


# Lifting Dichotomies

Yaroslav Alekseev  

Technion – Israel Institute of Technology, Haifa, Israel

Yuval Filmus  

Technion – Israel Institute of Technology, Haifa, Israel

Alexander Smal  

Technion – Israel Institute of Technology, Haifa, Israel

---

## Abstract

Lifting theorems are used for transferring lower bounds between Boolean function complexity measures. Given a lower bound on a complexity measure  $A$  for some function  $f$ , we compose  $f$  with a carefully chosen *gadget* function  $g$  and get essentially the same lower bound on a complexity measure  $B$  for the *lifted* function  $f \diamond g$ . Lifting theorems have a number of applications in many different areas such as circuit complexity, communication complexity, proof complexity, etc. One of the main questions in the context of lifting is how to choose a suitable gadget  $g$ . Generally, to get better results, i.e., to minimize the losses when transferring lower bounds, we need the gadget to be of a constant size (number of inputs). Unfortunately, in many settings we know lifting results only for gadgets of size that grows with the size of  $f$ , and it is unclear whether it can be improved to a constant size gadget. This motivates us to identify the properties of gadgets that make lifting possible.

In this paper, we systematically study the question “For which gadgets does the lifting result hold?” in the following four settings: lifting from decision tree depth to decision tree size, lifting from conjunction DAG width to conjunction DAG size, lifting from decision tree depth to parity decision tree depth and size, and lifting from block sensitivity to deterministic and randomized communication complexities. In all the cases, we prove the complete classification of gadgets by exposing the properties of gadgets that make lifting results hold. The structure of the results shows that there is no intermediate cases – for every gadget there is either a polynomial lifting or no lifting at all. As a byproduct of our studies, we prove the log-rank conjecture for the class of functions that can be represented as  $f \diamond \text{OR} \diamond \text{XOR}$  for some function  $f$ .

In this extended abstract, the proofs are omitted. Full proofs are given in the full version [2].

**2012 ACM Subject Classification** Theory of computation → Communication complexity; Theory of computation → Oracles and decision trees

**Keywords and phrases** decision trees, log-rank conjecture, lifting, parity decision trees

**Digital Object Identifier** 10.4230/LIPIcs.CCC.2024.9

**Related Version** *Full Version:* <https://eccc.weizmann.ac.il/report/2024/037/> [2]

**Funding** This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 802020-ERC-HARMONIC. Alexander Smal has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No 852870-ERC-SUBMODULAR.

## 1 Introduction

For  $f: \{0, 1\}^n \rightarrow V$  and  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ , a (*block-*)*composition*  $f \diamond g: \{0, 1\}^{n \times m} \rightarrow V$  is defined by

$$(f \diamond g)(z_1, z_2, \dots, z_n) := f(g(z_1), g(z_2), \dots, g(z_n)),$$

where each  $z_i \in \{0, 1\}^m$ . Usually *lifting theorems* have the following general form:

$$B(f \diamond g) = \Omega(A(f)),$$



© Yaroslav Alekseev, Yuval Filmus, and Alexander Smal;  
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 9; pp. 9:1–9:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



where  $A$  and  $B$  are two complexity measures. Note that the hidden constant in  $\Omega(\cdot)$  may depend on  $g$ . In this context, we call the function  $g$  a *gadget* and say that *there is a (linear) lifting from  $A$  to  $B$* . One of the first examples of a lifting theorem appeared in [19] where Raz and McKenzie proved a separation for the hierarchy of monotone circuit complexity classes within NC using a query-to-communication lifting theorem.

The need for such theorems is due to the fact that, in many cases, communication complexity lower bounds are much more difficult to prove compared to query complexity lower bounds. Lifting theorems proved to be useful in many other scenarios: proving communication complexity separations [10, 11, 12, 5], proof complexity separations [9, 14, 10, 7], monotone circuit complexity separations [19, 8], etc.

While the lifting technique is widely used in different areas we still do not quite understand its limitations. In the query-to-communication lifting theorems, we want to lower bound the deterministic/randomized communication complexity of the lifted function by deterministic/randomized query complexity of the original function. So, these theorems usually have the following form:

$$D(f \diamond g) = DT(f) \cdot \Theta(\log n),$$

where the gadget  $g$  has at most logarithmic communication complexity. In all the lifting theorems of this type, the *size* (the number of inputs) of the gadget  $g$  grows with the number of inputs of  $f$  (e.g., in [17] the size of the gadget is  $n^{1+\epsilon}$ , where  $n$  is a number of inputs of the lifted function). Even though some results do not depend on the gadget size (e.g., [11]), in many scenarios the use of non-constant size gadgets leads to weaker results in the applications (e.g., when lifting theorems are used to prove monotone circuit lower bounds via communication complexity). It is unknown whether it is possible to prove a query-to-communication lifting theorem with a constant size gadget, but we tend to believe that such lifting exists.

At the same time, for other complexity measures there are lifting theorems that can accommodate constant size gadgets. Here are some examples of such lifting theorems:

1. Lifting decision tree depth to decision tree size (XOR gadget [21]).
2. Lifting decision tree depth to parity decision tree depth/size (stifling gadgets [6]; INDEX gadget [3]).
3. Lifting from critical block sensitivity to communication complexity (VER gadget [10]).
4. Lifting parity decision tree depth to communication complexity (XOR gadget [13]).
5. Lifting AND decision tree depth to communication complexity (AND gadget [15]).
6. Lifting block sensitivity to randomized communication complexity (AND gadget [22]).

Note that in examples 4 and 5, the communication complexity is lower bounded by *some power* of the parity decision tree complexity or the AND decision tree complexity (in the latter case, with an additional  $\log n$  factor). In such cases, we say that there is a *polynomial lifting*.

As mentioned earlier, lifting theorems which lower bound communication complexity by a *linear* function of query complexity are only known for non-constant size gadgets. But in case of a polynomial lifting, there are query to communication complexity lifting theorems with constant size gadgets. E.g., in example 6 the author lifts block sensitivity to randomized communication complexity with a constant size gadget. Given that block sensitivity is polynomially related to query complexity, this gives a polynomial lifting from query to communication complexity.

All the examples of the lifting theorems with constant size gadgets that we mentioned above use specific and simple gadgets. But what happens if we plug some other gadget? The same proof might not work, but would the lifting result still be true? That is not always clear. And this reflects our lack of understanding of what properties of the gadget make lifting possible. For these theorems it is natural to pose the following question:

*For which gadgets does the lifting result hold?*

A systematical study of this question will help us to better understand how lifting works and what are the requirements for the gadget. Especially it would be interesting to understand for which gadgets lifting fails. This is what we need, for example, if we want to find a good candidate for query-to-communication linear lifting with constant size gadget.

One of the areas that might benefit from new lifting theorem is the study of *the log-rank conjecture* [18]. The log-rank conjecture states that for any function  $f$  the deterministic communication complexity of a function is polynomially related to the logarithm of the real rank of its associated communication matrix. Currently there is an exponential gap between the lower bound  $\Omega(\log^2(\text{rank}(f)))$  [11] and the upper bound  $O(\sqrt{\text{rank}(f)} \cdot \log \text{rank}(f))$  [16]. It seems that the general case of this problem is out of reach now. So most of research in this area has concentrated on the study of various complexity measures and special cases such as composed functions (e.g., see [15]). Therefore, lifting in this area is one of the main proof techniques.

