2019

Optimal Separation And Strong Direct Sum For Randomized Query Complexity

E. Blais

Joshua Brody
Swarthmore College, brody@cs.swarthmore.edu

Recommended Citation

E. Blais and Joshua Brody. (2019). "Optimal Separation And Strong Direct Sum For Randomized Query Complexity". *34th Computational Complexity Conference (CCC 2019)*. Volume 137, DOI: 10.4230/LIPIcs.CCC.2019.29
https://works.swarthmore.edu/fac-comp-sci/107

Optimal Separation and Strong Direct Sum for Randomized Query Complexity

Eric Blais
University of Waterloo, ON, Canada
eric.blais@uwaterloo.ca

Joshua Brody
Swarthmore College, PA, USA
brody@cs.swarthmore.edu

Abstract

We establish two results regarding the query complexity of bounded-error randomized algorithms.

Bounded-error separation theorem. There exists a total function $f : \{0,1\}^n \to \{0,1\}$ whose ϵ-error randomized query complexity satisfies $\overline{\mathrm{R}}_\varepsilon(f) = \Omega(\mathrm{R}(f) \cdot \log \frac{1}{\epsilon})$.

Strong direct sum theorem. For every function f and every $k \geq 2$, the randomized query complexity of computing k instances of f simultaneously satisfies $\overline{\mathrm{R}}_\varepsilon(f^k) = \Theta(k \cdot \overline{\mathrm{R}}_{\frac{\varepsilon}{k}}(f))$.

As a consequence of our two main results, we obtain an optimal superlinear direct-sum-type theorem for randomized query complexity: there exists a function f for which $\mathrm{R}(f^k) = \Theta(k \log k \cdot \mathrm{R}(f))$. This answers an open question of Drucker (2012). Combining this result with the query-to-communication complexity lifting theorem of Göös, Pitassi, and Watson (2017), this also shows that there is a total function whose public-coin randomized communication complexity satisfies $\mathrm{R}^{\mathrm{cc}}(f^k) = \Theta(k \log k \cdot \mathrm{R}^{\mathrm{cc}}(f))$, answering a question of Feder, Kushilevitz, Naor, and Nisan (1995).

2012 ACM Subject Classification Theory of computation → Probabilistic computation; Theory of computation → Oracles and decision trees

Keywords and phrases Decision trees, query complexity, communication complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.29

Acknowledgements The first author thanks Alexander Belov and Shalev Ben-David for enlightening discussions and helpful suggestions. The second author thanks Peter Winkler for insightful discussions. Both authors wish to thank the anonymous referees for valuable feedback and for the reference to [26].

1 Introduction

We consider two fundamental questions related to the query complexity of functions in the bounded-error randomized setting: how the randomized query complexity of total functions scales with the allowable error ϵ (the *separation* problem), and how the query complexity of computing k instances of a function scales with the complexity of computing only 1 instance of the same function (the *direct sum* problem). Standard folklore arguments give upper bounds on how much the randomized query complexity can depend on ϵ and on k in these two problems; the results described below show that these well-known upper bounds are tight in general.

A randomized algorithm $\mathcal{A}$ *computes* a function $f : \mathcal{X}^n \to \{0,1\}$ over a finite set $\mathcal{X}^n$ *with error* $\epsilon \geq 0$ if for every input $x \in \mathcal{X}^n$, the algorithm outputs the value $f(x)$ with probability at least $1 - \epsilon$. The *query cost* of $\mathcal{A}$ is the maximum number of coordinates of x that it queries, with the maximum taken over both the choice of input x and the internal randomness of $\mathcal{A}$. The *ϵ-error (worst-case) randomized query complexity* of f (also known as the *randomized decision tree complexity* of f) is the minimum query complexity of an

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 29; pp. 29:1–29:17

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

algorithm $\mathcal{A}$ that computes f with error at most ϵ. We denote this complexity by $\mathrm{R}_\epsilon(f)$, and we write $\mathrm{R}(f) := \mathrm{R}_{\frac{1}{3}}(f)$ to denote the $\frac{1}{3}$-error randomized query complexity of f.

Another natural measure for the query cost of a randomized algorithm $\mathcal{A}$ is the *expected* number of coordinates of an input x that it queries. Taking the maximum expected number of coordinates queried by $\mathcal{A}$ over all inputs yields the *average query cost* of $\mathcal{A}$. The minimum average query complexity of an algorithm $\mathcal{A}$ that computes a function f with error at most ϵ is the *average ϵ-error query complexity* of f, which we denote by $\overline{\mathrm{R}}_\epsilon(f)$. We again write $\overline{\mathrm{R}}(f) := \overline{\mathrm{R}}_{\frac{1}{3}}(f)$. Note that $\overline{\mathrm{R}}_0(f)$ corresponds to the standard notion of *zero-error randomized query complexity* of f.

