Lifting with Sunflowers

Shachar Lovett[*]
UC San Diego
slovett@cs.ucsd.edu

Raghu Meka
UC Los Angeles
raghuvardhan@gmail.com

Ian Mertz[†]
University of Toronto
mertz@cs.toronto.edu

Toniann Pitassi[‡]
University of Toronto and IAS
toni@cs.toronto.edu

Jiapeng Zhang
University of Southern California
jiapengz@usc.edu

December 1, 2020

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). Whereas previous proofs used sophisticated Fourier analytic techniques, our proof uses elementary counting together with a novel connection to the sunflower lemma.

In addition to a simplified proof, our approach also gives quantitative improvements in terms of *gadget size*. 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 poly logarithmic, similar to the known bounds for the corresponding robust sunflower lemmas.

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 [RM99, GPW15, dRNV16, WYY17, CKLM17], nondeterministic [GLM+16, Göö15], randomized [GPW17, CFK+19] degree-to-rank [She11, PR17, PR18, RPRC16] and nonnegative degree to nonnegative rank [CLRS16, KMR17]. In these papers and others, lifting

[*]Research supported by NSF Award DMS-1953928.
[†]Research supported by NSERC.
[‡]Research supported by NSF Award CCF-1900460 and NSERC.

theorems have been applied to simplify and resolve some longstanding open problems, including new separations in communication complexity [GP18, GPW15, GPW17, CKLM17, CFK+19], proof complexity [GLM+16, HN12, GP18, dRNV16, dRMN+19, GKMP20] monotone circuit complexity [GGKS18], monotone span programs and linear secret sharing schemes [RPRC16, PR17, PR18], and lower bounds on the extension complexity of linear and semi-definite programs [CLRS16, KMR17, LRS15]. Furthermore within communication complexity most functions of interest—e.g. equality, set-disjointness, inner product, gap-hamming (c.f. [Kus97, Juk12])—are lifted functions.

At the heart of these proofs is a *simulation theorem*. A communication protocol for the lifted function can "mimic" a decision tree for the original function by taking $\log m$ steps to calculate each variable queried by the decision tree in turn. For $m = n^{O(1)}$ and for every f the deterministic simulation theorem [RM99, GPW15] 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 [RM99], applying to a wider range of gadgets [CFK+19], and with more sharpened results giving somewhat improved parameters and simulation theorems for the more difficult settings of randomized and dag-like lifting. However, nearly all proofs of even the basic deterministic simulation theorem use tools from the Fourier analysis of Boolean functions, together with somewhat intricate counting arguments.

Lifting using the sunflower lemma. The primary purpose 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 [ER60] 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. [ALWZ20] proves the sunflower lemma with significantly improved parameters, making a huge step towards resolving the longstanding open problem obtaining optimal parameters.

Both the original Sunflower Lemma as well as Rossman's robust version [Ros10] 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 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 [ALWZ20] for a nice survey of the many applications to Computer Science.)

In all of these lower bounds, the central idea is to use the sunflower lemma in order to "tame" a protocol or algorithm, in order to show that at each step of the computation, the underlying set of inputs consistent so far can be partitioned into a structured part and a random part. This allows the algorithm to be massaged into a simpler form, where the lower bound is easier to prove. Since lifting theorems are attempting to do precisely the same thing, it is natural to expect that there should be a connection between the two lines of research. Indeed, [LLZ18] made an explicit connection between sunflowers and randomness extractors where the latter is again a primary tool used in many if not all of the proofs of lifting.

Additionally these same tools can be used to prove a lifting theorem for *dag-like* communication protocols, as originally proven in [GGKS18]. Again we follow the same proof as before but for the main invariant we rely on a connection to the sunflower lemma. In fact while the central lemma needed for the invariant in the dag-like case is stronger than in the case of the basic lifting theorem, it actually follows *more* directly from the connection to sunflowers. We note that our results extend straightforwardly to the real communication setting as well.[1]

Gadget size. In spite of the tremendous progress in lifting theorems, most generic lifting theorems require gadget sizes that are polynomial in n.[2] Define the gadget size of $g : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ as $\min(|\mathcal{X}|, |\mathcal{Y}|)$. For example, we now have lifting theorems for the three classical models of communication: deterministic, non-deterministic, and randomized. In each of these results and others that use Fourier-analytic arguments, the gadget size has to be at least $\Omega(n^2)$. Can we circumvent this quadratic barrier?

Gadget size is a fundamental parameter in lifting theorems and their applications. This is because often in applications, one loses factors that depend polynomially on the gadget size (as defined above). 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 $2^{\tilde{\Omega}(\sqrt{n})}$ [GJW18, HR00]. Improving the gadget size from poly n to $O(1)$ (or even poly $\log n$) would improve the best known lower bounds for extended formulations and monotone circuit size to $2^{\tilde{\Omega}(n)}$.

Luckily the quadratic bottleneck comes from the core lemma that we reprove in this paper, and as a nice side effect our simplified proof immediately gives us a gadget of size $n^{1+\epsilon}$. 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, up to a gadget of size $O(n \log n)$. Our approach does not seem to have the same bottleneck as previous approaches, and we conjecture that it can be pushed to poly-logarithmic gadget sizes (similar to the improvements made for the sunflower lemma in [ALWZ20]).

We also use our sunflower method to give a new proof (and with tightened parameters) of [GKMP20] 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 with query complexity with gadgets of comparable. Here we use a more involved argument using an older iteration of the sunflower lemma. 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 [GKMP20].

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. In Section 4 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 5 we show that the gadget size m can be improved to $n^{1+\epsilon}$, and by sacrificing in the strength of the lifting theorem we can even push it down to $O(n \log n)$. In Section 6 we show that we can lift dag-like query complexity to dag-like communication complexity. In Section 7 we show that m can be made $\text{poly}(\mathbf{P}^{dt}(f))$ with no other dependence on n. 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 conclude with directions for further research Section 8.

[1] In most query-to-communication settings it is fairly simple to extend results for communication complexity to the real communication setting [Kra98]; we refer readers to e.g. [dRNV16, GGKS18] for examples of these techniques and applications of lifting to real communication complexity.

[2] Some notable exceptions for models of communication with better gadget size are [She11, She14, GP18, PR17].

2 Preliminaries

We will use n to denote the length of the input, $N \leq n$ to denote an arbitrary number, 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.

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 $Y_I \in (\mathcal{U}^m)^I$ likewise. For a set system $\mathcal{F}$ over $\mathcal{U}$ and a set $S \subseteq \mathcal{U}$, we define $\mathcal{F}_{\bar{S}} := \{\gamma \setminus S : \gamma \in \mathcal{F}, S \subseteq \gamma\}$.

Definition 2.1. 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 $s \in (\mathcal{U}^m)^N$ each $(i, a) \in \gamma$ indicates an element $s[i, a] \in \mathcal{U}$. We sometimes say (i, a) is a *pointer*.

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 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 $\mathrm{free}(\rho) \subseteq [N]$ the variables assigned a star, and define $\mathrm{fix}(\rho) := [N] \setminus \mathrm{free}(\rho)$. If we have two restrictions ρ, ρ' such that $\mathrm{fix}(\rho) \cap \mathrm{fix}(\rho') = \emptyset$, then we define $\rho \cup \rho'$ to be the restriction which assigns $\mathrm{fix}(\rho)$ to $\rho[\mathrm{fix}(\rho)]$ and $\mathrm{fix}(\rho')$ to $\rho'[\mathrm{fix}(\rho')]$, 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 $S \subseteq \mathcal{U}^N$ and $I \subseteq [N]$ we denote by $\boldsymbol{S}_I$ the marginal distribution over coordinates I of the uniform distribution over S; note that the random draw is taken over the original set S before marginalizing to the coordinates I, rather than being the uniform distribution over S_I.

Definition 2.2. 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 $\mathbf{D}_\infty(\boldsymbol{s}) := \log |S| - \mathbf{H}_\infty(\boldsymbol{s}) \geq 0$. When $\boldsymbol{s}$ is chosen from a set $S \subseteq \mathcal{U}^N$, we define its *blockwise min-entropy* by $\min_{\emptyset \neq I \subseteq [N]} \frac{1}{|I|} \mathbf{H}_\infty(\boldsymbol{s}_I)$, 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 2.3. 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 N separate instances of the index gadget, which we denote by $\textsc{Ind}_m^N$ and which is a 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* $f \circ \textsc{Ind}^n_m$ is a bipartite search problem defined by $\mathcal{X} := [m]^n$, $\mathcal{Y} := (\{0,1\}^m)^n$, and $f \circ \textsc{Ind}^n_m(x,y) = \{o \in \mathcal{O} : o \in f(\textsc{Ind}^n_m(x,y))\}$.

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

$$\mathbf{P}^{dt}(f) \ := \ \text{least } \textit{depth} \text{ of a decision tree solving } f.$$

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

$$\mathbf{P}^{cc}(F) \ := \ \text{least } \textit{depth} \text{ of a communication protocol solving } F.$$

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 (p,ϵ)-*satisfying* if

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

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

We say that $\mathcal{F}$ is a (p,ϵ)-*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,ϵ)-satisfying. In other words,

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

In this paper we will always be using $p = 1/2$, and so for convenience we simply write $\boldsymbol{y} \subseteq \mathcal{U} \smallsetminus S$ instead of $\subseteq_{1/2}$ and call $\mathcal{F}$ an ϵ-robust sunflower instead of an $(1/2,\epsilon)$-robust sunflower. An analogue of the famed sunflower lemma of Erdős was proved for robust sunflowers by Rossman [Ros10]:

Lemma 2.1 (Robust Sunflower Lemma). *Let $s \in \mathbb{N}$ and let $\epsilon > 0$. Let $\mathcal{F}$ be a set system over $\mathcal{U}$ such that a) $|\gamma| \leq s$ for all $\gamma \in \mathcal{F}$; and b) $|\mathcal{F}| \geq (Cs\log 1/\epsilon)^s$ for some absolute constant $C > 0$. Then $\mathcal{F}$ contains an ϵ-robust sunflower.*

A recent breakthrough result (proved in [ALWZ20] and simplified in [Rao19]) proves the sunflower lemma with significantly improved parameters. As a stepping stone they also prove an improvement on Robust Sunflower Lemma assuming a condition called *spreadness*, but which we will state in the following way.

Lemma 2.2 (**Blockwise Robust Sunflower Lemma).** *Let $s \in \mathbb{N}$ and let $\epsilon > 0$. Let $\mathcal{F}$ be a set system over $\mathcal{U}$ such that a) $|\gamma| \leq s$ for all $\gamma \in \mathcal{F}$; and b) $\mathcal{F}$ has blockwise min-entropy at least $\log(K \log s/\epsilon)$ for some absolute constant $K > 0$. Then $\mathcal{F}$ is ϵ-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 2.3. *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 $\overline{x_i}$ in $\mathcal{C}$ with x_i. 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 $\overline{x_i}$ in $\mathcal{C}$ with 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 $x_i \vee A$, $x_i \vee B$, and C. If $\mathcal{C}^i(\beta^{-i}, 0) = 1$, 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 $\mathcal{C}^i(\beta^{-i}, 1) = 0$, 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 $\mathcal{C}^i(\beta^{-i}, 0) = 0$ and $\mathcal{C}^i(\beta^{-i}, 1) = 1$. 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 The basic lifting theorem: proof overview

The following is our basic deterministic lifting theorem. An earlier version was originally proven in [RM99] with $m = n^{20}$, and more recently in [GPW15] with $m = n^2$. We improve this to a near linear dependence on n.

Theorem 3.1 (**Basic Lifting Theorem).** *Let f be a search problem over $\{0,1\}^n$, and let $m = n^{1.1}$. Then*

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

In this section we will sketch out the technical ideas that go into proving Theorem 3.1, along with some of the innovations that have helped simplify the proof since [RM99]. 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 instead 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 coordinates 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 more than $d \log m$ 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. Also of note is that 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 to reduce the entropy of X_i to below the threshold, and since we ultimately want to shave an $\Omega(\log m)$ factor off the height of the communication protocol in our decision tree 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 assigment $x[I_1] = \alpha_1$ that violates $0.95 \log m$ blockwise min-entropy, 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 toss out all X^j sets or to pick one and query $z[I_j]$.[3]

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 intersection with R (in order to avoid stumbling into a situation where many coordinates get fixed for free) 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

[3] As described in Section 4.1, unlike in [GPW17] in our proof we truncate this procedure before X is empty, but the same basic principle applies.

