v(x,y) = 0$ (and right otherwise), terminates at a leaf u with $o_u \in F(x,y)$. We define

$\mathbf{P}^{cc}(F) \coloneqq$ least *depth* of a communication protocol solving F.

An alternative characterization of communication protocols, which will be useful for proving our main theorem, is as follows. Each non-leaf node v is labeled with a *(combinatorial) rectangle* $R_v = X_v \times Y_v \subseteq \mathcal{X} \times \mathcal{Y}$, such that if v_ℓ and v_r are the children of v, R_{v_ℓ} and R_{v_r} partition R_v. Furthermore this partition is either of the form $X_{v_\ell} \times Y_v \sqcup X_{v_r} \times Y_v$ or $X_v \times Y_{v_\ell} \sqcup X_v \times Y_{v_r}$. The unique root-to-leaf path on input (x,y) is generated by walking to whichever child v of the current node satisfies $(x,y) \in R_v$.

Sunflowers

Let $\mathcal{F}$ be a set system over some universe $\mathcal{U}$. We say that $\mathcal{F}$ is a *sunflower* if there exists some set $S \subseteq \mathcal{U}$ such that $\gamma_1 \cap \gamma_2 = S$ for any two distinct sets $\gamma_1, \gamma_2 \in \mathcal{F}$. We refer to the sets in $\mathcal{F}_{\overline{S}}$ as the *petals* and S as the *core*. The famed sunflower lemma of Erdős and Rado states the following.

▶ **Lemma 5** (Sunflower Lemma). *Let $s \in \mathbb{N}$ and let $k \in \mathbb{N}$. Let $\mathcal{F}$ be a set system over $\mathcal{U}$ such that a) $|\gamma| \leq s$ for all $\gamma \in \mathcal{F}$; and b) $|\mathcal{F}| \geq s!(k-1)^s$. Then $\mathcal{F}$ contains a sunflower with k petals.*

In this paper we will be mostly concerned with a set system that approximately reflects the behavior of a sunflower. Let $\mathcal{F}$ be a set system over some universe $\mathcal{U}$, and let $p, \kappa \in (0,1]$. We say that $\mathcal{F}$ is (p,κ)-*satisfying* if

$\mathbf{Pr}_{\boldsymbol{y} \subseteq_p \mathcal{U}}(\forall \gamma \in \mathcal{F} : \gamma \not\subseteq y) \leq \kappa$

where $\subseteq_p$ means that each element is added to y independently with probability p.

We say that $\mathcal{F}$ is a (p,κ)-*robust sunflower* (sometimes called an *approximate sunflower* or a *quasi-sunflower*) if it satisfies the following. Let $S = \cap_{T \in \mathcal{F}} T$ be the common intersection of all sets in $\mathcal{F}$. We require that $\mathcal{F}_{\overline{S}}$ is (p,κ)-satisfying. In other words,

$\mathbf{Pr}_{\boldsymbol{y} \subseteq_p \mathcal{U} \setminus S}(\forall \gamma \in \mathcal{F} : \gamma \setminus S \not\subseteq y) \leq \kappa.$

In this paper we will always be using $p = 1/2$, and so for convenience we simply write $\boldsymbol{y} \subseteq \mathcal{U} \setminus S$ instead of $\subseteq_{1/2}$ and call $\mathcal{F}$ an κ-robust sunflower instead of an $(1/2, \kappa)$-robust sunflower. An analogue of the classic sunflower lemma was proved for robust sunflowers by Rossman [33], and in a recent breakthrough result [1] (simplified in [29]) obtained an improvement in the parameters:

▶ **Lemma 6** (Robust Sunflower Lemma). *There exists an absolute constant K such that the following holds: Let $s \in \mathbb{N}$ and $\kappa > 0$. Let $\mathcal{F}$ be a set system over $\mathcal{U}$ such that a) $|\gamma| \leq s$ for all $\gamma \in \mathcal{F}$; and b) $|\mathcal{F}| \geq (K \log(s/\kappa))^s$. Then $\mathcal{F}$ contains a κ-robust sunflower.*

As a stepping stone they also prove an improvement on Lemma 6 assuming a condition called *spreadness*, but which we will state in the following way.

▶ **Lemma 7** (Blockwise Robust Sunflower Lemma). *There exists an absolute constant K such that the following holds: let $s \in \mathbb{N}$ and $\kappa > 0$. Let $\mathcal{F}$ be a block-respecting set system over $(\mathcal{U}^m)^N$ such that a) $|\gamma| \leq s$ for all $\gamma \in \mathcal{F}$; and b) $\mathcal{F}$ has blockwise min-entropy at least $\log(K \log(s/\kappa))$. Then $\mathcal{F}$ is κ-satisfying.*

In our main argument we will use a simple and general statement about the satisfiability of monotone CNFs in order to connect sunflowers to restrictions.

▷ Claim 8. Let $\mathcal{C} = C_1 \wedge \ldots \wedge C_m$ be a CNF on the variables $x_1 \ldots x_n$ such that no clause contains both the literals x_i and $\overline{x_i}$ for any i. Let $\mathcal{C}^{mon}$ be the result of replacing, for every i, every occurrence of x_i in $\mathcal{C}$ with $\overline{x_i}$.[6] Then

$$|\{x \in \{0,1\}^n : \mathcal{C}(x) = 1\}| \leq |\{x \in \{0,1\}^n : \mathcal{C}^{mon}(x) = 1\}|$$

Proof. Let $\mathcal{C}^i$ be the result of replacing every occurrence of x_i in $\mathcal{C}$ with $\overline{x_i}$. It is enough to show that for any i, $\mathcal{C}^i(x)$ is satisfied by at least as many assignments $\beta \in \{0,1\}^n$ to x as $\mathcal{C}(x)$ is, as we can then apply the argument inductively for $i = 1 \ldots n$. Let $\beta^{-i} \in \{0,1\}^{[n] \setminus \{i\}}$ be an assignment to every variable except x_i. We claim that for every β^{-i}, $\mathcal{C}^i(\beta^{-i}, x_i)$ is satisfied by at least as many assignments $\beta_i \in \{0,1\}$ to x_i as $\mathcal{C}(\beta^{-i}, x_i)$.

Since there are no clauses with both x_i and $\overline{x_i}$, each clause in $\mathcal{C}$ is of the form $x_i \vee A$, $\overline{x_i} \vee B$, or C, where A, B, and C don't depend on x_i; the corresponding clauses in $\mathcal{C}^i$ are $\overline{x_i} \vee A$, $\overline{x_i} \vee B$, and C. If $\mathcal{C}^i(\beta^{-i}, 1) = 1$, then $A(\beta^{-i}) = B(\beta^{-i}) = C(\beta^{-i}) = 1$ for all A, B, and C, and so $\mathcal{C}^i(\beta^{-i}, x_i)$ is always satisfied. If $\mathcal{C}^i(\beta^{-i}, 0) = 0$, then it must be that $C(\beta^{-i}) = 0$ for some C, and so $\mathcal{C}(\beta^{-i}, x_i)$ has no satisfying assignments. Finally assume neither of these cases hold, and so $\mathcal{C}^i(\beta^{-i}, 1) = 0$ and $\mathcal{C}^i(\beta^{-i}, 0) = 1$. Then it must be that either $A(\beta^{-i}) = 0$ for some A, in which case $\mathcal{C}(\beta^{-i}, 0) = 0$, or $B(\beta^{-i}) = 0$ for some B, in which case $\mathcal{C}(\beta^{-i}, 1) = 0$. Therefore $\mathcal{C}(\beta^{-i}, x_i)$ has at least one falsifying assignment, while $\mathcal{C}^i(\beta^{-i}, x_i)$ has exactly one. ◁

3 Proof overview

In this section we will sketch out the technical ideas that go into proving the basic deterministic lifting theorem, along with some of the innovations that have helped simplify the proof since [30]. We also sketch the changes that are required to prove our other lifting theorems, i.e. dag-like lifting, lifting with smaller gadgets (Subsection 6.1), and lifting whose gadget size scales with the decision tree depth (Subsection 6.2).

[6] Intuitively $\mathcal{C}^{mon}$ is the monotone version of $\mathcal{C}$, and note that it does not matter whether our monotone version has all variables occurring positively or negatively. This version will happen to be more suggestive later.

3.1 The basic lifting theorem

The following is our basic deterministic lifting theorem. For simplicity of exposition we focus on a concrete gadget size $m = n^{1.1}$, and later show how to adjust the parameters to get $m = n^{1+\epsilon}$ for any $\epsilon > 0$.

▶ **Theorem 9** (Basic Lifting Theorem). *Let f be a search problem over $\{0,1\}^n$, and let $m = n^{1.1}$. Then*

$$\mathbf{P}^{cc}(f \circ \mathit{IND}_m^n) \;=\; \mathbf{P}^{dt}(f) \cdot \Theta(\log m)$$

We prove that a) a decision tree of depth d for f can be simulated by a communication protocol of depth $O(d \log m)$ for the composed problem $f \circ \textsc{Ind}_m^n$, and b) a communication protocol of depth $d \log m$ for the composed problem $f \circ \textsc{Ind}_m^n$ can be simulated by a decision-tree of depth $O(d)$ for f. Let $\{z_i\}_i$ be the variables of f and let $\{x_i\}_i, \{y_i\}_i$ be the variables of $f \circ \textsc{Ind}_m^n$; recall that each z_i takes values in $\{0,1\}$, x_i takes values in $[m]$, and y_i takes values in $\{0,1\}^m$. The forward direction of the theorem is obvious: given a decision tree T for f, Alice and Bob can simply trace down T and compute the appropriate variable z_i at each node $v \in T$ visited, spending $\log m$ bits to compute $\textsc{Ind}_m(x_i, y_i)$ to do so. Thus we focus on simulating a communication protocol Π of depth $d \log m$.