The other area craving for new lifting theorems is proof complexity. There is a tight connection between proof complexity and lifting theorems. For example, lifting decision tree depth to parity decision tree depth/size from [6, 3] provides a systematic way to prove tree-like  $\text{Res}(\oplus)$  size lower bounds. It is important to mention that for proof complexity we usually need lifting theorems that hold for relations.

Finally, we think that a large number of different applications makes lifting a technique that is worth exploring on its own. A deeper understanding of how lifting works and what it requires from gadgets can lead to new applications and results.

## 1.1 Our results and methods

In this paper, we systematically explore the question “*For which gadgets does the lifting result hold?*” in several different settings. We prove complete classifications of gadgets in the following four settings:

- lifting from decision tree depth to decision tree size,
- lifting from certificate complexity to conjunction DAG size,
- lifting from decision tree depth to parity decision tree depth and size,
- lifting from block sensitivity to deterministic and randomized communication complexities.

All these results are formulated in the form of dichotomies (a trichotomy in one of the cases) that essentially state that there is either polynomial lifting or no lifting at all (no intermediate cases). As a byproduct of our studies, we prove the log-rank conjecture for the class of functions that can be represented as  $f \diamond \text{OR} \diamond \text{XOR}$  for some function  $f$ .

Now we describe the results in more detail and give an overview of the proof methods. The proofs are omitted from this extended abstract. Full proofs are given in the full version [2].

### 1.1.1 Decision tree depth to size

In Section 3, we define a class of *resistant* gadgets (defined in Section 3.1) and prove a decision tree depth to decision tree size lifting theorem for this class of gadgets (see Theorem 2). We give two different proofs illustrating the ideas of two different approaches: proof by simulation

and proof using random projections. The proofs later in the paper refer to the proofs in this simple case. The proof by simulation is constructive – it shows how given a decision tree for the lifted function  $f \diamond g$  one can construct a decision tree for the original function  $f$ , such that the depth of the new tree is bounded by a logarithm of the size of the given tree. In the proof using random projections, we use probabilistic method to show that there is a projection that converts the decision tree for  $f \diamond g$  into a shallow decision tree for  $f$ .

Our goal is to prove a classification theorem, so we need to find a class of gadgets such that lifting works only for gadgets in this class. It appears that the class of resistant gadgets is too small for this. We define a wider class of *weakly resistant* gadgets (defined in Section 3.2) and prove a certificate complexity to decision tree size lifting theorem (see Theorem 4). The proof uses the random projections method. Note that the decision tree size is upper bounded by the certificate complexity squared, so as a corollary we get a polynomial lifting from decision tree depth to decision tree size (see Corollary 5).

Finally, we give a complete classification of gadget functions in the context of polynomial lifting from decision tree depth to decision tree size. It is stated as a dichotomy result (see Theorem 7): either a gadget is weakly resistant and there is a polynomial lifting or there is no lifting at all.

In Section 3.4, we briefly discuss that some of the results above can be generalized to the case of search problems. Unfortunately, that does not give a classification theorem because decision tree depth of a search problem can be exponentially greater than its certificate complexity. However, we state a conjecture there is a decision tree depth to decision tree size polynomial lifting for weakly resistant gadgets.

### 1.1.2 Conjunction DAG width to size

In Section 4, we generalize the results from the previous section to decision conjunction DAGs (defined in Section 4.1). First of all, in Section 4.2, we show that similarly to decision trees where depth and size are exponentially separated, there is an exponential separation between conjunction DAG width and size. The separation is achieved for the Tribes function. In Section 4.3, we show that there is an exponential separation between decision tree size and conjunction DAG size in the case of relations. Indeed, conjunction DAGs can capture the structure of Resolution proofs, while decision trees can only capture the structure of tree-like Resolution proofs. Thus, the separation between these proof systems implies the separation between the measures under consideration. Finally, we argue that the lifting theorems from Section 3 can be generalized to the case of conjunction DAGs (see Theorem 8 and Theorem 9) using essentially the same proofs. Thus, we have a dichotomy result (see Theorem 10).

### 1.1.3 Decision tree depth to parity decision tree depth and size

Recently, Chattopadhyay et al. [6] showed that if  $g$  is *stifling* (defined in Section 5.1) then  $\log \text{DTSize}_{\oplus}(f \diamond g) = \Theta(\text{DT}(f))$ . In Section 5, we extend their result providing the complete classification of the gadgets for decision tree depth to parity decision tree depth and size lifting. In Section 5.2, we state and prove a “minimum weight lemma” (see Lemma 13), the technical lemma that we will use several times. This lemma states that if the certificate complexity of some function  $f$  is large enough in comparison to the parity certificate complexity of the lifted function  $f \diamond g$  at some input, then there is a substitution such that the minimal parity certificate of  $f \diamond g$  at this input has a large Hamming weight.

In Section 5.3, we consider the most challenging case of this setting – the case of OR gadget. In Theorem 14, we show that there is at least a quadratic gap in the certificate to parity certificate complexity lifting for OR gadget. If we assume that the lifting result

in [6] holds for OR gadget as well then the quadratic separation is tight (see Proposition 15). In Section 5.3.2, we show a polynomial (cubic) lifting for OR gadget (see Theorem 16). The proof consists of three ingredients. First, we show that if the minimum weight of a parity certificate for the lifted function is at least the size of the parity certificate then this certificate covers the all-1 input for the inner function. Then we use it to upper bound the minimum weight of a parity certificate in terms of the sizes of parity certificates for 0 and 1. And finally, we apply the “minimum weight lemma”. In the proof of Theorem 16, we assume that the certificate complexity of the original function is much larger than the parity certificate complexity of the lifted function, but due to the “minimum weight lemma” it would contradict the upper bound on the minimum weight of the parity certificate.

In Section 5.4, we prove a lifting from certificate complexity to parity decision tree size for AND/OR gadgets, the gadgets that affine project to both binary AND and binary OR (see Theorem 17). The proof is by simulation similar to the simulation proof in [6].

Finally, in Section 5.5, we prove the trichotomy result that gives us a complete classification of gadgets (see Theorem 19). The classification is based on the fact that if a gadget is not AND/OR then it is a disjunction or a conjunction of affine forms (see Lemma 18). We show that there are only three cases possible: (1) there are simultaneously a lifting from decision tree depth to parity decision tree depth and a lifting from certificate complexity to parity certificate complexity (the case of AND/OR gadgets); (2) there is a lifting from decision tree depth to parity decision tree depth but no lifting between certificate complexities (the case of gadgets that affine project to OR); (3) there is no lifting (all other gadgets, constant or affine).

In Section 5.6, we show an alternative proof for the lifting from decision tree depth to parity decision tree depth using function degree and sparsity (see Theorem 20). In the proof we show that the degree of a function is upper bounded by the logarithm of the sparsity of the function lifted with OR gadget, and compose it with a number of previously known inequalities. As a byproduct, we get a proof of the log-rank conjecture for the class of functions that can be represented as  $f \diamond \text{OR} \diamond \text{XOR}$  (see Theorem 21).

#### 1.1.4 Block sensitivity to communication complexity

In Section 6, we provide a complete classification of gadgets for lifting from block sensitivity to deterministic and randomized communication complexities. Note that this classification also gives us a classification of gadgets for query-to-communication polynomial lifting since block sensitivity and query complexity are polynomially related to each other in the case of total functions. However, since the proof goes through block sensitivity, we classify block sensitivity to communication complexity lifting.