1.1 Our Results

Bounded-Error Separation Theorem for Query Complexity

One of the first tricks that one learns in the study of randomized algorithm is *success amplification*: it is possible to cheaply reduce the error of a randomized algorithm from $\frac{1}{3}$ to any $\varepsilon > 0$ by running the algorithm $O(\log \frac{1}{\varepsilon})$ times and outputting the most frequent answer. In the context of randomized query complexity, this means that for every function $f : \{0,1\}^n \to \{0,1\}$,

$$\mathrm{R}_\varepsilon(f) = O\big(\mathrm{R}(f) \cdot \log \tfrac{1}{\varepsilon}\big). \tag{1}$$

When considering partial functions, it is easy to see that the success amplification trick is optimal, as there are partial functions for which this relationship is tight (see Section 2.2). However, in the case of total functions, for many natural functions such as the majority function, parity function, dictator function, etc., the stronger bound $\mathrm{R}_\epsilon(f) = O\big(\mathrm{R}(f)\big)$ holds and until now it was not known whether the bound in (1) is tight for *any* total function. In fact, even separations between zero-error and $\frac{1}{3}$-error randomized query complexity were not known until very recently, when Ambainis et al. [2] showed that there exists a total function f for which $\overline{\mathrm{R}}_0(f) = \widetilde{\Omega}(\mathrm{R}(f)^2)$. Similarly, other separations between randomized query complexity and other measures of complexity have also only been established very recently [23, 1, 3, 4, 2].

In this work, we give the first separation within the bounded-error randomized query complexity setting. Our separation shows that the bound in (1) is optimal in general.

▶ **Theorem 1.** *For infinitely many values of n and every $2^{-(\frac{n}{\log n})^{1/3}} < \varepsilon \leq \frac{1}{3}$, there exists a total function $f : \{0,1\}^n \to \{0,1\}$ with randomized query complexity*

$$\overline{\mathrm{R}}_\varepsilon(f) = \Omega\big(\mathrm{R}(f) \cdot \log \tfrac{1}{\varepsilon}\big).$$

Note that by the trivial relation $\overline{\mathrm{R}}_\varepsilon(f) \leq \mathrm{R}_\varepsilon(f)$ between average and worst-case randomized query complexity, Theorem 1 implies the existence of a function f for which $\mathrm{R}_\varepsilon(f) \geq \Omega\big(\mathrm{R}(f) \cdot \log \frac{1}{\varepsilon}\big)$ and $\overline{\mathrm{R}}_\varepsilon(f) \geq \Omega\big(\overline{\mathrm{R}}(f) \cdot \log \frac{1}{\varepsilon}\big)$, giving optimal separations in both the worst-case randomized query complexity and average query complexity settings.

Strong Direct Sum Theorem

The *direct sum problem* asks how the cost of computing a function f scales with the number k of instances of the function that we need to compute. This problem has received a considerable amount of attention in the context of query complexity [18, 7, 24, 25, 19, 8, 13], communication complexity [20, 14, 11, 5, 21, 6], and beyond.

Given a function $f : \{0,1\}^n \to \{0,1\}$ and a parameter $k \geq 2$, define $f^k : \{0,1\}^{n \cdot k} \to \{0,1\}^k$ by setting $f^k(x^{(1)}, \ldots, x^{(k)}) = \big(f(x^{(1)}), \ldots, f(x^{(k)})\big)$. A simple union bound argument shows that the randomized query complexity of f^k is bounded above by

$$R_\epsilon(f^k) = O\big(k \cdot R_{\frac{\epsilon}{k}}(f)\big) \tag{2}$$

since we can run a randomized algorithm $\mathcal{A}$ that computes f with error at most $\frac{\epsilon}{k}$ on each of the k instances. An analogous upper bound holds in the average query complexity setting as well.

Jain, Klauck, and Santha [19] first considered the problem of showing a direct sum theorem for randomized query complexity. They showed that for every function f and for small enough constant $\delta > 0$, $\mathrm{R}_\epsilon(f^k) \geq \delta^2 k \cdot \mathrm{R}_{\frac{\epsilon}{1-\delta}+\delta}(f)$. Note that in this inequality, the allowable error on the right-hand side of the equation is *larger* than the ϵ error parameter, in contrast to the upper bound where it is (much) smaller. Ben-David and Kothari [8] obtained an improved direct sum theorem holds, showing that $\overline{\mathrm{R}}_\epsilon(f^k) \geq k \cdot \overline{\mathrm{R}}_\epsilon(f)$ holds for every function. This result is formally stronger since it implies the Jain–Klauck–Santha bound, but it also does not show that the error parameter on the right-hand-side of the inequality needs to be smaller than ϵ, as it is in the upper bound (2).

