Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 43 (2019)

Nondeterministic and Randomized Boolean Hierarchies in Communication Complexity

Toniann Pitassi* University of Toronto

Morgan Shirley† University of Toronto

Thomas Watson‡ University of Memphis

December 31, 2020

Abstract

We investigate the power of randomness in two-party communication complexity. In particular, we study the model where the parties can make a constant number of queries to a function that has an efficient one-sided-error randomized protocol. The complexity classes defined by this model comprise the Randomized Boolean Hierarchy, which is analogous to the Boolean Hierarchy but defined with one-sided-error randomness instead of nondeterminism. Our techniques connect the Nondeterministic and Randomized Boolean Hierarchies, and we provide a complete picture of the relationships among complexity classes within and across these two hierarchies. In particular, we prove that the Randomized Boolean Hierarchy does not collapse, and we prove a query-to-communication lifting theorem for all levels of the Nondeterministic Boolean Hierarchy and use it to resolve an open problem stated in the paper by Halstenberg and Reischuk (CCC 1988) which initiated the study of this hierarchy.

1 Introduction

A classic example of the power of randomness in communication is the Equality function: Alice gets an n-bit string x, Bob gets an n-bit string y, and they want to know whether x equals y. Though Equality is maximally hard for deterministic communication [Yao79], it can be solved by a randomized protocol using $O(1)$ bits of communication (in the public-coin model) using the fingerprinting technique. Although this example (known for over 40 years) demonstrates the power of randomized communication in the standard two-party setting, many questions remain about the exact power of randomness in communication.

Much is still not understood about the power of randomness in other important communication settings beyond the standard two-party model. For example, in the Number-on-Forehead (NOF) model, even for three parties, no explicit function is known to exhibit a superpolylogarithmic separation between randomized and deterministic communication [LPS19] (despite the fact that linear lower bounds for randomized NOF protocols were proven over 30 years ago in the seminal paper [BNS89]). Another example concerns randomness in the context of *nondeterministic* two-party protocols (so-called Arthur–Merlin and Merlin–Arthur models). While strong lower bounds are known for Merlin–Arthur protocols [Kla03] (though even here, explicit linear lower bounds remain elusive), no strong lower bounds are known for Arthur–Merlin protocols computing any explicit function—such bounds are necessary for making progress on rigidity bounds and circuit lower bounds, and also important for delegation [Raz89, GPW16, AW17].

We wish to highlight that even in the standard setting of plain randomized two-party communication protocols, many fundamental questions remain poorly understood. Our goal in this paper is to make progress on some of these questions. Much is known about the *limitations* of randomness—e.g., strong (indeed, linear) lower bounds are known for the classic Set-Disjointness function [BFS86, KS92, Raz92, BYJKS04], which can be viewed as showing that $\mathsf{coNP}^{\mathsf{cc}} \not\subseteq \mathsf{BPP}^{\mathsf{cc}}$ (where we use $^{\mathsf{cc}}$ superscripts to indicate the communication analogues of classical complexity classes). However, surprisingly little is known about the *power* of

*Supported by NSERC, NSF Grant No. CCF-1900460 and the IAS School of Mathematics.
†Supported by NSERC.
‡Supported by NSF grants CCF-1657377 and CCF-1942742.

ISSN 1433-8092

randomness. Most known efficient randomized protocols for other functions, such as GREATER-THAN, can be viewed as oracle reductions to the aforementioned EQUALITY upper bound: GREATER-THAN $\in \mathsf{P}^{\text{EQUALITY}}$. Until recently, it was not even known whether $\mathsf{BPP}^{\mathsf{cc}} = \mathsf{P}^{\text{EQUALITY}}$ (assuming the classes are defined to contain only total two-party functions), i.e., whether EQUALITY is the "only" thing randomness is good for. Chattopadhyay, Lovett, and Vinyals [CLV19] answered this question in the negative by exhibiting a total function that is in $\mathsf{BPP}^{\mathsf{cc}}$ (indeed, in $\mathsf{RP}^{\mathsf{cc}}$) but not in $\mathsf{P}^{\text{EQUALITY}}$ (though the upper bound is still a form of fingerprinting). Since EQUALITY $\in \mathsf{coRP}^{\mathsf{cc}}$, we have $\mathsf{P}^{\text{EQUALITY}} \subseteq \mathsf{P}^{\mathsf{RPcc}}$ where the latter class contains functions with efficient deterministic protocols that can make (adaptive) oracle queries to any function in $\mathsf{RP}^{\mathsf{cc}}$. In fact, [CLV19] exhibited a strict infinite hierarchy of classes within $\mathsf{P}^{\mathsf{RPcc}}$, with $\mathsf{P}^{\text{EQUALITY}}$ at the bottom, and with subsequent levels having increasingly powerful specific oracle functions.

However, it remains open whether $\mathsf{BPP}^{\mathsf{cc}} = \mathsf{P}^{\mathsf{RPcc}}$ for total functions (intuitively, whether two-sided error can be efficiently converted to oracle queries to one-sided error). It is even open whether $\mathsf{BPP}^{\mathsf{cc}} \subseteq \mathsf{P}^{\mathsf{NPcc}}$ for total functions [GPW18b], although this is known to be false if the classes are defined to allow partial functions [PSS14]. We obtain a more detailed understanding of the structure of $\mathsf{P}^{\mathsf{RPcc}}$ by focusing on *restricting the number of oracle queries* (rather than restricting the $\mathsf{RP}^{\mathsf{cc}}$ function as in [CLV19]). For constants $q = 0, 1, 2, 3, \ldots$, the class $\mathsf{P}_{\parallel}^{\mathsf{RP}[q]\mathsf{cc}}$ consists of all two-party functions with an efficient (polylogarithmic communication cost) deterministic protocol that can make q many nonadaptive queries to an oracle for a function in $\mathsf{RP}^{\mathsf{cc}}$. (In contrast, the class $\mathsf{P}^{\mathsf{RP}[q]\mathsf{cc}}$ allows adaptive queries.) Even if partial functions are allowed, it was not known whether these classes form a strict infinite hierarchy, i.e., whether $\mathsf{P}_{\parallel}^{\mathsf{RP}[q]\mathsf{cc}} \subsetneq \mathsf{P}_{\parallel}^{\mathsf{RP}[q+1]\mathsf{cc}}$ for all q. One of our main contributions (Theorem 1) implies that this is indeed the case, even for total functions.

With $\mathsf{NP}^{\mathsf{cc}}$ in place of $\mathsf{RP}^{\mathsf{cc}}$, $\bigcup_q \mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}}$ forms the communication version of the Boolean Hierarchy from classical complexity, which was previously studied by Halstenberg and Reischuk [HR88, HR90, HR93]. We also prove results (Theorem 2 and Theorem 3) that resolve a 31-year-old open question posed in their work. $\bigcup_q \mathsf{P}_{\parallel}^{\mathsf{RP}[q]\mathsf{cc}}$ can be viewed as the communication version of the Randomized Boolean Hierarchy, which has not been studied explicitly in previous works. Overall, we obtain a complete understanding of the relationships among the classes within and across the Nondeterministic and Randomized Boolean Hierarchies in communication complexity. In the following subsection we discuss the relevant communication complexity classes and describe our theorems in detail.

1.1 Background and our contributions

Forgetting about communication complexity for a moment, the Boolean Hierarchy in classical complexity theory consists of problems that have a polynomial-time algorithm making a constant number of queries to an NP oracle. This hierarchy has an intricate relationship with other complexity classes, and its second level (DP) captures the complexity of certain "exact" versions of optimization problems. It consists of an infinite sequence of complexity classes $\mathsf{NP}(q)$ for $q = 1, 2, 3, \ldots$ (where $\mathsf{NP}(1) = \mathsf{NP}$ and $\mathsf{NP}(2) = \mathsf{DP}$). There are several equivalent ways of defining these classes [Wec85, CH86, KSW87, Wag88], which we review in Section 2. As illustrated in Figure 1, it is known that these levels are intertwined with the classes $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]}$ of all decision problems solvable in polynomial time using q nonadaptive NP queries (for constant q) [KSW87, Wag88, Bei91]:

$$\mathsf{NP}(q) \subseteq \mathsf{P}_{\parallel}^{\mathsf{NP}[q]} \subseteq \mathsf{NP}(q+1) \qquad \text{and} \qquad \mathsf{coNP}(q) \subseteq \mathsf{P}_{\parallel}^{\mathsf{NP}[q]} \subseteq \mathsf{coNP}(q+1)$$

(by closure of $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]}$ under complementation). Here, $\mathsf{coNP}(q)$ is the class of decision problems whose complement is in $\mathsf{NP}(q)$.

Analogous to the above *Nondeterministic* Boolean Hierarchy, one can define the *Randomized* Boolean Hierarchy by using RP (one-sided error randomized polynomial time) instead of NP in the definitions [BBJ+89]. The analogous inclusions like in Figure 1 hold among all the classes $\mathsf{RP}(q)$, $\mathsf{coRP}(q)$, and $\mathsf{P}_{\parallel}^{\mathsf{RP}[q]}$, by similar arguments. Although the (suitably defined) *Polynomial* Hierarchy over RP is known to collapse to its second level, which equals BPP [ZH86], the *Boolean* Hierarchy over RP has not been widely studied.

Recall the basic (deterministic) model of communication [Yao79, KN97], where Alice is given an input x and Bob is given an input y, and they wish to collaboratively evaluate some function $F(x, y)$ of their joint input by engaging in a protocol that specifies how they exchange bits of information about their

$$\begin{tikzcd}
 & \mathsf{NP} = \mathsf{NP}(1) \arrow[dr] & & \mathsf{DP} = \mathsf{NP}(2) \arrow[dr] & & \mathsf{NP}(3) \arrow[dr] & & \mathsf{NP}(4) \arrow[dr] & & & & \\
\mathsf{P} \arrow[ur] \arrow[dr] & & \mathsf{P}^{\mathsf{NP}[1]} \arrow[ur] \arrow[dr] & & \mathsf{P}_{\parallel}^{\mathsf{NP}[2]} \arrow[ur] \arrow[dr] & & \mathsf{P}_{\parallel}^{\mathsf{NP}[3]} \arrow[ur] \arrow[dr] & & \mathsf{P}_{\parallel}^{\mathsf{NP}[4]} & \cdots & \mathsf{P}_{\parallel}^{\mathsf{NP}} \arrow[r] & \mathsf{P}^{\mathsf{NP}} \\
 & \mathsf{coNP}(1) = \mathsf{coNP} \arrow[ur] & & \mathsf{coNP}(2) = \mathsf{coDP} \arrow[ur] & & \mathsf{coNP}(3) \arrow[ur] & & \mathsf{coNP}(4) \arrow[ur] & & & &
\end{tikzcd}$$

Figure 1: Relations between classes in the Boolean Hierarchy. Here, $\mathcal{C}_1 \to \mathcal{C}_2$ represents $\mathcal{C}_1 \subseteq \mathcal{C}_2$.

inputs. Many classical complexity classes (P, RP, NP, and so on) have natural two-party communication analogues [BFS86] (including the classes in the Nondeterministic and Randomized Boolean Hierarchies). The area of structural communication complexity, which concerns the properties of and relationships among these classes, is undergoing a renaissance and has turned out to yield new techniques and perspectives for understanding questions in a variety of other areas (circuit complexity, proof complexity, data structures, learning theory, delegation, fine-grained complexity, property testing, cryptography, extended formulations, etc.) [GPW18b]. For any classical time-bounded complexity class $\mathcal{C}$, we use $\mathcal{C}^{\mathsf{cc}}$ to denote its communication complexity analogue—the class of all two-party functions on n bits that admit a protocol communicating at most polylog(n) bits, in a model defined by analogy with the classical $\mathcal{C}$.

Halstenberg and Reischuk [HR88, HR90] initiated the study of the Nondeterministic Boolean Hierarchy in two-party communication complexity. They observed that the inclusions shown in Figure 1 hold for the communication versions of the classes, by essentially the same proofs as in the time-bounded setting. They also proved that $\mathsf{NP}(q)^{\mathsf{cc}} \neq \mathsf{coNP}(q)^{\mathsf{cc}}$, which simultaneously implies that each of the inclusions is strict: $\mathsf{NP}(q)^{\mathsf{cc}} \subsetneq \mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}} \subsetneq \mathsf{NP}(q+1)^{\mathsf{cc}}$.

The communication version of the Randomized Boolean Hierarchy has not been explicitly studied as far as we know, but as mentioned earlier it is interesting since Equality $\in \mathsf{coRP}^{\mathsf{cc}}$ and many randomized protocols have been designed by reduction to this fact (such as Greater-Than $\in \mathsf{P}^{\mathsf{RPcc}}$). What can we say about the power of a fixed number of queries to an $\mathsf{RP}^{\mathsf{cc}}$ oracle? Our first contribution strengthens the aforementioned separation due to Halstenberg and Reischuk.

Theorem 1. *For total functions,* $\mathsf{coRP}(q)^{\mathsf{cc}} \not\subseteq \mathsf{NP}(q)^{\mathsf{cc}}$ *for every constant* q.

Since $\mathsf{RP}^{\mathsf{cc}} \subseteq \mathsf{NP}^{\mathsf{cc}}$, Theorem 1 simultaneously implies that each of the inclusions in the Randomized Boolean Hierarchy is strict: $\mathsf{RP}(q)^{\mathsf{cc}} \subsetneq \mathsf{P}_{\parallel}^{\mathsf{RP}[q]\mathsf{cc}} \subsetneq \mathsf{RP}(q+1)^{\mathsf{cc}}$, and thus the hierarchy does not collapse. Previously, no separation beyond the first level seemed to be known in the literature. Our proof of Theorem 1 is completely different from (and more involved than) Halstenberg and Reischuk's proof of $\mathsf{coNP}(q)^{\mathsf{cc}} \not\subseteq \mathsf{NP}(q)^{\mathsf{cc}}$, which used the "easy-hard argument" of [Kad88].

In [HR88, HR90], Halstenberg and Reischuk also explicitly asked whether the inclusion $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}} \subseteq \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}}$ is strict. When $q = 0$, this is answered by the familiar results that $\mathsf{P}^{\mathsf{cc}} = \mathsf{NP}^{\mathsf{cc}} \cap \mathsf{coNP}^{\mathsf{cc}}$ when the classes are defined to contain only total functions [HR93], whereas $\mathsf{P}^{\mathsf{cc}} \subsetneq \mathsf{NP}^{\mathsf{cc}} \cap \mathsf{coNP}^{\mathsf{cc}}$ (indeed, $\mathsf{P}^{\mathsf{cc}} \subsetneq \mathsf{ZPP}^{\mathsf{cc}}$) holds when partial functions (promise problems) are allowed. For $q > 0$, we resolve this 31-year-old open question by proving that the situation is analogous to the $q = 0$ case.

Theorem 2. *For total functions,*

$$\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}} = \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}} \qquad \textit{and} \qquad \mathsf{P}_{\parallel}^{\mathsf{RP}[q]\mathsf{cc}} = \mathsf{RP}(q+1)^{\mathsf{cc}} \cap \mathsf{coRP}(q+1)^{\mathsf{cc}}$$

for every constant q.

Theorem 3. *For partial functions,* $\mathsf{RP}(q+1)^{\mathsf{cc}} \cap \mathsf{coRP}(q+1)^{\mathsf{cc}} \not\subseteq \mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}}$ *for every constant* q.

Since $\mathsf{RP}^{\mathsf{cc}} \subseteq \mathsf{NP}^{\mathsf{cc}}$, Theorem 3 implies that

$$\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}} \subsetneq \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}} \qquad \text{and} \qquad \mathsf{P}_{\|}^{\mathsf{RP}[q]\mathsf{cc}} \subsetneq \mathsf{RP}(q+1)^{\mathsf{cc}} \cap \mathsf{coRP}(q+1)^{\mathsf{cc}}$$

for partial functions. Taken together, Theorem 1, Theorem 2, and Theorem 3 complete the picture of the relationships among the classes within and across both hierarchies, for both total and partial functions.

1.2 Query-to-communication lifting

Our proof of Theorem 3 uses the paradigm of *query-to-communication lifting* [RM99, GLM+16, Göö15, GPW18a, GKPW19, GPW20, Wat20]. This approach to proving communication lower bounds has led to breakthroughs on fundamental questions in communication complexity and many of its application areas. The idea consists of two steps:

(1) First prove an analogous lower bound in the simpler setting of decision tree depth complexity (a.k.a. query complexity). This step captures the combinatorial core of the lower bound argument without the burden of dealing with the full power of communication protocols.