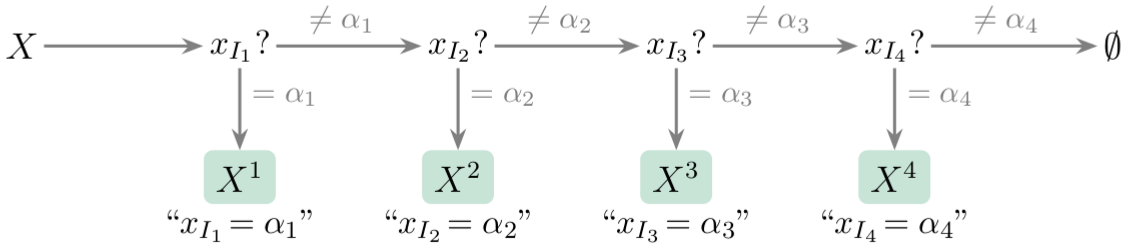

Figure 1: Rectangle Partition procedure (figure from [GPW17]).

such that $\boldsymbol{X}$ has blockwise min-entropy at least $0.95 \log m$ on $\overline{I_j}$, and Y has size at least $2^{mn-\text{poly}(n)}$ 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 and have queried some coordinates I, we need to ensure that we know enough to output the same answer as Π. Our second key lemma states that if X and Y are fixed on the coordinates $I \subseteq [n]$, $\boldsymbol{X}$ has min-entropy at least $0.95 \log m$ on $\overline{I}$, and Y has size at least $2^{m \cdot |\overline{I}| - \text{poly}(n)}$, then $\textsc{Ind}_m^{\overline{I}}(X, Y) = \{0,1\}^{\overline{I}}$; thus $R \subseteq R_\ell$ has every option left for $\overline{I}$, and so it cannot be that one of those assignments gives a different answer that the one covering R_ℓ.

Key lemmas through sunflowers. Up until this point, everything we've stated is as it appears in [GPW17]. 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 $\boldsymbol{X}_{\overline{I}}$ has high blockwise min-entropy and Y is large, there is a *single* row $x^* \in X$ such that $\textsc{Ind}_m^{\overline{I}}(x^*, Y) = \{0,1\}^{\overline{I}}$. 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, we can simply throw out those ys; then since some row x^* still has the full range of values available, whichever X^j part it appears in must not have had any ys thrown out 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 Blockwise Robust Sunflower Lemma. To illustrate this with a simple (but ultimately completely general) case, assume the all-ones vector is missing from $\textsc{Ind}_m^N(x, Y)$ for all x, or in other words there is no (x, y) such that $y[x] = 1^N$. Consider the universe $[mN]$, and let S_x be the set of size n defined by the values x points to. Since $\boldsymbol{X}$ has high blockwise min-entropy, by Blockwise Robust Sunflower Lemma 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 $y[x] \neq 1^N$. Thus $\mathbf{Pr}_{\boldsymbol{y}}[\forall x : y[x] \neq 1^x]$ is very low, or in other words a sufficiently large set $Y \subseteq (\{0,1\}^m)^n$ must contain some y such that $y[x] = 1^x$ for some x. This gives us our contradiction since we assumed Y was large.

One key aspect is that in the rectangle partition we've switched from proving an *extractor-like* property—showing $Y[I_j, \alpha_j]$ is close to uniform—to proving a *disperser-like* property—showing $\textsc{Ind}_m^n(x^*, Y)$ has every option in its support without need to show any sort of uniformity. This was an obstruction to getting small gadgets before, as Fourier arguments tend to be fairly coarse in terms of the polynomials involved.

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 let $v = \textit{root}$. 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 $|Y_{\overline{J}}| \geq 2^{m \cdot |J| - n \log n}$. 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.

4 The basic lifting theorem: full proof

To prove Basic Lifting Theorem, 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 variable queried by the decision tree. Our proof will follow the basic structure of previous works [GPW17, GGKS18]. 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 and 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^*, or in other words $\textsc{Ind}_m^N(x^*, Y) = \{0,1\}^N$. This appears as Lemma 7 in [GGKS18] for dag-like lifting and is stronger than is necessary for proving Basic Lifting Theorem, but the proof highlights our new counting strategy and will be a useful tool throughout the rest of the paper.[4]

Lemma 4.1 (Full Range Lemma). *Let $m \geq n^{1.1}$ and let $N \leq n$. Let $X \times Y \subseteq [m]^N \times (\{0,1\}^m)^N$ be such that $\boldsymbol{X}$ has blockwise min-entropy at least $0.95 \log m - O(1)$ and $|Y| > 2^{mN - n \log m}$. Then there exists an $x^* \in X$ such that for every $\beta \in \{0,1\}^N$, there exists a $y_\beta \in Y$ such that $\textsc{Ind}_m^N(x^*, y_\beta) = \beta$.*

Proof. Assume for contradiction that for all x there exists a β_x 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 2.3 we see that $|\{y \in (\{0,1\}^m)^N : \forall x, y[x] \neq \beta_x\}|$ is maximized when $\beta_x = 0^N$. Thus because $Y \subseteq (\{0,1\}^m)^N$,

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

Consider the space $[mN]$ where each element is indexed by $(i, \alpha) \in [N] \times [m]$. For each $x \in X$, let $\mathcal{S}_x \subseteq [mN]$ be the set defined by including (i, α) iff $x[i] = \alpha$, and let $\mathcal{S}_X = \{\mathcal{S}_x : x \in X\}$. By the fact that $m^{0.95} \gg O(n \log m)$ and $N \leq n$, $\mathcal{S}_{\mathbf{X}}$ has blockwise min-entropy $0.95 \log m - O(1) > \log(Kn \log m)) \geq \log(K \log(N/\epsilon))$, where $\epsilon := 2^{-n \log m}$ and K is the constant given by Blockwise Robust Sunflower Lemma. Thus we can apply Blockwise Robust Sunflower Lemma to $\mathcal{S}_X$ and get that $\mathbf{Pr}_{S_y \subseteq [mN]}(\forall \mathcal{S}_x \in \mathcal{S}_X, \mathcal{S}_x \not\subseteq \mathcal{S}_y) \leq \epsilon$,[5] and if we look at y as being the indicator vector for $\mathcal{S}_y$ then we get that $\mathbf{Pr}_{\boldsymbol{y} \sim \{0,1\}^{mN}}(\forall x \in X, y[x] \neq 0^x) \leq \epsilon$. Thus by counting we get

$$|Y| \leq |\{y \in (\{0,1\}^m)^N : \forall x, y[x] \neq 0^N\}| \leq \epsilon \cdot 2^{mN} = 2^{mN - n \log m}$$

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

[4] While we simplify things in this section by using $m = n^{1.1}$, our improved gadget size (see Section 5) crucially uses the improvements in Blockwise Robust Sunflower Lemma over the basic Robust Sunflower Lemma; the same improvements also give us a very short proof of our main lemma. However, these improvements aren't strictly necessary for our techniques; in Section 7 we provide an alternate proof just using Robust Sunflower Lemma.

[5] Recall that it does not matter that $\mathcal{S}_y$ is not necessarily block-respecting.

4.1 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 [GPW17]. Let $N \leq n$ and let $X \times Y \subseteq [m]^N \times (\{0,1\}^m)^N$. 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 "good" in a similar sense to the statement of Full Range Lemma. More specifically, for each $X^j \times Y^{j,\beta}$ there is some set of coordinates $J \subseteq [N]$ such that X and Y are completely fixed on J and "very unfixed" on $[N] \setminus J$. For X this means high blockwise min-entropy of $\boldsymbol{X}_{\bar{J}}$, meaning that every joint setting of some set of free coordinates is roughly equally likely. For Y the universe $(\{0,1\}^m)^N$ is so large in comparison to $[m]^N$ that a lower bound on $|Y^{j,\beta}_{\bar{J}}|$ is enough to assure $Y^{j,\beta}$ is free enough in the unfixed coordinates.

Definition 4.1. Let $N \leq n$ and let $\rho \in \{0,1,*\}^N$ be a partial assignment with $J := \mathrm{fix}(\rho) \subseteq [N]$. A rectangle $R = X \times Y \subseteq [m]^N \times (\{0,1\}^m)^N$ is *ρ-structured* if the following conditions hold:

- X and Y are fixed on the blocks J and $\textsc{Ind}_m^J(X_J, Y_J) = \rho[J]$
- $\boldsymbol{X}_{\bar{J}}$ has blockwise min-entropy at least $0.95 \log m$
- $|Y_{\bar{J}}| > 2^{m \cdot |\bar{J}| - n \log m}$

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 $I_j \subseteq [N]$ and has blockwise min-entropy $0.95 \log m$ on $\overline{I_j}$—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 assigment (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 $[N] \setminus I_j$ by our first goal, so clearly $\mathrm{fix}(\rho^{j,\beta}) = I_j$ for any k. 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.

Algorithm 1: Rectangle Partition

```
Initialize F = ∅, j = 1, and X^{≥1} := X;
PHASE I (X^j): while |X^{≥j}| ≥ |X|/2 do
    Let I_j be a maximal (possibly empty) subset of [N] such that X^{≥j} violates
      0.95 log m-blockwise min-entropy on I_j, and let α_j ∈ [m]^{I_j} be an outcome witnessing
      this: Pr_{x~X^{≥j}}(x[I_j] = α_j) > 2^{-0.95|I_j| log m};
    Define X^j := {x ∈ X^{≥j} : x[I_j] = α_j};
    Update F ← F ∪ {(I_j, α_j)}, X^{≥j+1} := X^{≥j} ∖ X^j, and j ← j + 1;
end
PHASE II (Y^{j,β}): for (I_j, α_j) ∈ F, β ∈ {0,1}^{I_j} do
    Let Y' = {y ∈ Y : y[I_j, α_j] = β}, and let η^{j,β} ∈ ({0,1}^m)^{|I_j|} be the string which
      maximizes |{y ∈ Y' : y[I_j] = η^{j,β}}|;
    Define Y^{j,β} := {y ∈ Y : y[I_j] = η^{j,β}};
end
return F, {X^j}_j, {Y^{j,β}}_{j,β};
```

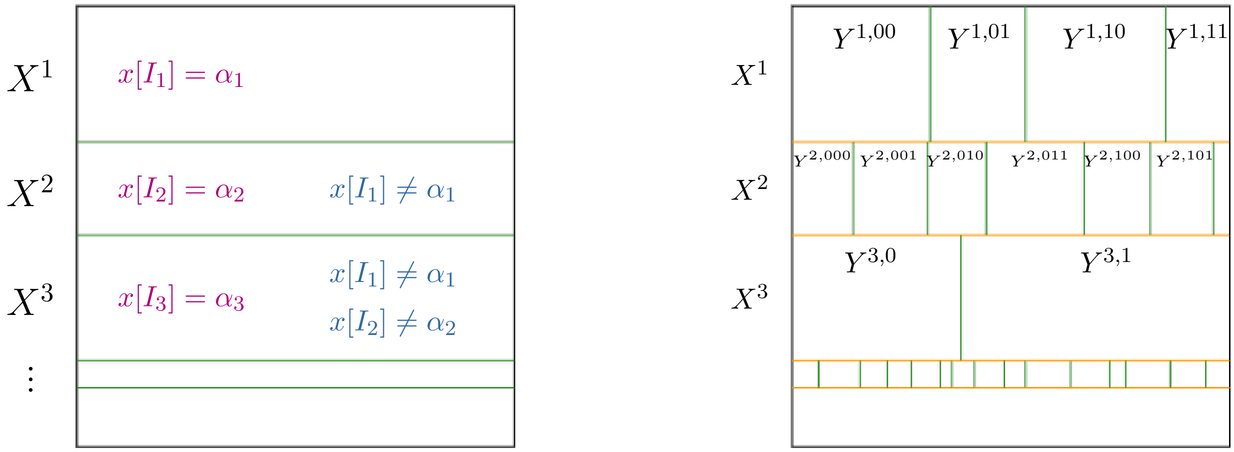

Figure 2: Phases I and II of Rectangle Partition. In each $X^j \times Y^{j,\beta}$, $x[I_j]$ is fixed to α_j and $y[I_j]$ is fixed so that $\textsc{Ind}_m^{I_j}(X^j_{I_j}, Y^{j,\beta}_{I_j}) = \beta$.