We start with the lifting theorem of Zhang [22] (see Theorem 23) that works for gadget  $g$  iff both AND and OR reduce to  $g$  via a communication complexity reduction (defined in Section 6.1). Then we show that if there is no reduction from OR to a gadget  $g$  then the communication matrix of  $g$  is (up to rearrangement) block diagonal (see Lemma 24). This gives us a classification of gadgets in the context of reductions from OR, AND, and XOR (see Corollary 26 and Corollary 27).

In Section 6.2, we prove a dichotomy result for randomized communication complexity (see Theorem 28). The proof is based on the fact that if OR does not reduce to a gadget  $g$  then  $\text{AND}_n \diamond g$  is essentially an instance of the equality function, and hence its randomized communication complexity is logarithmic. If AND does not reduce to a gadget, the situation is symmetrical (see Lemma 29). Thus, the theorem of Zhang covers all gadgets for which there is a lifting.

In Section 6.3, we prove the dichotomy result for deterministic communication complexity (see Theorem 31). The proof of the dichotomy depends on a block sensitivity to deterministic communication complexity lifting theorem that works for gadget  $g$  iff both AND and XOR reduce to  $g$  (see Theorem 30). Together with the Zhang’s lifting theorem that give us a complete classification of gadgets: if at least two of three functions AND, OR, XOR reduce to gadget  $g$  then there is a block sensitivity to deterministic communication complexity polynomial lifting. Otherwise, if at least two functions from this list do not reduce to  $g$  then  $g$  is either a constant function or (essentially) one of the functions AND, OR, XOR. The proof of the lifting theorem requires a number of ingredients. First of all, we compose two results from [13, 23] to get a parity certificate to deterministic communication complexity polynomial lifting for XOR gadget. Then we show two lower bounds for specific cases: we show that the deterministic communication complexity of  $f \diamond \text{XOR}$  is lower bounded by the size of any minimal sensitive block of the function  $f$ , and that the deterministic communication complexity of  $f \diamond \text{AND}$  is lower bounded by the block sensitivity of  $f$  at the all-0 input. One of the ingredients is again the “minimum weight lemma”. And the last one, is the lemma that shows that parity certificates of high minimum weight always intersect with (regular) certificates of sufficiently small size. Putting all the ingredients together we get the desired lifting result.

## 2 Prerequisites

### 2.1 Notation

All the logarithms are base 2. We use  $\text{OR}_n$ ,  $\text{AND}_n$ ,  $\text{XOR}_n$  to denote, respectively, logical “and”, “or” and “exclusive or” of  $n$  Boolean inputs. For simplicity we also define  $\text{OR} := \text{OR}_2$ ,  $\text{AND} := \text{AND}_2$ , and  $\text{XOR} := \text{XOR}_2$ . We use the notation  $x^B$  for  $x \in \{0, 1\}^n$  and  $B \subseteq [n]$  to denote  $x$  with all the coordinates in  $B$  flipped, i.e.,  $x_i = x_i^B \iff i \notin B$ . For  $g: \{0, 1\}^k \rightarrow \{0, 1\}$ , we define<sup>1</sup> a function  $g^n: \{0, 1\}^{n \times k} \rightarrow \{0, 1\}^n$  such that

$$g^n(x_1, x_2, \dots, x_n) := (g(x_1), g(x_2), \dots, g(x_n)).$$

For a function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , an input  $x \in \{0, 1\}^n$ , and a partial assignment  $\sigma \in \{0, 1, *\}^n$  that agrees with  $x$  on all non- $*$  coordinates, we use  $f|_\sigma$  to denote a restriction of  $f$  to  $\sigma$  and  $x|_\sigma$  to denote a projection of  $x$  to the  $*$ -coordinates of  $\sigma$ . For a gadget  $g: \{0, 1\}^k \rightarrow \{0, 1\}$  and a *blockwise* partial assignment  $\sigma \in (\{0, 1\}^k \cup \{*\}^k)^n$  (i.e., in any block of variables that corresponds to one copy of  $g$ , the variables are either all set or all stars), we use  $g^n(\sigma)$  to denote the partial assignment in  $\{0, 1, *\}^n$  induced by applying  $g$  to non- $*$  blocks of  $\sigma$ . We use “ $\sqcup$ ” instead of “ $\cup$ ” to indicate a union of disjoint sets.

### 2.2 Complexity measures

Throughout the text, we will consider several complexity measures.

#### Decision tree complexity

A *decision tree* is a rooted binary tree with internal nodes labeled by input variables (represent queries), edges labeled with 0 and 1, and leaves labeled by some values. Decision tree evaluation is defined in the natural way: given an assignment for the input variables,

<sup>1</sup> In some papers lifting theorems are formulated for the classical composition operation “ $\circ$ ” rather than for the block-composition “ $\diamond$ ”. Note that for  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and  $g: \{0, 1\}^k \rightarrow \{0, 1\}$ ,  $f \diamond g \equiv f \circ g^n$ .

we traverse the tree starting from the root and then sequentially choose every next edge according to the given assignments until we reach some leaf. The label of this leaf is the result of the evaluation. A decision tree computes some function  $f$  if for all possible assignments to the input variables the decision tree evaluates to the value of the function. For a function  $f$ ,  $\text{DT}(f)$  denotes the minimal depth of a decision tree computing  $f$ , and  $\text{DTSize}(f)$  denotes the minimal number of leaves in a decision tree computing  $f$ .

### Parity decision tree complexity

A *parity decision tree* is a generalization of a decision tree where the queries are arbitrary linear combinations of the input variables. It can be described as a rooted binary tree with internal nodes labeled by linear combinations of the input variables, edges labeled with 0 and 1, and leaves labeled by some values. The evaluation is defined analogously. For a function  $f$ ,  $\text{DT}_{\oplus}(f)$  denotes the minimal depth of a parity decision tree computing  $f$ , and  $\text{DTSize}_{\oplus}(f)$  denotes the minimal number of leaves in a parity decision tree computing  $f$ .

### Certificate complexity

A *certificate complexity* is a non-deterministic analogue of the decision tree complexity. A *certificate* for an input  $x \in \{0, 1\}^n$  to a function  $f$  is a set  $S \subseteq [n]$  of indices such that  $f$  restricted to all inputs that match  $x$  on  $S$  is constant, i.e.,  $f(y) = f(x)$  whenever  $y|_S = x|_S$ . The certificate complexity  $C(f, x)$  of  $f$  at input  $x$  is the size of the smallest certificate for  $x$ . Finally, the certificate complexity of function  $f$  is defined as  $C(f) := \max_{x \in \{0, 1\}^n} C(f, x)$ .

### Parity certificate complexity

A *parity certificate* for an input  $x \in \{0, 1\}^n$  to a function  $f$  is a set  $A$  of linear forms such that  $f$  is constant on the affine subspace defined by  $A = A(x)$ . A *parity certificate complexity*  $C_{\oplus}(f, x)$  of function  $f$  at input  $x$  is the size (number of linear forms) of the smallest parity certificate for  $x$ . The parity certificate complexity of  $f$  is defined as  $C_{\oplus}(f) := \max_{x \in \{0, 1\}^n} C_{\oplus}(f, x)$ .