We show that the bound in (2) is tight in the average-case query complexity model.

▶ **Theorem 2.** *For every function $f : \{0,1\}^n \to \{0,1\}$, every $k \geq 2$, and every $0 \leq \epsilon \leq \frac{1}{20}$,*

$$\overline{\mathrm{R}}_\epsilon(f^k) = \Omega\left(k \cdot \overline{\mathrm{R}}_{\frac{\epsilon}{k}}(f)\right).$$

We establish Theorem 2 by proving a corresponding strong direct sum theorem in the distributional setting, as we discuss in more details in Section 1.3. It remains open to determine whether a similar strong direct sum theorem holds in the worst-case randomized query complexity model. However, in that setting Shaltiel [25] has shown that a proof of such a direct sum theorem *can't* be obtained via a corresponding theorem in the distributional setting, as a counterexample shows that direct sum theorems do not hold in this setting in general.

1.2 Applications

Superlinear Direct-Sum-Type Theorem for Query Complexity

Combining (1) and (2), we obtain a bound on the cost of computing k instances of a function f with bounded (constant) error and the cost of computing a single instance of the same function:

$$\mathrm{R}(f^k) = O\big(k \log k \cdot \mathrm{R}(f)\big). \tag{3}$$

Drucker [13, Open problem 2] asked if the superlinear dependence on k in (3) is necessary for any total function f. Theorems 1 and 2 give a positive answer to this question.

▶ **Corollary 3.** *There exists a total function $f : \{0,1\}^n \to \{0,1\}$ such that for all $1 \leq k \leq 2^{(\frac{n}{\log n})^{1/3}}$,*

$$\mathrm{R}(f^k) = \Theta\big(k \log k \cdot \mathrm{R}(f)\big).$$

Note that Corollary 3 stands in contrast to the quantum query complexity setting, where such a superlinear dependence on k is not required [10].

Superlinear Direct-Sum-Type Theorem for Communication Complexity

Let $\mathrm{R}^{\mathrm{cc}}(f)$ denote the minimum amount of communication required of a public-coin randomized protocol that computes a function $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}$ with error at most $\frac{1}{3}$. As in the query complexity model, the communication complexity of the function f^k is bounded above by

$$\mathrm{R}^{\mathrm{cc}}(f^k) = O\big(k \log k \cdot \mathrm{R}^{\mathrm{cc}}(f)\big). \tag{4}$$

Feder, Kushilevitz, Naor, and Nisan [14] showed that this upper bound is not tight in general, as the equality function satisfies $\mathrm{R}^{\mathrm{cc}}(\mathrm{EQ}^k) = O\big(k \cdot \mathrm{R}^{\mathrm{cc}}(\mathrm{EQ})\big)$.[1] They then asked whether $\mathrm{R}^{\mathrm{cc}}(f^k) = O(k \cdot \mathrm{R}^{\mathrm{cc}}(f))$ holds for all functions or not [14, Open problem 2 in §7].

In the last few years, there has been much work on related direct sum questions. Molinaro, Woodruff, and Yaroslavtsev [21, 22] showed that in the *one-way* communication complexity model, the equality function does satisfy the superlinear direct sum bound $\mathrm{R}^{\mathrm{cc},\to}(\mathrm{EQ}^k) = \Theta\big(k \log k \cdot \mathrm{R}^{\mathrm{cc},\to}(\mathrm{EQ})\big)$. In the two-way communication complexity model that we consider, Barak, Braverman, Chen, and Rao [6] showed that every function f satisfies the direct sum $R(f^k) = \widetilde{\Omega}\big(\sqrt{k}\, R(f)\big)$, and this bound remains the state of the art as far as we know. Using the connection between information complexity and amortized communication complexity of Braverman and Rao [9], Ganor, Kol, and Raz [15] also showed that there is a partial function whose distributional communication complexity is exponentially larger than its amortized distributional communication complexity, showing that a tight direct sum theorem cannot hold in general in this setting. None of these results, however, answer Feder et al.'s original question.

Corollary 3 combined with the randomized query-to-communication lifting theorem of Göös, Pitassi, and Watson [17] answers Feder et al.'s question by showing that there is a function f for which the bound in (4) is tight.