Our algorithm is formally described in Rectangle Partition.[6] Let $X \subseteq [m]^N$, let $Y \subseteq (\{0,1\}^m)^N$, and let $\mathcal{F}$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$ be the outputs of the rectangle partition 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.

Claim 4.2. *For all j and for all $\beta \in \{0,1\}^{I_j}$, define $\rho^{j,\beta} \in \{0,1,*\}^N$ to be the restriction where $\text{fix}(\rho^{j,\beta}) = I_j$ and $\rho^{j,\beta}[I_j] = \beta$. Then $X^j \times Y^{j,\beta}$ is fixed on I_j and outputs $\textsc{Ind}_m^{I_j}(X^j_{I_j}, Y^{j,\beta}_{I_j}) = \rho^{j,\beta}[I_j]$.*

Proof. By definition X^j is fixed to α_j on the coordinates I_j, while $Y^{j,\beta}$ is fixed to $\eta^{j,\beta}$ on the blocks I_j. Since $\eta^{j,\beta} \in \{0,1\}^{I_j}$ clearly satisfies $\eta^{j,\beta}[\alpha_j] = \beta$, it holds that $\textsc{Ind}_m^{I_j}(X^j_{I_j}, Y^{j,\beta}_{I_j}) = \beta = \rho^{j,\beta}[I_j]$. $\square$

Claim 4.3. *For all j, $\mathbf{X}^j_{\overline{I_j}}$ has blockwise min-entropy at least $0.95 \log m$.*

Proof. Assume for contradiction that $I^* \subseteq [N] \setminus I_j$ 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. $\square$

Before moving to the third condition, the size of $Y^{j,\beta}_{\overline{I_j}}$, 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 4.4. *For all $(I_j, \alpha_j) \in \mathcal{F}$, $\mathbf{D}_\infty(\boldsymbol{X}^j_{\overline{I_j}}) \;\leq\; \mathbf{D}_\infty(\boldsymbol{X}) - 0.05|I_j| \log m + 1$.*

[6] For those familiar with previous works [GPW17], Rectangle Partition varies in two ways: 1) we truncate Phase I once we've partitioned at least half of X; and 2) in Phase II we fix the rest of $Y^{j,\beta}$ inside the blocks I_j.

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 a simple calculation

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

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 $|Y^{j,\beta}_{\overline{I_j}}| = |Y^{j,\beta}|$ 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 $|Y^{j,\beta}| \approx |Y|/2^{m \cdot |I_j|}$; for us it is enough that the smallest $Y^{j,\beta}$ be a factor of $2^{-O(|I_j| \log m)}$ away from this. We add two new assumptions on $X \times Y$: 1) $\boldsymbol{X}$ starts with blockwise min-entropy very close to $0.95 \log m$; and 2) Y is initially large. 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. This lemma is a fairly direct application of Full Range Lemma.

Lemma 4.5. *Let $X := \cup_j X^j$ be such that $\boldsymbol{X}$ has blockwise min-entropy $0.95 \log m - O(1)$, and let Y be such that $|Y| > 2^{mN - n \log m + 1}$. Then there is a j such that for all $\beta \in \{0,1\}^{I_j}$,*

$$|Y^{j,\beta}_{\overline{I_j}}| \geq |Y|/2^{m \cdot |I_j| + 2|I_j| \log m}$$

Proof. We will show that there exists a j such that for every $\beta \in \{0,1\}^{I_j}$, $|\{y \in Y : y[I_j, \alpha_j] = \beta\}| \geq |Y|/2^{2|I_j| \log m}$. If this is true, then by averaging there is some assignment to I_j—aka $\eta^{j,\beta}$—such that

$$|Y^{j,\beta}_{\overline{I_j}}| = |Y^{j,\beta}| \geq (|Y|/2^{2|I_j| \log m})/2^{m \cdot |I_j|} = |Y|/2^{m \cdot |I_j| + 2|I_j| \log m}$$

Assume for contradiction that for every j there exists a β_j such that $|\{y \in Y : y[I_j, \alpha_j] = \beta_j\}| < |Y|/2^{2|I_j| \log m}$. Define $Y_= := \{y \in Y : \exists j, y[I_j, \alpha_j] = \beta_j\}$ and $Y_{\neq} := Y \setminus Y_= = \{y \in Y : \forall j, y[I_j, \alpha_j] \neq \beta_j\}$. We will show that $|Y_=| < |Y|/2$; if this is the case then for it must be that $|Y_{\neq}| \geq |Y|/2 > 2^{mN - n \log m}$. By Full Range Lemma there must exist some $x^* \in X$ such that for every $\beta \in \{0,1\}^N$ there exists $y_\beta \in Y_{\neq}$ such that $y_\beta[x^*] = \beta$. Since $x^* \in X$, $x^* \in X^j$ for some j, and so for any $\beta \in \{0,1\}^N$ such that $\beta[I_j] = \beta_j$, there exists a $y_\beta \in Y_{\neq}$ such that $y_\beta[x^*] = \beta$. 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 $\mathcal{F}(k) := \{(I_j, \alpha_j) \in \mathcal{F} : |I_j| = k\}$. Clearly $|\mathcal{F}(k)| \leq 2^{1.9k \log m - 1}$ since there are at most $\binom{N}{k} \ll 2^{0.9k \log m - 1}$ possible sets I_j, and for each there are m^k possible assignments α_j. Furthermore $\mathcal{F}(0)$ must be empty, because if the empty restriction where $I_j = \emptyset$ is in $\mathcal{F}$, then the corresponding β_j would be empty and $\{y \in Y : y[\emptyset, \emptyset] = \emptyset\} = Y$ would be of size $|Y|/2^0$, which contradicts our choice of β_j. Since we assumed that $|\{y \in Y : y[I_j, \alpha_j] = \beta\}| < |Y|/2^{2|I_j| \log m}$ for all j, by our bound on $|\mathcal{F}(k)|$

$$\begin{aligned}
|Y_=| &< \sum_{k=1}^{N} (2^{1.9k \log m - 1} \cdot \frac{|Y|}{2^{2k \log m}}) \\
&\leq \frac{|Y|}{2} \cdot \sum_{k=1}^{N} (2^{0.1 \cdot \log m})^{-k} \\
&< \frac{|Y|}{2} \cdot \sum_{k=1}^{\infty} 2^{-k} = \frac{|Y|}{2}
\end{aligned}$$

which completes the proof.

$\square$

4.2 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 $f \circ \textsc{Ind}_m^n$ 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 $f \circ \textsc{Ind}_m^n$ we want to construct a decision-tree of depth $O(d)$ for f. Note that we can assume that $d = o(n)$ as the theorem is trivial otherwise. 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 $\textsc{Ind}_m^n(X \times Y)$ 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 4.5, 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 Simulation Protocol. 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.

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 Rectangle Partition on $(X \cap X_v)_{\bar{J}} \times (Y \cap Y_v)_{\bar{J}}$ and let $\mathcal{F} = \{(I_j, \alpha_j)\}_j$, $\{X^j\}_j$, $\{Y^{j,\beta}\}_{j,\beta}$ be the outputs;
>
> Apply Lemma 4.5 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 $X_{\bar{J}} \leftarrow X^j$ and $Y_{\bar{J}} \leftarrow Y^{j,\beta}$, and update $\rho \leftarrow \rho \cup \rho^{j,\beta}$ (recall that $\rho^{j,\beta} \in \{0,1,*\}^n$ is the restriction where $\mathrm{fix}(\rho^{j,\beta}) = I_j$ and $\rho^{j,\beta}[I_j] = \beta$);

end

Output the same value as v does;

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

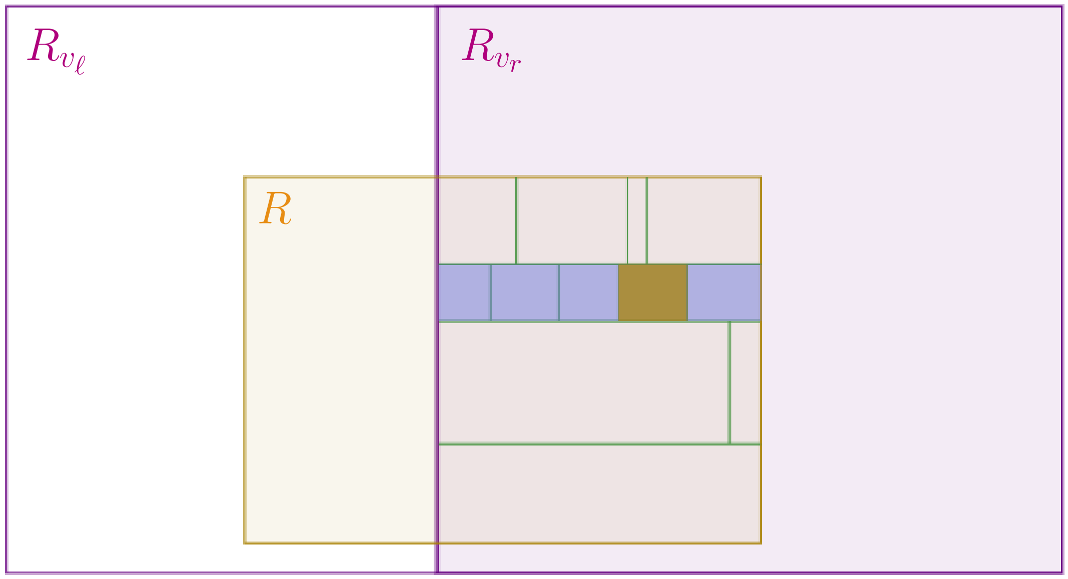

Figure 3: One iteration of Simulation Protocol. We perform Rectangle Partition (green lines) on the larger half of R after moving from v to its child (shaded in purple), use Lemma 4.5 to identify a part j (shaded in blue), and then query I_j and set R to $X^j \times Y^{j,\beta}$ for the result $z[I_j] = \beta$ (shaded in brown).

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 $|Y \cap Y_v| \geq |Y|/2$.

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.[7] Again for convenience we write $J^t := \text{fix}(\rho^t)$. Let (I^t, α^t) be the (possibly empty) assignment returned by Lemma 4.5 corresponding to index j^t, and let β^t be the result of querying $z[I^t]$. Note that $J^{t+1} = J^t \sqcup I^t$, $X^{t+1}_{\overline{J^{t+1}}} = X^{j^t}_{\overline{I^t}}$, and $Y^{t+1}_{\overline{J^{t+1}}} = Y^{j^t,\beta^t}_{\overline{I^t}}$. Also note that $t \leq d \log m$ by the depth of the protocol Π.

We show that our precondition that R^t is ρ^t-structured holds for all t, assuming for the moment that that $|J^t| \leq O(d)$ for all t. We show this by the following three invariants:

(i) X^t, Y^t are fixed on J^t and $\textsc{Ind}_m^{J^t}(X^t_{J^t}, Y^t_{J^t}) = \rho^t[J^t]$
(ii) $\boldsymbol{X}^t_{\overline{J^t}}$ has blockwise min-entropy at least $0.95 \log m$
(iii) $|Y^t_{\overline{J^t}}| \geq 2^{m \cdot |\overline{J^t}| - t - 2|J^t| \log m}$.

This is enough to show R^t is ρ^t structured as $2^{m \cdot |\overline{J^t}| - t - 2|J^t| \log m} > 2^{m \cdot |\overline{J^t}| - n \log m}$ by assumption on $|J^t|$. 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 invariants holds for the t-th iteration. After applying Rectangle Partition invariant *(i)* follows by Claim 4.2 and invariant *(ii)* follows by Claim 4.3. For invariant *(iii)* we first show that it is valid to apply Lemma 4.5 in the $(t+1)$-th

[7] 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.