High level idea: Tracing the "important" coordinates

What does it mean to "simulate" a communication protocol for $f \circ \textsc{Ind}_m^n$ by a decision tree for f? When we look at the communication matrix for $f \circ \textsc{Ind}_m^n$, we label the (x, y) entry with the solutions $o \in \mathcal{O}$ satisfying $(x, y) \in (f \circ \textsc{Ind}_m^n)^{-1}(o)$. However we have no control over f, and so in some sense what we really care about is the z variables. So instead we will think of the (x, y) entry as storing $z = \textsc{Ind}_m^n(x, y)$, and then instead of having to reason about f we can ask "what does the set of z values that make it to any given leaf of Π look like?"

For each leaf we want to split the coordinates into two categories: the "important" coordinates where the z values are (jointly) nearly fixed, and the rest where every possibility is still open. Hopefully this means that knowing the important coordinates is enough to declare the answer. Applying the same logic to the internal nodes we can query variables as they cross the threshold from unfixed to important, which leads us down to the leaves in a natural way. To do this efficiently, we have to define "importance" in a way that satisfies all these conditions while also ensuring that no leaf contains more than $O(d)$ important variables.

Blockwise min-entropy

In order to prove this formally, we will trace down the communication protocol node by node, at each step looking for the z variables that are fairly "well determined" by the current rectangle. We focus exclusively on the X side of the current rectangle, since Y is so large that it would take $m \gg n \log m$ rounds just to fix a single y_i. Our measure of coordinate i being well-determined will be the min-entropy of the uniform distribution on X marginalized to the coordinate i. At the start of the protocol, every coordinate will have min-entropy $\log m$, while each round can drop the min-entropy of a coordinate by at most 1. Once a coordinate i falls below a certain min-entropy threshold, say $0.95 \log m$, we can consider the coordinate important enough to query in the decision tree. We can think of Π as having "paid" for the coordinate i; since min-entropy can only drop by 1 each round, it took $0.05 \log m$ rounds

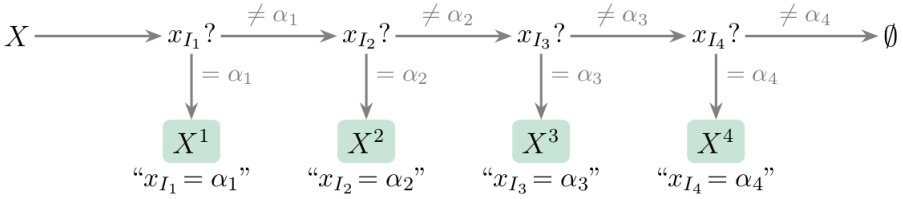

Figure 1 Rectangle Partition procedure (figure from [18]).

to reduce the entropy of X_i to below the threshold. Since we ultimately want to shave an $\Omega(\log m)$ factor off the height of the communication protocol in our decision tree, once Π has spent $\Omega(\log m)$ steps transmitting information about coordinate i we can feel satisfied giving up the rest of the information about X_i and Y_i for free.

In fact we will use the generalization of min-entropy to blockwise min-entropy, and so instead of tracking individual coordinates we stop whenever a *set* of coordinates I has a joint assignment $x[I] = \alpha$ which violates $0.95 \log m$ blockwise min-entropy. In addition we will use an entropy-restoring procedure called the *rectangle partition*. Whenever we find an assignment $x[I_1] = \alpha_1$ that "violates" $0.95 \log m$ blockwise min-entropy – in other words, I_1, α_1 such that $\mathbf{Pr}(x[I_1] = \alpha_1) > 2^{-0.95|I_1| \log m}$ – we split X into two pieces: $X^1 = \{x : x[I_1] = \alpha_1\}$ and $X - X^1 = \{x : x[I_1] \neq \alpha_1\}$. Next we repeat for $X - X^1$; if there is an assignment $x[I_2] = \alpha_2$ that violates $0.95 \log m$ blockwise min-entropy, then we split $X - X^1$ into X^2 and $(X - X^1) - X^2$. We repeat until there are no more assignments, and then we can make a decision to pick one and query $z[I_j]$.[7]

We now describe our high level procedure using this partitioning subroutine. In addition to the rectangles R_v at each node v of Π, we maintain a subrectangle $R = X \times Y$ – initially full – which will be our guide for how to proceed down Π. Starting at the root, we go down to the child v with the larger rectangle $R \cap R_v$ – which guarantees that the blockwise min-entropy of $\boldsymbol{X} \cap \boldsymbol{X}_v$ goes down by at most 1 from $\boldsymbol{X}$ – and update R to be R_v for whichever child v we picked. We continue going down the protocol and taking the child with the larger intersection with R until we find that a set of coordinates has blockwise min-entropy less than $0.95 \log m$ in R. After running the rectangle partition, we will need to decide which assignment to query; ultimately once we've chosen the assignment $x[I_j] = \alpha_j$, we will query $z[I_j]$ and restrict R to be consistent with the result. Our first key lemma states that if we run the rectangle partition on X such that $\boldsymbol{X}$ has blockwise min-entropy at least $0.95 \log m$ on $\overline{I_j}$, and Y has size at least $2^{mn - n \log m}$, then there is always some choice of j such that for every possible result $z[I_j] = \beta_j$, the resulting rectangle R is large on the Y side.

As mentioned before, our choice of min-entropy will be enough to guarantee that at every step, our rectangle R will have every assignment to z consistent with the current path in the decision tree available. When we reach a leaf ℓ in Π and have queried some coordinates I, we want to show that we know enough information to output an answer in the decision tree. To do this we show that we can output the same answer o as Π outputs at ℓ. Our second key lemma states that if X and Y are fixed on the coordinates $J \subseteq [n]$, $\boldsymbol{X}$ has min-entropy at least $0.95 \log m$ on $\overline{J}$, and Y has size at least $2^{mn - n \log m}$, then $\textsc{Ind}_m^{\overline{J}}(X, Y) = \{0,1\}^{\overline{J}}$; thus $R \subseteq R_\ell$ has every option left for the z variables in the coordinates $\overline{J}$. Thus if we consider any assignment α to all the z values consistent with the assignment to J in the current path in our decision tree, there must be some $(x, y) \in R_\ell$ such that $\textsc{Ind}_m^n(x, y) = \alpha$, and so o must be a correct answer for $z = \alpha$ for any α consistent with our path in the decision tree.

[7] As described in Subsection 5.1, unlike in [18] in our proof we truncate this procedure before X is empty, but the same basic principle applies.

Key lemmas through sunflowers

Up until this point, everything we've stated is as it appears in [18]. For our new proof we unify our two key lemmas with a more challenging but ultimately more straightforward lemma: given X and Y such that $\boldsymbol{X}_{\bar{J}}$ has high blockwise min-entropy and Y is large, there is some $x^* \in X$ such that x^* by itself has the full range of the index gadget available, or in other words $\textsc{Ind}_m^{\bar{J}}(x^*, Y) = \{0,1\}^{\bar{J}}$.[8] Given this statement both claims are easy to see. In the rectangle partition, for every I_j, α_j such that some value β_j has few ys consistent with it, remove those ys from Y;[9] by the lemma there is some x^* which still has the full range of values available. Thus whichever X^j part that x^* appears in must not have had any ys thrown out of Y on its account, and so $y[I_j, \alpha_j]$ should be fairly uniform. At the leaves, if a single x^* gives the full range, then so does $X \ni x^*$.

Despite seeming more challenging, this unifying lemma follows almost immediately from the Lemma 7. To illustrate this with a simple (but ultimately completely general) case, assume the all-ones vector is missing from $\textsc{Ind}_m^{\bar{J}}(x, Y)$ for all x, or in other words there is no (x, y) such that $y[x] = 1^{\bar{J}}$. Consider the universe $[mn]$, and let S_x be the set of size $|\bar{J}|$ defined by the values x points to. Since $\boldsymbol{X}$ has high blockwise min-entropy, by Lemma 7 a random set $\mathbf{S}_y \subseteq [mn]$ will contain some S_x with high probability. If we look at the incidence vector of our random S_y, it is a string $y \in \{0,1\}^{mn} = (\{0,1\}^m)^n$, and for S_y to not contain S_x is equivalent to saying that $y[x] \neq 1^{\bar{J}}$. Thus $\mathbf{Pr}_{\boldsymbol{y}}[\forall x : y[x] \neq 1^{\bar{J}}]$ is very low, or in other words a sufficiently large set $Y \subseteq (\{0,1\}^m)^N$ must contain some y such that $y[x] = 1^{\bar{J}}$ for some x. This gives us our contradiction since we assumed Y was large.

Recap

Summing up, our final procedure will be as follows. For all $v \in \Pi$ let R_v be the rectangle associated with node v, let $R = [m]^n \times (\{0,1\}^m)^n$, and at the start of the simulation, let $v = root$. At each step we go to the child v' of v maximizing $R \cap R_{v'}$. Then we perform the rectangle partition on X, query $z[I_j]$ for I_j from the key lemma (possibly empty) to get the answer β_j, and fix R to be consistent with $x[I_j] = \alpha_j$ and $y[I_j, \alpha_j] = \beta_j$. As an invariant we have that at the start of each round R is fixed on the coordinates J queried in our decision tree, $\boldsymbol{X}_{\overline{J}}$ has blockwise min-entropy $0.95 \log m$, and $|Y| \geq 2^{mn - n \log m}$. When we reach a leaf we apply the key lemma one last time to get that all possible z values consistent with our path in the decision tree are still available, and so we can return the same answer as Π.