### Block sensitivity

A block  $B \subseteq [n]$  is *sensitive* for a function  $f$  at  $x$  iff  $f(x) \neq f(x^B)$ . The *block sensitivity*  $\text{bs}(f, x)$  of  $f$  at  $x$  is the maximum number of disjoint sensitive blocks for  $f$  at  $x$ . The block sensitivity  $\text{bs}(f)$  of  $f$  is the maximum sensitivity  $\text{bs}(f, x)$  over all points  $x \in \{0, 1\}^n$ .

### Degree and sparsity of the function

Every Boolean function defined on  $\{0, 1\}^n \rightarrow \{0, 1\}$  can be considered as a function  $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$  by substituting 0 and 1 with 1 and  $-1$ , respectively. Every such function has a unique representation as a multilinear polynomial over  $\mathbb{R}$  of the following form

$$f(x) = \sum_{S \subseteq [n]} c_S x_S, \quad \text{where } x_S = \prod_{i \in S} x_i.$$

*Degree* of  $f$  is defined to be the degree of the corresponding polynomial, we denote it by  $\text{deg}(f)$ . *Sparsity* of  $f$  is the number of nonzero coefficients in this representation, we denote it by  $\text{spar}(f)$ .



### Deterministic communication complexity and rank

For a function  $f: X \times Y \rightarrow Z$ , let's consider the following game for two players, Alice and Bob, that want to compute  $f$ . Alice and Bob are given  $x \in X$  and  $y \in Y$ , respectively. Their goal is to compute  $f(x, y)$ . In order to do it, the players exchange information about their parts of the input using a simple communication channel that allows sending bit messages. Before the game the players come up with a *communication protocol* that determines their behavior on all possible inputs. The cost of a protocol is the maximum total number of bits sent by the players over all possible inputs  $x \in X, y \in Y$ . The *deterministic communication complexity* of  $f$  is the minimal cost of a deterministic communication protocol that computes  $f$ . We denote it by  $D(f)$ .

A *communication matrix* of the function  $f$  is a matrix  $M_f \in Z^{X \times Y}$  such that  $(M_f)_{x,y} := f(x, y)$  for all  $x \in X, y \in Y$ . We define  $\text{rank}(f)$  to be the rank of matrix  $M_f$ .

### Randomized communication complexity

In a *randomized communication game*, Alice and Bob have access to an unlimited amount of random bits and they can use it to decide which bit to send next. We say that a (*private coin*) *randomized communication protocol*  $\Pi$  computes a function  $f: X \times Y \rightarrow Z$  with error  $\varepsilon$  if

$$\Pr_{r_A, r_B} [\Pi(x, y, r_A, r_B) = f(x, y)] \geq 1 - \varepsilon, \quad \forall x \in X, y \in Y,$$

where  $r_A$  and  $r_B$  are the strings of random bits used by Alice and Bob, respectively. The cost of a randomized protocol is the maximum number of bits that can be sent. We denote by  $R_\varepsilon(f)$  the minimal cost of a private coin randomized communication protocol that computes  $f$  with error  $\varepsilon$ . A *randomized communication complexity* of  $f$  is defined as  $R(f) := R_{1/3}(f)$ .

## 3 Decision tree depth to size

We start by exploring perhaps one of the simplest scenarios, depth-to-size lifting in decision trees and state a lifting theorem for the class of *resistant* gadgets.

### 3.1 Resistant gadgets

Urquhart [21] proved that for any function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ ,

$$\log \text{DTSize}(f \diamond \text{XOR}) = \Omega(\text{DT}(f)).$$

We generalize this result for the class of *resistant* gadgets.

► **Definition 1.** A gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  is resistant if for every  $i \in [m]$  and  $b \in \{0, 1\}$ , the function obtained by fixing the  $i$ th input to  $b$  is not constant. Equivalently, the minimum certificate complexity at inputs is larger than 1. E.g., XOR function is resistant while AND function is not.

► **Theorem 2.** For any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and a resistant gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ ,

$$\log \text{DTSize}(f \diamond g) = \Omega(\text{DT}(f)).$$



### 3.2 Weakly resistant gadgets

There are gadgets, such as  $x \vee (y \wedge z)$ , which are clearly not resistant, but for which the lifting still holds as we will show below. To capture these cases, we define a more general class of *weakly resistant* gadgets.

► **Definition 3.** A gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  is weakly resistant if for every certificate  $\alpha$  (a partial assignment which sets the value of  $g$ ) there is a partial assignment  $y_j = b$  which conflicts with  $\alpha$  and does not make  $g$  constant.

Every resistant gadget is trivially weakly resistant: we can take any variable mentioned in  $\alpha$  and substitute the opposite value. The aforementioned gadget  $h = x \vee (y \wedge z)$  is weakly resistant (as we show below) but not resistant, since  $h|_{x=1} = 1$ . The gadget  $x \vee y$  is not even weakly resistant due to the certificate  $x = y = 0$ : if we substitute  $x = 1$  or  $y = 1$  then the gadget becomes constant.

Let us verify that  $h$  is weakly resistant, by considering all possible certificates:

- $x = 1$ : take  $x = 0$ .
- $y = z = 1$ : take  $y = 0$  or  $z = 0$ .
- $x = y = 0$ : take  $y = 1$ .
- $x = z = 0$ : take  $z = 1$ .

We prove the following lifting theorem:

► **Theorem 4.** For any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and a weakly resistant gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  the following holds:

$$\log \text{DTSize}(f \diamond g) = \Omega(\text{C}(f)).$$

Decision tree complexity is at most certificate complexity squared, thus we get the following corollary.

► **Corollary 5.** For any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and a weakly resistant gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  the following holds:

$$\log \text{DTSize}(f \diamond g) = \Omega(\sqrt{\text{DT}(f)}).$$

We do not know whether the lower bound is tight, even for  $h = x \vee (y \wedge z)$ . At the moment we can't even rule out  $\log \text{DTSize}(f \diamond h) = \Omega(\text{DT}(f))$ . Sherstov [20, Theorem 6.4] proved that

$$\max(\log \text{rank}(f \diamond \text{AND}), \log \text{rank}(f \diamond \text{OR})) \geq \deg(f).$$

Since rank lower bounds decision tree size, this implies that

$$\log \text{DTSize}(f \diamond g) \geq \deg(f).$$

Note that we are not aware of any separation between  $\text{DT}(f)$  and  $\max(\text{C}(f), \deg(f))$ .

### 3.3 Gadget classification

The dichotomy result of this section is based on the following classification of gadgets.

► **Lemma 6.** For every  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ , one of the following cases holds:

1. Function  $g$  is a (possibly empty) conjunction or a disjunction of literals.
2. Function  $g$  is weakly resistant.

The two cases in following dichotomy theorem correspond to the two cases of Lemma 6.

- **Theorem 7.** *For every gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ , one of the following cases holds:*
1. *There is an infinite family of functions  $f_n$  with  $\text{DT}(f_n) \rightarrow \infty$  and  $\text{DTSize}(f \diamond g) = O(\text{DT}(f))$ .*
  2. *For every function  $f$ , we have  $\log \text{DTSize}(f \diamond g) = \Omega(\text{DT}(f)^{\Omega(1)})$ .*

### 3.4 Generalization to search problems

A relation  $f$  is a subset of  $\{0, 1\}^n \times V$ . For a gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ , we define a composition  $f \diamond g \subseteq (\{0, 1\}^m)^n \times V$  as a relation, such that

$$(z_1, z_2, \dots, z_n, r) \in (f \diamond g) \iff (g(z_1), g(z_2), \dots, g(z_n), r) \in f.$$

One can observe that the proofs of Theorems 2 and 4 work as well when we let  $f$  to be a relation. However, the Corollary 5 does not hold for the relations since  $\text{DT}(f)$  can be exponentially greater than  $C(f)$ . As an example of this, one can take  $f$  to be a *falsified clause problem* corresponding to the Pigeonhole Principle Formula over an expander graph. It is known [4] that resolution width of any refutation for this formula at least  $\Omega(n)$  (which is greater or equal than  $\text{DT}(f)$ ), but the certificate complexity is constant (since each of the clauses of the formula is constant-sized).