(2) Then apply a *lifting theorem*, which translates the query lower bound into a communication lower bound for a related two-party function. This step encapsulates the general-purpose machinery for dealing with protocols, and can be reused from one argument to the next.

The availability of a lifting theorem greatly simplifies the task of proving certain communication lower bounds, because it divorces the problem-specific aspects from the generic aspects. The format of a lifting theorem is that if $f\colon \{0,1\}^n \to \{0,1\}$ is any partial function and $g\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ is a certain "small" two-party function called a gadget, then the communication complexity of the two-party composed function $f \circ g^n\colon \mathcal{X}^n \times \mathcal{Y}^n \to \{0,1\}$—in which Alice gets $x = (x_1, \ldots, x_n)$, Bob gets $y = (y_1, \ldots, y_n)$, and their goal is to evaluate $(f \circ g^n)(x, y) \coloneqq f(g(x_1, y_1), \ldots, g(x_n, y_n))$—should be approximately the query complexity of the outer function f. One direction is generally straightforward: given a query upper bound for f, a communication upper bound for $f \circ g^n$ is witnessed by a protocol that simulates the decision tree for f and evaluates $g(x_i, y_i)$ whenever it queries the i^{th} bit of the input to f; the number of bits of communication is at most the number of queries made by the decision tree times the (small) cost of evaluating a copy of g. The other direction is the challenging part: despite Alice and Bob's ability to send messages that depend in arbitrary ways on all n coordinates, they nevertheless cannot do much better than just simulating a decision tree, which involves "talking about" one coordinate at a time.

A lifting theorem must be stated with respect to a particular model of computation, such as deterministic, one-sided error randomized, nondeterministic, etc., which we associate with the corresponding complexity classes. Indeed, lifting theorems are known for P [RM99, GPW18a], RP [GPW20], NP [GLM+16, Göö15], and many other classes. It is convenient to recycle complexity class names to denote the complexity of a given function in the corresponding model, e.g., $\mathsf{P}^{\mathsf{dt}}(f)$ is the minimum worst-case number of queries made by any decision tree that computes f, and $\mathsf{P}^{\mathsf{cc}}(F)$ is the minimum worst-case communication cost of any protocol that computes F. With this notation, the deterministic lifting theorem from [RM99, GPW18a] can be stated as: for all f, $\mathsf{P}^{\mathsf{cc}}(f \circ g^n) = \mathsf{P}^{\mathsf{dt}}(f) \cdot \Theta(\log n)$ where $g\colon [m] \times \{0,1\}^m \to \{0,1\}$ is the "index" gadget defined by $g(x, y) = y_x$ with $m \coloneqq n^{20}$. (Note that $\mathsf{P}^{\mathsf{cc}}(g) = O(\log n)$ since Alice can send her $\log m$-bit "pointer" to Bob, who responds with the pointed-to bit from his string.) The index gadget has also been used in lifting theorems for several other complexity classes.

We prove lifting theorems for all classes in the Nondeterministic Boolean Hierarchy, with the index gadget.

Theorem 4. *For every partial function $f\colon \{0,1\}^n \to \{0,1\}$ and every constant q,*

(i) $\mathsf{NP}(q)^{\mathsf{cc}}(f \circ g^n) \;=\; \mathsf{NP}(q)^{\mathsf{dt}}(f) \cdot \Theta(\log n)$

(ii) $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}}(f \circ g^n) \;=\; \mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{dt}}(f) \cdot \Theta(\log n)$

where $g\colon [m] \times \{0,1\}^m \to \{0,1\}$ is the index gadget defined by $g(x, y) = y_x$ with $m \coloneqq n^{20}$.

Only part *(ii)* is needed for proving Theorem 3, but part *(i)* forms an ingredient in the proof of *(ii)* and is of independent interest.

The most closely related lifting theorem to Theorem 4 is the one for P^{NP} [GKPW19], corresponding to computations that make an unbounded number of adaptive queries to an NP oracle. In that paper, the overall idea was to approximately characterize P^{NP} complexity in terms of *decision lists* (DL), and then prove a lifting theorem directly for DLs. Briefly, a conjunction DL (introduced by [Riv87]) is a sequence of small-width conjunctions each with an associated output bit, and the output is determined by finding the first conjunction in the list that accepts the given input. A rectangle DL is similar but with combinatorial rectangles instead of conjunctions. The proof from [GKPW19] shows how to convert a rectangle DL for $f \circ g^n$ into a conjunction DL for f.

The gist of our arguments for both parts of Theorem 4 is to approximately characterize (via different techniques than for P^{NP}) these classes in terms of DLs with a bounded number of *alternations* (how many times the associated output bit flips as we walk down the entire DL). The DL lifting argument from [GKPW19] does not preserve the number of alternations, but we show how it can be adapted to do this. Our techniques also yield an approximate lifting theorem for $\mathsf{P}_{\parallel}^{\mathsf{NP}}$ (corresponding to computations that make an unbounded number of nonadaptive NP oracle queries) because our $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]}$ lifting theorem works when q is not constant, albeit with a $\mathrm{poly}(q)$ factor loss.

2 Preliminaries

We assume familiarity with deterministic computation in query and communication complexity [Juk12, KN97]. Recall the following standard definitions of nondeterministic and one-sided error randomized models:

- An $\mathsf{NP}^{\mathsf{dt}}$ decision tree is a DNF formula Φ. Given an input z, the output of such a decision tree is Φ evaluated on z. A function f is computed by Φ if $f(z) = \Phi(z)$ on all inputs z for which $f(z)$ is defined. The cost of Φ is the maximum width (number of literals) in any conjunction in Φ.

- An $\mathsf{NP}^{\mathsf{cc}}$ protocol is a set $\mathcal{R}$ of combinatorial rectangles. Given an input (x, y), the output of such a protocol is $\mathcal{R}(x, y) \coloneqq 1$ iff there exists an $R \in \mathcal{R}$ containing (x, y). A two-party function F is computed by $\mathcal{R}$ if $F(x, y) = \mathcal{R}(x, y)$ on all inputs (x, y) for which $F(x, y)$ is defined. The cost of $\mathcal{R}$ is $\lceil \log(|\mathcal{R}| + 1)\rceil$, which intuitively represents the number of bits required to specify a rectangle in $\mathcal{R}$ or indicate that the input is in no such rectangle.

- An $\mathsf{RP}^{\mathsf{dt}}$ decision tree is a distribution $\boldsymbol{T}$ over deterministic decision trees. Given an input z, the output of such a decision tree is computed by sampling a deterministic decision tree T from $\boldsymbol{T}$ and evaluating $T(z)$. A function f is computed by $\boldsymbol{T}$ if for all $z \in f^{-1}(0)$, $\Pr_{T\sim\boldsymbol{T}}[T(z) = 1] = 0$ and for all $z \in f^{-1}(1)$, $\Pr_{T\sim\boldsymbol{T}}[T(z) = 1] \geq 1/2$. The cost of $\boldsymbol{T}$ is the maximum number of bits queried in any T in its support.

- An $\mathsf{RP}^{\mathsf{cc}}$ protocol is a distribution $\boldsymbol{\Pi}$ over deterministic communication protocols. Given an input (x, y), the output of such a protocol is computed by sampling a deterministic protocol Π from $\boldsymbol{\Pi}$ and evaluating $\Pi(x, y)$. A function F is computed by $\boldsymbol{\Pi}$ if for all $(x, y) \in F^{-1}(0)$, $\Pr_{\Pi\sim\boldsymbol{\Pi}}[\Pi(x, y) = 1] = 0$ and for all $(x, y) \in F^{-1}(1)$, $\Pr_{\Pi\sim\boldsymbol{\Pi}}[\Pi(x, y) = 1] \geq 1/2$. The cost of $\boldsymbol{\Pi}$ is the maximum number of bits exchanged in any Π in its support.

Let $\mathcal{C}$ be an arbitrary complexity class name representing a model of computation (such as NP or RP). We let $\mathcal{C}^{\mathsf{cc}}(F)$ denote the communication complexity of a two-party function F in the corresponding model: the minimum cost of any $\mathcal{C}^{\mathsf{cc}}$ protocol computing F. We let $\mathcal{C}^{\mathsf{dt}}(f)$ denote the query complexity of a Boolean function f in the corresponding model: the minimum cost of any $\mathcal{C}^{\mathsf{dt}}$ decision tree computing f. Often we will abuse notation by having F or f refer to an infinite *family* of functions, where there is at most one function in the family for each possible input length. In this case, $\mathcal{C}^{\mathsf{cc}}(F)$ or $\mathcal{C}^{\mathsf{dt}}(f)$ will be the complexity parameterized by the input length n; we typically express this with asymptotic notation. When written by itself, $\mathcal{C}^{\mathsf{cc}}$ or $\mathcal{C}^{\mathsf{dt}}$ denotes the class of all families of functions with complexity at most polylogarithmic in n, in the corresponding model. We will always clarify whether a class $\mathcal{C}^{\mathsf{cc}}$ or $\mathcal{C}^{\mathsf{dt}}$ is meant to contain partial functions or just total functions, since this is not explicit in the notation.

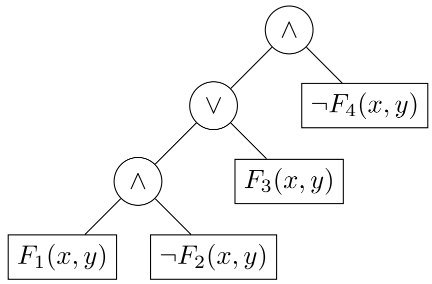

Figure 2: A visualization of a $\mathcal{C}(4)^{\mathsf{cc}}$ protocol, where each F_i is the function computed by $\mathcal{C}^{\mathsf{cc}}$ protocol Π_i.

For $\mathsf{RP}^{\mathsf{cc}}$ and $\mathsf{RP}^{\mathsf{dt}}$, the constant 1/2 in the success probability is arbitrary: by amplification, choosing a different positive constant in the definition would only affect the complexity of any function by a constant factor. Also note that $\mathsf{NP}^{\mathsf{dt}}(f) \le \mathsf{RP}^{\mathsf{dt}}(f)$ for all f, and since we defined $\mathsf{RP}^{\mathsf{cc}}$ using the public-coin model, we have $\mathsf{NP}^{\mathsf{cc}}(F) \le \mathsf{RP}^{\mathsf{cc}}(F) + O(\log n)$ for all F (by decreasing the number of random bits for sampling a protocol to $O(\log n)$ and using nondeterminism to guess an outcome that results in output 1).

2.1 Nondeterministic and Randomized Boolean Hierarchies

In the following definitions, restrict $\mathcal{C}$ to be either NP or RP. We will use two different but equivalent definitions of the constituent levels of the Nondeterministic and Randomized Boolean Hierarchies. Our "official" definition is in terms of the following "decision list functions" (also known as "odd-max-bit"):

Definition 5. $\Delta_q \colon \{0,1\}^q \to \{0,1\}$ is defined inductively as follows:

- $\Delta_1(z_1) \coloneqq z_1$.
- If q is odd, $\Delta_q(z_1,\ldots,z_{q-1},z_q) \coloneqq \Delta_{q-1}(z_1,\ldots,z_{q-1}) \vee z_q$.
- If q is even, $\Delta_q(z_1,\ldots,z_{q-1},z_q) \coloneqq \Delta_{q-1}(z_1,\ldots,z_{q-1}) \wedge (\neg z_q)$.

In other words, letting $\oplus \colon \mathbb{N} \to \{0,1\}$ denote the parity function, we have $\Delta_q(z) \coloneqq \oplus(i)$ where i is the greatest index such that $z_i = 1$ (or $i = 0$ if z is all zeros).

Definition 6. A $\mathcal{C}(q)^{\mathsf{cc}}$ protocol is an ordered list of q many $\mathcal{C}^{\mathsf{cc}}$ protocols $\Pi = (\Pi_1,\ldots,\Pi_q)$ which compute (total or partial) functions $F_1,\ldots,F_q$ respectively. Given an input (x,y), the output of the protocol is $\Pi(x,y) \coloneqq \Delta_q(F_1(x,y),\ldots,F_q(x,y))$. The cost of a $\mathcal{C}(q)^{\mathsf{cc}}$ protocol is the sum of the costs of the component $\mathcal{C}^{\mathsf{cc}}$ protocols.

See Figure 2 for a visualization. When the constituent $\mathcal{C}^{\mathsf{cc}}$ protocols compute partial functions, those partial functions should all have the same domain.

Note that we define the output of $\Pi(x,y)$ as the function Δ_q applied to the outputs of the functions $F_1(x,y),\ldots,F_q(x,y)$, not the protocols $\Pi_1(x,y),\ldots,\Pi_q(x,y)$. This is important only for randomized models of computation, as the output of a randomized protocol may differ from the function it computes. In nondeterministic models of computation, the output of the protocol always agrees with the function computed by that protocol, and so we often make the notation simpler by using protocols and functions interchangeably.

The Nondeterministic Boolean Hierarchy is $\bigcup_{\text{constant } q} \mathsf{NP}(q)^{\mathsf{cc}}$ and the Randomized Boolean Hierarchy is $\bigcup_{\text{constant } q} \mathsf{RP}(q)^{\mathsf{cc}}$. We are also interested in the complement classes $\mathsf{coNP}(q)^{\mathsf{cc}}$ and $\mathsf{coRP}(q)^{\mathsf{cc}}$. As is standard, when we write $\mathsf{co}\mathcal{C}(q)^{\mathsf{cc}}$ we refer to the class $\mathsf{co}(\mathcal{C}(q)^{\mathsf{cc}})$ (that is, functions that are the negations of functions in $\mathcal{C}(q)^{\mathsf{cc}}$) as opposed to $(\mathsf{co}\mathcal{C})(q)^{\mathsf{cc}}$ (that is, where the component protocols are $\mathsf{co}\mathcal{C}^{\mathsf{cc}}$ protocols).

There are analogous definitions in query complexity:

Definition 7. A $\mathcal{C}(q)^{\mathsf{dt}}$ decision tree is an ordered list of q many $\mathcal{C}^{\mathsf{dt}}$ decision trees $T = (T_1,\ldots,T_q)$ which compute (total or partial) functions $f_1,\ldots,f_q$ respectively. Given an input z, the output of the decision tree is $T(z) \coloneqq \Delta_q(f_1(z),\ldots,f_q(z))$. The cost of a $\mathcal{C}(q)^{\mathsf{dt}}$ decision tree is the sum of the costs of the component $\mathcal{C}^{\mathsf{dt}}$ decision trees.

Our alternative definition of the Nondeterministic and Randomized Boolean Hierarchies simply replaces Δ_q with the parity function $\oplus_q\colon \{0,1\}^q \to \{0,1\}$. In Appendix A.1 we provide the standard proof that these official and alternative definitions are equivalent:

Lemma 8. *For $\mathcal{C} \in \{\mathsf{NP}, \mathsf{RP}\}$, if the definitions of $\mathcal{C}(q)^{\mathsf{cc}}$ and $\mathcal{C}(q)^{\mathsf{dt}}$ are changed to use $\oplus_q$ in place of Δ_q, it only affects the complexity measures $\mathcal{C}(q)^{\mathsf{cc}}(F)$ and $\mathcal{C}(q)^{\mathsf{dt}}(f)$ by a constant factor (depending on q).*

We use both definitions in this paper. We found that the Δ_q definition makes it easier to prove the lifting theorems, and the $\oplus_q$ definition makes it easier to prove concrete upper and lower bounds.

2.2 Decision lists

The reason we call Δ_q a "decision list function" is that it highlights the connection between the Boolean Hierarchy classes and the *decision list* models of computation:

Definition 9. A rectangle decision list $\mathcal{L}_{\mathrm{R}}$ is an ordered list of pairs $(R_1, \ell_1), (R_2, \ell_2), \ldots$ where each R_i is a combinatorial rectangle, $\ell_i \in \{0,1\}$ is a label, and the final rectangle in the list contains all inputs in the domain. For an input (x, y), the output $\mathcal{L}_{\mathrm{R}}(x, y)$ is ℓ_i where i is the first index for which $(x, y) \in R_i$. The cost of $\mathcal{L}_{\mathrm{R}}$ is the log of the length of the list.