3.2 Further results

Dag-like lifting

In [11] they show that a lifting theorem exists for the appropriate notions of dag-like query complexity (decision dags) and communication complexity (rectangle dags) as well. This proof is very similar to our basic lifting theorem [10], and so we reprove this theorem using our new sunflower strategy. See the full version of our paper for all definitions and the exact statement of the dag-like lifting theorem.

---

[8] This lemma was also proven in [11], as it was necessary to prove their result for lifting in the dag-like case. We use it to simplify the proof of the tree-like result as well.

[9] We note one other seemingly minor but very useful feature of our proof, which is that our union bound for the sets of y's removed during the rectangle partition does not require a large gadget size. This removes the other bottleneck for the gadget size, and consolidates all issues of the choice of m to Lemma 7.

[10] In fact the move from min-entropy to blockwise min-entropy was necessary for this generalization, and so while we feel the approached outlined above simplifies the original proofs of [30, 17] for tree-like lifting, this was not the original motivation.

The main idea is the same as before; at every step our rectangle R has some number of variables fixed, while we have $0.95 \log m$ blockwise min-entropy on the rest. When we move to a new node we apply the rectangle partition to get a list of too-likely assignments, using our key lemma to show that at least one will be safe to query. The main difference from the tree-like proof is that Π is allowed to "forget" information along a path to the leaves, which potentially allows it to run for many more rounds. To handle this, when moving to node v we apply the rectangle partition to R_v itself instead of $R \cap R_v$, and we include all coordinates instead of just the ones unfixed in R. When we find a good assignment and associated rectangle in $R \cap R_v$, we set R to be this new rectangle and allow our decision dag to forget any assignments it was remembering that are not fixed in the new assignment. We use the same key lemma as in the tree-like case, first to make sure that after we remove all ys where $\{y \in Y : y[I_j, \alpha_j] = \beta_j\}$ is too small – and for technical reasons we now do this in advance to every R_v in the protocol in a bottom-up fashion, creating a not-too-large set of bad y values that we throw out before even starting – we can still find a good row x^*; and second, to make sure that at the leaves we are done.

Even smaller gadgets

The reader may have noticed that the choice of the constant 0.95 in the blockwise min-entropy threshold $0.95 \log m$ was arbitrary; the important thing is that the number of steps of the communication protocol required to reduce the blockwise min-entropy of our rectangle below the threshold is $\Omega(\log m)$ per coordinate. On the other hand, our gadget size $m = n^{1.1}$ will directly be a function of this constant: in order to apply Lemma 7 we need that $0.95 \log m > \log(n \log m) + O(1)$, or in other words $m^{0.95} > O(n \log m)$. Taking these two facts together, it turns out that taking the constant 0.95 arbitrarily close to 1 allows us to drive down the gadget size m in tandem, to $n^{1+\epsilon}$ for any $\epsilon > 0$.

In fact we can even take our blockwise min-entropy threshold to be $(1 - o(1)) \log m \geq O(n \log m)$ and get even closer to a linear sized gadget. However now we no longer have that the number of steps of the communication protocol required to reduce the blockwise min-entropy of our rectangle below the threshold is $\Omega(\log m)$ per coordinate. Thus we get a smooth tradeoff between the gadget size and the strength of the simulation, up to $m = O(n \log n)$.

▶ **Theorem 10 (Minimum Gadget Size Lifting Theorem).** *Let f be a search problem over $\{0,1\}^n$, and let $m = \Omega(n \log n)$. Then*

$$\mathbf{P}^{cc}(f \circ \mathit{IND}_m) \;\geq\; \Omega(\mathbf{P}^{dt}(f))$$

An analoguous theorem hold for dag-like lifting. We make these statements more formal in Subsection 6.1.

Lifting that scales with the query complexity

In many applications (e.g. tree-like Cutting Planes automatizability lower bounds) we want to lift very small decision tree lower bounds to communication lower bounds, and even having a gadget of size $m = n^{\epsilon}$ is too large to get anything useful. In [14] they prove a lifting theorem where the only restriction on m is that it be polynomial in $\mathbf{P}^{dt}(f)$, rather than having any direct dependence on n. We refer to this as *graduated lifting*.

In Subsection 6.2 we reprove this theorem for both tree-like lifting and dag-like lifting, the latter of which is new. In fact, almost nothing is required beyond the basic proofs, only a small observation in the choice of parameters for our unified key lemma. As a result we also

push the gadget size down to $m = (\mathbf{P}^{dt}(f))^{1+\epsilon}$ (and the equivalent statement for dag-like lifting), which generalizes all our previous results. There is a minor catch due to the case of the leaves, which restricts us to choosing $m = \Omega(\log^{1+\epsilon} n)$ unless we have some (natural) structure on the type of search problem we are lifting.[11]

Real lifting

Finally it is fairly trivial to extend all of our results to the real communication setting, further generalizing all previous results. The only result which cannot be extended is the case of dag-like graduated lifting, although it is not clear that this cannot be achieved with a small modification to the proof. See Subsection 6.3 for more details.

4 Towards polylogarithmic gadget size

As discussed above, the key issue in improving gadget size with current techniques is to prove the extractor or disperser like analogues of our key lemma (Lemma 12) for small gadget sizes. To this end, we pose the following concrete conjecture:

▶ **Conjecture 11.** *There exist a constant c such that for all large enough m the following holds. Let X, Y be distributions on $[m]^N$, $(\{0,1\}^m)^N$ with entropy deficiency at most Δ each. Then, $\textsc{Ind}_m^N(X,Y)$ contains a subcube of co-dimension at most $c\Delta$. That is, there exists $I \subseteq [N], |I| \leq c\Delta$, and $\alpha \in \{0,1\}^I$ such that for all $z \in \{0,1\}^N$ with $z_I = \alpha$, we have*

$$\mathbf{Pr}_{X,Y}[\textsc{Ind}_m^N(X,Y) = z] > 0.$$

Proving the above statement seems necessary for obtaining better lifting theorems with current techniques. Further, while there are other obstacles to be overcome, proving the conjecture for smaller gadget-sizes would be a significant step toward improving gadget size (e.g., at least in the non-deterministic setting as considered in [15]).

The robust-sunflower theorem of [1] can be seen as proving a related statement: For gadget-size $m = \text{poly}(\log n)$, if X has deficiency at most Δ, Y is the p-biased distribution, then we get the stronger guarantee that for some $I \subseteq [n], |I| = O(\Delta), \alpha \in \{0,1\}^I$ we have that for all z with $z_I = \alpha_I$, $\mathbf{Pr}_Y[\exists x \in X, \textsc{Ind}_m^n(X,Y) = z] = 1 - o(1)$. Note that the conclusion is stronger in the latter statement compared to the conjecture (the conjecture only asks for non-zero probability); however, the assumption on Y is incomparable in the robust-sunflower lemma (i.e., Y is a p-biased distribution, whereas in the conjecture Y has high min-entropy). Nevertheless, the present proof uses the robust sunflower lemma to prove the conjecture for $m = O(n \log n)$, whereas previous techniques needed $m \gg n^2$. We believe that these arguments could be useful in proving the above conjecture when the gadget-size is $m = \text{poly}(\log n)$.

5 The basic lifting theorem: full proof

To prove Theorem 9, we prove that if there exists a communication protocol Π of depth $d \log m$ for the composed problem $f \circ \textsc{Ind}_m^n$, then there exists a decision tree of depth $O(d)$ for f; the other direction is trivial as a communication protocol can simply compute each

[11] Such a restriction was also inherent in [14], who work with the canonical search problem on unsatisfiable CNFs; this is an example of a class which has such structure.

variable queried by the decision tree.[12] Our proof will follow the basic structure of previous works [18, 11]. We first define a procedure, called the rectangle partition, which forms the main technical tool in our simulation. We then prove that with this tool and a few useful facts about its output, we can efficiently simulate the protocol Π by a decision tree T, using a number of invariants to show the efficiency and correctness of T.

Before we begin, we prove a very useful lemma that shows that if $\boldsymbol{X}$ has high blockwise min-entropy outside some set of coordinates J, and furthermore Y is large, then it's possible to find an $x^* \in X$ such that the full image of the index gadget is available to x^* outside J, or in other words $\textsc{Ind}_m^{\bar{J}}(x^*, Y) = \{0,1\}^{\bar{J}}$. This appears as Lemma 7 in [11] for dag-like lifting and is stronger than is necessary for proving Theorem 9, but the proof highlights our new counting strategy and will be a useful tool throughout the rest of the paper.[13] We also emphasize that this is the only place in the proof of Theorem 9 where we use the size of the gadget.