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 $|(Y^t \cap Y_v)_{\overline{J^t}}| \geq |Y^t|/2 > 2^{m \cdot |\overline{J^t}| - t - 2|J^t| \log m - 1} > 2^{m \cdot |\overline{J^t}| - O(d \log m)}$, and recall that $d = o(n)$. Thus we can apply Lemma 4.5 and we get

$$\begin{aligned}
|Y^{t+1}_{\overline{J^{t+1}}}| &= |Y^{j^t, \beta^t}_{\overline{I^t}}| \\
&\geq |(Y^t \cap Y_v)_{\overline{J^t}}| / 2^{m \cdot |I^t| + 2|I^t| \log m} \\
&\geq 2^{m \cdot |\overline{J^t}| - t - 2|J^t| \log m - 1} / 2^{m \cdot |I^t| + 2|I^t| \log m} \\
&\geq 2^{m \cdot (|\overline{J^t}| - |I_t|) - t - 1 - 2(|J^t| + I^t|) \log m} = 2^{m \cdot |\overline{J^{t+1}}| - (t-1) - 2|J^{t+1}| \log m}
\end{aligned}$$

To show that $|J^t| \leq O(d)$—and by extension that our simulation is guaranteed to be efficient—it is enough to show that $\mathbf{D}_\infty(\boldsymbol{X}^t_{\overline{J^t}}) \leq 2t - 0.05|J^t| \log m$ for every $t \leq d \log m$, as this gives a bound of $|J^t| \leq 2t/0.05 \log m = O(d)$ by the non-negativity of deficiency. When $t = 0$ then J^t is empty and $\mathbf{D}_\infty(\boldsymbol{X}) = 0$. Now in the t-th iteration recall that we query the set I^t, and by Claim 4.4 we get that

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

We finally have to argue that if we reach a leaf v of Π while maintaining R and ρ, 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 $z[\text{fix}(\rho)] = \rho[\text{fix}(\rho)]$. 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 $\textsc{Ind}_m^{\text{fix}(\rho)}(x,y) = \rho[\text{fix}(\rho)] = \beta[\text{fix}(\rho)]$ for all $(x,y) \in R$, we focus on $\text{free}(\rho)$, and let $N := |\text{free}(\rho)|$. Since R is ρ-structured, $\boldsymbol{X}_{\text{free}(\rho)}$ has blockwise min-entropy $0.95 \log m$ and $|Y_{\text{free}(\rho)}| > 2^{m \cdot |\text{free}(\rho)| - n \log m}$. Thus applying Full Range Lemma, we know that that there exists $(x,y) \in R$ such that $\textsc{Ind}_m^n(x,y) = \beta$, which is a contradiction as $R \subseteq R_v \subseteq (f \circ \textsc{Ind}_m^n)^{-1}(o)$. □

5 Optimizing the gadget size

In Section 4 we loosely chose $m = n^{1.1}$ for the purpose of showing the basic lifting statement. In this section we improve from $n^{1.1}$; more specifically we show a tradeoff between the gadget size and the strength of the lifting theorem. Ultimately our tradeoff gives an optimal gadget size of m being quasilinear in n.

Theorem 5.1 (Improved Lifting Theorem). *Let f be a search problem over $\{0,1\}^n$, and let $m = \Omega(n \log n)$. Then*

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

Warm-up: $m = n^{1+\epsilon}$. First we improve on Basic Lifting Theorem 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 Full Range Lemma, and in both cases we can apply Blockwise Robust Sunflower Lemma 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 $m \geq n^{1+\epsilon}$ with an appropriate choice of δ. 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 4.4 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 $(2/\delta) \cdot d = O(d)$ 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 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 Improved Basic Lifting Theorem as a special case.

Theorem 5.2 (**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. 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 Rectangle Partition we set the blockwise min-entropy threshold for a violating assignment (I_j, α_j) at $(1-\delta)\log m$ as well.

To prove Full Range Lemma, note that we can apply Claim 2.3 regardless of m and N, and we can still apply Blockwise Robust Sunflower Lemma as long as we can choose m such that $(1-\delta)\log m - 2 > \log(Kn \log m)$. Thus for this altered rectangle partition procedure, by the same proofs as before, Claim 4.3 states that $\boldsymbol{X}^j_{\bar{I}_j}$ has blockwise min-entropy at least $(1-\delta)\log m$, Claim 4.4 states that $\mathbf{D}_\infty(\boldsymbol{X}^j_{\bar{I}_j}) \;\leq\; \mathbf{D}_\infty(\boldsymbol{X}) - |I_j| \cdot \delta \log m + 1$, and Lemma 4.5 states that if $\boldsymbol{X}$ has blockwise min-entropy $(1-\delta)\log m - 2$ and $|Y| > 2^{mN - n\log m}$, then there exists a j such that for all β, $|Y^{j,\beta}| \geq |Y|/2^{m \cdot |I_j| + 2|I_j| \log m}$.

Now our simulation procedure is the same as Simulation Protocol. 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 4.4 we get that $\mathbf{D}_\infty(\boldsymbol{X}^t_{\overline{J^t}}) \leq 2t - |J^t| \cdot \delta \log m$, which implies $|J^t| \leq 2t/\delta \log m \leq 2d$. Our precondition holds by applying the new versions of Claim 4.2, Claim 4.3, and Lemma 4.5 as before. Finally our simulation is correct again by the invariants and Full Range Lemma. ☐

6 Lifting for dag-like protocols

In this section we show that we can perform our lifting theorem in the dag-like model, going from decision dags to communication dags. This was originally proven by Garg et al. [GGKS18] using an alternate proof of Full Range Lemma, and we follow their proof exactly; in fact the only difference is that the parameters in Full Range Lemma require them to define ρ-structured with $|Y| \geq 2^{mn-n^3}$, whereas our definition of ρ-structured is the stricter $|Y| \geq 2^{mn - n\log m}$, which will again allow us to show the same improvements as in Section 5.

6.1 Decision Trees and Dags

To begin we generalize the definition of decision trees and communication protocols for solving search problems [Raz95, Pud10, Sok17]. Let $f \subseteq \mathcal{Z} \times \mathcal{O}$ be a search problem where $\mathcal{Z} = \{0,1\}^n$ and $\mathcal{O}$ is the set of potential solutions to the search problem. Let $\mathcal{Q}$ be a family of functions from $\mathcal{Z}$ to $\{0,1\}$. A $\mathcal{Q}$-decision tree T for f is a tree where each internal vertex v of T is labeled with a function $q_v \in \mathcal{Q}$, each leaf vertex of T is labelled with some $o \in \mathcal{O}$ and satisfying the following properties:

- $q_v^{-1}(1) = \mathcal{Z}$ when v is the root of T
- $q_v^{-1}(1) \subseteq q_u^{-1}(1) \cup q_w^{-1}(1)$ for any node v with children u and w
- $q_v^{-1}(1) \subseteq f^{-1}(o)$ for any leaf node v labeled with $o \in \mathcal{O}$

We can see that ordinary decision trees are $\mathcal{Q}$-decision trees where $\mathcal{Q}$ is the set of juntas (i.e., conjunctions of literals). The root corresponds to the trivially satisfiable junta 1, and the leaves are labeled with some junta that is sufficient to guarantee some answer $o \in \mathcal{O}$. At any node v with children u and w, since $v(z)$, $u(z)$, and $w(z)$ are all juntas and $q_v^{-1}(1) \subseteq q_u^{-1}(1) \cup q_w^{-1}(1)$, it is not hard to see that there is some variable z_i such that $u(z)$ is a relaxation of $q_v(z) \wedge z_i$ and $q_w(z)$ is a relaxation of $q_v(z) \wedge \overline{z_i}$, or vice-versa.

This notion also generalizes to communication complexity search problems $F \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$, where now $\mathcal{Q}$ is the family of functions from $\mathcal{X} \times \mathcal{Y}$ to $\{0,1\}$ corresponding to combinatorial rectangles $X \times Y \subseteq \mathcal{X} \times \mathcal{Y}$; more specifically $q^{X \times Y}(x,y) = 1$ iff $(x,y) \in X \times Y$. Since $q_v^{-1}(1) \subseteq q_u^{-1}(1) \cup q_w^{-1}(1)$, it must be that the rectangles we test membership for at u and w cover the rectangle being tested at v, and again it is not hard to see that this corresponds to a relaxation of testing membership in $X_v = X_u \sqcup X_w$ or $Y_v = Y_u \sqcup Y_w$.

Now we generalize this notion to dags. For a search problem $f \subseteq \mathcal{Z} \times \mathcal{O}$ and a family of functions $\mathcal{Q}$ from $\mathcal{Z}$ to $\{0,1\}$, a $\mathcal{Q}$-*dag* is a directed acyclic graph D where each internal vertex v of the dag is labeled with a function $q_v(z) \in \mathcal{Q}$ and each leaf vertex is labelled with some $o \in \mathcal{O}$ and satisfying the following properties:

- $q_v^{-1}(1) = \mathcal{Z}$ when v is the root of D
- $q_v^{-1}(1) \subseteq q_u^{-1}(1) \cup q_w^{-1}(1)$ for any node v with children u and w
- $q_v^{-1}(1) \subseteq f^{-1}(o)$ for any leaf node v labeled with $o \in \mathcal{O}$

For $\mathcal{Z} = \{0,1\}^n$ a *conjunction dag* D solving f is a $\mathcal{Q}$-dag where $\mathcal{Q}$ is the set of all juntas over $\mathcal{Z}$. [8] For conjunction dags our measure of complexity will be a bit different than size. The *width* of Π is the maximum number of variables occurring in any junta $v(z)$. We define

$$w(f) \;:=\; \text{least } \textit{width} \text{ of a conjunction dag solving } f.$$

For a communication complexity search problem $F \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ and a family of functions $\mathcal{Q}$ from $\mathcal{X} \times \mathcal{Y}$ to $\{0,1\}$, we define a $\mathcal{Q}$-dag solving F analogously. A *rectangle dag* Π solving F is a $\mathcal{Q}$-dag where $\mathcal{Q}$ is the set of all indicator vectors of rectangles $X \times Y \subseteq \mathcal{X} \times \mathcal{Y}$. We define

$$\textsf{rect-dag}(F) \;:=\; \text{least } \textit{size} \text{ of a rectangle dag solving } F.$$

[8] As noted above the terms "conjunction" and "junta" are closely related, but conjunctions are usually thought of as syntactic objects while juntas are functions. We keep the term conjunction dag from [GGKS18] for consistency even though we switch to using junta for the functions in $\mathcal{Q}$.

6.2 Main theorem

The following is our dag-like deterministic lifting theorem. Again an earlier version was originally proven in [GGKS18] with $m = n^2$, and we improve this to a near linear dependence on n.

Theorem 6.1 (**Dag-like Lifting Theorem** [GGKS18])**.** *Let f be a search problem over $\{0,1\}^n$, and let $m = n^{1+\epsilon}$ for some constant $\epsilon > 0$. Then*

$$\log \mathsf{rect\text{-}dag}(f \circ \textsc{Ind}_m^n) \;=\; w(f) \cdot \Theta(\log m)$$

In fact one can easily check that the same scaling argument as Scaling Basic Lifting Theorem can also be applied to the proof of Dag-like Lifting Theorem, as noted above.

Proof. For n sufficiently large let $m = n^{1+\epsilon}$ and let $d \leq n$. Again one direction is simple; given a conjunction dag D for f of width d we can construct a rectangle dag Π for $f \circ \textsc{Ind}_m^n$ of size $m^{O(d)}$ by simply replacing each edge in D with a short protocol that queries all variables fixed by the edge. Thus we will prove that given a rectangle dag Π for $f \circ \textsc{Ind}_m^n$ of size m^d we can construct a conjunction dag D of width $O(d)$ for f (again we can assume $d = o(n)$ since the problem is trivial otherwise).

Our procedure is similar to before, maintaining a ρ-structured rectangle $R \subseteq R_v$ at every step, but now there's a slight twist: the protocol may have depth greater than d and can decide to "forget" some bits at each stage, at which point we will have to make sure the assignment ρ we maintain also stays small.

This presents two problems. First off, it won't be enough to find a subrectangle of our current rectangle R, since R has some bits fixed that may be forgotten by the protocol. We circumvent this by applying the rectangle partition procedure to the *actual* rectangle R_v, which allows us to find the "important bits" as before, and then shift to a good rectangle $X^j \times Y^{j,\beta}$, leaving R behind.