Definition 10. A conjunction decision list $\mathcal{L}_{\mathrm{C}}$ is an ordered list of pairs $(C_1, \ell_1), (C_2, \ell_2), \ldots$ where each C_i is a conjunction, $\ell_i \in \{0,1\}$ is a label, and the final conjunction in the list accepts all inputs in the domain. For an input z, the output $\mathcal{L}_{\mathrm{C}}(z)$ is ℓ_i where i is the first index for which $C_i(z) = 1$. The cost of $\mathcal{L}_{\mathrm{C}}$ is the maximum width of any conjunction in the list.

Note that the restriction on the final rectangle/conjunction is without loss of generality. The complexity measures $\mathsf{DL}^{\mathsf{cc}}(F)$ and $\mathsf{DL}^{\mathsf{dt}}(f)$ are the minimum cost of any rectangle/conjunction decision list computing F or f, and the classes $\mathsf{DL}^{\mathsf{cc}}$ and $\mathsf{DL}^{\mathsf{dt}}$ contain those functions with complexity at most $\mathrm{polylog}(n)$.

We now define *q-alternating decision lists* to have the additional restriction that the sequence of output labels $\ell_1, \ell_2, \ldots$ only flips between 0 and 1 at most q times, and furthermore the last label is 0. This restriction partitions the list into contiguous *levels* where all labels in the same level are equal; without loss of generality the last level consists only of the final "catch-all" entry. For convenience, in the list entries we replace the labels with the level numbers themselves.

Definition 11. A q-alternating rectangle decision list $\mathcal{L}_{\mathrm{R}}$ is an ordered list of pairs $(R_1, \ell_1), (R_2, \ell_2), \ldots$ where each R_i is a combinatorial rectangle, $\ell_i \in \{0, 1, \ldots, q\}$ is a level such that $\ell_i \geq \ell_{i+1}$ for all i, and the final rectangle in the list contains all inputs in the domain and is the only rectangle at level 0. For an input (x, y), the output $\mathcal{L}_{\mathrm{R}}(x, y)$ is $\oplus(\ell_i)$ where i is the first index for which $(x, y) \in R_i$. The cost of $\mathcal{L}_{\mathrm{R}}$ is the log of the length of the list.

Definition 12. A q-alternating conjunction decision list $\mathcal{L}_{\mathrm{C}}$ is an ordered list of pairs $(C_1, \ell_1), (C_2, \ell_2), \ldots$ where each C_i is a conjunction, $\ell_i \in \{0, 1, \ldots, q\}$ is a level such that $\ell_i \geq \ell_{i+1}$ for all i, and the final conjunction in the list accepts all inputs in the domain and is the only conjunction at level 0. For an input z, the output $\mathcal{L}_{\mathrm{C}}(z)$ is $\oplus(\ell_i)$ where i is the first index for which $C_i(z) = 1$. The cost of $\mathcal{L}_{\mathrm{C}}$ is the maximum width of any conjunction in the list.

The complexity measures $\mathsf{DL}(q)^{\mathsf{cc}}(F)$ and $\mathsf{DL}(q)^{\mathsf{dt}}(f)$ are the minimum cost of any q-alternating rectangle/conjunction decision list computing F or f, and the classes $\mathsf{DL}(q)^{\mathsf{cc}}$ and $\mathsf{DL}(q)^{\mathsf{dt}}$ contain those functions with complexity at most $\mathrm{polylog}(n)$.

It turns out that q-alternating decision lists are equivalent to $\mathsf{NP}(q)$ in both communication and query complexity. This follows almost immediately from the definition of Δ_q — in fact, $\mathsf{NP}(q)$ is just $\mathsf{DL}(q)$ without an ordering on entries in the same level. For completeness, we include the argument in Appendix A.2.

Lemma 13. *$\mathsf{DL}(q)^{\mathsf{cc}}(F) = \Theta(\mathsf{NP}(q)^{\mathsf{cc}}(F))$ and $\mathsf{DL}(q)^{\mathsf{dt}}(f) = \Theta(\mathsf{NP}(q)^{\mathsf{dt}}(f))$ for every constant q. Thus, $\mathsf{DL}(q)^{\mathsf{cc}} = \mathsf{NP}(q)^{\mathsf{cc}}$ and $\mathsf{DL}(q)^{\mathsf{dt}} = \mathsf{NP}(q)^{\mathsf{dt}}$ for partial functions.*

This can be contrasted with the lemma from [GKPW19] stating that $\mathsf{DL}^{\mathsf{cc}} = \mathsf{P}^{\mathsf{NPcc}}$ and $\mathsf{DL}^{\mathsf{dt}} = \mathsf{P}^{\mathsf{NPdt}}$ for partial functions.

2.3 Parallel queries

In the following definitions, restrict $\mathcal{C}$ to be either NP or RP.

Definition 14. A $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{cc}}$ protocol consists of a deterministic protocol Π_{det} that maps an input (x, y) to two things: a function $\mathsf{out}\colon \{0,1\}^q \to \{0,1\}$ and an ordered list of q many $\mathcal{C}^{\mathsf{cc}}$ protocols $(\Pi_1, \ldots, \Pi_q)$ which compute (total or partial) functions $F_1, \ldots, F_q$ respectively. The output is then $\mathsf{out}(F_1(x, y), \ldots, F_q(x, y))$. The cost of a $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{cc}}$ protocol is the communication cost (depth) of Π_{det} plus the maximum over (x, y) of the sum of the costs of the $\mathcal{C}^{\mathsf{cc}}$ protocols produced by $\Pi_{\text{det}}(x, y)$.

Definition 15. A $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{dt}}$ decision tree consists of a deterministic decision tree T_{det} that maps an input z to two things: a function $\mathsf{out}\colon \{0,1\}^q \to \{0,1\}$ and an ordered list of q many $\mathcal{C}^{\mathsf{dt}}$ decision trees $(T_1, \ldots, T_q)$ which compute (total or partial) functions $f_1, \ldots, f_q$ respectively. The output is then $\mathsf{out}(f_1(z), \ldots, f_q(z))$. The cost of a $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{dt}}$ decision tree is the query cost (depth) of T_{det} plus the maximum over z of the sum of the costs of the $\mathcal{C}^{\mathsf{dt}}$ decision trees produced by $T_{\text{det}}(z)$.

The following lemma states that at each leaf of Π_{det} or T_{det}, we can replace the q "$\mathcal{C}$ oracle queries" with one "$\mathcal{C}(q)$ oracle query" (where some leaves may output the oracle's answer, while other leaves output the negation of it). This was shown in classical time-bounded complexity using the so-called "mind-change argument" [Bei91], and this argument can be translated directly to communication and query complexity. For example, [HR90] used this method to show that $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}} \subseteq \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}}$. For completeness, we provide the proof in Appendix A.3. We will only need to use the result for $\mathcal{C} = \mathsf{NP}$.

Lemma 16. *For $\mathcal{C} \in \{\mathsf{NP}, \mathsf{RP}\}$, we have $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{cc}}(F) = \Theta(\mathsf{P}^{\mathcal{C}(q)[1]\mathsf{cc}}(F))$ and $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{dt}}(f) = \Theta(\mathsf{P}^{\mathcal{C}(q)[1]\mathsf{dt}}(f))$ for every constant q.*

3 Separations

Restatement of Theorem 1. *For total functions, $\mathsf{coRP}(q)^{\mathsf{cc}} \not\subseteq \mathsf{NP}(q)^{\mathsf{cc}}$ for every constant q.*

In Section 3.1 we prove a weaker version of Theorem 1 that holds for partial functions, by showing the analogous query complexity separation $\mathsf{coRP}(q)^{\mathsf{dt}} \not\subseteq \mathsf{NP}(q)^{\mathsf{dt}}$ then applying our lifting theorem for $\mathsf{NP}(q)$, Theorem 4.*(i)*. (The query complexity separation is known to be false for total functions: $\mathsf{coRP}(q)^{\mathsf{dt}} \subseteq \mathsf{P}^{\mathsf{RPdt}} \subseteq \mathsf{BPP}^{\mathsf{dt}} \subseteq \mathsf{P}^{\mathsf{dt}} \subseteq \mathsf{NP}(q)^{\mathsf{dt}}$ [Nis91].) This serves two purposes: it is a warmup for our direct proof of Theorem 1 for total functions in Section 3.2, and it forms a component in our proof of Theorem 3 in Section 3.3.

3.1 Proof of Theorem 1 for partial functions

Fix any constant q. Let $\oplus_q\textsc{GapOr}\colon (\{0,1\}^n)^q \to \{0,1\}$ be the partial function where the input is divided into q blocks $z = (z_1, \ldots, z_q)$ having the promise that each $z_i \in \{0,1\}^n$ is either all zeros or at least half ones (call such an input *valid*), and which is defined by $\oplus_q\textsc{GapOr}(z) := 1$ iff an odd number of blocks i are such that z_i is nonzero. Note that $\mathsf{RP}(q)^{\mathsf{dt}}(\oplus_q\textsc{GapOr}) = O(1)$ by Lemma 8 and the fact that $\mathsf{RP}^{\mathsf{dt}}(\textsc{GapOr}) = 1$. Letting $g\colon [m] \times \{0,1\}^m \to \{0,1\}$ be the index gadget with $m := (qn)^{20}$, this implies that $\mathsf{RP}(q)^{\mathsf{cc}}(\oplus_q\textsc{GapOr} \circ g^{qn}) = O(\log n)$ (here we have used the "easy direction" of $\mathsf{RP}(q)$ lifting; the "hard direction" remains an open problem) and thus $\overline{\oplus_q\textsc{GapOr}} \circ g^{qn} \in \mathsf{coRP}(q)^{\mathsf{cc}}$. We will now prove that $\mathsf{NP}(q)^{\mathsf{dt}}(\overline{\oplus_q\textsc{GapOr}}) = \Omega(n)$, which by Theorem 4.*(i)* implies that $\mathsf{NP}(q)^{\mathsf{cc}}(\overline{\oplus_q\textsc{GapOr}} \circ g^{qn}) = \Omega(n \log n)$.

Suppose for contradiction $\overline{\oplus_q\textsc{GapOr}}$ has an $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree of cost $k \le n/2$, say $T = (\Phi_1, \ldots, \Phi_q)$ where each Φ_i is a DNF. By Lemma 8 we may assume the decision tree outputs 1 iff an odd number of these DNFs accept the input, in other words, $T(z) := \oplus_q(\Phi_1(z), \ldots, \Phi_q(z))$.

We will iteratively construct a sequence of partial assignments $\rho^j \in (\{0,1,*\}^n)^q$ for $j = 0, \ldots, q$, where each ρ^{j+1} is a refinement of ρ^j, such that *(i)* there are at least j many values of i for which Φ_i accepts all inputs that are consistent with ρ^j, and *(ii)* ρ^j is consistent with a valid input where exactly j blocks are nonzero. We obtain the contradiction when $j = q$: by *(ii)* some valid input z consistent with ρ^q has

z_i nonzero for all i and thus $\overline{\oplus_q\textsc{GapOr}}(z) = 1 - \oplus(q)$, yet by *(i)* we have $\Phi_i(z) = 1$ for all i and thus $T(z) = \oplus(q)$, contradicting the supposed correctness of T.

We will actually maintain stronger invariants than the above *(i)* and *(ii)*: For *(i)*, we will actually have for some j values of i—we assume they are $1, \dots, j$ for notational convenience—some individual conjunction C_i of Φ_i accepts all inputs consistent with ρ^j. For *(ii)*, ρ^j will actually have the following form: for some fixed assignment $l^j = (l_1, \dots, l_j) \in (\{0,1\}^n)^j$ such that l_i is at least half ones for all $i \in [j]$ ("left"), and for some partial assignment $r^j \in (\{0,*\}^n)^{q-j}$ ("right"), we have $\rho^j := l^j r^j$. The valid input z^j obtained by filling in a 0 for each $*$ in ρ^j has exactly j nonzero blocks, as needed for *(ii)*.

In fact, we will maintain that r^j has at least half stars in each block. Specifically, the total number of zeros in r^j will be at most the sum of the widths of the conjunctions $C_1, \dots, C_j$, which is at most the sum over all $i \in [q]$ of the maximum width of Φ_i, which equals $k \le n/2$. At the end, $r^q \in (\{0,*\}^n)^{q-q}$ will be the empty tuple, which means ρ^q will be the fixed valid input $l^q = z^q$, which has all blocks nonzero.

We start with $\rho^0 = r^0 = (*^n)^q$, which indeed has no zeros. Now supposing we already have l^j and r^j satisfying all the properties from the previous two paragraphs, we explain how to obtain $l_{j+1} \in \{0,1\}^n$ and $r^{j+1} \in (\{0,*\}^n)^{q-(j+1)}$ so these properties again hold (with $l^{j+1} := l^j l_{j+1}$).

We first observe that $z^j := l^j(0^n)^{q-j}$, which is consistent with ρ^j, must be accepted by at least one conjunction from $\Phi_{j+1}, \dots, \Phi_q$. This is because z^j is already accepted by the conjunctions $C_1, \dots, C_j$, and these cannot be the only values of i such that $\Phi_i(z^j) = 1$ since otherwise we would have $T(z^j) = \oplus(j)$ while $\overline{\oplus_q\textsc{GapOr}}(z^j) = 1 - \oplus(j)$, contradicting the supposed correctness of T. Now pick any one conjunction from $\Phi_{j+1}, \dots, \Phi_q$ that accepts z^j, and assume this conjunction is C_{j+1} from Φ_{j+1} for notational convenience. Defining the partial assignment $\tilde{r}^j$ from r^j by filling in a 0 for each $*$ corresponding to a negative literal in C_{j+1} (note that C_{j+1} has no positive literals among the last $q-j$ blocks since it accepts z^j), we see that the total number of zeros in $\tilde{r}^j$ is at most the sum of the widths of the conjunctions $C_1, \dots, C_{j+1}$.

Let r^{j+1} be all but the first block of $\tilde{r}^j$, and obtain l_{j+1} by replacing the remaining stars in the first block of $\tilde{r}^j$ with ones, noting that l_{j+1} is at least half ones since the first block of $\tilde{r}^j$ was at least half stars. Now $\rho^{j+1} := l^{j+1} r^{j+1}$ maintains the desired properties: *(ii)* is maintained since each block of l^{j+1} is at least half ones, and r^{j+1} has at most (sum of widths of $C_1, \dots, C_{j+1}$) many zeros and the rest stars. To see that *(i)* is maintained, consider any input z consistent with ρ^{j+1}. Then z is also consistent with ρ^j and is thus accepted by each of $C_1, \dots, C_j$. Moreover, $C_{j+1}(z) = 1$ since $C_{j+1}(z^j) = 1$ and z and z^j differ only in locations that are stars in $\tilde{r}^j$ and therefore do not appear in C_{j+1}.

3.2 Proof of Theorem 1 for total functions

Fix any constant q. Let $\oplus_q\textsc{NonEq}\colon (\{0,1\}^n)^q \times (\{0,1\}^n)^q \to \{0,1\}$ be the two-party total function where Alice's and Bob's inputs are divided into q blocks $x = (x_1, \dots, x_q)$ and $y = (y_1, \dots, y_q)$ with each $x_i, y_i \in \{0,1\}^n$, and which is defined by $\oplus_q\textsc{NonEq}(x,y) := 1$ iff there are an odd number of blocks i such that $x_i \ne y_i$. Note that $\mathsf{RP}(q)^{\mathsf{cc}}(\oplus_q\textsc{NonEq}) = O(1)$ by Lemma 8 and the standard fact that $\mathsf{RP}^{\mathsf{cc}}(\textsc{NonEq}) = O(1)$. Thus $\overline{\oplus_q\textsc{NonEq}} \in \mathsf{coRP}(q)^{\mathsf{cc}}$. We will now prove that $\mathsf{NP}(q)^{\mathsf{cc}}(\overline{\oplus_q\textsc{NonEq}}) = \Omega(n)$.

Suppose for contradiction $\overline{\oplus_q\textsc{NonEq}}$ has an $\mathsf{NP}(q)^{\mathsf{cc}}$ protocol of cost $k \le n/2^q$, say $\Pi = (\mathcal{R}_1, \dots, \mathcal{R}_q)$ where each $\mathcal{R}_i$ is a nonempty set of rectangles. By Lemma 8 we may assume the protocol outputs 1 iff the input is contained in an odd number of the rectangle unions $\bigcup_{R \in \mathcal{R}_i} R$ for $i \in [q]$, in other words, $\Pi(x,y) := \oplus_q(\mathcal{R}_1(x,y), \dots, \mathcal{R}_q(x,y))$. Note that, assuming $\mathcal{R}_i$ has cost k_i, the total number of rectangles in these unions is at most $\sum_i |\mathcal{R}_i| \le \prod_i (|\mathcal{R}_i| + 1) \le \prod_i 2^{k_i} = 2^k$.