▶ **Lemma 12** (Full Range Lemma). *Let $m \geq n^{1.1}$ and let $J \subseteq [n]$. Let $X \times Y \subseteq [m]^{\bar{J}} \times (\{0,1\}^m)^n$ be such that $\boldsymbol{X}$ has blockwise min-entropy at least $0.95 \log m - O(1)$ and $|Y| > 2^{mn - 2n \log m}$. Then there exists an $x^* \in X$ such that for every $\beta \in \{0,1\}^{\bar{J}}$, there exists a $y_\beta \in Y$ such that $\textsc{Ind}_m^{\bar{J}}(x^*, y_\beta) = \beta$.*

Proof. Assume for contradiction that for all x there exists a $\beta_x \in \{0,1\}^{\bar{J}}$ such that $|\{y \in Y : y[x] = \beta_x\}| = 0$, or in other words for all $(x,y) \in X \times Y$, $y[x] \neq \beta_x$. Consider the CNF over $y_1 \dots y_{mn}$ where clause C_x is the clause uniquely falsified by $y[x] = \beta_x$; then by Claim 8 we see that $|\{y \in (\{0,1\}^m)^n : \forall x, y[x] \neq \beta_x\}|$ is maximized when $\beta_x = 1^{\bar{J}}$. Thus because $Y \subseteq (\{0,1\}^m)^n$,

$$|\{y \in Y : \forall x, y[x] \neq \beta_x\}| \leq |\{y \in (\{0,1\}^m)^n : \forall x, y[x] \neq 1^{\bar{J}}\}|$$

Consider the space $[mn]$ where each element is indexed by $(i, \alpha) \in [n] \times [m]$. For each $x \in X$, let $S_x \subseteq [mn]$ be the set defined by including (i,α) iff $x[i] = \alpha$, and let $\mathcal{S}_X = \{S_x : x \in X\}$. By the fact that $m^{0.95} \gg O(n \log m)$ and $|\bar{J}| \leq n$, $\mathcal{S}_{\mathbf{X}}$ has blockwise min-entropy $0.95 \log m - O(1) > \log(O(n \log m)) > \log(K \log(|\bar{J}|/\kappa))$, where $\kappa := 2^{-3n \log m}$ and K is the constant given by Lemma 7. Thus we can apply Lemma 7 to $\mathcal{S}_X$ and get that $\mathbf{Pr}_{\mathbf{S}_{\mathbf{y}} \subseteq [mn]}(\forall S_x \in \mathcal{S}_X, S_x \not\subseteq S_y) \leq \kappa$,[14] and if we look at y as being the indicator vector for S_y then we get that $\mathbf{Pr}_{\boldsymbol{y} \sim \{0,1\}^{mn}}(\forall x \in X, y[x] \neq 1^{\bar{J}}) \leq \kappa$. Thus by counting we get

$$\begin{aligned} |Y| &= |\{y \in Y : \forall x, y[x] \neq \beta_x\}| \\ &\leq |\{y \in (\{0,1\}^m)^n : \forall x, y[x] \neq 1^{\bar{J}}\}| \\ &\leq \kappa \cdot 2^{mn} = 2^{mn - 3n \log m} \end{aligned}$$

which is a contradiction as $|Y| > 2^{mn - 2n \log m}$ by assumption. ◀

[12] We can assume that $d = o(n)$ as the theorem is trivial otherwise, but this fact will not be necessary for our proof.

[13] While we simplify things in this section by using $m = n^{1.1}$, our improved gadget size (see Section 6) crucially uses the improvements in Lemma 7 over the basic Lemma 6; the same improvements also give us a very short proof of our main lemma. However, these improvements aren't strictly necessary for our techniques; using the parameters of the original robust sunflower from [33], we obtain a gadget size of $n^{2+\epsilon}$, matching previous constructions.

[14] Recall that it does not matter that S_y is not necessarily block-respecting.

5.1 Density-restoring partition

Before going into the simulation, we define our essential tool, which is usually called the *density-restoring partition* or *rectangle partition* as per [18]. To understand how this will be used to define our core invariant on rectangles $X \times Y$, we need the following definition. Intuitively it states that there is some set of coordinates $J \subseteq [n]$ such that $\textsc{Ind}_m^n(X, Y)$ is fixed on J and "very unfixed" on $\bar{J}$. For the rest of this section, recall that $d \leq n$ is a parameter such that Π has depth $d \log m$.

▶ **Definition 13.** *Let m, n, d be as defined above, and let $\rho \in \{0, 1, *\}^n$ be a partial assignment with $J := \text{fix}(\rho) \subseteq [n]$, A rectangle $R = X \times Y \subseteq [m]^n \times (\{0,1\}^m)^n$ is ρ-structured if the following conditions hold:*

- $\textsc{Ind}_m^J(X_J, Y_J) = \{\rho[J]\}$
- *X_J is fixed to a single value α, and $\boldsymbol{X}_{\bar{J}}$ has blockwise min-entropy at least $0.95 \log m$*
- $|Y| \geq 2^{mn - d \log m - |J| \log m}$

If the second condition only holds for $0.95 \log m - O(1)$, we say R is ρ-almost structured.

In this section our goal will be to "restore" an almost-good (almost structured) rectangle R into a good (structured) rectangle R' inside it, fixing coordinates as necessary. Let $J \subseteq [n]$, let ρ be some restriction fixing exactly the coordinates in J, and let $X \times Y \subseteq [m]^n \times (\{0,1\}^m)^n$ be ρ-almost structured. Our goal will be to output a set of rectangles $X^j \times Y^{j,\beta}$ which cover most of $X \times Y$ such that each $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$-structured for some $\rho^{j,\beta}$ extending ρ.

To perform the partition we will need to find the sets $X^j \times Y^{j,\beta}$ along with a corresponding assignment $\rho^{j,\beta}$ for which they are $\rho^{j,\beta}$-structured. This is done in two phases. Our goal in Phase I will be to break up X into disjoint parts X^j, such that each X^j is fixed on some set $I_j \subseteq \bar{J}$ and has blockwise min-entropy $0.95 \log m$ on $\bar{J} \setminus I_j$ – hence this partition is "density-restoring" when $\boldsymbol{X}$ starts off with blockwise min-entropy below $0.95 \log m$. To do this, the procedure iteratively finds a maximal partial assignment (I_j, α_j) such that the assignment $x[I_j] = \alpha_j$ violates $0.95 \log m$ blockwise min-entropy in $\boldsymbol{X}$, splits the remaining X into the part X^j satisfying this assignment and the part $X \setminus X^j$ not satisfying it, and recurses on the latter part. We do this until we've covered at least half of X by X^j subsets.

Our goal in Phase II will be to break up Y into disjoint parts $Y^{j,\beta}$ for each X^j from Phase I, such that each $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$-structured for some restriction $\rho^{j,\beta}$. We already have the blockwise min-entropy of X^j in the coordinates $\bar{J} \setminus I_j$ by our first goal, and clearly we will choose $\rho^{j,\beta}$ such that $\text{fix}(\rho^{j,\beta}) = J \cup I_j$ for each j. Thus we need to fix the coordinates of Y within the blocks I_j, and within each $Y^{j,\beta}$ it should be the case that $y[I_j, \alpha_j] = \beta$ for all $y \in Y^{j,\beta}$, at which point $\rho^{j,\beta}$ can be fixed to β on I_j and left free everywhere else in $\bar{J}$ (with the coordinates of J being fixed by assumption).

Our algorithm is formally described in Algorithm 1. Let $X \times Y$ be ρ-almost structured for some ρ with $\text{fix}(\rho) = J$ where $|J| = O(d)$, and let $\mathcal{F}$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$ be the result of Algorithm 1 on $X \times Y$. Recall that our goal was to break $X \times Y$ up into $\rho^{j,\beta}$-structured rectangles $X^j \times Y^{j,\beta}$; the following simple claims show that the obvious choice of $\rho^{j,\beta}$ achieves two of the three conditions needed (outside of the part of X that we never touch before the procedure ends).

▷ Claim 14. For all j and for all $\beta \in \{0,1\}^{I_j}$, define $\rho^{j,\beta} \in \{0,1,*\}^n$ to be the restriction extending ρ by $\rho^{j,\beta}[I_j] = \beta$. Then $X^j_{I_j} = \{\alpha_j\}$ and $\textsc{Ind}_m^{J \cup I_j}(X^j, Y^{j,\beta}) = \rho^{j,\beta}[J \cup I_j]$.

Proof. By definition X^j is fixed to α_j on the coordinates I_j, while $Y^{j,\beta}$ only contains values y such that $y[\alpha_j] = \beta$. ◁

Algorithm 1 Rectangle Partition.

Initialize $\mathcal{F} = \emptyset$, $j = 1$, and $X^{\geq 1} := X$;
PHASE I (X^j): while $|X^{\geq j}| \geq |X|/2$ **do**
  Let I_j be a maximal subset of $\bar{J}$ such that $\boldsymbol{X}^{\geq j}$ violates $0.95 \log m$-blockwise min-entropy on I_j, or let $I_j = \emptyset$ if no such subset exists;
  Let $\alpha_j \in [m]^{I_j}$ be an outcome such that $\mathbf{Pr}_{x \sim \boldsymbol{X}^{\geq j}}(x[I_j] = \alpha_j) > 2^{-0.95|I_j| \log m}$;
  Define $X^j := \{x \in X^{\geq j} : x[I_j] = \alpha_j\}$;
  Update $\mathcal{F} \leftarrow \mathcal{F} \cup \{(I_j, \alpha_j)\}$, $X^{\geq j+1} := X^{\geq j} \setminus X^j$, and $j \leftarrow j + 1$; [15]
end
PHASE II ($Y^{j,\beta}$): for $(I_j, \alpha_j) \in \mathcal{F}$, $\beta \in \{0,1\}^{I_j}$ **do**
  Define $Y^{j,\beta} := \{y \in Y : y[I_j, \alpha_j] = \beta\}$;
end
return $\mathcal{F}$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$;

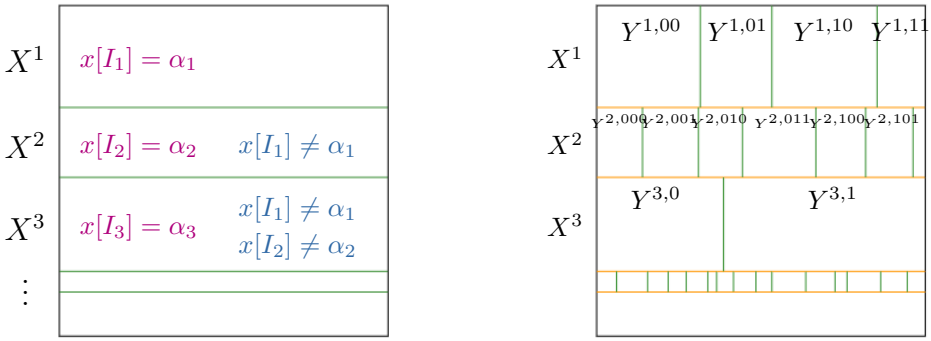

Figure 2 Phases I and II of Algorithm 1. In each $X^j \times Y^{j,\beta}$, $x[I_j]$ is fixed to α_j and $y[I_j]$ is fixed so that $\textsc{Ind}_m^{I_j}(X^j_{I_j}, Y^{j,\beta}_{I_j}) = \beta$.