The second challenge is that whenever we apply Rectangle Partition we need to ensure that every set I_j we find is of size $O(d)$. The Rectangle Lemma is the main technical lemma of [GGKS18], establishing extra properties of Rectangle Partition. We give a new proof of the Rectangle Lemma, showing that that by slightly modifying Rectangle Partition we can remove some "error sets" from X and Y and afterwards assume that all our rectangles $X^j \times Y^{j,\beta}$ are ρ-structured for some *small* restriction ρ, aka one that fixes $O(d)$ coordinates. Here we don't require that $\boldsymbol{X}$ has high blockwise min-entropy or Y is large; recall that in Rectangle Partition these conditions were only needed to find a "good" j. We prove this at the end of the section.

Lemma 6.2 (**Rectangle Lemma**, [GGKS18])**.** *Let $R = X \times Y \subseteq [m]^N \times \{0,1\}^{mN}$ and let $d < n$. Then there exists a procedure which outputs $\{X^j \times Y^{j,\beta}\}_{j,\beta}, X_{err}, Y_{err}$, where $X_{err} \subseteq X$ and $Y_{err} \subseteq Y$ have density $2^{-2d\log m}$ in $[m]^n$ and $(\{0,1\}^m)^n$ respectively, and for each j, β one of the following holds:*

- **structured***: $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$ structured for some $\rho^{j,\beta}$ of width at most $O(d)$*
- **error***: $X^j \times Y^{j,\beta} \subseteq X_{err} \times \{0,1\}^{mn} \cup [m]^n \times Y_{err}$*

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

With the Rectangle Lemma at hand, the simulation algorithm (Algorithm 3) and proof of correctness essentially follows [GGKS18]. In particular Algorithm 3 starts with a preprocessing step where for each vertex v in the communication dag, we apply Lemma 6.2. Then in a bottom-up fashion. and for each v we remove from R_v all error sets appearing in *descendants* of v.

Algorithm 3: Dag-like Simulation Protocol

PREPROCESSING: initialize $X^*_{err} = \emptyset$ and $Y^*_{err} = \emptyset$, and for all $v \in \Pi$ let $R_v := X_v \times Y_v$ be the rectangle corresponding to v;

for $v \in \Pi$ *starting from the leaves and going up to the root* **do**

  Update $X_v \leftarrow X_v \setminus X^*_{err}$ and $Y_v \leftarrow Y_v \setminus Y^*_{err}$;

  Apply Lemma 6.2 to $X_v \times Y_v$ and let $\{X^j_v\}_j$, $\{Y^{j,\beta}_v\}_{j,\beta}$, X_{err}, Y_{err}, $\{I_x\}_x$ be the outputs;

  Update $X^*_{err} \leftarrow X^*_{err} \cup X_{err}$ and $Y^*_{err} \leftarrow Y^*_{err} \cup Y_{err}$;

end

Initialize $v :=$ root of Π; $R := R_v$; $\rho = *^n$;

while v *is not a leaf* **do**

  Precondition: $R = X \times Y$ *is* ρ*-structured, for convenience define* $J := \mathrm{fix}(\rho)$*, and furthermore* $|J| \leq O(d)$;

  Apply Full Range Lemma to $X_{\bar{J}} \times Y_{\bar{J}}$ to get $x^* \in X$;

  Let v_ℓ, v_r be the children of v, let j_ℓ, j_r be the indices such that $x^* \in X^{j_\ell}_{v_\ell}$ and $x^* \in X^{j_r}_{v_r}$, and let I_{j_ℓ} and I_{j_r} be the query alignment sets I_{x^*} for v_ℓ and v_r respectively;

  Query each variable z_i for every $i \in (I_{j_\ell} \cup I_{j_r}) \setminus J$, let $\beta_{j_\ell} \in \{0,1\}^{I_{j_\ell}}$ be the result concatenated with $\rho[J]$ and restricted to I_{j_ℓ}, and let $\beta_{j_r} \in \{0,1\}^{I_{j_r}}$ be defined analogously;

  Let $y^* \in Y$ be such that $\textsc{Ind}^{I_{j_\ell}}_m(x^*, y^*) = \beta_{j_\ell}$ and $\textsc{Ind}^{I_{j_r}}_m(x^*, y^*) = \beta_{j_r}$, and let $c \in \{\ell, r\}$ be such that $(x^*, y^*) \in X^{j_c}_{v_c} \times Y^{j_c,\beta_{j_c}}_{v_c}$;

  Update $X \times Y = X^{j_c}_{v_c} \times Y^{j_c,\beta_{j_c}}_{v_c}$ and $\rho \leftarrow \rho^{j_c,\beta_{j_c}}$;

end

Output the same value as v does;

After preprocessing to remove the error sets, we enter the main while loop of the algorithm, which iteratively walks down the communication dag, maintaining the invariant that when processing node v, we have a ρ-structured rectangle R associated with v with at most $O(d)$ bits fixed. We apply Full Range Lemma to R to find some x^*. Since all error sets were removed in the preprocessing step, we are guaranteed that x^* is contained not only in R (the rectangle associated with v), but also in the rectangles associated with the children of v. That is, $x^* \in X^{j_\ell}_{v_\ell}$ and $x^* \in X^{j_r}_{v_r}$ for some $X^{j_\ell}_{v_\ell}$ generated in the left child and $X^{j_r}_{v_r}$ generated in the right child. Furthermore x^* has full range in $R \subseteq R_{v_\ell} \cup R_{v_r}$, and so there cannot be any $Y^{j_\ell,\beta}_{v_\ell}$ or $Y^{j_r,\beta}_{v_r}$ that is missing.

We use the query alignment property for I_{j_ℓ} and I_{j_r} corresponding to $X^{j_\ell}_{v_\ell}$ and $X^{j_r}_{v_r}$, and query all unknown bits for both sets. Then because of the full range of x^* we find a y^* compatible with all bits fixed, and move to the (structured) rectangle output by the partition at whichever child of v contains y^*, since R is in the union of the rectangles at v's children. Thus when we move down to a child of v, we maintain our invariant of being in a ρ-structured rectangle with at most $O(d)$ fixed bits. We state the algorithm (Algorithm 3) formally in the next page.

We briefly go over the invariants needed to run Dag-like Simulation Protocol. For the preprocessing step consider any node v. Since the number of descendants of v is at most $|\Pi| = m^d$, we know that after having removed all error sets below the current node v we've only lost a $m^d \times 2^{-2d \log m} \ll 1/2$ fraction of X_v and Y_v. At the root of Π, after processing R_v in total we've lost an $m^d \cdot 2^{-2d \log m} \ll 1/2$ fraction of $[m]^n$ and $(\{0,1\}^m)^n$ each, meaning we start with $|X_v| = m^n/2$ and $|Y_v| = 2^{mn}/2$. After this the rectangle associated with the root we will never encounter an error rectangle in our procedure. This will be the only place where we use the fact that $|\Pi| = m^d$.

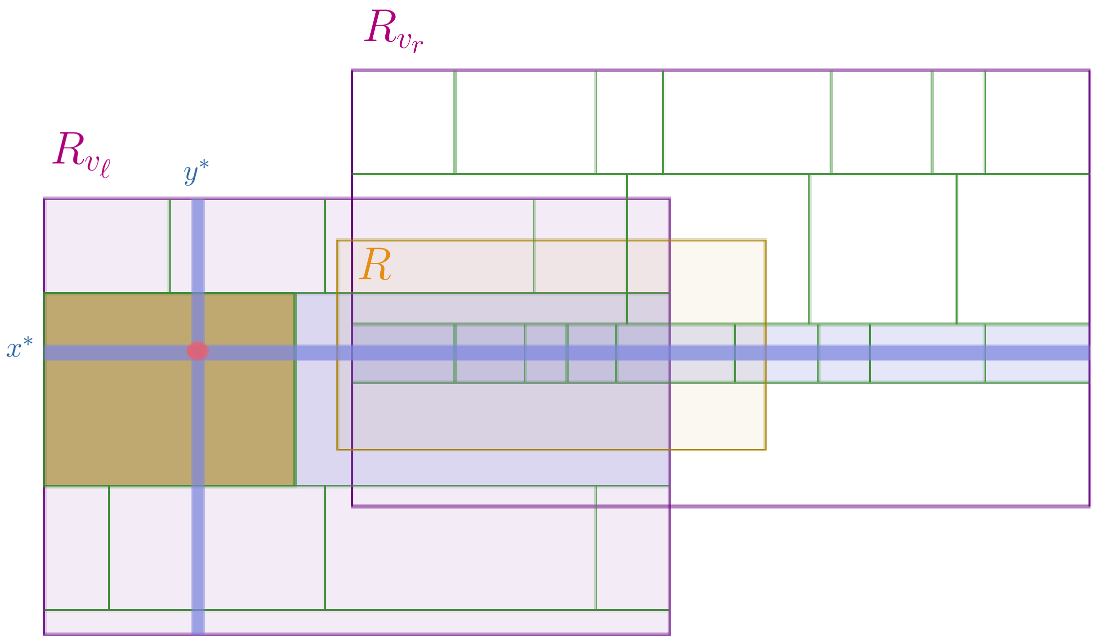

Figure 4: One iteration of Dag-like Simulation Protocol. We perform Rectangle Partition (green lines) on both R_{v_ℓ} and R_{v_r} separately, use Full Range Lemma find an $x^* \in R$ with full range, query all bits in the sets I_{j_ℓ} and I_{j_r} corresponding to $X^{j_\ell}, X^{j_r} \ni x^*$ (shaded in blue), find a y^* for which $\textsc{Ind}_m^n(x^*, y^*)$ matches the result, and set R to $X^{j_c} \times Y^{j_c,\beta_c} \ni (x^*, y^*)$ (shaded in brown) for $c \in \{\ell, r\}$ (shaded in purple).

In the main procedure, assuming the precondition of R being ρ-structured holds we meet all conditions for applying Full Range Lemma. Since x^* has full range we know that every $Y_{v_\ell}^{j_\ell,\beta_{j_\ell}}$ and $Y_{v_r}^{j_r,\beta_{j_r}}$ exists, and since we removed all error sets the rectangle $X_{v_c}^{j_c} \times Y_{v_c}^{j_c,\beta_c}$ we end up in must be in the "structured" case of Lemma 6.2. Thus again end up in an R which is ρ'-structured for some ρ' which fixes at most $O(d)$ coordinates, and so we've met the preconditions for the next round.

Our argument at the leaves is identical to the proof of Basic Lifting Theorem, but we restate it for completeness. We have to argue that if we reach a leaf v of Π while maintaining R and ρ, then the solution $o \in \mathcal{O}$ output by Π is also valid solution to the values of z, of which the decision-dag knows that $z[\mathrm{fix}(\rho)] = \rho[\mathrm{fix}(\rho)]$. 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 $\textsc{Ind}_m^{\mathrm{fix}(\rho)}(x, y) = \rho[\mathrm{fix}(\rho)] = \beta[\mathrm{fix}(\rho)]$ for all $(x, y) \in R$, we focus on $\mathrm{free}(\rho)$, and let $N := |\,\mathrm{free}(\rho)|$. Since R is ρ-structured, $\boldsymbol{X}_{\mathrm{free}(\rho)}$ has blockwise min-entropy $0.95 \log m$ and $|Y_{\mathrm{free}(\rho)}| > 2^{m \cdot |\,\mathrm{free}(\rho)| - n \log m}$. Thus applying Full Range Lemma, we know that that there exists $(x, y) \in R$ such that $\textsc{Ind}_m^n(x, y) = \beta$, which is a contradiction as $R \subseteq R_v \subseteq (f \circ \textsc{Ind}_m^n)^{-1}(o)$.

□

Proof of Lemma 6.2. Our procedure for generating rectangles $X^j \times Y^{j,\beta}$ will be nearly the same as Rectangle Partition, with the one small caveat that we run Phase I until $X^{\geq j}$ is empty instead of stopping after partitioning half of X.[9] Let $R \subseteq [m]^n \times \{0,1\}^{mn}$, let $\{X^j \times Y^{j,\beta}\}_{j,\beta}$ be the output

[9] Our procedure doesn't require a drop in deficiency anymore, since it's enough to maintain the invariant that we've fixed at most $O(d)$ coordinates.