▶ **Corollary 4.** *There is a constant $c > 0$ and a total function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ such that for all $1 \le k \le 2^{n^c}$,*

$$\mathrm{R}^{\mathrm{cc}}(f^k) = \Theta\big(k \log k \cdot \mathrm{R}^{\mathrm{cc}}(f)\big).$$

1.3 Proof Overviews

Bounded-Error Separation Theorem

The proof of Theorem 1 is established by following the general approach used to great effect by Ambainis et al. [2]: first, identify a partial function f for which the query complexity separation holds, then design a variant of the Göös–Pitassi–Watson (GPW) pointer function [16] that "embeds" the partial function into a total function and preserves the same separation.

The first step in this plan is accomplished by observing that the partial *gap identity* function $\textsc{GapID} : \{0,1\}^m \to \{0,1,*\}$ defined by

$$\textsc{GapID}(x) = \begin{cases} 1 & \text{if } |x| = 0, \\ 0 & \text{if } |x| = \lfloor \frac{m}{2} \rfloor, \\ * & \text{otherwise} \end{cases}$$

satisfies $\overline{\mathrm{R}}_{\varepsilon}(\textsc{GapID}) = \Theta\big(\mathrm{R}(\textsc{GapID}) \cdot \log \frac{1}{\varepsilon}\big)$ for every $\epsilon \ge 2^{-m}$.

[1] In fact, Feder et al. showed that the *private-coin* randomized communication complexity of EQ satisfies the stronger relation $\mathrm{R}^{\mathrm{cc,priv}}(\mathrm{EQ}^k) = o\big(k \cdot \mathrm{R}^{\mathrm{cc,priv}}(\mathrm{EQ})\big)$; their construction also directly establishes the result stated in the main text.

Ambainis et al. [2] also used (essentially) the same gap identity function to establish the separation $\overline{\mathrm{R}}_0(f) = \widetilde{\Omega}\big(\mathrm{R}(f)^2\big)$. In constructing a GPW pointer function analogue of the GapID function, however, Ambainis et al. lose a few logarithmic factors: their construction shows that there exists a total function $f : \{0,1\}^n \to \{0,1\}$ with ϵ-error randomized query complexity that satisfies $\mathrm{R}_\epsilon(f) = O(\sqrt{n}\log^2 n \log\frac{1}{\varepsilon})$ and $\mathrm{R}_\epsilon(f) = \Omega(\sqrt{n}\log\frac{1}{\epsilon})$. The polylogarithmic gap between those two bounds is not particularly important when comparing this query complexity to the zero-error randomized query complexity $\overline{\mathrm{R}}_0(f) = \widetilde{\Omega}(n)$ of the same function, but it makes it impossible to obtain any separation at all between $\mathrm{R}(f)$ and $\mathrm{R}_\varepsilon(f)$ whenever $\epsilon = \Omega(n^{-\log n})$. To prove Theorem 1, we need a new variant of the GPW pointer function whose analysis avoids *any* gap that is a non-constant function of n.

At a high-level, GPW pointer functions are constructed by defining an $n \times m$ array of cells, whose values are taken from some (typically fairly large) alphabet Σ. The first logarithmic gap in Ambainis et al.'s upper and lower bounds occurs because the upper bound is measured in terms of the number of *bits* queried by the algorithm while the lower bound is in terms of the number of *cells* queried by an algorithm. To eliminate this gap, we must either reduce the size of the alphabet from $|\Sigma| = O(\log n)$ to a constant size or modify the analysis so that both the upper and lower bounds are in terms of bit-query complexity. We do the latter, using the notion of *resilient functions* [12] to show that an algorithm must query a constant fraction of the bits of a cell to learn *anything* about the contents of that cell. Resilient functions were introduced by Chor et al. [12], who gave an essentially optimal construction using basic linear algebra and the probabilistic method. Sherstov recently created a gadget [26] resilient to approximate polynomial degree. This gadget is both similar in construction to [12] and in motivation to our work; it too removes some loss due to function inputs coming from large alphabets.

The second logarithmic gap in Ambainis et al.'s construction occurs because the location of the "special" cells that an algorithm seeks to discover in the GPW pointer function can be found by following a binary tree structure; the upper bound accounts for the $\log n$ cell queries an algorithm requires to follow this structure while the lower bound holds even if an algorithm finds these special cells in a single query. We bypass this problematic issue with a simple but powerful observation: in our setting, once we use resilient functions to encode the contents of each cell, there is no longer any requirement to keep the size $|\Sigma|$ of the alphabet for each cell in the GPW pointer function to be polylogarithmic in n and so we can include a *lot* more information in each cell without affecting the query complexity gap. We use this flexibility to replace pointers to the root of a binary tree structure with direct pointers to all the special cells in its leaves.