We will iteratively construct a sequence of rectangles Q^j for $j = 0, \dots, q$, where each Q^{j+1} is contained in Q^j, such that *(i)* there are at least j many values of i for which $Q^j \subseteq \bigcup_{R \in \mathcal{R}_i} R$, and *(ii)* Q^j contains an input where exactly j blocks are unequal. We obtain the contradiction when $j = q$: by *(ii)* some input $(x,y) \in Q^q$ has $x_i \ne y_i$ for all i and thus $\overline{\oplus_q\textsc{NonEq}}(x,y) = 1 - \oplus(q)$, yet by *(i)* we have $\mathcal{R}_i(x,y) = 1$ for all i and thus $\Pi(x,y) = \oplus(q)$, contradicting the supposed correctness of Π.

We will actually maintain stronger invariants than the above *(i)* and *(ii)*: For *(i)*, we will actually have for some j values of i—we assume they are $1, \dots, j$ for notational convenience—some individual rectangle $R_i \in \mathcal{R}_i$ contains Q^j. For *(ii)*, Q^j will actually have the following form: for some fixed strings $a^j = (a_1, \dots, a_j) \in (\{0,1\}^n)^j$ and $b^j = (b_1, \dots, b_j) \in (\{0,1\}^n)^j$ such that $a_i \ne b_i$ for all $i \in [j]$, and for some nonempty set $S^j \subseteq (\{0,1\}^n)^{q-j}$, we have $Q^j := \{a^j s : s \in S^j\} \times \{b^j s : s \in S^j\}$, which we abbreviate as $a^j S^j \times b^j S^j$. Defining a *diagonal* input in Q^j to be one of the form $(a^j s, b^j s)$ for any particular $s \in S^j$, we see that each diagonal input has exactly j unequal blocks, as needed for *(ii)*. (In our notation, R and Q always refer to rectangles, while S is a set of strings associated with the diagonal entries of some rectangle.)

In fact, we will maintain not just that S^j is nonempty, but that it is sufficiently large. Specifically, the *deficiency* of S^j, defined as $\mathbf{D}_\infty(S^j) := n(q-j) - \log|S^j|$, will be at most $(2^j-1)(2k+1)$. At the end, since $(2^q-1)(2k+1) < \infty$, this guarantees that S^q will contain at least one element from $(\{0,1\}^n)^{q-q}$. The latter set only has one element, namely the empty tuple, so this means Q^q will contain the single input (a^q, b^q), which has all blocks unequal.

We start with $S^0 = (\{0,1\}^n)^q$, which indeed has $\mathbf{D}_\infty(S^0) = 0 = (2^0-1)(2k+1)$, and thus Q^0 is the rectangle of all possible inputs. Now supposing we already have a^j, b^j, and S^j satisfying all the properties from the previous two paragraphs, we explain how to obtain $a_{j+1}, b_{j+1} \in \{0,1\}^n$ and $S^{j+1} \subseteq (\{0,1\}^n)^{q-(j+1)}$ so these properties again hold (with $a^{j+1} := a^j a_{j+1}$ and $b^{j+1} := b^j b_{j+1}$).

We first observe that each diagonal input in Q^j must be contained in at least one rectangle from $\mathcal{R}_{j+1} \cup \cdots \cup \mathcal{R}_q$. This is because such an input (x,y) is already contained in the rectangles $R_1 \in \mathcal{R}_1, \ldots, R_j \in \mathcal{R}_j$, and these cannot be the only values of i such that $\mathcal{R}_i(x,y) = 1$ since otherwise we would have $\Pi(x,y) = \oplus(j)$ while $\overline{\oplus_q \textsc{NonEq}}(x,y) = 1 - \oplus(j)$, contradicting the supposed correctness of Π. Now pick one of the (at most 2^k) rectangles from $\mathcal{R}_{j+1} \cup \cdots \cup \mathcal{R}_q$ that contains the largest fraction (at least $1/2^k$) of diagonal inputs from Q^j, and assume this rectangle is $R_{j+1} \in \mathcal{R}_{j+1}$ for notational convenience. Defining $\tilde{S}^j := \{s \in S^j : (a^j s, b^j s) \in R_{j+1}\}$, we see that $\mathbf{D}_\infty(\tilde{S}^j) \le \mathbf{D}_\infty(S^j) + k \le (2^j-1)(2k+1) + k$.

Since R_{j+1} is a rectangle, it must in fact contain the entire rectangle $a^j\tilde{S}^j \times b^j\tilde{S}^j$. Since $a^j\tilde{S}^j \times b^j\tilde{S}^j \subseteq a^j S^j \times b^j S^j = Q^j$, by assumption it is also contained in each of $R_1, \ldots, R_j$. In the end, we will ensure Q^{j+1} is a subrectangle of $a^j\tilde{S}^j \times b^j\tilde{S}^j$, which will maintain property *(i)*: Q^{j+1} is contained in each of $R_1, \ldots, R_{j+1}$.

To maintain *(ii)*, we will find some $a_{j+1} \neq b_{j+1}$ and then define $S^{j+1} := \{s : a_{j+1}s \in \tilde{S}^j \text{ and } b_{j+1}s \in \tilde{S}^j\}$. Then $a_{j+1}S^{j+1} \subseteq \tilde{S}^j$ and $b_{j+1}S^{j+1} \subseteq \tilde{S}^j$ ensure that $Q^{j+1} := a^{j+1}S^{j+1} \times b^{j+1}S^{j+1}$ is indeed a subrectangle of $a^j\tilde{S}^j \times b^j\tilde{S}^j$, as we needed for *(i)*. The fact that this can be done with a not-too-small S^{j+1} is encapsulated in the following technical lemma, which we prove shortly:

Lemma 17. *Consider any bipartite graph with left nodes U and right nodes V, and suppose $1 \ge \varepsilon \ge 2/|U|$. If an ε fraction of all possible edges are present in the graph, then there exist distinct nodes $u, u' \in U$ that have at least $(\varepsilon^2/2) \cdot |V|$ common neighbors.*

Specifically, take $U := \{0,1\}^n$ and $V := (\{0,1\}^n)^{q-(j+1)}$ (so $|V| = 1$ if $j = q-1$, but that is fine), put an edge between $u \in U$ and $v \in V$ iff $uv \in \tilde{S}^j$, and let $\varepsilon := |\tilde{S}^j|/2^{n(q-j)} = 1/2^{\mathbf{D}_\infty(\tilde{S}^j)}$. Notice that $\varepsilon \ge 2/|U|$ holds since $\mathbf{D}_\infty(\tilde{S}^j) \le (2^j-1)(2k+1) + k \le 2^{j+1}k - 1 \le n-1$ follows from our assumption that $k \le n/2^q$. Thus Lemma 17 guarantees we can pick strings $a_{j+1} \neq b_{j+1}$ (corresponding to the nodes u, u') such that S^{j+1} (the set of common neighbors) has size at least $(\varepsilon^2/2) \cdot 2^{n(q-(j+1))}$. Thus

$$\begin{aligned}\mathbf{D}_\infty(S^{j+1}) \;&:=\; n(q-(j+1)) - \log|S^{j+1}| \;\le\; \log(2/\varepsilon^2) \;=\; 2\mathbf{D}_\infty(\tilde{S}^j) + 1 \\ &\le\; 2\big((2^j-1)(2k+1) + k\big) + 1 \;=\; \big(2(2^j-1)+1\big)(2k+1) \;=\; (2^{j+1}-1)(2k+1)\end{aligned}$$

as we needed for *(ii)*. This finishes the proof of Theorem 1.

Proof of Lemma 17. Let d_u and d_v denote the degrees of nodes $u \in U$ and $v \in V$, and let $d_{u,u'}$ denote the number of common neighbors of $u, u' \in U$. The sum of common neighbors over ordered pairs u, u' of not-necessarily-distinct left nodes is equivalent to the sum of pairs of not-necessarily-distinct neighbors over right nodes v, and so we have

$$\sum_{u,u' \in U} d_{u,u'} \;=\; \sum_{v \in V} d_v^2 \;\ge\; \Big(\sum_{v \in V} d_v\Big)^2/|V| \;=\; \varepsilon^2 \cdot |U|^2 \cdot |V|$$

by Cauchy–Schwarz and the assumption $\sum_{v \in V} d_v = \varepsilon \cdot |U| \cdot |V|$. Now sampling u, u' independently uniformly at random from U, we have

$$\varepsilon^2 \cdot |V| \;\le\; \mathop{\mathrm{E}}_{u,u'}[d_{u,u'}] \;\le\; \mathop{\mathrm{E}}_{u,u'}[d_{u,u'} \mid u \neq u'] + \mathop{\mathrm{E}}_{u}[d_u] \cdot \Pr_{u,u'}[u = u']$$

(the conditioning is valid by the assumption $|U| \ge 2$). Since $\mathrm{E}_u[d_u] = \varepsilon \cdot |V|$ and $\Pr_{u,u'}[u = u'] = 1/|U|$, rearranging gives

$$\mathop{\mathrm{E}}_{u,u'}[d_{u,u'} \mid u \neq u'] \;\ge\; \varepsilon^2 \cdot |V| - \varepsilon \cdot |V|/|U| \;\ge\; (\varepsilon^2/2) \cdot |V|$$

where the last inequality holds by the assumption $1/|U| \le \varepsilon/2$. Thus there must be some $u \neq u'$ such that $d_{u,u'}$ is at least this large. $\square$

3.3 Proof of Theorem 3

Restatement of Theorem 3. *For partial functions,* $\mathsf{RP}(q+1)^{\mathsf{cc}} \cap \mathsf{coRP}(q+1)^{\mathsf{cc}} \not\subseteq \mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}}$ *for every constant* q*.*

Fix any constant q. Let $\textsc{Which}\oplus_{q+1}\textsc{GapOr}\colon (\{0,1\}^{2n})^{2(q+1)} \to \{0,1\}$ be the following partial function: The input is divided into two halves $z = (z^0, z^1)$, and each half is divided into $q+1$ blocks $z^h = (z^h_1, \ldots, z^h_{q+1})$ having the promise that each $z^h_i \in \{0,1\}^{2n}$ is either all zeros or at least a *quarter* ones, and moreover it is promised that the number of nonzero blocks in z^0 has the opposite parity as the number of nonzero blocks in z^1 (call such an input *valid*). The partial function is defined by

$$\textsc{Which}\oplus_{q+1}\textsc{GapOr}(z) = \begin{cases} 1 & \text{if the number of nonzero blocks is odd in } z^0 \text{ and even in } z^1 \\ 0 & \text{if the number of nonzero blocks is even in } z^0 \text{ and odd in } z^1 \end{cases}.$$

We henceforth abbreviate $\textsc{Which}\oplus_{q+1}\textsc{GapOr}$ as f. Note that $\mathsf{RP}(q+1)^{\mathsf{dt}}(f) = O(1)$ by applying the $\mathsf{RP}(q+1)^{\mathsf{dt}}$ decision tree for $\oplus_{q+1}\textsc{GapOr}$ on z^0 (adapted for the different block length and different threshold for fraction of ones in a block). By symmetry (focusing on z^1), we also have $\mathsf{RP}(q+1)^{\mathsf{dt}}(\overline{f}) = O(1)$. Letting $g\colon [m] \times \{0,1\}^m \to \{0,1\}$ be the index gadget with $m \coloneqq N^{20}$ where $N \coloneqq 4(q+1)n$, this implies that

$$\mathsf{RP}(q+1)^{\mathsf{cc}}(f \circ g^N) = O(\log n) \qquad \text{and} \qquad \mathsf{coRP}(q+1)^{\mathsf{cc}}(f \circ g^N) = O(\log n)$$

(by the "easy direction" of $\mathsf{RP}(q+1)$ lifting) and thus $f \circ g^N \in \mathsf{RP}(q+1)^{\mathsf{cc}} \cap \mathsf{coRP}(q+1)^{\mathsf{cc}}$. We will now prove that $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{dt}}(f) = \Omega(n)$, which by Theorem 4.*(ii)* implies that $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}}(f \circ g^N) = \Omega(n \log n)$.

We show this by reduction from the fact that $\mathsf{NP}(q)^{\mathsf{dt}}(\overline{\oplus_q\textsc{GapOr}}) = \Omega(n)$. We henceforth abbreviate $\overline{\oplus_q\textsc{GapOr}}$ as f', and as in Section 3.1 we assume f' has block length n and threshold half. Supposing f has a $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{dt}}$ decision tree T of cost $k \le n/2$, say with deterministic phase T_{det}, we will use it to construct an $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree T' of cost at most k for f'.

By Lemma 16 we may assume that each leaf of T_{det} produces a single $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree and chooses whether to output the same or opposite answer as that decision tree. Follow the root-to-leaf path in T_{det} where all queries are answered with zero. Let $\rho \in (\{0,*\}^{2n})^{2(q+1)}$ be the partial assignment with at most $k \le n/2$ zeros that records these queries (so an input leads to this leaf iff it is consistent with ρ). Let $T_{\text{leaf}} = (\Phi_1, \ldots, \Phi_q)$ be the $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree of cost at most k produced at this leaf, where each Φ_i is a DNF. In the following, we assume that this leaf chooses to output the same answer as T_{leaf}. If instead the leaf chooses the opposite answer, simply swap the roles of z^0 and z^1 in the remainder of the proof.

Given any valid input z' to f', we show how to map it to a valid input z to f such that *(i)* $f'(z') = f(z)$, *(ii)* z is consistent with ρ, and *(iii)* each bit of z either is fixed or is some preselected bit of z'. Since *(iii)* implies that $T_{\text{leaf}}(z)$ can be viewed as an $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree $T'(z')$ by substituting a constant or variable of z' in for each variable of z (which does not increase the width of any conjunction), and T' would correctly compute f' since

$$f'(z') \;=\; f(z) \;=\; T(z) \;=\; T_{\text{leaf}}(z) \;=\; T'(z')$$

by *(i)*, correctness of T, *(ii)*, and *(iii)* respectively, this would show that $\mathsf{NP}(q)^{\mathsf{dt}}(f') \le k \le n/2$, which we know is false from Section 3.1.

To define z, we start with ρ (that is, we place zeros everywhere ρ requires, thus ensuring *(ii)*). Since ρ has at most n zeros in each block (indeed, at most $n/2$ zeros total), we can then place more zeros in such a way that each block now has exactly n zeros and n stars. Next we replace the stars in z^0_{q+1} with ones and replace the stars in z^1_{q+1} with zeros. Finally, for each $i \in [q]$, we fill in the stars of z^0_i with a copy of z'_i and fill in the stars of z^1_i with another copy of z'_i. This construction satisfies *(iii)*.

To verify *(i)*, first observe that since each block of z' is either all zeros or at least half ($n/2$) ones, this ensures each block of z is either all zeros or at least a quarter ($2n/4$) ones. Furthermore, if z' has exactly ℓ nonzero blocks then the number of nonzero blocks is $\ell+1$ in z^0 (since z^0_{q+1} is nonzero) and ℓ in z^1 (since z^1_{q+1} is all zeros). Hence if $f'(z') = 1$ (ℓ is even) then $f(z) = 1$ (since $\ell+1$ is odd and ℓ is even), and if

$f'(z') = 0$ (ℓ is odd) then $f(z) = 0$ (since $\ell + 1$ is even and ℓ is odd). Thus $f'(z') = f(z)$, and this finishes the proof of Theorem 3.

4 Total function collapse

Restatement of Theorem 2. *For total functions,*

$$\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}} = \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}} \qquad \textit{and} \qquad \mathsf{P}_{\parallel}^{\mathsf{RP}[q]\mathsf{cc}} = \mathsf{RP}(q+1)^{\mathsf{cc}} \cap \mathsf{coRP}(q+1)^{\mathsf{cc}}$$

for every constant q.