▷ Claim 15. For all j, $\boldsymbol{X}^j_{J \cup I_j}$ has blockwise min-entropy at least $0.95 \log m$.

Proof. Assume for contradiction that $I^* \subseteq \bar{J} \setminus I_j$ such that $\boldsymbol{X}^j$ violates $0.95 \log m$-blockwise min-entropy on I^*, and let α^* be an outcome witnessing this. Then

$$\begin{aligned} \mathbf{Pr}_{x \sim \boldsymbol{X}^{\geq j}}(x[I_j] = \alpha_j \wedge x[I^*] = \alpha^*) &> 2^{-0.95|I_j| \log m} \cdot \mathbf{Pr}_{x \sim \boldsymbol{X}^j}(x[I^*] = \alpha^*) \\ &> 2^{-0.95|I_j| \log m - 0.95|I^*| \log m} = 2^{-0.95|I_j \cup I^*| \log m} \end{aligned}$$

which contradicts the maximality of I_j. ◁

Claims 14 and 15 do not use the fact that $X \times Y$ was ρ-almost structured, while our next two claims will. Before moving to the third condition, the size of $Y^{j,\beta}$, we show that the deficiency of each $\boldsymbol{X}^j$ drops by $\Omega(|I_j| \log m)$. This will be used later to show the efficiency of our simulation.

▷ Claim 16. For all $(I_j, \alpha_j) \in \mathcal{F}$, $\mathbf{D}_\infty(\boldsymbol{X}^j) \leq \mathbf{D}_\infty(\boldsymbol{X}) - 0.05|I_j| \log m + 1$.

Proof. By our choice of (I_j, α_j) it must be that $|X^j| = |X^{\geq j}| \cdot \mathbf{Pr}_{x \sim \boldsymbol{X}^{\geq j}}(x[I_j] = \alpha_j) \geq |X^{\geq j}| \cdot 2^{-0.95 \log m}$. Then by the fact that X^j is fixed on $J \cup I_j$ and and X is fixed on J,

$$\begin{aligned}
\mathbf{D}_\infty(\boldsymbol{X}^j) &= |\overline{J \cup I_j}| \log m - \log |X^j| \\
&\leq (n - |J \cup I_j|) \log m - \log(|X^{\geq j}| \cdot 2^{-0.95|I_j| \log m}) \\
&\leq ((n - |J|) - |I_j|) \log m - \log |X^{\geq j}| + 0.95|I_j| \log m - \log |X| + \log |X| \\
&= (|\bar{J}| \log m - \log |X|) - 0.05|I_j| \log m + \log(|X|/|X^{\geq j}|) \\
&\leq \mathbf{D}_\infty(\boldsymbol{X}) - 0.05|I_j| \log m + 1
\end{aligned}$$

where the last step used the fact that $|X^{\geq j}| \geq |X|/2$, since we terminate as soon as $|X^{\geq j}| < |X|/2$ at the start of the j-th iteration. ◁

For our last lemma before going into the simulation, instead of showing that $|Y^{j,\beta}|$ is large for *every* j and every β, we want to show that $|Y^{j,\beta}|$ is large for *some* j and every β. If every β were equally likely then $|Y^{j,\beta}| \approx |Y|/2^{|I_j|}$; for us it is enough that the smallest $Y^{j,\beta}$ be has size at least $|Y|/2^{|I_j| \log m}$. For convenience we redefine X to only be the union of the X^j parts – since we terminate after $|X^{\geq j}| < |X|/2$ we can do this and only decrease the blockwise min-entropy of $\boldsymbol{X}$ by 1 – and furthermore we restrict down to the free coordinates $\bar{J}$.

We assume otherwise, and that for every j there exists a bad β_j for which Y^{j,β_j} is too small. We split Y into two parts: y's that are in *some* bad Y^{j,β_j}, and y's that are in *no* bad Y^{j,β_j}. Our contradiction will be to show that both sets are much smaller than $|Y|/2$. While this strategy was implicit in previous works, our contribution in this paper is to improve both. For the first set, we use a simple but novel union bound argument which works *independent of the gadget size* m; previous union bound strategies relied on the fact that $m = \text{poly}(n)$. Bounding the second set is our central contribution, and is a fairly direct application of Lemma 12.

▶ **Lemma 17.** *Let $X \times Y$ be ρ-almost structured for some ρ with $\text{fix}(\rho) = J \subseteq [n]$ and let $\mathcal{F}$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$ be the result of Algorithm 1 on $X \times Y$. Let $X' := (\cup_j X^j)_{\bar{J}}$ be such that $\boldsymbol{X}'$ has blockwise min-entropy $0.95 \log m - O(1)$, and let Y be such that $|Y| \geq 2^{mn - d \log m - |J| \log m}$. Then there is a j such that for all $\beta \in \{0,1\}^{I_j}$,*

$$|Y^{j,\beta}| \geq 2^{mn - d \log m - |J \cup I_j| \log m}$$

Proof. We will show that there is a j such that for all $\beta \in \{0,1\}^{I_j}$, $|Y^{j,\beta}| \geq |Y|/2^{|I_j| \log m}$, which is sufficient by our bound on $|Y|$. Assume for contradiction that for every j there exists a β_j such that $|Y^{j,\beta_j}| < |Y|/2^{|I_j| \log m}$. Define $Y_= := \{y \in Y : \exists j, y[I_j, \alpha_j] = \beta_j\}$ and $Y_{\neq} := Y \setminus Y_= = \{y \in Y : \forall j, y[I_j, \alpha_j] \neq \beta_j\}$.

Assume for the moment that $|Y_=| < |Y|/2$; if this is the case then it must be that $|Y_{\neq}| \geq |Y|/2 \geq 2^{mn - d \log m - |J| \log m} > 2^{mn - 2n \log m}$. By Lemma 12 on $X'_{\bar{J}} \times Y_{\neq}$, there must exist some $x^* \in X'$ such that for every $\beta \in \{0,1\}^{\bar{J}}$ there exists $y_\beta \in Y_{\neq}$ such that $y_\beta[\bar{J}, x^*[\bar{J}]] = \beta$. Since $x^* \in X'$, there exists some j such that $x^* \in X^j$, and thus for any $\beta \in \{0,1\}^{\bar{J}}$ such that $\beta[I_j] = \beta_j$, there exists a $y_\beta \in Y_{\neq}$ such that $y_\beta[\bar{J}, x^*[\bar{J}]] = \beta$. But since $x^* \in X^j$, $x^*[I_j] = \alpha_j$, so $y_\beta[I_j, \alpha_j] = \beta_j$ which is a contradiction since $Y_{\neq} = \{y \in Y : \forall j, y[I_j, \alpha_j] \neq \beta_j\}$.

We now show that $|Y_=| < |Y|/2$. Define $\mathcal{F}(k) := \{(I_j, \alpha_j) \in \mathcal{F} : |I_j| = k\}$. Assume that there exists some k such that $|\mathcal{F}(k)| > 2m^{0.95k}$. Note that every set $(I_j, \alpha_j) \in \mathcal{F}(k)$ corresponds to an assignment to X which occurs with probability greater than $2^{-0.95k \log m}$ in $X^{\geq j}$, which has size at least $|X|/2$ by construction, and so by a union bound we get that

$$|X| > |\mathcal{F}(k)| \cdot (2^{-0.95k \log m} \frac{|X|}{2}) > (\frac{1}{2} \cdot 2^{0.95 \cdot k \log m + 1} \cdot 2^{-0.95 \cdot k \log m})|X| = |X|$$

which is clearly a contradiction. Thus we can assume that $|\mathcal{F}(k)| \leq 2m^{0.95k}$ for all k, then because we assumed $|Y^{j,\beta_j}| < |Y|/2^{|I_j| \log m}$ we get that

$$\begin{aligned} |Y_=| &< \sum_{k=1}^{n} (2m^{0.95k} \cdot \frac{|Y|}{2^{k \log m}}) \\ &< \sum_{k=1}^{n} (2^{0.96k \log m - 1} \cdot \frac{|Y|}{2^{k \log m}}) \\ &= \frac{|Y|}{2} \cdot \sum_{k=1}^{n} (2^{0.04 \log m})^{-k} \\ &< \frac{|Y|}{2} \cdot \sum_{k=1}^{\infty} 2^{-k} = \frac{|Y|}{2} \end{aligned}$$

which completes the proof. ◀

5.2 Simulation

Proof of Basic Lifting Theorem. For n sufficiently large let $m = n^{1.1}$ and let $d \leq n$. As stated in Section 1 given a decision tree T for f of depth d we can build a communication protocol for $f \circ \text{IND}_m^n$ of depth $d \log m$; Alice sends the entirety of x_j for whatever variable z_j the decision tree queries, Bob sends back $y_j[x_j]$, and they go down the appropriate path in the decision tree. Thus we show the other direction: given a protocol Π of depth $d \log m$ for the composed problem $f \circ \text{IND}_m^n$ we want to construct a decision-tree of depth $O(d)$ for f.