The details of the proof of Theorem 1 are presented in Section 2.

Strong Direct Sum Theorem

Our proof of the strong direct sum theorem proceeds by establishing an analogous result in the setting of distributional query complexity. The *ϵ-error distributional complexity* of $f : \{0,1\}^n \to \{0,1\}$ with respect to the distribution μ on $\{0,1\}^n$, denoted by $\mathrm{D}^\mu_\epsilon(f)$, is the minimum query complexity of a deterministic algorithm that computes the value $f(x)$ correctly with probability at least $1-\epsilon$ when x is drawn from μ.

The distributional complexity approach is also the one used in prior work on direct sum theorems for query complexity [19, 8]. The challenge with this approach, however, is that a strong direct sum theorem for distributional query complexity does *not* hold in general, as Shaltiel [25] demonstrated (see also §4 in [13]): there exists a function f and a distribution μ on f's domain for which $\mathrm{D}^{\mu^k}_\epsilon(f^k) = O\big(\epsilon k \mathrm{D}^\mu_\epsilon(f)\big)$.

A similar barrier to strong direct sum theorems exists in the communication complexity setting. Molinaro, Woodruff, and Yaroslavstev [21, 22] bypassed this barrier by considering randomized protocols that are allowed to abort with some bounded probability. They were then able to show that the information complexity of such communication protocols (in both the one-way and two-way communication models) satisfies a strong direct sum property.

Following an analogous approach, we consider randomized algorithms that are allowed to *abort* (or, equivalently, to output some value $\perp$ that corresponds to "don't know") with some probability at most δ. The *ϵ-error, δ-abort randomized query complexity* of a function f is denoted by $R_{\delta,\epsilon}(f)$. With a natural extension of Yao's minimax principle, we can obtain bounds on this randomized query complexity by considering the corresponding *ϵ-error, δ-abort distributional complexity* $\mathrm{D}^{\mu}_{\delta,\epsilon}(f)$ of a function f, which is the minimum query complexity of deterministic algorithms must err with probability at most ϵ and abort with probability at most δ when inputs are drawn from the distribution μ. We show that a strong direct sum theorem does hold in this setting.

▶ **Lemma 5.** *There exists a constant c such that for every function $f : \{0,1\}^n \to \{0,1\}$, every distribution μ on $\{0,1\}^n$, and every $0 \le \delta, \epsilon \le \frac{1}{40}$,*

$$\mathrm{D}^{\mu^k}_{\delta,\epsilon}(f^k) = \Omega\left(k \cdot \mathrm{D}^{\mu}_{\frac{1}{3},\frac{c\cdot\epsilon}{k}}(f)\right).$$

The proof of Theorem 2 is then obtained from this lemma by showing that an analogue of Yao's minimax principle holds for algorithms that can both err and abort. The full details of the proofs of Lemma 5 and Theorem 2 are presented in Section 3.

2 Bounded-Error Separation Theorem

We complete the proof of Theorem 1 in this section. In Section 2.1, we first define the pointer function PtrFcn at the heart of the proof. In Sections 2.2–2.4, we establish a lower bound on the query complexity of the PtrFcn function via reductions from the GapID function, and in Section 2.5, we provide a matching upper bound on this query complexity. We complete the proof of Theorem 1 in Section 2.6 by combining these results with the use of resilient functions.

2.1 Pointer Function

The total function at the heart of the proof of Theorem 1 is a variant of the Göös–Pitassi–Watson pointer function PtrFcn that we define below. Let $[n]$ denote the set $\{1,\dots,n\}$.

Define $\Gamma = \{0,1\} \times ([n] \cup \{\perp\})^m \times ([m] \cup \{\perp\})$ to be the set of symbols σ that encode a *value* that we denote by $\textsc{value}(\sigma)$, m *row pointers* that we denote by $\textsc{row}_1(\sigma),\dots,\textsc{row}_m(\sigma)$, and one *column pointer* that we denote $\textsc{col}(\sigma)$.