We start with the intuition for the nondeterministic case of Theorem 2. This is proved in a way similar to the specific result for $q = 0$ (that is, for total functions, $\mathsf{P}^{\mathsf{cc}} = \mathsf{NP}^{\mathsf{cc}} \cap \mathsf{coNP}^{\mathsf{cc}}$). In that proof, Alice and Bob can use the fact that the rectangles in the $\mathsf{NP}^{\mathsf{cc}}$ protocol's 1-monochromatic covering of F are disjoint from the rectangles in the $\mathsf{coNP}^{\mathsf{cc}}$ protocol's 0-monochromatic covering. Specifically, if $F(x, y) = 1$, then (x, y) is in some 1-rectangle, which is row-disjoint or column-disjoint from each 0-rectangle. (If a 1-rectangle and 0-rectangle shared a row and a column, they would intersect, which is not possible for a total function.) Therefore, Alice and Bob can repeatedly eliminate from consideration at least half of the remaining 0-rectangles, by identifying a 1-rectangle that either has x in its row set but is row-disjoint from at least half the remaining 0-rectangles, or has y in its column set but is column-disjoint from at least half the remaining 0-rectangles. If (x, y) is indeed in a 1-rectangle, then this process can always continue until there are no 0-rectangles left. If (x, y) is in a 0-rectangle, then this process will never eliminate that rectangle, so the process will halt with a nonempty set of 0-rectangles.

We repeat a similar argument, but using the "top level" of the $\mathsf{NP}(q+1)^{\mathsf{cc}}$ and $\mathsf{coNP}(q+1)^{\mathsf{cc}}$ protocols for F as our monochromatic rectangle sets. Here we think of a $\mathsf{coNP}(q+1)^{\mathsf{cc}}$ protocol as computing F by applying $\overline{\Delta}_{q+1}$ (with negations pushed to the leaves) to the indicators for $q+1$ rectangle unions. Depending on the parity of q, the rectangle union $\mathcal{R}_{q+1}$ queried at depth 1 of the $\mathsf{NP}(q+1)^{\mathsf{cc}}$ protocol will correspond to either 1-monochromatic rectangles or 0-monochromatic rectangles for F. The rectangle union $\mathcal{R}'_{q+1}$ queried at depth 1 of the $\mathsf{coNP}(q+1)^{\mathsf{cc}}$ protocol will be the opposite color of monochromatic rectangles. Crucially, this means that no input is in a rectangle from both of these sets (as we are assuming F is total). See Figure 3 for an illustration.

A key observation is that a deterministic protocol similar to the one used in the $q = 0$ case, ran using these top-level rectangle sets, will return the correct answer *under the promise that (x, y) is in one of these rectangles*. Say, for example, that (x, y) is in some rectangle in the 1-monochromatic top-level set. Then the deterministic protocol will successfully eliminate all 0-rectangles from the other top-level set, and will announce that the answer is 1. If (x, y) was in one of the 0-rectangles, then that rectangle will never be eliminated, and so the protocol would announce that the answer is 0.

If (x, y) is in the top-level rectangle union for one of the protocols, then (x, y) is not in the top-level rectangle union of the *other* protocol, so $F(x, y)$ can be computed by the other protocol but where the top level is skipped (resulting in only q many $\mathsf{NP}^{\mathsf{cc}}$ oracle queries). This boils down to the observation that $\Delta_{q+1}(z_1, \ldots, z_q, 0) = \Delta_q(z_1, \ldots, z_q)$.

What if (x, y) is in neither top-level rectangle union? Then we can make no guarantees about the behavior of the deterministic protocol—it might answer 0 or 1 (which we interpret as merely a "guess" for $F(x, y)$). However, in this case *both* protocols correctly compute $F(x, y)$ even if the top level is skipped. Therefore, we will still get the correct answer no matter which guess is produced by the deterministic protocol.

What follows is the formal proof.

Proof of Theorem 2 for the nondeterministic case. We already know $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}} \subseteq \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}}$ [HR90]. Let $F \colon \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ be a two-party total function with $\max\{\mathsf{NP}(q+1)^{\mathsf{cc}}(F), \mathsf{coNP}(q+1)^{\mathsf{cc}}(F)\} = k$. Consider any $\mathsf{NP}(q+1)^{\mathsf{cc}}$ protocol for F with cost at most k, say $\Pi = (\mathcal{R}_1, \ldots, \mathcal{R}_{q+1})$ where each $\mathcal{R}_i$ is a nonempty set of rectangles. Consider any $\mathsf{coNP}(q+1)^{\mathsf{cc}}$ protocol for F with cost at most k, say $\Pi' = (\mathcal{R}'_1, \ldots, \mathcal{R}'_{q+1})$ meaning that $F(x, y) = \overline{\Delta}_{q+1}(\mathcal{R}'_1(x, y), \ldots, \mathcal{R}'_{q+1}(x, y))$.

Out of the two protocols Π and Π', let $\Pi^{\wedge} = (\mathcal{R}^{\wedge}_1, \ldots, \mathcal{R}^{\wedge}_{q+1})$ be whichever one has $\wedge$ as its root gate, and let $\Pi^{\vee} = (\mathcal{R}^{\vee}_1, \ldots, \mathcal{R}^{\vee}_{q+1})$ be whichever one has $\vee$ as its root gate (after pushing negations to the leaves

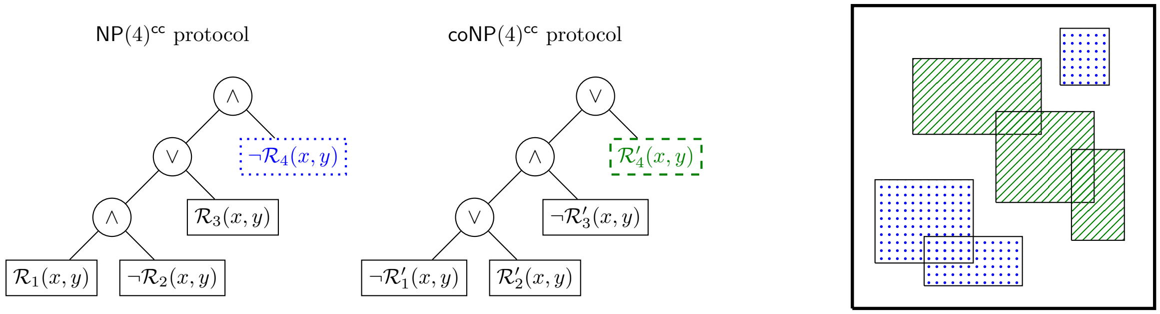

Figure 3: If a total function has an $\mathsf{NP}(4)^{\mathsf{cc}}$ protocol and a $\mathsf{coNP}(4)^{\mathsf{cc}}$ protocol, then the rectangle unions from the $\mathsf{NP}^{\mathsf{cc}}$ functions at depth one of each protocol are disjoint.

in Π'). Similarly, out of the two functions Δ_q and $\overline{\Delta}_q$, let $\Delta_q^\wedge$ be whichever one has $\wedge$ as its root gate, and let $\Delta_q^\vee$ be whichever one has $\vee$ as its root gate. In other words:

	$\Pi^\wedge$	$\Pi^\vee$	$\Delta_q^\wedge$	$\Delta_q^\vee$
if q is odd	Π	Π'	$\overline{\Delta}_q$	Δ_q
if q is even	Π'	Π	Δ_q	$\overline{\Delta}_q$

We claim that Algorithm 1 is a $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}}$ protocol with cost $O(k^2)$ that correctly computes F.

Claim 18. *Algorithm 1 is a $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}}$ protocol with cost $O(k^2)$.*

Proof of Claim 18. Guess is a deterministic protocol. In each iteration of the loop, either at least half of the remaining rectangles in $\mathcal{R}^0$ get removed, or the loop terminates. Since $|\mathcal{R}^0| \leq 2^k$ at the beginning, there can be at most $k+1$ iterations. Each iteration involves up to $k+1$ bits of communication, to specify one of the at most 2^k rectangles in $\mathcal{R}^1$ (or indicate the non-existence of a suitable rectangle in $\mathcal{R}^1$). Thus Guess has communication cost $O(k^2)$. Also, the q many nonadaptive $\mathsf{NP}^{\mathsf{cc}}$ oracle queries on line 14 or line 15 contribute at most k to the cost since they are just part of Π or Π', so the overall cost is still $O(k^2)$. $\square$

Claim 19. *If $\mathcal{R}_{q+1}^\wedge(x,y) = 1$ then* Guess $= 0$*. If $\mathcal{R}_{q+1}^\vee(x,y) = 1$ then* Guess $= 1$.

Proof of Claim 19. Assume $\mathcal{R}_{q+1}^\wedge(x,y) = 1$, so $(x,y) \in R$ for some $R \in \mathcal{R}_{q+1}^\wedge$. Note that Guess only removes a rectangle from $\mathcal{R}^0$ when it is certain that (x,y) is not in that rectangle (because x is not in its row set or y is not in its column set). Thus R will never be removed from $\mathcal{R}^0$ throughout Guess, which means $\mathcal{R}^0$ will never be empty, and Guess will eventually return 0.

Assume $\mathcal{R}_{q+1}^\vee(x,y) = 1$, so $(x,y) \in R'$ for some $R' \in \mathcal{R}_{q+1}^\vee$. Observe that each rectangle in $\mathcal{R}_{q+1}^\wedge$ is 0-monochromatic for F (since $\Pi^\wedge$ outputs 0 on such inputs) and each rectangle in $\mathcal{R}_{q+1}^\vee$ is 1-monochromatic for F (since $\Pi^\vee$ outputs 1 on such inputs). Since F is total, R' is disjoint—and hence either row-disjoint or column-disjoint—from each rectangle $R \in \mathcal{R}_{q+1}^\wedge$ (since R' contains only 1-inputs and R contains only 0-inputs). Thus in each iteration of the loop, either at least half the remaining rectangles in $\mathcal{R}^0$ are row-disjoint from $R' \in \mathcal{R}^1$ (which Alice would notice) or at least half are column-disjoint from R' (which Bob would notice). Either way, the loop will continue with one party announcing a rectangle (not necessarily R') and shrinking $\mathcal{R}^0$, which means the loop will not halt until $\mathcal{R}^0 = \emptyset$, and Guess will return 1. $\square$

Claim 20. *For all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, Algorithm 1 correctly computes $F(x,y)$.*

Proof of Claim 20. If Guess $= 0$ then $\mathcal{R}_{q+1}^\vee(x,y) = 0$ by Claim 19, and thus $F(x,y) = \Pi^\vee(x,y) = \Delta_q^\wedge(\mathcal{R}_1^\vee(x,y), \ldots, \mathcal{R}_q^\vee(x,y)) \vee \mathcal{R}_{q+1}^\vee(x,y)$ equals the output on line 14. If Guess $= 1$ then $\mathcal{R}_{q+1}^\wedge(x,y) = 0$ by Claim 19, and thus $F(x,y) = \Pi^\wedge(x,y) = \Delta_q^\vee(\mathcal{R}_1^\wedge(x,y), \ldots, \mathcal{R}_q^\wedge(x,y)) \wedge \neg\mathcal{R}_{q+1}^\wedge(x,y)$ equals the output on line 15. $\square$

Together, Claim 18 and Claim 20 show that Algorithm 1 witnesses $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}}(F) \leq O(k^2)$. This completes the proof that for total functions, $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}} = \mathsf{NP}(q+1)^{\mathsf{cc}} \cap \mathsf{coNP}(q+1)^{\mathsf{cc}}$. $\square$

Algorithm 1 $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}}$ protocol for F

function GUESS
  $\mathcal{R}^0 \leftarrow \mathcal{R}^{\wedge}_{q+1}$ (set of 0-monochromatic rectangles)
  $\mathcal{R}^1 \leftarrow \mathcal{R}^{\vee}_{q+1}$ (set of 1-monochromatic rectangles)
  loop
    if $\mathcal{R}_0 = \emptyset$ **then return** 1
    else if there exists an $R \in \mathcal{R}_1$ whose row set contains x and which is row-disjoint from at least half of the rectangles in $\mathcal{R}_0$ **then**
      Alice announces such an R
      remove from $\mathcal{R}_0$ all rectangles that are row-disjoint from R
    else if there exists an $R \in \mathcal{R}_1$ whose column set contains y and which is column-disjoint from at least half of the rectangles in $\mathcal{R}_0$ **then**
      Bob announces such an R
      remove from $\mathcal{R}_0$ all rectangles that are column-disjoint from R
    else return 0

function MAIN
  if GUESS $= 0$ **then return** $\Delta^{\wedge}_q(\mathcal{R}^{\vee}_1(x,y), \ldots, \mathcal{R}^{\vee}_q(x,y))$
  if GUESS $= 1$ **then return** $\Delta^{\vee}_q(\mathcal{R}^{\wedge}_1(x,y), \ldots, \mathcal{R}^{\wedge}_q(x,y))$

The corresponding argument for the Randomized Boolean Hierarchy is very similar. Note that since $\mathsf{RP}^{\mathsf{cc}} \subseteq \mathsf{NP}^{\mathsf{cc}}$, we can simply interpret the $(q+1)^{\text{st}}$ components of our $\mathsf{RP}(q+1)^{\mathsf{cc}}$ and $\mathsf{coRP}(q+1)^{\mathsf{cc}}$ protocols as $\mathsf{NP}^{\mathsf{cc}}$ protocols. Then Algorithm 1 would work in exactly the same way, except instead of using $\mathcal{R}^{\vee}_1, \ldots, \mathcal{R}^{\vee}_q$ or $\mathcal{R}^{\wedge}_1, \ldots, \mathcal{R}^{\wedge}_q$ as the oracle queries at the end, these would be replaced by $\mathsf{RP}^{\mathsf{cc}}$ protocols (the first q components of our $\mathsf{RP}(q+1)^{\mathsf{cc}}$ or $\mathsf{coRP}(q+1)^{\mathsf{cc}}$ protocols).

We remark that a similar approach shows that $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{dt}} = \mathsf{NP}(q+1)^{\mathsf{dt}} \cap \mathsf{coNP}(q+1)^{\mathsf{dt}}$ for total functions. (The corresponding result for randomized query complexity follows anyway from $\mathsf{P}^{\mathsf{dt}} = \mathsf{BPP}^{\mathsf{dt}}$ [Nis91].)

5 Query-to-communication lifting for $\mathsf{NP}(q)$

Restatement of Theorem 4.*(i)*. *For every partial function $f\colon \{0,1\}^n \to \{0,1\}$ and every constant q,*

$$\mathsf{NP}(q)^{\mathsf{cc}}(f \circ g^n) \;=\; \mathsf{NP}(q)^{\mathsf{dt}}(f) \cdot \Theta(\log n)$$

where $g\colon [m] \times \{0,1\}^m \to \{0,1\}$ is the index gadget defined by $g(x,y) = y_x$ with $m \coloneqq n^{20}$.

The big-O direction follows immediately from the same fact for NP: for every f, $\mathsf{NP}^{\mathsf{cc}}(f \circ g^n) = \mathsf{NP}^{\mathsf{dt}}(f) \cdot O(\log n)$ holds by replacing each of the $n^{O(k)}$ conjunctions in a width-k DNF with m^k rectangles (each of which contains inputs where the gadget outputs satisfy the conjunction), for a total of $n^{O(k)} m^k = 2^{k \cdot O(\log n)}$ rectangles. In the rest of this section we prove the big-Ω direction. By Lemma 13 it suffices to show

$$\mathsf{DL}(q)^{\mathsf{cc}}(f \circ g^n) \;=\; \mathsf{DL}(q)^{\mathsf{dt}}(f) \cdot \Omega(\log n).$$

5.1 Technical preliminaries

Our proof is closely related to the P^{NP} lifting theorem of Göös, Kamath, Pitassi, and Watson [GKPW19], so we start by recalling some definitions and lemmas that were used in that work. We will need to tweak some of the statements and parameters, though.

Define $G\colon [m]^n \times (\{0,1\}^m)^n \to \{0,1\}^n$ as $G \coloneqq g^n$. This partitions the input domain into 2^n slices $G^{-1}(z) = \{(x,y) : g(x_i, y_i) = z_i \text{ for all } i \in [n]\}$, one for each $z \in \{0,1\}^n$. For a set $Z \subseteq \{0,1\}^n$, let $G^{-1}(Z) \coloneqq \bigcup_{z \in Z} G^{-1}(z)$.