The decision-tree is naturally constructed by starting at the root of Π and taking a walk down the protocol tree guided by occasional queries to the variables $z = (z_1, \ldots, z_n)$ of f. During the walk, we maintain a ρ-structured rectangle $R = X \times Y \subseteq [m]^n \times (\{0,1\}^m)^n$ which will be a subset of the inputs that reach the current node in the protocol tree, where ρ corresponds to the restriction induced by the decision tree at the current step. Thus our goal is to ensure that the image $\text{IND}_m^n(X \times Y)$ has some of its bits fixed according to the queries to z made so far, and no information has been leaked about the remaining free bits of z.

To choose which bits to fix, we use the density restoring partition to identify any assignments to some of the x variables that have occurred with too high a probability; by the way the rectangle partition is defined the corresponding sets X^j regain blockwise min-entropy. Then using Lemma 17, we pick one of these assignments and query all the corresponding z variables, and for the resulting β we know $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$-structured since the size of $Y^{j,\beta}$ doesn't decrease too much. With the blockwise min-entropy of $\boldsymbol{X}$ restored and the size of Y kept high, we can update ρ to include $\rho^{j,\beta}$ and continue to run the rectangle partition at the next node, and so we proceed in this way down the whole communication protocol.

We describe our query simulation of the communication protocol Π in Algorithm 2. For all $v \in \Pi$ let $R_v = X_v \times Y_v$ be the rectangle induced at node v by the protocol Π. The query and output actions listed in bold are the ones performed by our decision tree.

Before we prove the correctness and efficiency of our algorithm, we note that we make no distinction between Alice speaking and Bob speaking in our procedure. Here we note that each R_v is a rectangle induced by the protocol Π, and so updating v only splits X or Y – corresponding to when Alice and Bob speak respectively – but not both, and so since $R \subseteq R_v$ we get that $|X \cap X_v| \geq |X|/2$ and $|Y \cap Y_v| \geq |Y|/2$.

Algorithm 2 Simulation protocol.

Initialize $v :=$ root of Π; $R := [m]^n \times (\{0,1\}^m)^n$; $\rho = *^n$;
while v *is not a leaf* **do**
  Precondition: $R = X \times Y$ *is* ρ*-structured; for convenience define* $J := \mathrm{fix}(\rho)$;
  Let v_ℓ, v_r be the children of v, and update $v \leftarrow v_\ell$ if $|R \cap R_{v_\ell}| \geq |R|/2$ and $v \leftarrow v_r$ otherwise;
  Execute Algorithm 1 on $(X \cap X_v) \times (Y \cap Y_v)$ and let $\mathcal{F} = \{(I_j, \alpha_j)\}_j$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$ be the outputs;
  Apply Lemma 17 to $\mathcal{F}$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$ to get some index j corresponding to $(I_j, \alpha_j) \in \mathcal{F}$;
  Query each variable z_i for every $i \in I_j$, and let $\beta \in \{0,1\}^{I_j}$ be the result;
  Update $X \leftarrow X^j$ and $Y \leftarrow Y^{j,\beta}$;
  Update $\rho \leftarrow \rho^{j,\beta}$ (recall that $\rho^{j,\beta} \in \{0,1,*\}^n$ is the restriction extending ρ by $\rho^{j,\beta}[I_j] = \beta$);
end
Output the same value as v does;

Efficiency and correctness

To prove the efficiency and correctness of our algorithm, consider the start of the t-th iteration, where we are at a node v and maintaining $R^t = X^t \times Y^t$ and ρ^t.[16] Again for convenience we write $J^t := \mathrm{fix}(\rho^t)$. Let (I^t, α^t) be the (possibly empty) assignment returned by Lemma 17 corresponding to index j^t, and let β^t be the result of querying $z[I^t]$.

We show that our precondition that R^t is ρ^t-structured holds for all $t \leq d \log m$, as well as the fact that ρ^t fixes at most $O(d)$ coordinates:

1. $\mathrm{IND}_m^{J^t}(X^t_{J^t}, Y^t_{J^t}) = \rho^t[J^t]$
2. X_{J^t} is fixed to a single value and $\boldsymbol{X}^t_{\overline{J^t}}$ has blockwise min-entropy at least $0.95 \log m$
3. $|Y^t| \geq 2^{mn - t - |J^t| \log m}$.
4. $\mathbf{D}_\infty(\boldsymbol{X}^t_{\overline{J^t}}) \leq 2t - 0.05|J^t| \log m$, which implies $|J^t| \leq 40d$ by non-negativity of deficiency

All invariants hold at the start of the algorithm since $\rho^0 = *^n$ and $X^0 \times Y^0 = [m]^n \times (\{0,1\}^m)^n$. Inductively consider the $(t+1)$-th iteration assuming all invariant holds for the t-th iteration. After applying Algorithm 1 invariant 1 follows by Claim 14 and invariant 2 follows by Claims 14 and 15. For invariant 3 we first show that it is valid to apply Lemma 17 in the $(t+1)$-th iteration. First, because $|X^t \cap X_v| \geq |X^t|/2$ we know that the blockwise min-entropy of $(\boldsymbol{X}^t \cap \boldsymbol{X}_v)_{\overline{J^t}}$ is at most one less than the blockwise min-entropy of $\boldsymbol{X}^t_{\overline{J^t}}$, which is at least $0.95 \log m$. Second, we have

$$|(Y^t \cap Y_v)| \geq |Y^t|/2 \geq 2^{mn - t - |J^t| \log m - 1} = 2^{mn - (t+1) - |J^t| \log m} \geq 2^{mn - 41 d \log m}$$

recalling that $t + 1 \leq d \log m$. Thus we can apply Lemma 17 and we get

$$\begin{aligned} |Y^{t+1}| &= |Y^{j^t, \beta^t}| \\ &\geq 2^{mn - t - |J^t| \log m - 1 - |I^t| \log m} \\ &\geq 2^{mn - t - 1 - (|J^t| + I^t|) \log m} = 2^{mn - (t+1) - |J^{t+1}| \log m} \end{aligned}$$

[16] We understand that this notation is somewhat overloaded with X^j, $Y^{j,\beta}$, and $\rho^{j,\beta}$. Since the proof that the invariants hold is short and we only ever use t (or $t+1$) for the time stamps and j for the indices, hopefully this won't cause any confusion.

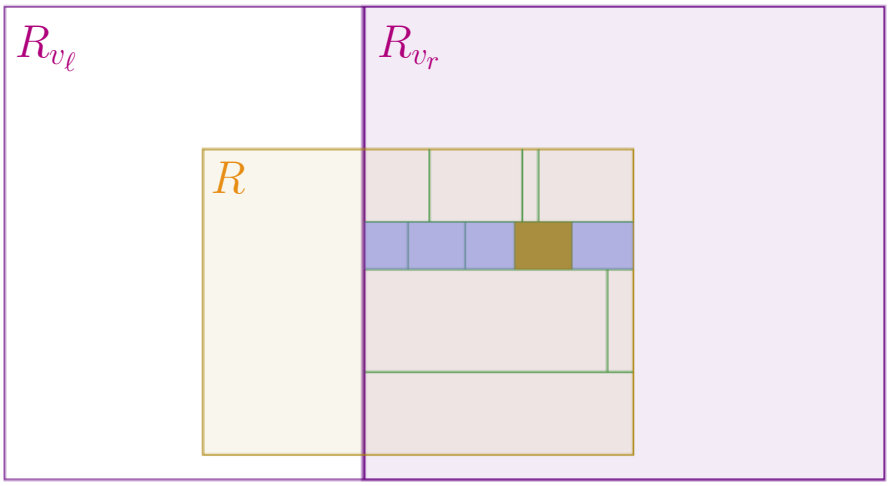

Figure 3 One iteration of Algorithm 2. We perform Algorithm 1 (green lines) on the larger half of R after moving from v to its child (shaded in purple), use Lemma 17 to identify a part j (shaded in blue), and then query I_j and set R to $X^j \times Y^{j,\beta}$ for the result $z[I_j] = \beta$ (shaded in brown).

For invariant 4, by Claim 16 and induction we get that

$$\begin{aligned}
\mathbf{D}_\infty(\boldsymbol{X}^{t+1}) &= \mathbf{D}_\infty(\boldsymbol{X}^{j^t}) \\
&\leq \mathbf{D}_\infty(\boldsymbol{X} \cap \boldsymbol{X}_v) - 0.05|I^t| \log m + 1 \\
&\leq (2t - 0.05|J^t| \log m + 1) - 0.05|I^t| \log m + 1 \\
&= 2(t+1) - 0.05|J^{t+1}| \log m
\end{aligned}$$

which completes the proof of our invariants. Thus our procedure is well-defined.

Lastly we have to argue that if we reach a leaf v of Π while maintaining R and ρ with fixed coordinates J, then the solution $o \in \mathcal{O}$ output by Π is also valid solution to the values of z, of which the decision-tree knows that $z[J] = \rho[J]$. Suppose Π outputs $o \in \mathcal{O}$ at the leaf v, and assume for contradiction that there exists $\beta \in \{0,1\}^n$ consistent with ρ such that $\beta \notin f^{-1}(o)$. Since $\textsc{Ind}_m^J(x,y) = \rho[J] = \beta[J]$ for all $(x,y) \in R$, we focus on $\bar{J} = \text{free}(\rho)$. Since R is ρ-structured, $\boldsymbol{X}_{\bar{J}}$ has blockwise min-entropy $0.95 \log m$ and $|Y| > 2^{mn - d \log m - |J| \log m} > 2^{mn - 2n \log m}$. Thus applying Lemma 12 to $X \times Y$, we know that that there exists $(x,y) \in R$ such that $\textsc{Ind}_m^n(x,y) = \beta$, which is a contradiction as $R \subseteq R_v \subseteq (f \circ \textsc{Ind}_m^n)^{-1}(o)$. ◀

6 Smaller gadgets

In Section 5 we loosely chose $m = n^{1.1}$ for the purpose of showing the basic lifting statement. In this section we improve from $n^{1.1}$.