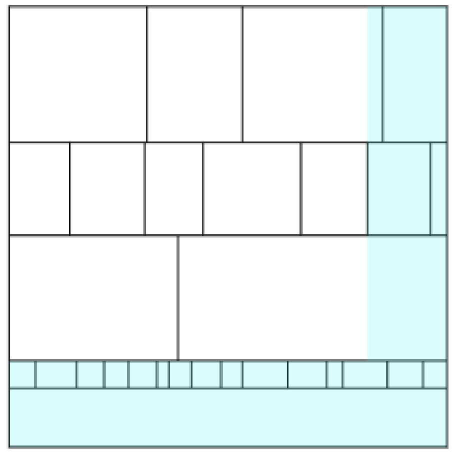

Figure 5: Error rectangles shaded in blue. X^j is added to X_{err} if I_j is too large (bottom), while $Y^{j,\beta}$ is added to Y_{err} if $Y^{j,\beta}$ is too small (right).

of this procedure on R, and let $\mathcal{F} = \{(I_j, \alpha_j)\}$ be the set of assignments found in the procedure as in Rectangle Partition. We first need to define the error rectangles X_{err} and Y_{err}. Intuitively every "structured" $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$ structured for some $\rho^{j,\beta}$, and furthermore we want to ensure that the number of bits fixed in $\rho^{j,\beta}$ is at most $O(d)$. For X this means ensuring that I_j is small, while for Y this means ensuring that $Y^{j,\beta}$ is large. We initialize $J_{good} = [|\mathcal{F}|]$, and we repeatedly find "bad" $j \in J_{good}$ and add either X^j or $Y^{j,\beta}$ to X_{err} or Y_{err}.

- X_{err}: while there exists $j \in J_{good}$ such that $|I_j| > 40d$, update $X_{err} \leftarrow X_{err} \cup X^j$ and $J_{good} \leftarrow J_{good} \setminus \{j\}$
- Y_{err}: while there exists $j \in J_{good}$ and β such that $|Y^{j,\beta} \setminus Y_{err}| < 2^{m \cdot |\overline{I_j}| - 5d \log m}$, update $Y_{err} \leftarrow Y_{err} \cup Y^{j,\beta}$ for all such β and $J_{good} \leftarrow J_{good} \setminus \{j\}$

We prove a series of short claims about X_{err} and Y_{err}, most of which immediately follow in the same way as Claim 4.2, Claim 4.3, and Lemma 4.5. The first puts these claims together to show that all rectangles corresponding to $j \in J_{good}$ fulfill the "structured" case of Lemma 6.2.

Claim 6.3. *For all $j \in J_{good}$ and all $\beta \in \{0,1\}^{I_j}$, $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$ structured for some $\rho^{j,\beta}$ which fixes at most $O(d)$ coordinates.*

Proof. As usual, for all j and for all $\beta \in \{0,1\}^{I_j}$, define $\rho^{j,\beta} \in \{0,1,*\}^N$ to be the restriction where $\mathrm{fix}(\rho^{j,\beta}) = I_j$ and $\rho^{j,\beta}[I_j] = \beta$. Then

- by Claim 4.2, $X^j \times Y^{j,\beta}$ is fixed on I_j and outputs $\textsc{Ind}_m^{I_j}(X^j_{I_j}, Y^{j,\beta}_{I_j}) = \rho^{j,\beta}[I_j]$.
- by Claim 4.3, $\boldsymbol{X}_{\overline{I_j}}$ has blockwise min-entropy $0.95 \log m$.
- since $j \in J_{good}$, it must be that $|Y^{j,\beta}| \geq 2^{m \cdot |\overline{I_j}| - 5d \log m} \geq 2^{m \cdot |\overline{I_j}| - n \log m}$. [10]

and so $X^j \times Y^{j,\beta}$ is $\rho^{j,\beta}$-structured. Furthermore, since $j \in J_{good}$ it must be the case that $|\mathrm{fix}(\rho^{j,\beta})| = |I_j| \leq 40d$. $\square$

[10] Note that even if some of $Y^{j,\beta}$ is lost when we expand Y_{err} for some j', the loop will not terminate until *every* $Y^{j,\beta}$ that is currently too small is added, and so if $Y^{j,\beta}$ survives then it must be large enough.

Finally we handle the density of the error rectangles. In our simulation this will be used to ensure we can apply Full Range Lemma at every step.

Claim 6.4. $|X_{err}| \leq m^n \cdot 2^{-2d\log m}$ *and* $|Y_{err}| \leq 2^{mn} \cdot 2^{-2d\log m}$

Proof. For X_{err} we have two cases: either X_{err} is empty, in which case the claim is trivial, or X_{err} is not empty and there is some minimal j such that X^j gets added to X_{err}, and by extension $|I_j| > 40d$. Recall that we showed $|X^j| \leq |X^{\geq j}| \cdot 2^{-0.95|I_j|\log m}$, and by extension $\mathbf{H}_\infty(\boldsymbol{X}^j) \geq \mathbf{H}_\infty(\boldsymbol{X}^{\geq j}) - |I_j| \cdot 0.95 \log m$. Then because X^j is a set in $[m]^n$ fixed on coordinates $I_j \subseteq n$, $\mathbf{H}_\infty(\boldsymbol{X}^j) \leq (n - |I_j|)\log m$. Combining these two bounds gives us $\mathbf{H}_\infty(\boldsymbol{X}^{\geq j}) \leq (n - 0.05|I_j|)\log m$, and note that $X_{err} \subseteq X^{\geq j}$. Thus by our choice of j we get that

$$|X_{err}| \leq |X^{\geq j}| < 2^{(n-0.05\cdot 40d)\log m} < m^n \cdot 2^{-2d\log m}$$

For Y_{err}, as in the proof of Lemma 4.5 for all $k \in [40d]$ there are $\binom{n}{k} \cdot m^k \cdot 2^k < 2^{3k\log m}$ choices of (I_j, α_j, β_j) such that $|Y^{j,\beta_j}| < 2^{m\cdot(N-k)-5d\log m}$, and taking a union bound we get that

$$|Y_{err}| \leq \sum_{k=1}^{40d} 2^{3k\log m} \cdot 2^{m\cdot(N-k)-5d\log m} \leq 40d \cdot 2^{m(N-1)-2d\log m} \ll 2^{mn} \cdot 2^{-2d\log m}$$

which completes the proof. □

The proof of Lemma 6.2 is now fairly immediate. The density of X_{err} and Y_{err} follows from Claim 6.4. For any $X^j \times Y^{j,\beta}$, if $j \in J_{good}$ then by Claim 6.3 this fulfills the structured case, while if $j \notin J_{good}$ then either $X^j \subseteq X_{err}$ or $Y^{j,\beta} \subseteq Y_{err}$ by definition. The query alignment property holds by taking $I_x = I_j$ for all $x \notin X_{err}$, where j is such that $x \in X^j$. □

7 Lifting that scales with simulation length

In this section we prove a variant on Basic Lifting Theorem, which allows us to set the gadget size m in terms of the target decision tree depth d. This theorem was originally proven in [GKMP20] but here we get a simpler proof via the sunflower lemma together with simple counting.

Theorem 7.1 (Low-depth Lifting Theorem [GKMP20]). *Let f be a search problem over $\{0,1\}^n$, and let $m \geq (\mathbf{P}^{dt}(f))^{5+\epsilon}$ for some constant $\epsilon > 0$. Then*

$$\mathbf{P}^{cc}(f \circ \text{I}\small{\text{ND}}_m) \;\geq\; \mathbf{P}^{dt}(f) \cdot \Omega(\log m)$$

Proof sketch. Again we start with a given real protocol Π of depth $d\log m$ for the composed problem $f \circ \text{I}\small{\text{ND}}^n_m$ and construct a decision-tree of depth $O(d)$ for f. Putting aside Full Range Lemma for the moment, we can apply Rectangle Partition as in Basic Lifting Theorem, and Claims 4.2, 4.3, and 4.4 hold with no change in proof. For Lemma 4.5 we impose the slightly stronger precondition that $|Y| \geq 2^{mN-O(d\log m)}$,[11] which will become useful when we go to prove Full Range Lemma. Our simulation proceeds according to to Simulation Protocol, and the same proofs hold to show our simulation's efficiency. For correctness note that with our new condition that $|Y| \geq 2^{m\cdot|\mathrm{free}(\rho)|-O(d\log m)}$ the same argument holds as before, as long as $m^{1-\delta} > O(d\log m)$, where $(1-\delta)\log m$ is our blockwise min-entropy threshold in Rectangle Partition.

[11]Since our invariants actually give $|Y^t| \geq 2^{m\cdot|\mathrm{free}(\rho^t)|-t-2d\log m}$ for $t \leq d\log m$, we could have stated Lemma 4.5 this way originally, but to simplify the presentation we omitted any reference to d in Rectangle Partition

Now we return to Full Range Lemma. To see why this is our main technical challenge, note that we can apply Claim 2.3 as before, but the rest of the argument no longer holds for $m \ll n$, as we cannot apply Blockwise Robust Sunflower Lemma for $(1-\delta)\log m < \log(K\log(N/\epsilon))$. We will prove a variant of Lemma 4.1, from which Lemma 4.5 follows using our additional constraint $|Y| > 2^{mN-O(d\log m)}$. As usual we work with the relatively simple blockwise min-entropy threshold $0.95\log m$ and gadget size d^7 as a matter of clarity, and at the end of the section we prove it more generally to get better parameters. For convenience we reuse the shorthand $\gamma_j := (I_j, \alpha_j)$.

Lemma 7.2. *Let $N \leq n$, let $d = o(n)$, and let $m > d^7$. Let X be such that $\boldsymbol{X}$ has blockwise min-entropy $0.95\log m$, and let $\mathcal{F} = \{\gamma_j\}_j$ be a block-respecting set system over $[m]^N$ such that 1) for all $x \in X$ there exists a $\gamma_j \in \mathcal{F}$ consistent with x, and 2) $|\gamma_j| \leq O(d)$ for all j. Then*

$$\mathbf{Pr}_{\boldsymbol{y}\subseteq[mN]}(\forall j : \gamma_j \not\subseteq y) < 2^{-\Omega(d\log m)}$$

We prove this lemma in Section 7.1, as it is our main technical contribution. □

7.1 Proof of Lemma 7.2

Proof sketch. Our goal will be to show that $\mathcal{F}$ contains an $2^{-\Omega(d\log m)}$-robust sunflower with an empty core. Note that in proving Full Range Lemma we invoked Blockwise Robust Sunflower Lemma, which stated that X contained an ϵ-robust sunflower with an empty core. However, while we still have blockwise min-entropy on X, for $m \ll N$ we cannot use Blockwise Robust Sunflower Lemma on X.

Instead we turn to $\mathcal{F}$, and since we don't have a notion of blockwise min-entropy for $\mathcal{F}$ we switch to using the basic Robust Sunflower Lemma, and instead use the blockwise min-entropy of X to ensure $\mathcal{F}$ is large. More specifically, because the blockwise min-entropy of X is at least $0.95\log m$, for any non-empty set $\gamma_j \in \mathcal{F}$ the set of all $x \in X$ consistent with γ_j can only cover a $2^{-0.95|\gamma_j|\log m}$ fraction of X. Since each $x \in X$ must be consistent with some γ_j, there must be a huge number of sets γ_j in $\mathcal{F}$, and so by Robust Sunflower Lemma $\mathcal{F}$ contains some ϵ-robust sunflower $\mathcal{F}_{\bar{S}}$ even for very small ϵ.

If $|S| = 0$ then we are done, but unfortunately using Robust Sunflower Lemma we have no control over $|S|$. Instead, we employ the strategy an iterative strategy where we drive down the size of the smallest core S for which $\mathcal{F}_{\bar{S}}$ is an ϵ-robust sunflower. For simplicity assume there is some $s \leq 20d$ such that every set in $\mathcal{F}$ has size s, and so in the worst case we can assume that every core S for which $\mathcal{F}_{\bar{S}}$ is an ϵ-robust sunflower has size $s-1$. We want to show now that there exist enough such cores S that the collection of these cores *itself* is an ϵ-robust sunflower, and so it must have a core S' of size at most $s-2$. If this is true then it turns out $\mathcal{F}_{\bar{S}'}$ is an ϵ'-robust sunflower for ϵ' only slightly larger than ϵ. From this we've made progress; by increasing ϵ slightly we've found a core of a smaller size.