Consider sets $A \subseteq [m]^n$ and $B \subseteq (\{0,1\}^m)^n$. For $I \subseteq [n]$, we let $A_I \coloneqq \{x_I : x \in A\}$ and $B_I \coloneqq \{y_I : y \in B\}$ be the *projections* onto the coordinates of I. The *min-entropy* of a random variable $\boldsymbol{x}$ is

$\mathbf{H}_\infty(\boldsymbol{x}) \coloneqq \min_x \log(1/\Pr[\boldsymbol{x} = x])$. We say A is *δ-dense* if the uniform random variable $\boldsymbol{x}$ over A satisfies the following: for every nonempty $I \subseteq [n]$, $\mathbf{H}_\infty(\boldsymbol{x}_I) \geq \delta|I|\log m$ (that is, the min-entropy of the marginal distribution of $\boldsymbol{x}$ on coordinates I is at least a δ fraction of the maximum possible for a distribution over $[m]^I$). The *deficiency* of B is $\mathbf{D}_\infty(B) \coloneqq mn - \log|B|$.

Lemma 21 ([GKPW19, Lemma 11])**.** *If $A \subseteq [m]^n$ is 0.8-dense and $B \subseteq (\{0,1\}^m)^n$ has deficiency at most n^4, then $G(A \times B) = \{0,1\}^n$, that is, for every $z \in \{0,1\}^n$ there are $x \in A$ and $y \in B$ with $G(x, y) = z$.*

Here the density parameter is $\delta = 0.8$ and the deficiency is $\mathbf{D}_\infty(B) \leq n^4$, instead of $\delta = 0.9$ and $\mathbf{D}_\infty(B) \leq n^2$ as in the original. Lemma 21 still holds because our gadget size has increased: we use $m \coloneqq n^{20}$, whereas [GKPW19] used $m \coloneqq n^4$. This can be verified by a simple substitution in the proof.

The next lemma is altered enough from the original that we will reprove it here.

Lemma 22 (A more general version of [GKPW19, Claim 12])**.** *Let $\mathcal{X} \subseteq [m]^n$ be 0.85-dense. If $A' \subseteq \mathcal{X}$ satisfies $|A'| \geq |\mathcal{X}|/2^{k+1}$ then there exist an $I \subseteq [n]$ of size $|I| < 20(k+1)/\log m$ and an $A \subseteq A'$ such that A is fixed on coordinates I and $A_{[n]\smallsetminus I}$ is 0.8-dense.*

The original version was for the special case $\mathcal{X} = [m]^n$. We observe that it is sufficient for $\mathcal{X}$ to be δ-dense, for some suitable constant δ greater than the desired density of the resulting set A (here we use $\delta = 0.85$).

Proof of Lemma 22. If A' is already 0.8-dense, then we can simply take $I \coloneqq \emptyset$ and $A \coloneqq A'$, so assume otherwise. Let $I \subseteq [n]$ be a maximal set of coordinates that violates 0.8-density: for the uniform random variable $\boldsymbol{x}$ over A', $\mathbf{H}_\infty(\boldsymbol{x}_I) < 0.8|I|\log m$. Since the uniform random variable $\boldsymbol{X}$ over $\mathcal{X}$ has $\mathbf{H}_\infty(\boldsymbol{X}_I) \geq 0.85|I|\log m$, and $|A'| \geq |\mathcal{X}|/2^{k+1}$, we also have $\mathbf{H}_\infty(\boldsymbol{x}_I) \geq 0.85|I|\log m - (k+1)$. Combining these, we get that $k+1 > 0.05|I|\log m$, which implies that $|I| < 20(k+1)/\log m$.

We can now simply choose some $\alpha \in [m]^I$ such that $\Pr[\boldsymbol{x}_I = \alpha] > 2^{0.8|I|\log m}$, and take $A \coloneqq \{x \in A' : x_I = \alpha\}$. This certainly satisfies that A is fixed on coordinates I. To see that A is 0.8-dense on all other coordinates, assume for contradiction there is some nonempty $J \subseteq [n] \smallsetminus I$ witnessing a 0.8-density violation of A. Then it is straightforward to check that $I \cup J$ would be a set of coordinates witnessing a 0.8-density violation of A', which contradicts the maximality of I. ☐

5.2 The simulation

We exhibit an algorithm that takes a q-alternating rectangle decision list $\mathcal{L}_\mathrm{R}$ for $f \circ g^n$ of cost k, and converts it to a q-alternating conjunction decision list $\mathcal{L}_\mathrm{C}$ for f of cost $O(k/\log n)$. The argument from [GKPW19] does exactly this except without preserving the bound on the number of alternations. In [GKPW19] the argument is formulated using a "dual" characterization of $\mathsf{DL}^{\mathsf{dt}}$, but it has the effect of building $\mathcal{L}_\mathrm{C}$ in order, obtaining each conjunction by "extracting" it from one of the rectangles in $\mathcal{L}_\mathrm{R}$. The trouble is that the rectangles are not necessarily "extracted from" in order: after extracting a conjunction from some rectangle, the *next* conjunction that gets put in $\mathcal{L}_\mathrm{C}$ may be extracted from a rectangle that is *earlier* in $\mathcal{L}_\mathrm{R}$. Thus $\mathcal{L}_\mathrm{C}$ may end up with more alternations than $\mathcal{L}_\mathrm{R}$.

To fix this, we convert the argument to a "primal" form and argue that it still works when we force the rectangles to be extracted from in order. The high-level view is that we iterate through the rectangles of $\mathcal{L}_\mathrm{R}$ in order, and for each we extract as many conjunctions as we can until the rectangle becomes "exhausted", at which time we remove the remaining "error" portion of the rectangle (by deleting few rows and columns) and move on to the next rectangle. With this modification, the rest of the technical details from [GKPW19] continue to work, and it now preserves the number of alternations.

At any step of this process, we let $X \times Y$ be the remaining rows and columns (after having removed the error portion of all previous rectangles in $\mathcal{L}_\mathrm{R}$), and we let $Z \subseteq \{0,1\}^n$ be the remaining inputs to f (which have not been accepted by any previous conjunctions we put in $\mathcal{L}_\mathrm{C}$). Suppose (R_i, ℓ_i) is our current entry in $\mathcal{L}_\mathrm{R}$. The goal is to find a subrectangle $A \times B \subseteq R_i \cap (X \times Y)$ that is "conjunction-like" in the sense that $G(A \times B)$ is exactly the inputs accepted by some small-width conjunction C, and such that among all remaining inputs $z \in Z$, C only accepts those with $f(z) = \oplus(\ell_i)$. These properties would ensure it is safe to put (C, ℓ_i) next in $\mathcal{L}_\mathrm{C}$.

Algorithm 2 Simulation algorithm

In: $\mathcal{L}_{\mathrm{R}} = (R_1, \ell_1), \ldots, (R_{2^k}, \ell_{2^k})$ and $\mathcal{X} \subseteq [m]^n$, $\mathcal{Y} \subseteq (\{0,1\}^m)^n$
Out: $\mathcal{L}_{\mathrm{C}}$

1: initialize $X \leftarrow \mathcal{X}$, $Y \leftarrow \mathcal{Y}$, $Z \leftarrow$ domain of f, $\mathcal{L}_{\mathrm{C}} \leftarrow$ empty list
2: **for** $i = 1$ to 2^k **do**
3:     **while** $Z \neq \emptyset$ **do**
4:         for each $x \in X$, let $Y_x := \{y \in Y : (x, y) \in R_i \cap G^{-1}(Z)\}$
5:         let $A' := \{x \in X : |Y_x| \geq 2^{mn-n^4}\}$
6:         **if** $|A'| \leq |\mathcal{X}|/2^{k+1}$ **then**
7:             update $X \leftarrow X \smallsetminus A'$
8:             update $Y \leftarrow Y \smallsetminus \bigcup_{x \in X \smallsetminus A'} Y_x$
9:             break out of inner loop
10:        **else** $|A'| > |\mathcal{X}|/2^{k+1}$
11:            let $A \subseteq A'$, $I \subseteq [n]$, $\alpha \in [m]^I$ be such that:
12:                 $|I| = O(k/\log n)$, A_I is fixed to α, and $A_{[n]\smallsetminus I}$ is 0.8-dense
13:            pick any $x' \in A$ and choose $\beta \in (\{0,1\}^m)^I$ to maximize the size of $B := \{y \in Y_{x'} : y_I = \beta\}$
14:            let C be the conjunction "$z_I = g^I(\alpha, \beta)$"
15:            update $\mathcal{L}_{\mathrm{C}}$ by appending (C, ℓ_i) to it
16:            update $Z \leftarrow Z \smallsetminus C^{-1}(1)$

Combining Lemma 21 and Lemma 22 (using $\mathcal{X} = [m]^n$) suggests an approach for finding a conjunction-like subrectangle: If A' is not too small, we can restrict it to A that is fixed on few coordinates I and dense on the rest (by Lemma 22). If B is also not too small (low deficiency) and fixed on coordinates I, then $G(A \times B)$ is fixed on I and takes on all possible values on the remaining coordinates (by Lemma 21, which still works with $[n] \smallsetminus I$ in place of $[n]$). In other words, $G(A \times B) = C^{-1}(1)$ for a small-width conjunction C, as desired.

The other property we needed to ensure is that if this C is the first conjunction in $\mathcal{L}_{\mathrm{C}}$ to accept a particular z, then $f(z) = \oplus(\ell_i)$ (so $\mathcal{L}_{\mathrm{C}}$ is correct). This will follow if we know there is some $(x, y) \in G^{-1}(z)$ such that R_i is the first rectangle in $\mathcal{L}_{\mathrm{R}}$ to contain (x, y), as that guarantees $f(z) = f(G(x, y)) = (f \circ g^n)(x, y) = \oplus(\ell_i)$. It turns out this will hold automatically if $A \times B \subseteq R_i \cap (X \times Y)$, because $A \times B$ touches the slice of every z that is accepted by C, and all inputs $(x, y) \in G^{-1}(Z)$ that were in some R_j with $j < i$ have already been removed from $X \times Y$.

Our algorithm for building $\mathcal{L}_{\mathrm{C}}$ from $\mathcal{L}_{\mathrm{R}}$ is shown in Algorithm 2. It is described as starting from some arbitrary initial rectangle $\mathcal{X} \times \mathcal{Y}$. For the purpose of proving Theorem 4.*(i)* we only need to take $\mathcal{X} = [m]^n$ and $\mathcal{Y} = (\{0,1\}^m)^n$, but when we invoke this as a component in the proof of Theorem 4.*(ii)* we will need to start from some $\mathcal{X} \times \mathcal{Y}$ that is merely "dense $\times$ large" rather than the full input domain, so we state this more general version now.

Lemma 23. *If $\mathcal{L}_{\mathrm{R}}$ computes $f \circ g^n$ on $\mathcal{X} \times \mathcal{Y}$ and has cost k, and if $\mathcal{X}$ is 0.85-dense and $\mathbf{D}_\infty(\mathcal{Y}) \leq n^3$, then $\mathcal{L}_{\mathrm{C}}$ produced by Algorithm 2 computes f and has cost $O(k/\log n)$. Moreover, if $\mathcal{L}_{\mathrm{R}}$ is q-alternating then so is $\mathcal{L}_{\mathrm{C}}$.*

Proof. To verify the cost, just note that lines 11 and 12 always succeed by Lemma 22 (since $\mathcal{X}$ is 0.85-dense and $|A'| \geq |\mathcal{X}|/2^{k+1}$), so when a conjunction is added to $\mathcal{L}_{\mathrm{C}}$ on lines 14 and 15, it has width $|I| < 20(k+1)/\log m = O(k/\log n)$. On line 13 we have $|B| \geq |Y_{x'}|/2^{m|I|} \geq 2^{mn-n^4-m|I|} = 2^{m(n-|I|)-n^4}$ (since $x' \in A \subseteq A'$) and therefore $\mathbf{D}_\infty(B_{[n]\smallsetminus I}) \leq n^4$ (relative to $(\{0,1\}^m)^{[n]\smallsetminus I}$). Thus by applying Lemma 21 to $A_{[n]\smallsetminus I}$ (which is 0.8-dense) and $B_{[n]\smallsetminus I}$ we have $g^{[n]\smallsetminus I}(A_{[n]\smallsetminus I} \times B_{[n]\smallsetminus I}) = \{0,1\}^{n-|I|}$ and therefore $G(A \times B) = C^{-1}(1)$. (Lemma 21 works with the same parameters even though the sets are now on fewer than n coordinates.)

The algorithm terminates because Z always shrinks on line 16: for any $y \in B$ we have $G(x', y) \in Z$ (from the definition of $Y_{x'}$) and $C(G(x', y)) = 1$ (since $x'_I = \alpha$ and $y_I = \beta$ and thus $G(x', y)_I = g^I(\alpha, \beta)$).

The algorithm maintains the invariant that for all $j < i$, $R_j \cap (X \times Y) \cap G^{-1}(Z) = \emptyset$. This vacuously holds at the beginning, and is clearly maintained in the **else** case because i stays the same and nothing gets

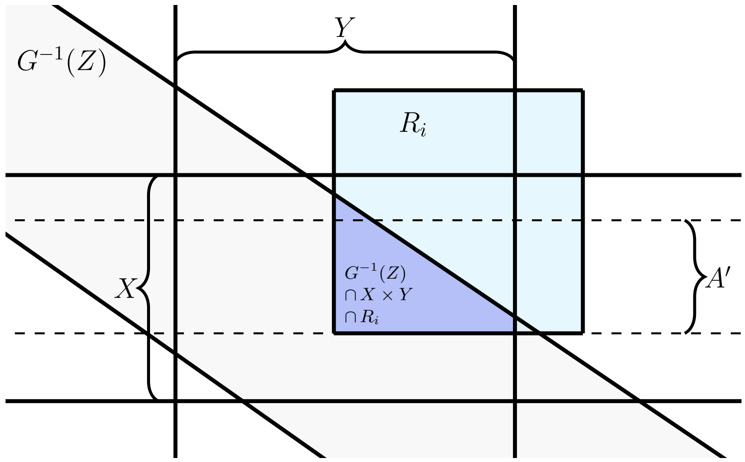

Figure 4: A visualization of a step of Algorithm 2. In this example, A' is large, and so lines 10–16 will be executed.

added to X, Y, or Z. Lines 7 and 8 maintain the invariant in the **if** case because the removed rows and columns cover all of $R_i \cap (X \times Y) \cap G^{-1}(Z)$ and i goes up by 1.

Next we argue that when the algorithm terminates, Z must be empty. In each iteration of the outer loop, we throw out at most $|\mathcal{X}|/2^{k+1}$ rows and at most $|\mathcal{X}| \cdot 2^{mn-n^4} \leq m^n \cdot 2^{mn-n^4} \leq 2^{mn-n^3}/2^{k+1} \leq |\mathcal{Y}|/2^{k+1}$ columns. (We throw out columns in Y_x for $x \notin A'$, all of these Y_x had the property $|Y_x| < 2^{mn-n^4}$, we do this for at most $|\mathcal{X}|$ values of x, and $n^4 - n\log m \geq n^3 + k + 1$.) Since the outer loop executes 2^k times, by the end at most half the rows of $\mathcal{X}$ and half the columns of $\mathcal{Y}$ have been discarded, so $|X| \geq |\mathcal{X}|/2$ and $|Y| \geq |\mathcal{Y}|/2$. This means X is essentially as dense as $\mathcal{X}$ (only a -1 loss in any $\mathbf{H}_\infty(\boldsymbol{x}_I)$) and Y is essentially as low-deficiency as $\mathcal{Y}$ (only a $+1$ loss in $\mathbf{D}_\infty$). Thus Lemma 21 (with a tiny perturbation of the parameters, which does not affect the result) shows that $G(X \times Y) = \{0,1\}^n$. However, the last rectangle that is processed, R_{2^k}, contains all of $\mathcal{X} \times \mathcal{Y}$ by definition (since we assume $\mathcal{L}_\mathrm{R}$ is correct on $\mathcal{X} \times \mathcal{Y}$). So, the invariant guarantees $(X \times Y) \cap G^{-1}(Z) = \emptyset$ at termination. This can only happen if $G^{-1}(Z) = \emptyset$ and thus $Z = \emptyset$ (since $G(X \times Y) = \{0,1\}^n$).