The function $\textsc{PtrFcn} : \Gamma^{n\times m} \to \{0,1\}$ is defined as follows. First, we represent an input $x \in \Gamma^{n\times m}$ as an $n \times m$ grid of *cells*. We say that a column $j^* \in [m]$ is *special* for x if $\textsc{value}(x_{i,j^*}) = 1$ for every $1 \le i \le n$. Then $\textsc{PtrFcn}(x) = 1$ if and only if

- There is a unique column j^* that is special for x;
- Within the special column j^*, there is a unique cell i^* called the *special cell*;
- $\textsc{row}_j(x_{i,j^*}) = \perp$ for all $i \ne i^*$ and all $j \ne j^*$;
- For all $j \ne j^*$, let $i_j := \textsc{row}_j(x_{i^*,j^*})$. Then, we have
  - $\textsc{value}(x_{i_j,j}) = 0$ (i.e., all cells pointed to by the special cell have value 0)
  - $|\{j \ne j^* : \textsc{col}(x_{i_j,j}) = j^* \wedge \textsc{row}_{j^*}(x_{i_j,j}) = i^*\}| = \lfloor \frac{m-1}{2} \rfloor$ (i.e., *half* the cells pointed to by the special cell point back to the special cell)

We call the cells (i_j, j) *linked cells*; linked cells that point back to the special cell are *good*. In summary, $\textsc{PtrFcn}(x) = 1$ if (i) there is a special column, (ii) within the special column, there is a special cell, (iii) all cells in the special column that are not the special cell have $\textsc{row}_j(x_{i,j*}) = \perp$ for all $j \neq j^*$, (iv) each linked cell has value 0, and (v) exactly half of the linked cells are good.

The following simple claim will be useful in obtaining the query complexity lower bound for $\textsc{PtrFcn}$.

▷ Claim 6. Let $\mathcal{A}$ be an ε-error randomized algorithm for $\textsc{PtrFcn}$. Let $z \in \textsc{PtrFcn}^{-1}(1)$, and let (i^*, j^*) be the special cell of z. Then $\mathcal{A}(z)$ probes (i^*, j^*) with probability at least $1 - 2\varepsilon$.

Proof. Let $\bar{z}$ be the same input as z except that $\textsc{value}(i^*, j^*) = 0$. Then $\textsc{PtrFcn}(z) \neq \textsc{PtrFcn}(\bar{z})$ but $z, \bar{z}$ differ only on the special cell. Whenever $\mathcal{A}$ doesn't probe the special cell, it must output the same value for z and $\bar{z}$, so it errs on either z or $\bar{z}$. By the error guarantee of $\mathcal{A}$ and a union bound, the probability that $\mathcal{A}$ doesn't probe cell (i^*, j^*) is at most 2ε. ◁

2.2 Lower Bound on the Query Complexity of GapID

We begin the proof of Theorem 1 by establishing a (simple, asymptotically optimal) lower bound on the average query complexity of the $\textsc{GapID}$ function.

▶ **Lemma 7.** *For every $m \geq 2$ and every $\epsilon < \frac{1}{2}$, $\overline{\mathrm{R}}_\epsilon(\textsc{GapID}) = \Omega\left(\min\{\log \frac{1}{\epsilon}, m\}\right)$.*

Proof. Fix any $\epsilon \geq 2^{-\frac{2}{3}m}$. We will show that $\overline{\mathrm{R}}_\epsilon(\textsc{GapID}) = \Omega\left(\log \frac{1}{\epsilon}\right)$. This suffices to complete the proof of the theorem since it implies that for any $\epsilon < 2^{-\frac{2}{3}m}$, $\overline{\mathrm{R}}_\epsilon(\textsc{GapID}) \geq \overline{\mathrm{R}}_{2^{-\frac{2}{3}m}}(\textsc{GapID}) = \Omega(m)$.

Let $\mathcal{A}$ be a randomized algorithm that computes $\textsc{GapID}$ with error probability at most ϵ. Let $Q \subseteq [m]$ be a random variable that denotes the set of coordinates queried by $\mathcal{A}$, and let $\xi := \xi(Q, x)$ denote the event that each coordinate of the input x queried by the algorithm has the value 0. Note that when the event $\xi(Q, x)$ occurs, $\mathcal{A}$ has the same behavior on input x as it does on the input 0^m. Since $\textsc{GapID}(0^m) = 1$ and $\mathcal{A}$ has error probability at most ϵ, this means that for every input $x \in \{0,1\}^m$,

$$\Pr[\mathcal{A}(x) = 0 \wedge \xi] = \Pr[\mathcal{A}(0^m) = 0 \wedge \xi] \leq \Pr[\mathcal{A}(0^m) = 0] \leq \epsilon$$

and so $\Pr[\mathcal{A}(x) = 1] \geq \Pr[\mathcal{A}(x) = 1 \wedge \xi] \geq \Pr[\xi] - \epsilon$.