However, we conjecture that lifting from decision tree depth to decision tree size can also be proved for weakly resistant gadgets. Equivalently, this will mean that such lifting holds for the gadget  $g(x, y, z) := x \vee (y \wedge z)$ .

## 4 Conjunction DAG width to size

### 4.1 Conjunction DAGs

Conjunction DAGs were first defined formally in [8], though they appear implicitly in previous work. A *conjunction DAG* over a set of variables is a single-rooted DAG with the following additional information:

- Each internal vertex is annotated with a variable, and it has two outgoing edges, one labeled 0 and the other one labeled 1.
- Each vertex  $v$  is annotated with a partial assignment  $\rho(v)$  with the following constraint. Suppose that  $v$  queries  $x_i$ , and the answer  $b$  leads to the vertex  $v_b$ . Then the partial assignment  $\rho(v_b)$  is a subset of the partial assignment  $\rho(v) \cup \{x_i \leftarrow b\}$ . (This is not identical to the definition in [8], but morally the same.)
- The partial assignment at the root is the empty assignment.
- Each leaf is annotated with some value.

A decision DAG *computes*  $f$  if for every leaf  $\ell$  annotated with  $y_\ell$ ,  $\rho(\ell)$  is a  $y_\ell$ -certificate of  $f$ .

Every decision tree is a decision DAG. There are three parameters of interest: the (total) size (number of vertices), the leaf size (number of leaves), and the width (maximum number of variables in any  $\rho(v)$ ).

### 4.2 Size vs width

A decision tree of depth  $d$  contains at most  $2^d$  leaves, and this is tight for the parity function, in the sense that the bound  $\text{DTSize}(f) \leq 2^{\text{DT}(f)}$  cannot be improved when  $f$  is the parity function.

Similarly, a conjunction DAG of width  $d$  contains at most  $\binom{n}{\leq d} = O(n^d)$  vertices, and this is tight for the following function (also known as the *Tribes* function)

$$f(x) = \bigvee_{i=1}^{\sqrt{n}} \bigwedge_{j=1}^{\sqrt{n}} x_{ij}.$$

We can construct a conjunction DAG of width  $O(\sqrt{n})$  for  $f$  as follows. We think of the input as a matrix, where  $i$  is the row number and  $j$  is the column number. We scan each row sequentially. If the current row consists only of 1s, we stop. Otherwise, we add the first 0 to  $\rho$ , and forget all the remaining entries of the row.

Conversely, consider the set of  $\sqrt{n}^{\sqrt{n}}$  inputs having a single 0 per row. No two inputs share the same certificate, and so every conjunction DAG for  $f$  must contain at least  $n^{\sqrt{n}/2}$  leaves.

### 4.3 Separation between decision tree size and conjunction DAG size

Let  $d(f)$  denotes the conjunction DAG width of a function  $f$ , which is the smallest width of a conjunction DAG for  $f$ . Since  $\text{DT}(f) \geq d(f) \geq C(f) = \Omega(\sqrt{\text{DT}(f)})$ , in case of functions conjunction DAG width and decision tree depth are polynomially related. In this section, we show that in case of relations decision tree size and conjunction DAG size can be far apart.

Alekhovich et al. [1] constructed a family of CNF contradictions  $\phi_n$  with  $\text{poly}(n)$  many variables and clauses which have a refutation in Resolution of size  $\text{poly}(n)$ , but such that any refutation in tree-like Resolution (even in regular Resolution) is of size  $2^{\Omega(n)}$ .

We can think of a Resolution proof as a conjunction DAG which solves the *falsified clause problem*: given a truth assignment, find a falsified clause. This is a relation rather than function. Similarly, a tree-like Resolution proof is a decision tree solving the same problem.

Now we are going to show that there is a function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  with a conjunction DAG of size  $\text{poly}(n)$  such that  $\text{DTSize}(f) = 2^{\Omega(n/\log n)}$ . Consider the contradictions  $\phi_n$  mentioned above. Index the clauses of  $\phi_n$  using bitstrings of length  $\ell = O(\log n)$  in some arbitrary way. Now consider a polynomial size Resolution proof of  $\phi_n$ , and let  $f_i$  be the function mapping a truth assignment to the  $i$ -th bit of the bitstring indexing the falsified clause found by the proof. By construction, each  $f_i$  has a conjunction DAG of size  $\text{poly}(n)$ . Given decision trees for  $f_1, \dots, f_\ell$ , we can construct a decision tree solving the falsified clause problem of size  $\prod_i \text{DTSize}(f_i)$ . Since this product must be at least  $2^{\Omega(n)}$ , we conclude that  $\max_i \text{DTSize}(f_i) = 2^{\Omega(n/\log n)}$ .

### 4.4 Gadget classification

For the case of conjunction DAGs, we can generalize Theorem 2:

► **Theorem 8.** *Let  $f \subseteq \{0, 1\} \times V$  be a relation and  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  be a resistant gadget. If there is a conjunction DAG of size  $S$  computing  $f \diamond g$ , then there is a conjunction DAG of width  $O(\log S)$  computing  $f$ .*

Moreover, for conjunction DAGs we can prove an analogue of Theorem 4:

► **Theorem 9.** *Let  $f \subseteq \{0, 1\} \times V$  be a relation and  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  be a weakly resistant gadget. If there is a conjunction DAG for  $f \diamond g$  with  $S$  leaves, then the certificate complexity of  $f$  is at most  $O(\log S)$ .*

Note that Theorem 9 gives us a lifting from certificate complexity to leaf size. Unfortunately, in the case of the falsified clause problem for CNF, this theorem cannot be effectively used to prove lower bounds since leaf size is usually small as well as certificate complexity. However, this theorem still implies a dichotomy result similar to one in Theorem 7.

- **Theorem 10.** *For every gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ , one of the following cases holds:*
1. *There exists an infinite family of functions  $f_n$  such that  $f_n \diamond g$  can be computed with a conjunction DAG having  $O(n)$  leaves, but  $C(f_n) = \Omega(n)$ .*
  2. *For every relation  $f$ , if we have a conjunction DAG for  $f \diamond g$  with  $S$  leaves, then the certificate complexity of  $f$  is at most  $O(\log S)$ .*

## 5 Decision tree depth to parity decision tree depth and size

In this section, we are going to classify gadgets in the context of decision tree depth to parity decision tree depth and size lifting. We start by restating recent result of Chattopadhyay et al. [6]. After that we are going to state the “minimum weight lemma” (Lemma 13) that is a technical tool we will use multiple times throughout this section and the following one. We use this lemma to prove the lifting from certificate complexity to parity certificate complexity with OR gadget, which is the most challenging case in the classification. Finally, we will discuss an alternative and much simpler proof for the lifting from decision tree depth to parity decision tree depth using degree and sparsity. In some sense, this proof should give us a better exponent for the decision tree lifting. The main point of considering certificate complexity lifting is that there is a non-trivial upper bound for OR gadget showing that it is impossible to prove a linear lifting in this setting (see Theorem 14). As a byproduct, we get a proof of the log-rank conjecture for the class of functions that can be represented as  $f \diamond \text{OR} \diamond \text{XOR}$ .