We now argue that $\mathcal{L}_\mathrm{C}$ is correct. Consider any z in the domain of f. Since Z is empty at termination, z must be accepted by some conjunction in $\mathcal{L}_\mathrm{C}$. Let (C, ℓ_i) be the first entry such that $C(z) = 1$, so $z \in Z$ during the iteration of the inner loop when this entry was added. Since in this iteration we have $G(A \times B) = C^{-1}(1)$ and $z \in C^{-1}(1)$, there is some $(x,y) \in A \times B$ with $G(x,y) = z$. Since $A \times B \subseteq R_i$ we have $(x,y) \in R_i$. Since $A \times B \subseteq X \times Y$, we have $(x,y) \in (X \times Y) \cap G^{-1}(Z)$ and thus (x,y) cannot be in R_j for any $j < i$ since $R_j \cap (X \times Y) \cap G^{-1}(Z) = \emptyset$ by the invariant. In summary, R_i is the first rectangle in $\mathcal{L}_\mathrm{R}$ that contains (x,y). By correctness of $\mathcal{L}_\mathrm{R}$ on $\mathcal{X} \times \mathcal{Y}$, we have $\oplus(\ell_i) = (f \circ g^n)(x,y) = f(G(x,y)) = f(z)$. Thus $\mathcal{L}_\mathrm{C}$ also correctly outputs $\oplus(\ell_i)$ on input z.

The "moreover" part is straightforward to verify: the levels assigned to conjunctions in $\mathcal{L}_\mathrm{C}$ come from the levels assigned to rectangles in $\mathcal{L}_\mathrm{R}$ (namely $\{0, \ldots, q\}$), in the same order (which is non-increasing). $\square$

6 Query-to-communication lifting for $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]}$

Restatement of Theorem 4.*(ii)*. *For every partial function $f\colon \{0,1\}^n \to \{0,1\}$ and every constant q,*

$$\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{cc}}(f \circ g^n) \;=\; \mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{dt}}(f) \cdot \Theta(\log n)$$

where $g\colon [m] \times \{0,1\}^m \to \{0,1\}$ is the index gadget defined by $g(x,y) = y_x$ with $m \coloneqq n^{20}$.

For the big-O direction, the deterministic phase of a $\mathsf{P}_{\parallel}^{\mathsf{NP}[q]\mathsf{dt}}$ decision tree can be simulated by a protocol that communicates $\log m + 1 = O(\log n)$ bits to evaluate $g(x_i, y_i)$ whenever the decision tree queries the i^{th}

bit of the input to f. The $\mathsf{NP}^{\mathsf{dt}}$ oracle queries can also be converted to $\mathsf{NP}^{\mathsf{cc}}$ oracle queries with $O(\log n)$ factor overhead in their contribution to the cost (as was the case for Theorem 4.*(i)*). In the rest of this section we prove the big-Ω direction. By Lemma 16 it suffices to show

$$\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{cc}}(f \circ g^n) \;=\; \mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{dt}}(f) \cdot \Omega(\log n).$$

6.1 Technical preliminaries

The high-level idea is to convert a $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{cc}}$ protocol for $f \circ g^n$ into a $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{dt}}$ decision tree for f by using the P lifting theorem of Raz and McKenzie [RM99, GPW18a] to handle the deterministic phase, followed by our $\mathsf{NP}(q)$ lifting theorem to handle the single $\mathsf{NP}(q)$ oracle query. To get these components to mesh, we need to review some technical concepts from the deterministic lifting theorem.

We say that $A \subseteq [m]^n$ is δ-*thick* if A is nonempty and for every $i \in [n]$ and every $\alpha_1 \cdots \alpha_{i-1}\alpha_{i+1} \cdots \alpha_n \in [m]^{[n] \smallsetminus \{i\}}$, if there is at least one value of $\alpha_i \in [m]$ for which the combined tuple α is in A, then there are at least m^δ many such values of α_i. Recall that for $A \subseteq [m]^n$, $B \subseteq (\{0,1\}^m)^n$, and $I \subseteq [n]$, $A_I \coloneqq \{x_I : x \in A\}$ and $B_I \coloneqq \{y_I : y \in B\}$ are the *projections* onto the coordinates of I. Also recall that the *deficiency* of B is $\mathbf{D}_\infty(B) \coloneqq mn - \log|B|$. For a partial assignment $\rho \in \{0,1,*\}^n$, let $\mathrm{free}(\rho) \coloneqq \rho^{-1}(*) \subseteq [n]$ denote its free coordinates and $\mathrm{fixed}(\rho) \coloneqq [n] \smallsetminus \mathrm{free}(\rho)$ denote its fixed coordinates. The rectangle $A \times B$ is called ρ-*consistent* if every assignment in $g^n(A \times B)$ is consistent with ρ.

The deterministic lifting theorem of [RM99, GPW18a] proves the following result.[1]

Lemma 24. *For every deterministic communication protocol Π_{det} of cost k (with the same input domain as g^n where g is the index gadget with $m \coloneqq n^{20}$), there exists a deterministic decision tree T_{det} of cost $\leq 40k/\log m$ (with input domain $\{0,1\}^n$) such that the following holds: For every leaf v of T_{det}, letting $\rho \in \{0,1,*\}^n$ be the partial assignment recording the results of the queries on the path to v (so $|\mathrm{fixed}(\rho)| \leq 40k/\log m$), there exists an associated leaf v' of Π_{det} and a rectangle $A \times B \subseteq [m]^n \times (\{0,1\}^m)^n$ such that:*

- *$A \times B$ is contained within the rectangle of v'.*
- *$A \times B$ is ρ-consistent.*
- *$A_{\mathrm{free}(\rho)}$ is 0.85-thick.*
- *$\mathbf{D}_\infty(B_{\mathrm{free}(\rho)}) \leq n^2$ (relative to $(\{0,1\}^m)^{\mathrm{free}(\rho)}$).*

The idea is to apply Lemma 24 to the deterministic phase of the $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{cc}}$ protocol, and then apply Lemma 23 to each of the leaves of the resulting deterministic decision tree. In order to do so, we need the following claim, which observes that 0.85-thickness implies 0.85-density.

Claim 25. *If $X \subseteq [m]^n$ is δ-thick then X is δ-dense.*

Proof. Let $\boldsymbol{x}$ be the uniform random variable over X. The basic intuition is that thickness tells us that for any single coordinate i, the probability that $\boldsymbol{x}_i$ takes on a particular value, given that all of the other coordinates are fixed, is at most $m^{-\delta}$. This implies that the probability that $\boldsymbol{x}_i$ takes on a particular value, conditioned on setting any subset of the other coordinates to some values, is still bounded by $m^{-\delta}$. Therefore, we can simply apply the chain rule to prove that thickness implies density.

Assume for notational simplicity that I is the first k coordinates, so $I = \{1, \dots, k\}$. Let $\alpha = \alpha_1 \cdots \alpha_k \in [m]^I$ be an assignment to the coordinates of I. By the chain rule,

$$\Pr[\boldsymbol{x}_I = \alpha_1 \cdots \alpha_k] \;=\; \Pr[\boldsymbol{x}_1 = \alpha_1] \cdot \Pr[\boldsymbol{x}_2 = \alpha_2 \mid \boldsymbol{x}_1 = \alpha_1] \cdot \ldots \cdot \Pr[\boldsymbol{x}_k = \alpha_k \mid \boldsymbol{x}_{I \smallsetminus \{k\}} = \alpha_1 \cdots \alpha_{k-1}]$$

and each conditioning is valid, assuming $\Pr[\boldsymbol{x}_I = \alpha] > 0$.

Thickness of X tells us that for any coordinate $i \in I$, any $\alpha_i \in [m]$, and any $\beta \in [m]^{[n] \smallsetminus \{i\}}$, we have $\Pr[\boldsymbol{x}_i = \alpha_i \mid \boldsymbol{x}_{[n] \smallsetminus \{i\}} = \beta] \leq m^{-\delta}$ if the conditioning is valid. Thus for any $I' \subseteq [n] \smallsetminus \{i\}$, and any assignment β' to the coordinates of I', by the law of total probability we have $\Pr[\boldsymbol{x}_i = \alpha_i \mid \boldsymbol{x}_{I'} = \beta'] \leq m^{-\delta}$ if the conditioning is valid. Therefore, each term in the chain rule expansion is at most $m^{-\delta}$. This gives $\Pr[\boldsymbol{x}_I = \alpha] \leq (m^{-\delta})^k = 2^{-\delta|I|\log m}$, which implies $\mathbf{H}_\infty(\boldsymbol{x}_I) \geq \delta|I|\log m$ since α was arbitrary. Since this holds for all nonempty subsets $I \subseteq [n]$, this means that X is δ-dense. ∎

[1] The exact statement of Lemma 24 does not appear in these papers, but an inspection of their proof reveals that the lemma indeed holds.

6.2 The simulation

Let Π be a $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{cc}}$ protocol for $f \circ g^n$ with cost k. We construct a $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{dt}}$ decision tree T for f with cost $O(k/\log n)$.

If $k = \Omega(n \log n)$, we can construct a deterministic decision tree for f with cost n that simply queries every bit in the input. This is a $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{dt}}$ decision tree for f (with a trivial $\mathsf{NP}(q)^{\mathsf{dt}}$ phase) with cost $O(k/\log n)$. In the following, assume that $k = o(n \log n)$.

Let Π_{det} be the deterministic phase of Π. Each leaf v of Π_{det} has an associated rectangle R_v and $\mathsf{NP}(q)^{\mathsf{cc}}$ protocol Π_v of cost $\leq k$ that computes either $f \circ g^n$ or its complement on inputs in R_v. Applying Lemma 24 to Π_{det}, which has communication cost $\leq k$, yields a deterministic decision tree T_{det} having query cost $\leq 40k/\log m = O(k/\log n)$, which will form the deterministic phase of T. Henceforth consider any leaf of T_{det}, say corresponding to partial assignment ρ, and from Lemma 24 let v be the associated leaf of Π_{det} and $A \times B \subseteq R_v$ be the associated rectangle. Assume without loss of generality that Π_v computes $f \circ g^n$ on R_v and hence on $A \times B$ (a symmetric argument handles the case where Π_v computes $\overline{f} \circ g^n$ on R_v). Abbreviate $\text{free}(\rho)$ as J, and let $n' \coloneqq |J|$. Since $|\text{fixed}(\rho)| \leq 40k/\log m \leq n/2$ (as we are assuming $k = o(n \log n)$), we have $n' \geq n/2$.

Since $A \times B$ is ρ-consistent, if we modify Π_v by intersecting each rectangle with $A \times B$ and then projecting to J, this yields an $\mathsf{NP}(q)^{\mathsf{cc}}$ protocol that correctly computes $f_\rho \circ g^J$ on $A_J \times B_J$, where $f_\rho \colon \{0,1\}^J \to \{0,1\}$ is the restriction of f to ρ. After converting this protocol to a q-alternating rectangle decision list of cost $O(k)$ by Lemma 13, we may apply Lemma 23 by substituting f_ρ for f, n' for n, A_J for $\mathcal{X}$ (since A_J is 0.85-thick), and B_J for $\mathcal{Y}$ (since $\mathbf{D}_\infty(B_J) \leq n^2 \leq (n')^3$),[2] to get a q-alternating conjunction decision list for f_ρ of cost $O(k/\log n') = O(k/\log n)$. By Lemma 13 again, there is an $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree T_v with cost $O(k/\log n)$ that correctly computes f_ρ. Therefore, we can complete T by having our arbitrary leaf of T_{det} use T_v as its oracle query and output the same answer. Thus T computes f and has total cost $O(k/\log n)$.

Acknowledgments

We thank Benjamin Rossman for helpful comments and discussions.

References

[AW17] Josh Alman and Ryan Williams. Probabilistic rank and matrix rigidity. In *Proceedings of the 49th Symposium on Theory of Computing (STOC)*, pages 641–652. ACM, 2017. doi:10.1145/3055399.3055484.

[BBJ+89] Alberto Bertoni, Danilo Bruschi, Deborah Joseph, Meera Sitharam, and Paul Young. Generalized Boolean hierarchies and Boolean hierarchies over RP. In *Proceedings of the 7th International Conference on Fundamentals of Computation Theory (FCT)*, pages 35–46. Springer, 1989. doi:10.1007/3-540-51498-8_4.

[Bei91] Richard Beigel. Bounded queries to SAT and the Boolean hierarchy. *Theoretical Computer Science*, 84(2):199–223, 1991. doi:10.1016/0304-3975(91)90160-4.

[BFS86] László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 337–347. IEEE, 1986. doi:10.1109/SFCS.1986.15.

[BNS89] László Babai, Noam Nisan, and Mario Szegedy. Multiparty protocols and logspace-hard pseudorandom sequences. In *Proceedings of the 21st Symposium on Theory of Computing (STOC)*, pages 1–11. ACM, 1989. doi:10.1145/73007.73008.

[BYJKS04] Ziv Bar-Yossef, T.S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004. doi:10.1016/j.jcss.2003.11.006.

[2] A subtlety is that Lemma 23 assumes $m \coloneqq (n')^{20}$ whereas here we have $m \coloneqq n^{20}$, but this tiny perturbation in parameters does not affect the result.

[CH86] Jin-Yi Cai and Lane Hemachandra. The Boolean hierarchy: Hardware over NP. In *Proceedings of the 1st Structure in Complexity Theory Conference (STRUCTURES)*, pages 105–124. Springer, 1986. doi:10.1007/3-540-16486-3_93.

[CLV19] Arkadev Chattopadhyay, Shachar Lovett, and Marc Vinyals. Equality alone does not simulate randomness. In *Proceedings of the 34th Computational Complexity Conference (CCC)*, pages 14:1–14:11. Schloss Dagstuhl, 2019. doi:10.4230/LIPIcs.CCC.2019.14.

[GKPW19] Mika Göös, Pritish Kamath, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for P^{NP}. *Computational Complexity*, 28(1):113–144, 2019. doi:10.1007/s00037-018-0175-5.

[GLM+16] 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.

[Göö15] Mika Göös. Lower bounds for clique vs. independent set. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1066–1076. IEEE, 2015. doi:10.1109/FOCS.2015.69.

[GPW16] Mika Göös, Toniann Pitassi, and Thomas Watson. Zero-information protocols and unambiguity in Arthur–Merlin communication. *Algorithmica*, 76(3):684–719, 2016. doi:10.1007/s00453-015-0104-9.

[GPW18a] 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.

[GPW18b] Mika Göös, Toniann Pitassi, and Thomas Watson. The landscape of communication complexity classes. *Computational Complexity*, 27(2):245–304, 2018. doi:10.1007/s00037-018-0166-6.

[GPW20] Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. *SIAM Journal on Computing*, 49(4):FOCS17–441–FOCS17–461, 2020. doi:10.1137/17M115339X.

[HR88] Bernd Halstenberg and Rüdiger Reischuk. Relations between communication complexity classes. In *Proceedings of the 3rd Structure in Complexity Theory Conference (STRUCTURES)*, pages 19–28. IEEE, 1988. doi:10.1109/SCT.1988.5259.

[HR90] Bernd Halstenberg and Rüdiger Reischuk. Relations between communication complexity classes. *Journal of Computer and System Sciences*, 41(3):402–429, 1990. doi:10.1016/0022-0000(90)90027-I.

[HR93] Bernd Halstenberg and Rüdiger Reischuk. Different modes of communication. *SIAM Journal on Computing*, 22(5):913–934, 1993. doi:10.1137/0222057.

[Juk12] Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*, volume 27 of *Algorithms and Combinatorics*. Springer, 2012. doi:10.1007/978-3-642-24508-4.

[Kad88] Jim Kadin. The polynomial time hierarchy collapses if the Boolean hierarchy collapses. *SIAM Journal on Computing*, 17(6):1263–1282, 1988. doi:10.1137/0217080.

[Kla03] Hartmut Klauck. Rectangle size bounds and threshold covers in communication complexity. In *Proceedings of the 18th Conference on Computational Complexity (CCC)*, pages 118–134. IEEE, 2003. doi:10.1109/CCC.2003.1214415.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[KS92] Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992. doi:10.1137/0405044.

[KSW87] Johannes Köbler, Uwe Schöning, and Klaus Wagner. The difference and truth-table hierarchies for NP. *Theoretical Informatics and Applications*, 21(4):419–435, 1987. doi:10.1051/ita/1987210404191.

[LPS19] Nati Linial, Toniann Pitassi, and Adi Shraibman. On the communication complexity of high-dimensional permutations. In *Proceedings of the 10th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 54:1–54:20. Schloss Dagstuhl, 2019. doi:10.4230/LIPIcs.ITCS.2019.54.

[Nis91] Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991. doi:10.1137/0220062.