Define μ to be the uniform distribution on all inputs $x \in \{0,1\}^m$ with $|x| = m/2$. To err with probability at most ϵ on those inputs, the algorithm $\mathcal{A}$ must satisfy $\Pr[\mathcal{A}(x) = 1] \leq \epsilon$ for every x in the support of μ. Combining this upper bound with the previous lower bound, we therefore have that

$$\Pr_{x\sim\mu,\, Q}[\xi] - \epsilon \leq \mathop{\mathrm{E}}_{x\sim\mu}\left[\Pr[\mathcal{A}(x) = 1]\right] \leq \epsilon \qquad \Longrightarrow \qquad \Pr_{x\sim\mu,\, Q}[\xi] \leq 2\epsilon. \tag{5}$$

For any value $1 \leq q \leq \frac{m}{3}$,

$$\Pr_{x\sim\mu,\, Q}\left[\xi \;\middle|\; |Q| = q\right] = \frac{\binom{m-q}{m/2}}{\binom{m}{m/2}} = \frac{\frac{m}{2}(\frac{m}{2}-1)\cdots(\frac{m}{2}-q+1)}{m(m-1)\cdots(m-q+1)}$$
$$> \left(\frac{\frac{m}{2}-q}{m-q}\right)^q > \left(\frac{1}{2} - \frac{q}{2(m-q)}\right)^q \geq 4^{-q}.$$

Therefore,

$$\Pr_{x\sim\mu,\,Q}\left[\xi \;\middle|\; |Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] > 4^{-\frac{1}{2}\log\frac{1}{4\epsilon}} = 4\epsilon.$$

Combining this inequality with (5), we obtain

$$2\epsilon \ge \Pr_{x\sim\mu,\,Q}[\xi] \ge \Pr\left[|Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] \cdot \Pr_{x\sim\mu,\,Q}\left[\xi \;\middle|\; |Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] > \Pr\left[|Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] \cdot 4\epsilon.$$

Rearranging the inequality yields $\Pr\left[|Q| \le \frac{1}{2}\log\frac{1}{4\epsilon}\right] < \frac{1}{2}$ and so the average query complexity of $\mathcal{A}$ is bounded below by

$$\mathrm{E}\left[|Q|\right] > \tfrac{1}{2}\log\tfrac{1}{4\epsilon} \cdot \Pr\left[|Q| > \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] > \tfrac{1}{4}\log\tfrac{1}{4\epsilon}. \qquad \blacktriangleleft$$

2.3 Lower Bound on the Query Complexity of BlueRed

We wish to relate the average query complexity of PtrFcn to that of the GapID function. We do this by relating both query complexities to that of another partial function that we call BlueRed.

Let $\Sigma := \{\textsc{black}, \textsc{blue}, \textsc{red}\}$, and call a symbol *colored* if it is not black. The input is an $n \times m$ grid of entries from Σ, with the promise that each column contains a unique colored entry, and either all colored entries are red, or half the colored entries are blue. Formally, we define $\textsc{BlueRed} : \Sigma^{n\times m} \to \{0, 1, *\}$ as follows:

$$\textsc{BlueRed}(x) = \begin{cases} 1 & \text{if each column has 1 colored entry \& all colored entries are } \textsc{red}, \\ 0 & \text{if each column has 1 colored entry \& exactly } \lfloor \tfrac{m}{2} \rfloor \text{ entries are } \textsc{blue}, \\ * & \text{otherwise.} \end{cases}$$

The following reduction shows that the average query complexity of BlueRed is $\Theta(n)$ times as large as that of the GapID function.

▶ **Lemma 8.** *For every $\epsilon > 0$, $\overline{\mathrm{R}}_\varepsilon(\textsc{BlueRed}) \ge \frac{n}{4} \cdot \overline{\mathrm{R}}_\varepsilon(\textsc{GapID})$.*

Proof. Fix any algorithm $\mathcal{A}$ that computes BlueRed with error at most ϵ and has expected query cost $c = \overline{\mathrm{R}}_\varepsilon(\textsc{BlueRed})$. We will use $\mathcal{A}$ to construct an algorithm $\mathcal{B}$ that computes GapID with error at most ϵ and expected cost $4c/n$.

Given an input $x \in \{0,1\}^m$, the algorithm $\mathcal{B}$ constructs an instance of the BlueRed problem in the following way. First, it generates indices $i_1, \ldots, i_m \in [n]$ independently and uniformly at random. Then it defines

$$y_{i,j} = \begin{cases} \textsc{red} & \text{if } i = i_j \text{ and } x_j = 0, \\ \textsc{blue} & \text{if } i = i_j \text{ and } x_j = 1, \\ \textsc{black} & \text{if } i \ne i_j. \end{cases}$$

Finally, the algorithm $\mathcal{B}$ emulates the algorithm $\mathcal{A}$ on input y, querying the value of x_j whenever $\mathcal{A}$ queries the bit (i_j, j) for some $j \le m$. This construction guarantees that $\mathcal{B}$ computes GapID with error at most ϵ; its query complexity corresponds to the number of red or blue entries that are queried by $\mathcal{A}$.