### 5.1 Stifling gadgets

Chattopadhyay et al. [6] defined the following notion.

► **Definition 11.** *A function  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  is  $k$ -stifling if for every set of  $k$  coordinates and  $b \in \{0, 1\}$  there is a way to set the remaining  $m - k$  coordinates so that the output is  $b$  (regardless of the value of the chosen  $k$  coordinates). A function is stifling if it is 1-stifling.*

Chattopadhyay et al. [6] showed that if  $g$  is stifling then  $\log \text{DTSize}_{\oplus}(f \diamond g) = \Theta(\text{DT}(f))$ , where the hidden constant can depend on  $g$ . See the full version of this paper [2] for an exposition of their proof.

### 5.2 Minimum weight lemma

A parity certificate  $C$  is a system of affine equations, so it can be represented by a matrix  $M$  and a vector  $v$  that define a linear subspace.

► **Definition 12.** *A minimum weight of a parity certificate  $C$  is the minimum Hamming weight of non-zero vectors in the row space of  $M$ .*

For example, one can consider a certificate  $\{x + y = 1, x + y + z = 0\}$ . The minimum weight of this certificate is equal to 1 and this corresponds to the equation  $z = 1$ . On the other hand, the minimum weight of a certificate  $\{x + y = 1, x + z = 0\}$  is equal to 2.

The following lemma is a key tool that we will use both in Section 5 and in Section 6. This lemma shows that if the certificate complexity of some function  $f$  is large enough in comparison to the parity certificate complexity of the lifted function  $f \diamond g$  at some input,

then there is a substitution such that the minimal parity certificate of  $f \diamond g$  at this input has large Hamming weight. Informally, this means that we can make a small enough substitution, such that the preimage of the parity certificate after the substitution will contain the points that we are interested in.

► **Lemma 13** (minimal weight lemma). *For functions  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and  $g: \{0, 1\}^m \rightarrow \{0, 1\}$ , suppose that  $f \diamond g$  has a parity certificate complexity at most  $k$  at some point  $x$ . Let  $K$  be a parameter. If certificate complexity of  $f$  at  $g^n(x)$  is greater than  $k \cdot K$  then we can find a blockwise partial assignment  $\sigma$  to the inputs of  $f \diamond g$  consistent with  $x$ , such that for some  $k' \leq k$ :*

- $(f \diamond g)|_\sigma$  has a parity certificate of size  $k'$  at  $x|_\sigma$  whose minimum weight is at least  $K$ ,
- $C(f|_{g^n(\sigma)}, g^n(x)|_{g^n(\sigma)}) > k'K$ .

## 5.3 OR gadget

### 5.3.1 Separation

Chattopadhyay et al. [6] showed that  $C_\oplus(f \diamond g) = \Theta(C(f))$  whenever  $g$  is stifling. This no longer holds when  $g$  is binary OR. The following theorem shows that there is at least a quadratic gap for OR gadget in the certificate to parity certificate complexity lifting.

► **Theorem 14.** *Let  $n = m^2$ , and let  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  be the function that accepts an  $m \times m$  Boolean matrix iff it has no rows of Hamming weight 1.*

1.  $C(f) = n$ ,
2.  $C_\oplus(f \diamond \text{OR}) \leq 2m$ .

Note that if we assume that the lifting result of [6] holds for the OR gadget as well then the quadratic separation is tight.

► **Proposition 15.** *Suppose that  $\text{DT}_\oplus(f \diamond \text{OR}) = \Theta(\text{DT}(f))$  for all  $f$ . Then every function  $f$  satisfies  $C(f) = O(C_\oplus(f \diamond \text{OR})^2)$ .*

### 5.3.2 Lifting

Given Lemma 13 we prove the following lifting theorem for the certificate complexity.

► **Theorem 16.** *For every function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , we have  $C(f) = O(C_\oplus(f \diamond \text{OR})^3)$ .*

This theorem leads us to the following natural question: can we improve this bound from cubic to quadratic, so the bound matches the separation provided by Proposition 15?

## 5.4 AND/OR gadgets

A function  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  affine projects to a function  $h: \{0, 1\}^p \rightarrow \{0, 1\}$  if there are affine functions  $\ell_1, \dots, \ell_m: \mathbb{Z}_2^p \rightarrow \mathbb{Z}_2$  such that

$$g(\ell_1(z), \dots, \ell_m(z)) = h(z).$$

A gadget  $g: \{0, 1\}^m \rightarrow \{0, 1\}$  is AND/OR if it affine projects to both binary AND and binary OR. Here are some examples of AND/OR gadgets:

- $g(x, y, z) := x \vee (y \wedge z)$ . The projections:  $g(0, a, b) = a \wedge b$  and  $g(a, b, 1) = a \vee b$ .
- $g(x, y, z) := [x + y + z = 1]$ . The projections:  $g(1, \bar{a}, \bar{b}) = a \wedge b$  and  $g(\bar{a}, \bar{b}, a \oplus \bar{b}) = a \vee b$ .

► **Theorem 17.** *If  $g$  is an AND/OR gadget then for all functions  $f: \{0, 1\}^n \rightarrow \{0, 1\}$*

$$\log \text{DTSize}_\oplus(f \diamond g) \geq C(f).$$

## 5.5 Gadget classification

We use the following lemma to classify gadgets which are not AND/OR.

► **Lemma 18.** *If  $g$  is not an AND/OR then  $g$  is a disjunction or a conjunction of affine forms in the inputs.*

Using these facts we prove the following classification of gadgets.

► **Theorem 19.** *For every gadget  $g$ , one of the following cases holds:*

1. *There is an infinite family of functions  $f_n$  with  $\text{DT}(f_n) \rightarrow \infty$  and  $\text{DTSize}_{\oplus}(f_n \diamond g) = O(1)$ .*
2. *For every function  $f$ ,  $\text{DT}_{\oplus}(f \diamond g) = \Omega(\text{DT}(f)^{\Omega(1)})$ . There is an infinite family of functions  $f_n$  with  $\text{DT}(f_n) \rightarrow \infty$  and  $\text{DTSize}(f_n \diamond g) = O(\text{DT}(f))$ .*
3. *For every function  $f$ ,  $\log \text{DTSize}_{\oplus}(f \diamond g) = \Omega(\text{DT}(f)^{\Omega(1)})$  and  $C_{\oplus}(f \diamond g) = \Omega(C(f)^{\Omega(1)})$ .*

The first case of theorem corresponds to affine or constant gadgets, the second case corresponds to gadgets that affine project to either AND or OR, and the third case corresponds to AND/OR gadgets.

## 5.6 Lifting for OR gadget and the log-rank conjecture

In this section, we discuss a proof of the following inequality:

► **Theorem 20.** *For any function  $f: \{0,1\}^n \rightarrow \{0,1\}$ ,*

$$\text{DT}(f) \leq O\left(\text{DT}_{\oplus}(f \diamond \text{OR})^{O(1)}\right).$$

This inequality is a corollary of Theorem 16 using the fact that  $\text{DT}$  is polynomially related to  $C$ , and  $\text{DT}_{\oplus}$  is polynomially related to  $C_{\oplus}$ . In [2], we present an alternative proof of this statement that gives as a byproduct a proof of the log-rank conjecture for a subclass of Boolean functions that can be represented as  $f \diamond \text{OR} \diamond \text{XOR}$  for arbitrary function  $f$ .