6.1 Optimizing the gadget size

First, we make direct improvements on Theorem 9 by showing that the gadget size can be improved to $m = n^{1+\epsilon}$ with only a small modification of the proof. Second we show that the same modification can be used to obtain a tradeoff between the gadget size and the strength of the lifting theorem, which gives an optimal gadget size of m being quasilinear in n for a slightly weaker lower bound.

Warm-up: $m = n^{1+\epsilon}$

First we improve on Theorem 9 to get a gadget of size $n^{1+\epsilon}$ for any $\epsilon > 0$, with no changes in the asymptotic strength of the lifting theorem nor anything non-trivial in the proof. This comes from two observations. First, we only use the size of m in the two places we apply Lemma 12, and in both cases we can apply Lemma 7 as long as $0.95 \log m - O(1) \geq \log(Kn \log m)$. Second, from the perspective of our simulation, the constant 0.95 is only used to set the blockwise min-entropy threshold for the density-restoring partition, and was chosen arbitrarily.

So for $\delta > 0$ we can instead choose to put the threshold at $(1-\delta)\log m$, at which point our condition on m changes to $(1-\delta)m \geq \log(Kn \log m)$. Clearly this can be made to fulfill our condition $m \geq n^{1+\epsilon}$ as long as $(1-\delta)(1+\epsilon) > 1$. The proof itself then simply becomes a matter of replacing 0.95 with $1-\delta$ and 0.05 with δ throughout the proof, as well as a few other constants. Since Claim 16 now gives a drop in deficiency of δ for every coordinate we query, the non-negativity of deficiency gives us $|\operatorname{fix}(\rho^t)| \leq 2t/\delta \log m$ at any time $t \leq d \log m$, which gives us a decision tree of depth $(2/\delta) \cdot d = O(d)$ – or for dag-like lifting, a decision dag of width $(2/\delta) \cdot d = O(d)$ – as required.

Near-linear gadget: $m = \Theta(n \log n)$

Building off the intuition from our warm-up, what happens if δ is chosen to be subconstant? We cannot hope to get a tight lifting theorem, as our decision tree/dag will be of depth $(2/\delta) \cdot d$. Furthermore choosing $\delta = o(1/\log m)$ makes our blockwise min-entropy threshold $(1-\delta)\log m$ trivial, as $\log m$ is the maximum possible blockwise min-entropy for $\boldsymbol{X}$. Thus by choosing $\delta = \Omega(1/\log m)$ we can get the following general lower bound, which gives Theorem 9 and Theorem 10 as special cases.

▶ **Theorem 18** (Scaling Basic Lifting Theorem). *Let f be a search problem over $\{0,1\}^n$, and let m, δ be such that $\delta \geq \Omega(\frac{1}{\log m})$ and $m^{1-\delta} \geq \Omega(n \log m)$. Then*

$$\mathbf{P}^{cc}(f \circ \textsc{Ind}_m) \;\geq\; \mathbf{P}^{dt}(f) \cdot \Omega(\delta \log m)$$

Proof sketch. We start with a given communication protocol Π of depth $d \cdot \delta \log m$ for the composed problem $f \circ \textsc{Ind}_m^n$ and construct a decision-tree of depth $O(d)$ for f.[17] We define a ρ-structured rectangle R as before except now with the condition that $\boldsymbol{X}$ has blockwise min-entropy $(1-\delta)\log m$. Then in Algorithm 1 we set the blockwise min-entropy threshold for a violating assignment (I_j, α_j) at $(1-\delta)\log m$ as well.

To prove Lemma 12, note that we can apply Claim 8 regardless of m and N, and we can still apply Lemma 7 as long as we can choose m such that $(1-\delta)\log m - 2 > \log(K \cdot 2n \log m)$. Thus for this altered rectangle partition procedure, by the same proofs as before, Claim 15 states that $\boldsymbol{X}^j_{\overline{J \cup I_j}}$ has blockwise min-entropy at least $(1-\delta)\log m$, Claim 16 states that $\mathbf{D}_\infty(\boldsymbol{X}^j) \;\leq\; \mathbf{D}_\infty(\boldsymbol{X}) - |I_j| \cdot \delta \log m + 1$, and Lemma 17 states that if $\boldsymbol{X}_{\bar{J}}$ has blockwise min-entropy $(1-\delta)\log m - O(1)$ and $|Y| > 2^{mn - d \log m - |J| \log m}$, then there exists a j such that for all β, $|Y^{j,\beta}| \geq 2^{mn - d \log m - |J \cup I_j| \log m}$.

Now our simulation procedure is the same as Algorithm 2. Again at the start of the t-th iteration we are maintaining $R^t = X^t \times Y^t$, ρ^t, and $J^t := \operatorname{fix}(\rho^t)$, where now $t \leq d \cdot \delta \log m$. By the same argument our procedure is well-defined as long as the precondition of R^t being ρ^t-structured holds, and by a deficiency argument using our new Claim 16 we get that

[17] This is a bit different than previously, as we are incorporating δ into the size of our communication protocol rather than our decision tree, but this is purely for readability's sake.

$\mathbf{D}_\infty(\boldsymbol{X}^t) \leq 2t - |J^t| \cdot \delta \log m$, which implies $|J^t| \leq 2t/\delta \log m \leq 2d$. Our precondition holds by applying the new versions of Claim 14, Claim 15, and Lemma 17 as before. Finally our simulation is correct again by the invariants and Lemma 12. ◀

6.2 Graduated lifting

In this section we prove a variant on Theorem 9, which allows us to set the gadget size m in terms of the target decision tree depth d. The tree-like theorem was originally proven in [14] but it also follows immediately from our proof of Theorem 9 with significant improvements to the gadget size. The only technical detail is that for arbitrary search problems we cannot allow the gadget size to go below $\Omega(\log^{1+\epsilon} n)$,[18] although future improvements on the statement of Lemma 7 could change this restriction. In particular, the *Robust Sunflower Conjecture* states that the precondition in Lemma 7 can be improved to $\log O(\log 1/\epsilon)$; if this held then it would remove our restriction on d.

▶ **Theorem 19** (Graduated Lifting Theorem, large d)**.** *Let f be a search problem over $\{0,1\}^n$ and let $m \geq \max(\mathbf{P}^{dt}(f), \log n)^{1+\epsilon}$ for some $\epsilon > 0$. Then*

$$\mathbf{P}^{cc}(f \circ \textsc{Ind}_m) \;\geq\; \mathbf{P}^{dt}(f) \cdot \Omega(\log m)$$

Proof sketch. We start with a given communication protocol Π of depth $d \cdot \delta \log m$ for the composed problem $f \circ \textsc{Ind}_m^n$, where δ is such that $(1-\delta)(1+\epsilon) > 1$, and we construct a decision-tree of depth $O(d)$ for f. We change the precondition on $|Y|$ in Lemma 12 to read "$|Y| \geq 2^{mn - 2d \log m}$", which is guaranteed by the preconditions of Lemma 17 and our ρ-almost structured invariant whenever it is applied. Accordingly, in the proof of Lemma 12 we set $\kappa = 2^{-3d \log m}$. By our choice of m we get that $(1-\delta)\log m - O(1) \geq \log(K \log n + K \cdot 3d \log m) \geq \log K \log(n/\kappa)$. ◀

We also note that for many natural search problems, such as the canonical search problem on CNF-UNSAT, the restriction $m = \Omega(\log n)$ can be removed. For simplicity, call a search problem f "nice" if the following condition holds: for any potential output $o \in \mathcal{O}$ to $f(z_1 \dots z_n)$, any partial assignment $\rho \in \{0,1,*\}^n$ to the z variables which admits o, and any partial assignment $\rho' \in \{0,1,*\}^n$ extending ρ certifying that f does not output o which is minimal with respect to the number of coordinates in $\operatorname{fix}(\rho') \cap \operatorname{free}(\rho)$, then $|\operatorname{fix}(\rho') \cap \operatorname{free}(\rho)| \leq 2^{O(d \log d)}$.[19]

▶ **Theorem 20** (Graduated Lifting Theorem, small d)**.** *Let f be a nice search problem over $\{0,1\}^n$ and let $m \geq (\mathbf{P}^{dt}(f))^{1+\epsilon}$ for some $\epsilon > 0$. Then*

$$\mathbf{P}^{cc}(f \circ \textsc{Ind}_m) \;\geq\; \mathbf{P}^{dt}(f) \cdot \Omega(\log m)$$

Proof of Theorem 20. We begin by apply all the changes to Lemma 12 as stated in the previous proof. In order to remove the difficulty of having the set size n in the statement of Lemma 7, we marginalize each x to subsets of size at most $2^{O(d \log m)}$.

[18] This necessarily holds whenever $d = \Omega(\log n)$, which can be considered the "natural" range of parameters as otherwise there is a variable that is never queried along *any* path of our decision tree, meaning f does not depend on all its variables.

[19] If we consider the canonical search problem on CNF-UNSAT, then for any output clause C and any ρ, either ρ totally falsifies C or it leaves at least one variable in C unfixed, at which point ρ' can simply set any unset variable in C to satisfy it. These are the only minimal extensions, and thus $|\operatorname{fix}(\rho') \cap \operatorname{free}(\rho)| = 1$.