[PSS14] Periklis Papakonstantinou, Dominik Scheder, and Hao Song. Overlays and limited memory communication. In *Proceedings of the 29th Conference on Computational Complexity (CCC)*, pages 298–308. IEEE, 2014. doi:10.1109/CCC.2014.37.

[Raz89] Alexander Razborov. On rigid matrices. Technical report, Steklov Mathematical Institute, 1989. In Russian.

[Raz92] Alexander Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992. doi:10.1016/0304-3975(92)90260-M.

[Riv87] Ronald Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987. doi:10.1007/BF00058680.

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

[Wag88] Klaus Wagner. Bounded query computations. In *Proceedings of the 3rd Structure in Complexity Theory Conference (STRUCTURES)*, pages 260–277. IEEE, 1988. doi:10.1109/SCT.1988.5286.

[Wat20] Thomas Watson. A $\mathrm{ZPP}^{\mathrm{NP}[1]}$ lifting theorem. *ACM Transactions on Computation Theory*, 12(4):27:1–27:20, 2020. doi:10.1145/3428673.

[Wec85] Gerd Wechsung. On the Boolean closure of NP. In *Proceedings of the 5th International Conference on Fundamentals of Computation Theory (FCT)*, pages 485–493. Springer, 1985. doi:10.1007/BFb0028832.

[Yao79] Andrew Yao. Some complexity questions related to distributive computing. In *Proceedings of the 11th Symposium on Theory of Computing (STOC)*, pages 209–213. ACM, 1979. doi:10.1145/800135.804414.

[ZH86] Stathis Zachos and Hans Heller. A decisive characterization of BPP. *Information and Control*, 69(1-3):125–135, 1986. doi:10.1016/S0019-9958(86)80044-4.

A Proofs from Section 2

In this appendix we prove the technical lemmas from Section 2, which provide alternative characterizations of the classes in the Nondeterministic and Randomized Boolean Hierarchies.

A.1 Decision list vs. parity

Restatement of Lemma 8. *For $\mathcal{C} \in \{\mathsf{NP}, \mathsf{RP}\}$, if the definitions of $\mathcal{C}(q)^{\mathsf{cc}}$ and $\mathcal{C}(q)^{\mathsf{dt}}$ are changed to use $\oplus_q$ in place of Δ_q, it only affects the complexity measures $\mathcal{C}(q)^{\mathsf{cc}}(F)$ and $\mathcal{C}(q)^{\mathsf{dt}}(f)$ by a constant factor (depending on q).*

Wagner [Wag88] showed that these alternative characterizations are equivalent for the classical Nondeterministic Boolean Hierarchy, and Halstenberg and Reischuk [HR90] observed the same (up to a constant factor) in communication complexity. This latter proof uses only the property that $\mathsf{NP}^{\mathsf{cc}}$ is closed under intersection and union; that is, if $\mathsf{NP}^{\mathsf{cc}}(F_1), \mathsf{NP}^{\mathsf{cc}}(F_2) \leq k$, then $\mathsf{NP}^{\mathsf{cc}}(F_1 \wedge F_2)$ and $\mathsf{NP}^{\mathsf{cc}}(F_1 \vee F_2)$ are both $O(k)$. We observe that since this property also holds for $\mathsf{RP}^{\mathsf{cc}}$, $\mathsf{NP}^{\mathsf{dt}}$, and $\mathsf{RP}^{\mathsf{dt}}$, their proof works for these models of computation as well. In fact, in all of these models, the cost of the intersection or union of i cost-k computations is at most ik.

Proof of Lemma 8. We prove this using the language of communication complexity. The query complexity proof is completely analogous.

For transforming Δ_q to $\oplus_q$, the idea is to observe that $\Delta_q = \oplus_q \circ h$ where $h\colon \{0,1\}^q \to \{0,1\}^q$ is defined by $h(z_1, \ldots, z_q) = (z_1 \vee \cdots \vee z_q,\ z_2 \vee \cdots \vee z_q,\ \ldots,\ z_{q-1} \vee z_q,\ z_q)$. In other words, given a protocol $\Pi = (\Pi_1, \ldots, \Pi_q)$ of cost k that computes $\Delta_q(F_1, \ldots, F_q)$, we can transform it into an equivalent protocol $\Pi' = (\Pi'_1, \ldots, \Pi'_q)$ that computes $\oplus_q(F'_1, \ldots, F'_q)$ where F'_i is the function "does there exist a $j \geq i$ for which F_j outputs 1?". This works because if i is the greatest index such that F_i outputs 1, then Π outputs $\oplus(i)$ and Π' also outputs $\oplus(i)$ since there are exactly i many values of j such that F'_j outputs 1, namely $1, \ldots, i$. For the implementation of Π'_i: If $\mathcal{C} = \mathsf{NP}$ then a witness for Π'_i can specify a $j \geq i$ together with a witness for Π_j, incurring a cost of $O(k)$; thus the cost of Π' is $O(qk) = O(k)$. If $\mathcal{C} = \mathsf{RP}$ then Π'_i can run Π_j for every $j \geq i$, each amplified to have error probability $\leq 1/2q$, and see whether at least one of them outputs 1, incurring a cost of $O(k \log q)$; thus the cost of Π' is $O(qk \log q) = O(k)$.

For transforming $\oplus_q$ to Δ_q, the idea is to observe that $\oplus_q = \Delta_q \circ h$ where $h\colon \{0,1\}^q \to \{0,1\}^q$ is defined by $h(z) = (\mathrm{thr}(z,1), \mathrm{thr}(z,2), \ldots, \mathrm{thr}(z,q))$ where $\mathrm{thr}(z,i)$ indicates whether the Hamming weight of z is at least i. In other words, given a protocol $\Pi = (\Pi_1, \ldots, \Pi_q)$ of cost k that computes $\oplus_q(F_1, \ldots, F_q)$, we can transform it into an equivalent protocol $\Pi' = (\Pi'_1, \ldots, \Pi'_q)$ that computes $\Delta_q(F'_1, \ldots, F'_q)$ where F'_i is the function "are there at least i many values of j for which F_j outputs 1?" This works because if there are exactly i values of j for which F_j outputs 1, then Π outputs $\oplus(i)$ and Π' also outputs $\oplus(i)$ since i is the greatest index such that F'_i outputs 1. For the implementation of Π'_i: If $\mathcal{C} = \mathsf{NP}$ then a witness for Π'_i can specify a set of i values of j together with a witness for each of those Π_j, incurring a cost of $O(k)$; thus the cost of Π' is $O(qk) = O(k)$. If $\mathcal{C} = \mathsf{RP}$ then Π'_i can run every Π_j, each amplified to have error probability $\leq 1/2q$, and see whether at least i of them output 1, incurring a cost of $O(k \log q)$; thus the cost of Π' is $O(qk \log q) = O(k)$. ☐

A.2 Boolean Hierarchy vs. decision list

Restatement of Lemma 13. *$\mathsf{DL}(q)^{\mathsf{cc}}(F) = \Theta(\mathsf{NP}(q)^{\mathsf{cc}}(F))$ and $\mathsf{DL}(q)^{\mathsf{dt}}(f) = \Theta(\mathsf{NP}(q)^{\mathsf{dt}}(f))$ for every constant q. Thus, $\mathsf{DL}(q)^{\mathsf{cc}} = \mathsf{NP}(q)^{\mathsf{cc}}$ and $\mathsf{DL}(q)^{\mathsf{dt}} = \mathsf{NP}(q)^{\mathsf{dt}}$ for partial functions.*

Proof. To see that $\mathsf{DL}(q)^{\mathsf{cc}}(F) \leq \mathsf{NP}(q)^{\mathsf{cc}}(F)$, consider any $\mathsf{NP}(q)^{\mathsf{cc}}$ protocol for F with cost k, say $\Pi = (\mathcal{R}_1, \ldots, \mathcal{R}_q)$ where each $\mathcal{R}_i$ is a nonempty set of rectangles. To form a q-alternating rectangle decision list, let level q be the rectangles of $\mathcal{R}_q$ in any order, then level $q-1$ be the rectangles of $\mathcal{R}_{q-1}$ in any order, and so on, and finally let level 0 be the rectangle containing all inputs. This has the same output as Π, so it correctly computes F. Assuming $\mathcal{R}_i$ has cost k_i, the length of the list is $\sum_i |\mathcal{R}_i| + 1 \leq \prod_i (|\mathcal{R}_i| + 1) \leq \prod_i 2^{k_i} = 2^k$, so the cost is at most k.

To see that $\mathsf{NP}(q)^{\mathsf{cc}}(F) = O(\mathsf{DL}(q)^{\mathsf{cc}}(F))$, consider any q-alternating rectangle decision list $\mathcal{L}_{\mathrm{R}}$ for F with cost k. To form an $\mathsf{NP}(q)^{\mathsf{cc}}$ protocol $\Pi = (\mathcal{R}_1, \ldots, \mathcal{R}_q)$, for each i let $\mathcal{R}_i$ be the set of rectangles at level i in $\mathcal{L}_{\mathrm{R}}$. This has the same output as $\mathcal{L}_{\mathrm{R}}$, so it correctly computes F. Since $|\mathcal{R}_i| \leq 2^k - 1$ for each i, the cost of Π is at most $\sum_i k = qk$.

To see that $\mathsf{DL}(q)^{\mathsf{dt}}(f) \leq \mathsf{NP}(q)^{\mathsf{dt}}(f)$, consider any $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree for f with cost k, say $T = (\Phi_1, \ldots, \Phi_q)$ where each Φ_i is a DNF. To form a q-alternating conjunction decision list, let level q be the conjunctions of Φ_q in any order, then level $q-1$ be the conjunctions of Φ_{q-1} in any order, and so on, and finally let level 0 be the conjunction that accepts all inputs. This has the same output as T, so it correctly computes f. Since every Φ_i has maximum width at most k, the list also has cost at most k.

To see that $\mathsf{NP}(q)^{\mathsf{dt}}(f) = O(\mathsf{DL}(q)^{\mathsf{dt}}(f))$, consider any q-alternating conjunction decision list $\mathcal{L}_{\mathrm{C}}$ for f with cost k. To form an $\mathsf{NP}(q)^{\mathsf{dt}}$ decision tree $T = (\Phi_1, \ldots, \Phi_q)$, for each i let Φ_i be the disjunction of all conjunctions at level i in $\mathcal{L}_{\mathrm{C}}$. This has the same output as $\mathcal{L}_{\mathrm{C}}$, so it correctly computes f. Since each Φ_i has maximum width at most k, the cost of T is at most $\sum_i k = qk$. ☐

A.3 Leaves with parallel queries

Restatement of Lemma 16. *For $\mathcal{C} \in \{\mathsf{NP}, \mathsf{RP}\}$, we have $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{cc}}(F) = \Theta(\mathsf{P}^{\mathcal{C}(q)[1]\mathsf{cc}}(F))$ and $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{dt}}(f) = \Theta(\mathsf{P}^{\mathcal{C}(q)[1]\mathsf{dt}}(f))$ for every constant q.*

Proof. It is trivial that $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{cc}}(F) \le \mathsf{P}^{\mathcal{C}(q)[1]\mathsf{cc}}(F)$ and $\mathsf{P}_{\|}^{\mathcal{C}[q]\mathsf{dt}}(f) \le \mathsf{P}^{\mathcal{C}(q)[1]\mathsf{dt}}(f)$ since one "$\mathcal{C}(q)$ oracle query" can be expressed with q nonadaptive "$\mathcal{C}$ oracle queries". For the other direction, we just show the argument for $\mathsf{NP}^{\mathsf{cc}}$, but essentially the same argument works for $\mathsf{RP}^{\mathsf{cc}}$, $\mathsf{NP}^{\mathsf{dt}}$, and $\mathsf{RP}^{\mathsf{dt}}$.

The main idea of this proof is to decompose the arbitrary out function at any leaf of a protocol into Δ_q applied to some functions which have low nondeterministic complexity. Consider a $\mathsf{P}_{\|}^{\mathsf{NP}[q]\mathsf{cc}}$ protocol Π for F of cost k. We convert it to a $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{cc}}$ protocol for F of cost $O(k)$. The deterministic phase Π_{det} stays the same. Henceforth fix any leaf of Π_{det} and its associated output function $\mathsf{out}\colon \{0,1\}^q \to \{0,1\}$ and q nonempty sets of rectangles $\mathcal{R}_1, \ldots, \mathcal{R}_q$. Also fix any input (x, y) that reaches this leaf, and define $w \in \{0,1\}^q$ by $w_i = \mathcal{R}_i(x, y)$ for each $i \in [q]$, so that $\mathsf{out}(w) = \Pi(x, y) = F(x, y)$.

For $u, v \in \{0,1\}^q$ we say $u \le v$ iff $u_i \le v_i$ for each index $i \in [q]$. Consider the q-dimensional *Hamming cube DAG*: this is the graph where the set of nodes is $\{0,1\}^q$ and there is a directed edge from u to v iff $u \le v$ and u and v only differ by one index. Each node v has a corresponding output value $\mathsf{out}(v)$, and on any directed path $v^1 \to v^2 \to \cdots \to v^j$, we say there is a *mind-change* on the path at node v^i iff $\mathsf{out}(v^i) \ne \mathsf{out}(v^{i-1})$. For any directed path in the Hamming cube DAG from 0^q to v, the number of mind-changes on the path is even if $\mathsf{out}(v) = \mathsf{out}(0^q)$ and odd if $\mathsf{out}(v) \ne \mathsf{out}(0^q)$. Thus, letting $\mathsf{mmc}(v)$ be the maximum number of mind-changes on any directed path from 0^q to v, we have $\oplus(\mathsf{mmc}(v))$ is the indicator for $\mathsf{out}(v) \ne \mathsf{out}(0^q)$.

Since $\mathsf{out}(0^q)$ is fixed at the leaf, it is sufficient for Alice and Bob to determine $\oplus(\mathsf{mmc}(w))$ in order to compute $\mathsf{out}(w) = F(x, y)$.

Claim 26. *For fixed $i \in [q]$, the cost of an $\mathsf{NP}^{\mathsf{cc}}$ protocol that determines "is $\mathsf{mmc}(w) \ge i$?" (given input (x, y) that reaches the leaf) is at most $q + k$.*

Before we prove Claim 26, we use it to finish the proof of Lemma 16. At the leaf, for each $i \in [q]$ we form an $\mathsf{NP}^{\mathsf{cc}}$ protocol of cost $\le q + k$ for determining if $\mathsf{mmc}(w) \ge i$, and we use these to form a single $\mathsf{NP}(q)^{\mathsf{cc}}$ oracle query. We output the same answer as the oracle if $\mathsf{out}(0^q) = 0$, and the opposite answer if $\mathsf{out}(0^q) = 1$; this is correct because $\mathsf{mmc}(w)$ is the maximum value of i for which the i^{th} $\mathsf{NP}^{\mathsf{cc}}$ protocol would output 1, and thus the oracle query returns $\oplus(\mathsf{mmc}(w))$, which is the indicator for $\mathsf{out}(w) \ne \mathsf{out}(0^q)$. The $\mathsf{NP}(q)^{\mathsf{cc}}$ oracle query has cost $\le q(q + k) = O(k)$. Thus the overall $\mathsf{P}^{\mathsf{NP}(q)[1]\mathsf{cc}}$ protocol for F has cost $O(k)$, and this concludes the proof of Lemma 16. ☐

Proof of Claim 26. The idea is to find some (possibly not proper) prefix of a directed path from 0^q to w in the Hamming cube DAG, where this prefix has i mind-changes. If we can find such a prefix, it confirms that some 0^q-to-w directed path has at least i mind-changes, and therefore $\mathsf{mmc}(w) \ge i$.

The witness itself is some $v \in \{0,1\}^q$, along with witnesses of $\mathcal{R}_j(x, y) = 1$ for every j where $v_j = 1$. This can be represented by simply giving a rectangle R' such that $(x, y) \in R'$ and R' is the intersection of (Hamming weight of v many) rectangles, one from each $\mathcal{R}_j$ with $v_j = 1$. The number of possibilities for R' is at most $\prod_j |\mathcal{R}_j| \le 2^k$, so R' contributes $\le k$ to the cost of the $\mathsf{NP}^{\mathsf{cc}}$ protocol. Alice and Bob can verify the witness by checking that there exists a 0^q-to-v path with i mind-changes (this does not depend on the input) and that $(x, y) \in R'$ and thus $v \le w$. ☐

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il