In this section, we need a composition of a function and a gadget (XOR in this case) in the communication complexity context. For a function  $f: \{0,1\}^n \rightarrow \{0,1\}$  we define a function  $f_{\oplus}: \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$ , such that  $f_{\oplus}(x, y) := f(x_1 \oplus y_1, \dots, x_n \oplus y_n)$ , and associate it with the following communication problem: Alice and Bob are given  $x$  and  $y$ , respectively, and their goal is to compute  $f(x, y)$ . For a subclass of such XOR functions, we prove the following version of log-rank conjecture:

► **Theorem 21.** *For any function  $f: \{0,1\}^n \rightarrow \{0,1\}$ ,*

$$D((f \diamond \text{OR})_{\oplus}) \leq \text{poly}(\log \text{rank}((f \diamond \text{OR})_{\oplus})).$$

The proof is due to the following chain of inequalities.

► **Lemma 22.** *For any function  $f: \{0,1\}^n \rightarrow \{0,1\}$ ,*

$$\begin{aligned} \text{DT}(f) &\leq 2 \deg(f)^4 \leq O((\log \text{spar}(f \diamond \text{OR}))^4) = O((\log \text{rank}((f \diamond \text{OR})_{\oplus}))^4) \\ &\leq O(D((f \diamond \text{OR})_{\oplus})^4) \leq O(\text{DT}_{\oplus}(f \diamond \text{OR})^4). \end{aligned}$$

## 6 Block sensitivity to communication complexity

In this section, we provide a complete classification of gadgets for lifting from block sensitivity to deterministic and randomized communication complexities. Note that this classification will also give us a classification of gadgets for query-to-communication polynomial lifting since block sensitivity and query complexity are polynomially related to each other in the case of total functions. However, since the proof goes through block sensitivity, we will classify block sensitivity to communication complexity lifting.

### 6.1 Reductions in communication complexity

Let  $f_i: \mathcal{X}_i \times \mathcal{Y}_i \rightarrow \{0, 1\}$  for  $i = 1, 2$  be two-party functions. We say that  $f_1$  reduces to  $f_2$ , denoted  $f_1 \leq f_2$ , if the communication matrix of  $f_1$  is a submatrix of the communication matrix of  $f_2$ . Equivalently,  $f_1 \leq f_2$  iff there exist one-to-one mappings  $\pi_A$  and  $\pi_B$  such that

$$f_1(x, y) = f_2(\pi_A(x), \pi_B(y)), \quad \forall (x, y) \in \mathcal{X}_1 \times \mathcal{Y}_1.$$

Zhang [22] proved the following theorem:

► **Theorem 23 (Zhang).** *If a two-party gadget  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  satisfies  $\text{AND}, \text{OR} \leq g$ , then for every function  $f: \{0, 1\}^n \rightarrow Q$ , the function  $f \diamond g$  has (constant error) randomized communication complexity  $\Omega(\text{bs}(f))$ .*

For the classification of gadgets, we will need the following lemma that shows that if  $\text{OR} \not\leq g$  then the communication matrix of  $g$  is block diagonal.

► **Lemma 24.** *If  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  satisfies that  $\text{OR} \not\leq g$  then there are partitions of  $\mathcal{X}$  and  $\mathcal{Y}$  into disjoint sets*

$$\mathcal{X} = \mathcal{X}_0 \sqcup \mathcal{X}_1 \sqcup \dots \sqcup \mathcal{X}_k \quad \text{and} \quad \mathcal{Y} = \mathcal{Y}_0 \sqcup \mathcal{Y}_1 \sqcup \dots \sqcup \mathcal{Y}_k,$$

such that

- If  $x \in \mathcal{X}_i, y \in \mathcal{Y}_j$ , where  $i \neq j$ , or  $i = 0$ , or  $j = 0$ , then  $g(x, y) = 0$ .
- If  $x \in \mathcal{X}_i, y \in \mathcal{Y}_i$ , where  $i \neq 0$ , then  $g(x, y) = 1$ .

► **Definition 25.** *We say that a gadget  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  is a blow-up of a gadget  $h: \mathcal{Z} \times \mathcal{W} \rightarrow \{0, 1\}$  if there are decompositions  $\mathcal{X} = \bigsqcup_{z \in \mathcal{Z}} \mathcal{X}_z$  and  $\mathcal{Y} = \bigsqcup_{w \in \mathcal{W}} \mathcal{Y}_w$ , with  $\mathcal{X}_z, \mathcal{Y}_w \neq \emptyset$ , such that all  $(x_z, y_w) \in \mathcal{X}_z \times \mathcal{Y}_w$  satisfy  $g(x_z, y_w) = h(z, w)$ .*

For example,  $g$  is a blow-up of XOR if (up to rearrangement) it has communication matrix of the following form

$$\begin{bmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & \dots & 1 \end{bmatrix}$$

If  $g$  is a blow-up of  $h$  then  $f \diamond g$  and  $f \diamond h$  have the same communication complexity in all models. Indeed, on the one hand,  $h$  is a restriction of  $g$ , and so a protocol for  $f \diamond g$  can be used to solve  $f \diamond h$ ; and on the other hand, by replacing  $x \in \mathcal{X}_z$  by  $z$  and  $y \in \mathcal{Y}_w$  by  $w$ , we can use a protocol for  $f \diamond h$  to solve  $f \diamond g$ .



► **Corollary 26.** *If  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  satisfies both  $\text{OR} \not\leq g$  and  $\text{AND} \not\leq g$  then either  $g$  is constant, or it is a blow-up of XOR.*

► **Corollary 27.** *If  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  satisfies both  $\text{OR} \not\leq g$  and  $\text{XOR} \not\leq g$  then either  $g$  is constant, or it is a blow-up of AND. Similarly, if  $\text{AND} \not\leq g$  and  $\text{XOR} \not\leq g$  then either  $g$  is constant, or it is a blow-up of OR.*

## 6.2 Gadget classification for randomized communication complexity

In this section we state the following dichotomy result for the randomized case.

► **Theorem 28.** *For every gadget  $g$ , one of the following cases holds:*

1. *There is an infinite family of functions  $f_n$  with  $\text{bs}(f_n) = \Omega(n)$  and  $\text{R}(f_n \diamond g) = O(\log n)$ .*
2. *For every function  $f$ , we have  $\text{R}(f \diamond g) = \Omega(\text{bs}(f)^{\Omega(1)})$ .*

The second case in the theorem corresponds to AND/OR gadgets, it holds due to Theorem 23. The first case is due to the following lemma.

► **Lemma 29.** *Let  $g, h: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  be gadgets such that  $\text{OR} \not\leq g$  and  $\text{AND} \not\leq h$ . Then*

$$\text{R}(\text{AND}_n \diamond g) = O(\log n), \quad \text{R}(\text{OR}_n \diamond h) = O(\log n).$$

## 6.3 Gadget classification for deterministic communication complexity

The following lifting theorem that implies the classification theorem.