Using this idea, at a high level we will perform an iterative procedure, where we repeat the following three steps until we find a sunflower with an empty core in $\mathcal{F}$: 1) repeatedly pluck ϵ-robust sunflowers from $\mathcal{F}$; 2) when we have enough sunflowers, pluck an robust sunflower from their cores; 3) increase ϵ enough so that the core of this new sunflower is the core of an ϵ-robust sunflower in $\mathcal{F}$ as well. In our actual calculations we will need to keep track of the sets of cores of each size, as well as to focus only on the sets in $\mathcal{F}$ of a certain size. This will allow us to know when we should pluck a sunflower from the cores, and will give us a measure of progress towards finding an empty core, which will allow us to choose our ϵ small enough to get $2^{-\Omega(d\log m)}$ at the end.

The last remaining piece is showing that we can actually pluck enough sunflowers from $\mathcal{F}$ to repeat this procedure enough to get an empty core, without running out of sets in $\mathcal{F}$. Unfortunately when we find a core S and pluck the sunflower $\mathcal{F}_{\bar{S}}$, we have no control over how many sets are actually in $\mathcal{F}_{\bar{S}}$, and so it seems hopeless to control how many rounds we can run for. However, note that the $0.95 \log m$ lower bound on the blockwise min-entropy of X holds for *any* S over $[m]^N$, which applies to a) the original sets $\gamma_j \in \mathcal{F}$, and b) the cores S that we pluck. Thus instead of arguing that each $\mathcal{F}_{\bar{S}}$ we find is small, we instead argue that the fraction of X covered by sets remaining in $\mathcal{F}$ is large, using the blockwise min-entropy of X for all (non-empty) cores S we've found so far. Then, again using the blockwise min-entropy of X on $\mathcal{F}$, we know that $\mathcal{F}$ must still have many sets to cover the remaining fraction of X, as we did when showing that $\mathcal{F}$ was originally big enough to apply Robust Sunflower Lemma. □

Proof. It is sufficient to show that $\mathcal{F}$ contains an ϵ-robust sunflower with an empty core for some $\epsilon \leq 2^{-\Omega(d \log m)}$. Again we assume $\emptyset \notin \mathcal{F}$ as the lemma is trivial otherwise, and so for all $s \in [O(d)]$ let $\mathcal{F}(s)$ be the set of all sets in $\mathcal{F}$ of size exactly s, and let $X(s)$ be the set of all $x \in X$ consistent with a set in $\mathcal{F}(s)$. Since every x is consistent with some $\gamma \in \mathcal{F} = \cup_s \mathcal{F}(s)$, we know that $X = \cup_s X(s)$. Therefore by averaging there must exist some $s \in [O(d)]$ such that $|X(s)| \geq \frac{1}{O(d)}|X|$, and so we fix an arbitrary such s.

We define an iterative procedure to find an robust sunflower with an empty core in $\mathcal{F}(s)$. Set $t = 0$, set $\epsilon_0 := 2^{-\Omega(d \log m) - s^2 \log m}$, and set $\mathcal{F}^0 := \mathcal{F}(s)$. For $k = 0 \ldots s-1$ set $\mathcal{S}^k := \emptyset$. We repeat the following until we ever add a set to $\mathcal{S}^0$:

0. **abort** *if the following invariants ever do not hold:*
   (a) $|\mathcal{S}^k| \leq 2^{0.5k \log m}$
   (b) $|\mathcal{F}^t| \geq 2^{0.5s \log m}$
   (c) for every k and every $S \in \mathcal{S}^k$, $|S| = k$ and $\mathcal{F}(s)_{\bar{S}}$ is an ϵ_t-robust sunflower
   (d) $\epsilon_t < 2^{-\Omega(d \log m)}$
1. let $\mathcal{F}^t_{\bar{S}}$ be an ϵ_t-robust sunflower in $\mathcal{F}^t$; if none exists, **abort**
2. increment t and set $\epsilon_t \leftarrow \epsilon_{t-1}$
3. set $\mathcal{S}^{|S|} \leftarrow \mathcal{S}^{|S|} \cup \{S\}$ and set $\mathcal{F}^t \leftarrow \mathcal{F}^{t-1} - \mathcal{F}^{t-1}_{\bar{S}}$
4. while there exists k such that $|\mathcal{S}^k| = 2^{0.5k \log m}$:
   (a) if $k = 0$, **exit** and return ϵ_t
   (b) let $\mathcal{S}^k_{\bar{S}}$ be an ϵ_0-robust sunflower in $\mathcal{S}^k$; if none exists, **abort**
   (c) increment t and set $\epsilon_t \leftarrow \epsilon_{t-1} + \epsilon_0$
   (d) set $\mathcal{S}^{|S|} \leftarrow \mathcal{S}^{|S|} \cup \{S\}$, set $\mathcal{S}^{k'} \leftarrow \mathcal{S}^{k'} - \mathcal{S}^{k'}_{\bar{S}}$ for all $k' > |S|$, and set $\mathcal{F}^t \leftarrow \mathcal{F}^{t-1} - \mathcal{F}^{t-1}_{\bar{S}}$

If this process exits without aborting, clearly by invariants (c) and (d), $\mathcal{F}(s)$ is a $2^{-\Omega(d \log m)}$-robust sunflower with an empty core as desired (note that when the procedure exits, $|\mathcal{S}^0| = 2^{0.5 \cdot 0 \log m} = 1$). Thus we prove that the process never aborts.

First we show that by Robust Sunflower Lemma, in steps 1 and 4b we always find an robust sunflower. Recall that $m > d^7$.[12] For step 1, by invariant (b) and the fact that $1/\epsilon_t \leq 1/\epsilon_0 = 2^{\Omega(d \log m) + s^2 \log m}$ we have

$$|\mathcal{F}^t| \geq 2^{0.5s \log m} = (m^{0.5})^s \gg (\Omega(d^3 \log m))^s \geq (s \cdot \log \exp(\Omega(d \log m) + s^2 \log m))^s \geq (s \cdot \log 1/\epsilon_t)^s$$

and for step 4b by the inner loop condition the same calculation shows

$$|\mathcal{S}^k| = 2^{0.5k \log m} = (m^{0.5})^k \gg (\Omega(d^3 \log m))^k \geq (k \cdot \log \exp(\Omega(d \log m) + s^2 \log m))^k = (k \cdot \log 1/\epsilon_0)^k$$

[12] Here is the first place where we use the gadget size.

We now prove that the invariants hold. For invariant (a), clearly after exiting the inner loop $|\mathcal{S}^k| < 2^{0.5k\log m}$ for all k. Before the inner loop runs we add at most one element to at most one set $\mathcal{S}^k$, and thus for that set $|\mathcal{S}^k| < 2^{0.5k\log m}+1$, or in other words $|\mathcal{S}^k| \leq 2^{0.5k\log m}$. At the start of each iteration of the inner loop at most one set $\mathcal{S}^k$ has size $2^{0.5k\log m}$, and since we remove at least one element from it and add at most one element to at most one other set we maintain that invariant.

For invariant (b), assume for contradiction that $|\mathcal{F}^t| < 2^{0.5s\log m}$. Recall that $\boldsymbol{X}$ has blockwise min-entropy at least $0.95\log m$, meaning that *every* set S over $[m]^N$ covers at most $2^{-0.95|S|\log m}\cdot|X|$ elements in X, and by extension in $X(s)$. In particular this applies to every set $\gamma_j \in \mathcal{F}^t$ as well as every set $S \in \mathcal{S}^k$. Lastly by assumption $|\mathcal{F}^t| < 2^{0.5s\log m}$, and likewise by invariant (a) we know that $|\mathcal{S}^k| < 2^{0.5k\log m}$ for every k. Therefore since $m > d^7$,[13]

$$\begin{aligned}
|X(s)| &\leq |\mathcal{F}^t|\cdot(2^{-0.95s\log m}\cdot|X|)+\sum_{k=1}^{s-1}|\mathcal{S}^k|\cdot(2^{-0.95k\log m}\cdot|X|)\\
&< 2^{0.5s\log m}\cdot 2^{-0.95s\log m}\cdot|X|+\\
&\quad \sum_{k=1}^{s-1}2^{0.5k\log m}\cdot 2^{-0.95k\log m}\cdot|X|\\
&= (\sum_{k=1}^{s}2^{-0.45k\log m})\cdot|X|\\
&\leq (s\cdot 2^{-0.45\log m})\cdot|X|\\
&\leq (s\cdot d^{-3})\cdot|X| = \frac{1}{\omega(d)}|X|
\end{aligned}$$

which is a contradiction of our choice of s.

For invariant (c), we first note the following simple observation about sunflowers.

Fact 7.3. *Let $\mathcal{F}$ and $\mathcal{H}$ be any two set systems such that $\mathcal{H} \subseteq \mathcal{F}$, let $\epsilon, \epsilon' > 0$ be such that $\epsilon \leq \epsilon'$, and let S be any set. Then if $\mathcal{H}_{\bar{S}}$ is an ϵ-robust sunflower, $\mathcal{F}_{\bar{S}}$ is also an ϵ' robust sunflower.*

Consider $S \in \mathcal{S}^k$. Clearly $|S| = k$ by construction, and so we show that $\mathcal{F}(s)_{\bar{S}}$ is an ϵ_t-robust sunflower. We consider only the value of t when S was added to $\mathcal{S}^k$, as ϵ_t only grows, and we do this by induction on t. First observe that for any t, if S was added to $\mathcal{S}^k$ in step 1 then the claim follows immediately since $\mathcal{F}^t \subseteq \mathcal{F}(s)$. This establishes the base case since at $t = 0$ we are at the start of the procedure, and so we consider $t > 0$. We show this with induction on k in reverse order from $s-1$ to 0. If $k = s-1$, since there is no $\mathcal{S}^{k'}$ for $k' > s-1$ it must have been added in step 1, and so again the claim follows immediately. Thus we consider $k < s-1$ and assume S was added in step 4b.

Let $k' > k$ be such that $\mathcal{S}^{k'}_{S}$ was the sunflower discovered in step 4b which made us add S to $\mathcal{S}^k$. We claim that $\mathcal{F}(s)_{\bar{S}}$ is an $(\epsilon_{t-1}+\epsilon_0)$-robust sunflower, which completes the claim since $\epsilon_t = \epsilon_{t-1}+\epsilon_0$. Consider the probability that a random set $y \subseteq [mN] - S$ doesn't contain any set in $\mathcal{F}(s)_{\bar{S}}$. For this to happen, for every set $S' \in \mathcal{S}^{k'}_{\bar{S}}$ either y contains no sets in $\mathcal{F}(s)_{\bar{S'}}$ or it does not contain S' itself. If there is some S' such that $S' \subseteq y$, then by the inductive hypothesis on t and k we know that $\mathcal{F}^{t'}_{S'}$ is an $\epsilon_{t'}$-robust sunflower, where $t' \leq t-1$ was the value of t when S' was added to $\mathcal{S}^k$. Since $\epsilon_{t'} \leq \epsilon_{t-1}$ and $\mathcal{F}^{t'} \subseteq \mathcal{F}(s)$, by extension y avoids every set in $\mathcal{F}(s)_{S'}$ with probability at most ϵ_{t-1}. In the other case where no such S' exists, then because $\mathcal{S}^{k'}_{\bar{S}}$ is an ϵ_0-robust sunflower y avoids every set $S' \in \mathcal{S}^{k'}_{\bar{S}}$ with probability at most ϵ_0. Taking a union bound over these two events gives us our claim.

[13]Here is the second place where we use the gadget size.