▶ **Lemma 21** (*d*-Full Range Lemma). *Let $m \geq d^{1+\epsilon}$ and let $J \subseteq [n]$. Let $X \times Y \subseteq [m]^{\bar{J}} \times (\{0,1\}^m)^n$ be such that $\boldsymbol{X}$ has blockwise min-entropy at least $(1-\delta)\log m - O(1)$ and $|Y| > 2^{mn-2d\log m}$. Then there exists an $x^* \in X$ such that for every constant C, every $J' \subseteq \bar{J}$ with $|J'| \leq 2^{Cd\log m}$, and every $\beta \in \{0,1\}^{J'}$, there exists a $y_\beta \in Y$ such that $\textsc{Ind}_m^{J'}(x^*, y_\beta) = \beta$.*

Proof. Assume for contradiction that for all x there exists a set $I_x \subseteq \bar{J}$ of size at most $2^{Cd\log m}$ and an assignment $\beta_x \in \{0,1\}^{I_x}$ such that $|\{y \in Y : y[I_x, x[I_x]] = \beta_x\}| = 0$. As before, by Claim 8 we can assume that $\beta_x = 1^{I_x}$. For each $x \in X$, let $S_x \subseteq [mn]$ be the set defined by including (i,α) iff $x[i] = \alpha$ and $i \in I_x$, and let $\mathcal{S}_X = \{S_x : x \in X\}$.

As above we set $\kappa = 2^{-3d\log m}$. By our choice of m we get that $(1-\delta)\log m - O(1) \geq \log(K(C+3)d\log m) \geq \log K \log(2^{Cd\log m}/\kappa)$. Thus by Lemma 7 we get that $\mathbf{Pr}_{\mathbf{S_y} \subseteq [mn]}(\forall S_x \in \mathcal{S}_X, S_x \not\subseteq S_y) \leq \kappa$, and if we look at y as being the indicator vector for S_y then we get that $\mathbf{Pr}_{\boldsymbol{y} \sim \{0,1\}^{mn}}(\forall x \in X, y[I_x, x[I_x]] \neq 1^{I_x}) \leq \kappa$. Thus by counting we get

$$\begin{aligned} |Y| &= |\{y \in Y : \forall x, y[I_x, x[I_x]] \neq \beta_x\}| \\ &\leq |\{y \in (\{0,1\}^m)^n : \forall x, y[I_x, x[I_x]] \neq 1^{I_x}\}| \\ &\leq \kappa \cdot 2^{mn} = 2^{mn-3d\log m} \end{aligned}$$

which is a contradiction as $|Y| > 2^{mn-2d\log m}$ by assumption. ◀

Marginalizing to sets of size $2^{O(d\log m)}$ causes no issue for us when we apply Lemma 12 to show that there exists a good j in the rectangle partition, as we can assume that all sets have size at most $O(d)$ by either deficiency or error sets. At the leaves we use the fact any falsifying assignment only depends on at most $2^{O(d\log m)}$ unset variables, since we can no longer assert that every joint assignment to *all* remaining free variables exists. ◀

6.3 Real lifting

Our results also generalize to the real lifting setting. At node v of a *real communication protocol*, Alice and Bob send $A_v(x) : \mathcal{X} \to \mathbb{R}$ and $B_v(y) : \mathcal{Y} \to \mathbb{R}$, respectively, and they go left iff $A_v(x) \geq B_v(y)$ and right otherwise. For a combinatorial view, we say a *triangle* is a set $T \subseteq X \times Y \subseteq \mathcal{X} \times \mathcal{Y}$ and an ordering $<_X$, $<_Y$ on X and Y respectively such that if $x_1 <_X x_2$ and $(x_2, y) \in T$, $(x_1, y) \in T$, and if $y_1 <_Y y_2$ and $(x, y_2) \in T$, $(x, y_1) \in T$.

▶ **Theorem 22** (Real Lifting Theorem). *Let f be a search problem over $\{0,1\}^n$ and let m, δ be such that $\delta \geq \Omega(\frac{1}{\log m})$, $m^{1-\delta} = \Omega(\mathbf{P}^{dt}(f)\log m)$, and either f is nice or $m^{1-\delta} = \Omega(\log n)$. Then*

$$\mathbf{P}^{rcc}(f \circ \textsc{Ind}_m) \geq \mathbf{P}^{dt}(f) \cdot \Omega(\delta \log m)$$

Proof sketch. Our results immediately extend to the case of real lifting. In Algorithm 2 at node v the children of v partition our current rectangle R into two triangles T_0, T_1; after going to the side which maximizes $|T_b|$, there exists a rectangle $X' \times Y' \subseteq T_b$ such that $|X'| \geq |X|/2$ and $|Y'| \geq |Y|/2$, which was already what we assumed in our invariants and when executing Algorithm 1. This also holds for the tree-like items of Theorems 18, 19, and 20, giving our first point of Theorem 22. ◀

References

1 Ryan Alweiss, Shachar Lovett, Kewen Wu, and Jiapeng Zhang. Improved bounds for the sunflower lemma. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 624–630. ACM, 2020.

2 Tolson Bell, Suchakree Chueluecha, and Lutz Warnke. Note on sunflowers. *Discrete Mathematics*, 344(7):112367, 2021.

3 Bruno Pasqualotto Cavalar, Mrinal Kumar, and Benjamin Rossman. Monotone circuit lower bounds from robust sunflowers. In *LATIN 2020: Theoretical Informatics - 14th Latin American Symposium, São Paulo, Brazil, January 5-8, 2021, Proceedings*, volume 12118 of *Lecture Notes in Computer Science*, pages 311–322. Springer, 2020.

4 Siu On Chan, James R. Lee, Prasad Raghavendra, and David Steurer. Approximate constraint satisfaction requires large LP relaxations. *J. ACM*, 63(4):34:1–34:22, 2016.

5 Arkadev Chattopadhyay, Yuval Filmus, Sajin Koroth, Or Meir, and Toniann Pitassi. Query-to-communication lifting using low-discrepancy gadgets. Technical Report TR19-103, Electronic Colloquium on Computational Complexity (ECCC), 2019. URL: `https://eccc.weizmann.ac.il/report/2019/103/`.

6 Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation theorems via pseudo-random properties. *Comput. Complex.*, 28(4):617–659, 2019.

7 Susanna de Rezende, Or Meir, Jakob Nordström, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with simple gadgets and applications to circuit and proof complexity. Technical Report TR19-186, Electronic Colloquium on Computational Complexity (ECCC), 2019. URL: `https://eccc.weizmann.ac.il/report/2019/186/`.

8 Susanna de Rezende, Jakob Nordström, and Marc Vinyals. How limited interaction hinders real communication (and what it means for proof and circuit complexity). In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 295–304. IEEE, 2016.

9 Paul Erdös and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35(1):85–90, 1960.

10 Keith Frankston, Jeff Kahn, Bhargav Narayanan, and Jinyoung Park. Thresholds versus fractional expectation-thresholds. *Annals of Mathematics*, 194(2):475–495, 2021.

11 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 902–911. ACM, 2018.

12 Mika Göös. Lower bounds for clique vs. independent set. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1066–1076. IEEE Computer Society, 2015.

13 Mika Göös, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM J. Comput.*, 47(1):241–269, 2018.

14 Mika Göös, Sajin Koroth, Ian Mertz, and Toniann Pitassi. Automating cutting planes is NP-hard. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 68–77. ACM, 2020.

15 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM J. Comput.*, 45(5):1835–1869, 2016.

16 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM Journal on Computing*, 47(5):1778–1806, 2018.

17 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1077–1088. IEEE, 2015.

18 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 132–143, 2017.

19 Danny Harnik and Ran Raz. Higher lower bounds on monotone size. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 378–387. ACM, 2000.

20 Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time–space trade-offs in proof complexity. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 233–248. ACM, 2012.

21 Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

22 Pravesh K. Kothari, Raghu Meka, and Prasad Raghavendra. Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of csps. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 590–603. ACM, 2017.

23 Jan Krajíček. Interpolation by a game. *Mathematical Logic Quarterly*, 44:450–458, 1998.

24 Eyal Kushilevitz. Communication complexity. In *Advances in Computers*, volume 44, pages 331–360. Elsevier, 1997.

25 James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 567–576. ACM, 2015.

26 Xin Li, Shachar Lovett, and Jiapeng Zhang. Sunflowers and quasi-sunflowers from randomness extractors. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018, August 20-22, 2018 - Princeton, NJ, USA*, volume 116 of *LIPIcs*, pages 51:1–51:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

27 Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1246–1255. ACM, 2017.

28 Toniann Pitassi and Robert Robere. Lifting nullstellensatz to monotone span programs over any field. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1207–1219. ACM, 2018.

29 Anup Rao. Coding for sunflowers. *Discrete Analysis*, 2:8, 2020.

30 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999.

31 Robert Robere. Unified lower bounds for monotone computation. *Ph.D Thesis*, 2018.

32 Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen A. Cook. Exponential lower bounds for monotone span programs. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 406–415. IEEE Computer Society, 2016.

33 Benjamin Rossman. The monotone complexity of k-clique on random graphs. *Proceedings of the 51st Symposium on Foundations of Computer Science (FOCS)*, pages 193–201, 2010.

34 Alexander A. Sherstov. The pattern matrix method. *SIAM J. Comput.*, 40(6):1969–2000, 2011.

35 Alexander A Sherstov. Communication lower bounds using directional derivatives. *Journal of the ACM (JACM)*, 61(6):1–71, 2014.

36 Xiaodi Wu, Penghui Yao, and Henry S. Yuen. Raz-McKenzie simulation with the inner product gadget. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:10, 2017.