► **Theorem 30.** *If a two-party gadget  $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  satisfies  $\text{AND}, \text{XOR} \leq g$ , then for every function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , the function  $f \diamond g$  has communication complexity  $\Omega(\text{bs}(f)^k)$  for some fixed constant  $k > 0$ . The same is true if  $g$  satisfies  $\text{OR}, \text{XOR} \leq g$ .*

Together with Theorem 23 this gives us a complete classification of gadgets.

► **Theorem 31.** *For every gadget  $g$ , one of the following cases holds:*

1. *There is an infinite family of functions  $f_n$  with  $\text{bs}(f_n) = \Omega(n)$  and  $\text{D}(f_n \diamond g) = O(1)$ .*
2. *For every function  $f$ , we have  $\text{D}(f \diamond g) = \Omega(\text{bs}(f)^{\Omega(1)})$ .*

If  $\text{AND}, \text{OR} \leq g$ , or  $\text{AND}, \text{XOR} \leq g$ , or  $\text{OR}, \text{XOR} \leq g$ , then Theorem 23 and Theorem 30 show that  $\text{D}(f \diamond g) = \Omega(\text{bs}(f)^{\Omega(1)})$ , so the second case holds.

If none of these cases holds, then at least two of the functions AND, OR, XOR do not reduce to  $g$ . Corollaries 26 and 27 show that either  $g$  is constant (and so the first case trivially holds) or it is a blow-up of one of the functions XOR, OR, AND. By taking  $f_n = \text{XOR}_n, \text{OR}_n, \text{AND}_n$  (respectively), we get the first case of the theorem.

## 7 Open problems

### Matching lower and upper bounds for the certificate complexity lifting

Theorem 14 shows that one there is a function  $f$  such that  $\text{C}(f) \geq \Omega(\text{C}_{\oplus}(f \diamond \text{OR})^2)$ . However, Theorem 16 shows only that  $\text{C}(f) \leq O(\text{C}_{\oplus}(f \diamond \text{OR})^3)$ . Can we show that  $\text{C}(f) \leq O(\text{C}_{\oplus}(f \diamond \text{OR})^2)$ ?

### Generalization of current results to relations

Almost all the results discussed above were proved for Boolean functions. However, the question of proving lifting dichotomies for relations is still open.

One motivation for studying relations instead of functions is the following: any tree-like Resolution refutation of a CNF formula corresponds to a *decision tree*, solving the *falsified clause problem* for this CNF (which is usually a relation rather than Boolean function). Similarly, any tree-like  $\text{Res}(\oplus)$  refutation of some CNF formula corresponds to a *parity decision tree*, solving the falsified clause problem for this CNF. So, any lifting theorem from decision trees to parity decision trees with constant size gadgets that holds for relations, should give us a new way of proving tree-like  $\text{Res}(\oplus)$  lower bounds.

### Conjunction DAG to parity conjunction DAG lifting

A parity conjunction DAG is defined similarly to a conjunction DAG, with the following two differences:

- Queries are linear forms rather than variables.
- Nodes are annotated by affine subspaces rather than partial assignments. The label  $\rho(v_b)$  of a child node  $v_b$ , that is attached to the parent node  $v$  via an edge labeled  $b$ , is a subspace of  $\rho(v) \cup \{\ell \leftarrow b\}$ , where  $\ell$  is the query in  $v$ .

Another possible direction of research is to prove a lifting theorem from conjunction DAG size to parity conjunction DAG size. The motivation for this kind of lifting also comes from the proof complexity. Dag-like Resolution refutation can be viewed as a conjunction DAG and dag-like  $\text{Res}(\oplus)$  refutation can be viewed as a parity conjunction DAG. So, proving such lifting theorems for the relations with small enough gadgets can be used to prove lower bounds for  $\text{Res}(\oplus)$  refutations, which is a long-standing open problem in proof complexity.

---

### References

- 1 Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. *Theory Comput.*, 3:81–102, 2007. doi:10.4086/toc.2007.v003a005.
- 2 Yaroslav Alekseev, Yuval Filmus, and Alexander Smal. Lifting dichotomies. *Electron. Colloquium Comput. Complex.*, pages TR24–037, 2024. URL: <https://eccc.weizmann.ac.il/report/2024/037>.
- 3 Paul Beame and Sajin Korothe. On disperser/lifting properties of the index and inner-product functions. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10–13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPICs*, pages 14:1–14:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.ITCS.2023.14.
- 4 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, March 2001. doi:10.1145/375827.375835.
- 5 Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation theorems via pseudo-random properties. *Comput. Complex.*, 28(4):617–659, December 2019. doi:10.1007/s00037-019-00190-7.
- 6 Arkadev Chattopadhyay, Nikhil S. Mande, Swagato Sanyal, and Suhail Sherif. Lifting to parity decision trees via stifling, 2022. doi:10.48550/arXiv.2211.17214.
- 7 Susanna de Rezende, Or Meir, Jakob Nordstrom, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with simple gadgets and applications to circuit and proof complexity. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 24–30, November 2020. doi:10.1109/FOCS46700.2020.00011.

- 8 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from Resolution. *Theory Comput.*, 16:Paper No. 13, 30, 2020. doi:10.4086/toc.2020.v016a013.
- 9 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. doi:10.1137/15M103145X.
- 10 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018. doi:10.1137/16M1082007.
- 11 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. *SIAM Journal on Computing*, 47(6):2435–2450, 2018. doi:10.1137/16M1059369.
- 12 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 132–143, 2017. doi:10.1109/FOCS.2017.21.
- 13 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for XOR functions. *SIAM J. Comput.*, 47(1):208–217, 2018. doi:10.1137/17M1136869.
- 14 Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity [extended abstract]. In *STOC'12 – Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 233–247. ACM, New York, 2012. doi:10.1145/2213977.2214000.
- 15 Alexander Knop, Shachar Lovett, Sam McGuire, and Weiqiang Yuan. Log-rank and lifting for AND-functions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, pages 197–208, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3406325.3450999.
- 16 Shachar Lovett. Communication is bounded by root of rank. *J. ACM*, 63(1), February 2016. doi:10.1145/2724704.
- 17 Shachar Lovett, Raghu Meka, Ian Mertz, Toniann Pitassi, and Jiapeng Zhang. Lifting with Sunflowers. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 104:1–104:24, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2022.104.
- 18 László Lovász and Michael Saks. Lattices, mobius functions and communications complexity. In *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science*, pages 81–90, 1988. doi:10.1109/SFCS.1988.21924.
- 19 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19, September 1999. doi:10.1007/s004930050062.
- 20 Alexander A. Sherstov. On quantum-classical equivalence for composed communication problems. *Quantum Inf. Comput.*, 10(5-6):435–455, 2010.
- 21 Alasdair Urquhart. The depth of resolution proofs. *Studia Logica: An International Journal for Symbolic Logic*, 99(1/3):349–364, 2011. URL: <http://www.jstor.org/stable/41475208>.
- 22 Shengyu Zhang. On the tightness of the Buhrman–Cleve–Wigderson simulation. In *Proceedings of the 20th International Symposium on Algorithms and Computation*, ISAAC '09, pages 434–440, Berlin, Heidelberg, 2009. Springer-Verlag. doi:10.1007/978-3-642-10631-6\_45.
- 23 Zhiqiang Zhang and Yaoyun Shi. On the parity complexity measures of Boolean functions. *Theoretical Computer Science*, 411(26):2612–2618, 2010. doi:10.1016/j.tcs.2010.03.027.