Finally for invariant (d), we claim that $t \leq 2^{s^2 \log m} - 1$ when the process ends. Putting this fact together with $\epsilon_0 := 2^{-\Omega(d \log m) - s^2 \log m}$ and $\epsilon_t \leq \epsilon_{t-1} + 2^{-\Omega(d \log m) - s^2 \log m}$ for all t gives us

$$\epsilon_t \leq \epsilon_0 + t \cdot \epsilon_0 \leq 2^{s^2 \log m} \cdot 2^{-\Omega(d \log m) - s^2 \log m} = 2^{-\Omega(d \log m)}$$

We associate each tuple $\mathcal{S} := (\mathcal{S}^k)_{k=1\ldots s-1}$ with the string $\tau(\mathcal{S}) = |\mathcal{S}^1|\#|\mathcal{S}^2|\#\ldots\#|\mathcal{S}^{s-1}|$. We claim that for every t there is a unique string τ_t corresponding to $\tau(\mathcal{S})$ at the time t was incremented. This is simply because in every round of the outer loop we increase the size of at least one set $\mathcal{S}^k$, and in every round of the inner loop that we cause some $\mathcal{S}^k$ to shrink in some round of the inner loop, we also cause some set $\mathcal{S}^{k'}$ to grow where $k' < k$. By invariant (a) and the inner loop condition, $|\mathcal{S}^k| \leq 2^{0.5k \log m}$ for every k whenever we updated t, and so as long as $|\mathcal{S}^0| = 0$—in other words for all t except the very last one—we have

$$t \leq |\tau(\mathcal{S})| = \prod_{k=1}^{s-1} 2^{0.5k \log m} = (2^{0.5 \log m})^{\sum_{k=1}^{s-1} k} < 2^{s^2 \log m} - 2$$

and so at the end of the procedure $t \leq 2^{s^2 \log m} - 1$. $\square$

7.2 Better gadget size

By balancing the parameters in Lemma 7.2 we can prove our scaling lifting theorem.

Lemma 7.4. *Let $N \leq n$, let $d = o(n)$, and let $m > d^{5+\epsilon}$ for any constant $\epsilon > 0$. Let X and δ be such that $\boldsymbol{X}$ has blockwise min-entropy $(1-\delta)\log m$, and let $\mathcal{F} = \{\gamma_j\}_j$ be a block-respecting set system over $[m]^N$ such that 1) for all $x \in X$ there exists a $\gamma_j \in \mathcal{F}$ consistent with x, and 2) $|\gamma_j| \leq O(d)$ for all j. Then*

$$\mathbf{Pr}_{\boldsymbol{y} \subseteq [mN]}(\forall j : \gamma_j \not\subseteq y) < 2^{-\Omega(d \log m)}$$

Proof. As usual we set our blockwise min-entropy threshold at $(1-\delta)\log m$, and furthermore we set our cutoff for the size of the sets $\mathcal{S}^k$ at $m^{(1-\delta')k}$. There are two conditions that need to be fulfilled: 1) to show we can apply the sunflower lemma to $\mathcal{F}^t$ and $\mathcal{S}^k$ we need $m^{1-\delta'} = \Omega(d^3 \log m)$; 2) to bound $|X(s)|$ we need $m^{\delta-\delta'} < \omega(d^{-2})$. Clearly our first step is to set δ to be smaller than an arbitrarily small constant ϵ'—recall that this does not affect the asymptotic strength of our lifting theorem—and for simplicity we replace $\log m$ with $d^{\epsilon'}$ in the first condition, and so we get $m^{1-\delta'} = d^{3+\epsilon'}$ and $m^{\delta'-\epsilon'} = d^2$. Setting $\delta' = 0.4$ gives us $m = d^{5+\epsilon}$ for some constant $\epsilon = O(\epsilon')$, and since we make ϵ' arbitrarily small we can do the same for ϵ. $\square$

8 Open problems

Towards poly(log) **gadget size.** As discussed in the paper, one of the core issues in improving gadget size with current techniques is to prove the extractor or disperser like analogues of Lemma 4.1 for small gadget sizes. To this end, we pose the following concrete conjecture:

Conjecture 1. 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 [GLM+16]). Our work proves the conjecture where $m = O(N \log N)$, whereas previous techniques needed $m \gg N^2$. The robust-sunflower theorem of [ALWZ20] can be seen as proving a related statement: For gadget-size $m = \mathrm{poly}(\log N)$, 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 $z_I = \alpha_I$, $\mathbf{Pr}_Y[\exists x \in X, \textsc{Ind}_m^N(X, Y) = z] \approx 1$. We believe that these arguments could be useful in proving the above conjecture when the gadget-size is $m = \mathrm{poly}(\log N)$.

Acknowledgements

The authors thank Paul Beame for comments.

A A Rosetta Stone for lifting and sunflower terminology

In this appendix we draw relations between concepts in lifting theorems and in sunflowers lemmas.

Spreadness and min-entropy. One way of understanding the jump in [GPW17] from single coordinates to blocks of coordinates is as a movement to a "higher moment method", similar to the one used in [ALWZ20] to improve the parameters in the classic Sunflower Lemma. For this we will be focusing on universes $\mathcal{U}^N$ split into N blocks. A set system $\mathcal{F}$ over $\mathcal{U}$ is r*-spread* if $|\mathcal{F}_{\bar{S}}| \le |\mathcal{F}|/r^{|S|}$ for every $\emptyset \ne S \subseteq \mathcal{U}$. Recall that the blockwise min-entropy of a set system $\mathcal{F}$ over $\mathcal{U}$ is $\min_{\emptyset \ne I \subseteq [N]} \frac{1}{|I|} \mathbf{H}_\infty(F_I)$. We will draw the following equivalence:

$$\mathcal{F} \text{ is } r-\text{spread} \Leftrightarrow \mathcal{F} \text{ has blockwise min-entropy } \log r$$

$\Rightarrow$: consider a set $I \subseteq [N]$ and a block-respecting subset S containing elements exactly from the blocks I. Since $|\mathcal{F}_{\bar{S}}| \le |\mathcal{F}|/r^{|S|}$ for every $\emptyset \ne S \subseteq \mathcal{U}$, it follows that $\mathbf{Pr}_{\gamma \sim \mathcal{F}}[S \subseteq \gamma] \le r^{-|S|}$. Applying this to every such S gives us $\mathbf{H}_\infty(F_I) \ge \frac{1}{|I|} \log r$, and taking the minimum over all I completes the proof.

$\Leftarrow$: for every set $\emptyset \ne S \subseteq \mathcal{U}$ we have two cases: either S is also block respecting or it is not. In the former case then by blockwise min-entropy we have $|\mathcal{F}_{\bar{S}}| \le |\mathcal{F}| \cdot 2^{-|S| \log r}$. In the latter case, note that $|\mathcal{F}_{\bar{S}}| = 0$ since every set $\gamma \in \mathcal{F}$ is block respecting and therefore S cannot be a subset of γ—in other words the blockwise-respecting nature of $\mathcal{F}$ is irrelevant, as claimed in Section 2. Either way S meets the spreadness condition.

We put a last note in, which is that we derived Blockwise Robust Sunflower Lemma from Lemma 4 of [Rao19] using this equivalence. There there is an additional condition that $|\mathcal{F}| \ge r^N$; if the blockwise min-entropy of $\mathcal{F}$ is $\log r$ then this follows by averaging, just considering all sets S of size N.

Disperser property and full range. In [GPW17] there was an improvement to Lemma 4.5 which showed that $\textsc{Ind}_m^N(\boldsymbol{x}, \boldsymbol{y})$ is multiplicatively close to uniform given blockwise min-entropy and largeness. In combinatorics, this uniformity is what is often called an *extractor property*. By contrast a *disperser property* is one that only ensures that $\textsc{Ind}_m^N(X,Y)$ has full range, similar to our key Full Range Lemma. While this coarseness means we cannot achieve a lifting theorem for BPP, it was the key to applying our sunflower techniques, and ultimately necessary for going to a quasilinear size gadget.

Covers and the rectangle partition. For two set systems $\mathcal{F}$ and X over $\mathcal{U}$, $\mathcal{F}$ is a *cover* of X if for every $x \in X$ there exists a $\gamma \in \mathcal{F}$ such that $\gamma \subseteq x$. Furthermore $\mathcal{F}$ is an *r-tight cover* if for every $\gamma \in \mathcal{F}$, $|X_\gamma| > k^{-|\gamma|}|X|$. Going by our translation from spreadness to blockwise min-entropy, it is clear that Rectangle Partition is designed to find a tight cover of X, since γ corresponds to an assignment that is too likely in X, and in each resulting part X^j we want to ensure that blockwise min-entropy is restored. There is a bit more work involved with turning a tight cover into a rectangle partition, but the principle is exactly the same.

Link of $\mathcal{F}$ at S. The *link* of $\mathcal{F}$ at S is $\{\gamma \setminus S : \gamma \in \mathcal{F}, S \subseteq \gamma\}$. While a comparatively minor point, it is worth pointing out that we use the terminology $\mathcal{F}_{\overline{S}}$ to refer to the link, while in sunflower papers this is often written as $\mathcal{F}_S$. We did so to keep consistency with the rest of our set notation.

References

[ALWZ20] Ryan Alweiss, Shachar Lovett, Kewen Wu, and Jiapeng Zhang. Improved bounds for the sunflower lemma. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *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. doi:10.1145/3357713.3384234.

[CFK+19] 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/.

[CKLM17] Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation theorems via pseudorandom properties. *CoRR*, abs/1704.06807, 2017. URL: http://arxiv.org/abs/1704.06807, arXiv:1704.06807.

[CLRS16] 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. doi:10.1145/2811255.

[dRMN+19] 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/.

[dRNV16] 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. doi:10.1109/FOCS.2016.40.

[ER60] Paul Erdös and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35(1):85–90, 1960.

[GGKS18] 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. doi:10.1145/3188745.3188838.

[GJW18] Mika Göös, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM J. Comput.*, 47(1):241–269, 2018. doi:10.1137/16M109884X.

[GKMP20] Mika Göös, Sajin Koroth, Ian Mertz, and Toniann Pitassi. Automating cutting planes is np-hard. In *Proceedings of the 52nd Symposium on Theory of Computing (STOC)*, 2020.

[GLM+16] 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.

[Göö15] Mika Göös. Lower bounds for clique vs. independent set. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1066–1076. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.69.

[GP18] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM Journal on Computing*, 47(5):1778–1806, 2018. doi:doi.org/10.1137/16M1082007.

[GPW15] 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. doi:10.1109/FOCS.2015.70.

[GPW17] 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. doi:10.1109/FOCS.2017.21.

[HN12] 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. doi:10.1145/2213977.2214000.

[HR00] Danny Harnik and Ran Raz. Higher lower bounds on monotone size. In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 378–387. ACM, 2000. doi:10.1145/335305.335349.

[Juk12] Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

[KMR17] Pravesh K. Kothari, Raghu Meka, and Prasad Raghavendra. Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of csps. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *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. doi:10.1145/3055399.3055438.

[Kra98] Jan Krajíček. Interpolation by a game. *Mathematical Logic Quarterly*, 44:450–458, 1998. doi:10.1002/malq.19980440403.

[Kus97] Eyal Kushilevitz. Communication complexity. In *Advances in Computers*, volume 44, pages 331–360. Elsevier, 1997.

[LLZ18] Xin Li, Shachar Lovett, and Jiapeng Zhang. Sunflowers and quasi-sunflowers from randomness extractors. In Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer, editors, *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. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.51.

[LRS15] 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. doi:10.1145/2746539.2746599.

[PR17] Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *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. doi:10.1145/3055399.3055478.

[PR18] Toniann Pitassi and Robert Robere. Lifting nullstellensatz to monotone span programs over any field. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *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. doi:10.1145/3188745.3188914.

[Pud10] Pavel Pudlák. On extracting computations from propositional proofs (a survey). *Leibniz International Proceedings in Informatics, LIPIcs*, 8:30–41, 01 2010.

[Rao19] Anup Rao. Coding for sunflowers. *CoRR*, abs/1909.04774, 2019.

[Raz95] A A Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya: Mathematics*, 59(1):205–227, feb 1995.

[RM99] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999. doi:10.1007/s004930050062.

[Ros10] Benjamin Rossman. Approximate sunflowers. *Manuscript*, 2010.

[RPRC16] Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen A. Cook. Exponential lower bounds for monotone span programs. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 406–415. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.51.

[She11] Alexander A. Sherstov. The pattern matrix method. *SIAM J. Comput.*, 40(6):1969–2000, 2011. doi:10.1137/080733644.

[She14] Alexander A Sherstov. Communication lower bounds using directional derivatives. *Journal of the ACM (JACM)*, 61(6):1–71, 2014.

[Sok17] Dmitry Sokolov. Dag-like communication and its applications. In *Proceedings of the 12th Computer Science Symposium in Russia (CSR)*, pages 294–307. Springer, 2017. doi:10.1007/978-3-319-58747-9_26.

[WYY17] 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. URL: https://eccc.weizmann.ac.il/report/2017/010.