Let $Q \subseteq [n] \times [m]$ be the random variable that denotes the set of indices queried by $\mathcal{A}$, and let $C \subseteq [m]$ denote the set of columns whose red or blue entry is queried by $\mathcal{A}$. Without loss of generality, we may assume that $\mathcal{A}$ does not query any entry of a column

after it finds the colored entry within that column. We partition C into two sets C_{early} and C_{late}, where C_{early} denotes the set of columns whose colored entry is found within the first $\frac{n}{2}$ queries to that column and C_{late} denotes the set of columns whose colored entry was found with more than $\frac{n}{2}$ queries to that column. Let $X_1, X_2, \ldots, X_{|Q|}$ be indicator variables where $X_k = 1$ if and only if the kth query (i, j) made by $\mathcal{A}$ is RED or BLUE *and* is one of the first $\frac{n}{2}$ queries to column j. Since each value i_j is drawn uniformly at random from $[n]$, each of these indicator variables has expected value $\mathrm{E}[X_k] \leq \frac{2}{n}$. Therefore,

$$\mathrm{E}\left[|C_{\text{early}}|\right] = \mathrm{E}\left[\sum_{i \leq |Q|} X_i\right] \leq \frac{2}{n}\,\mathrm{E}\left[|Q|\right].$$

Furthermore, by definition at least $\frac{n}{2}$ queries are made to each column in C_{late} so the expected size of this set is bounded by $\mathrm{E}\left[|C_{\text{late}}|\right] \leq \frac{2}{n}\,\mathrm{E}\left[|Q|\right]$ and

$$\mathrm{E}\left[|C|\right] = \mathrm{E}\left[|C_{\text{early}}|\right] + \mathrm{E}\left[|C_{\text{late}}|\right] \leq \frac{4}{n}\,\mathrm{E}\left[|Q|\right].$$

Thus, the expected query cost of $\mathcal{B}$ is at most $\frac{4}{n} \cdot \overline{\mathrm{R}}_{\varepsilon}(\textsc{BlueRed})$, as we wanted to show. ◀

2.4 Lower Bound on the Query Complexity of PtrFcn

▶ **Lemma 9.** *For every* $0 \leq \epsilon \leq \frac{1}{4}$, $\overline{\mathrm{R}}_{\epsilon}(\textsc{PtrFcn}) \geq \overline{\mathrm{R}}_{2\epsilon}(\textsc{BlueRed})$.

Proof. Let $\mathcal{A}$ be a randomized algorithm that computes PTRFCN with error at most ϵ and expected query cost $q := \overline{\mathrm{R}}_{\epsilon}(\textsc{PtrFcn})$. We use $\mathcal{A}$ to construct a randomized algorithm $\mathcal{B}$ that computes BLUERED with the same cost and error at most 2ϵ.

Let x be an input for BLUERED. Each time $\mathcal{A}$ queries a cell, $\mathcal{B}$ queries the corresponding entry in x. If the entry in x is BLACK, then $\mathcal{B}$ returns $\langle 1, \perp, \ldots, \perp \rangle$. If the entry in x is RED, then $\mathcal{B}$ returns $\langle 0, \perp, \ldots, \perp \rangle$. Finally, if the entry of x is BLUE, then $\mathcal{B}$ terminates the emulation and returns 0. If $\mathcal{A}$ reaches the end of the emulation without having been terminated, $\mathcal{B}$ outputs the same result as $\mathcal{A}$.

The query complexity of $\mathcal{B}$ is at most that of $\mathcal{A}$. It remains to show that $\mathcal{B}$ errs with probability at most 2ϵ. There are two cases to consider.

The first case is when $x \in \textsc{BlueRed}^{-1}(1)$. Then x maps directly to an input $z \in \textsc{PtrFcn}^{-1}(0)$ and hence $\mathcal{B}$ errs with probability at most ϵ on x.

The second case is when $x \in \textsc{BlueRed}^{-1}(0)$. Let z be an arbitrary 1-input for PTRFCN such that (i) $z_{i,j} = \langle 1, \perp, \ldots, \perp \rangle$ whenever $x_{i,j} =$ BLACK, (ii) $z_{i,j} = \langle 0, \perp, \ldots, \perp \rangle$ whenever $x_{i,j} =$ RED, and (iii) the special entry and good entries of z correspond to BLUE entries of x. It might not be possible to completely emulate $\mathcal{A}$ on input z without knowing the exact set of BLUE entries. However, $\mathcal{B}$ doesn't need to fully emulate $\mathcal{A}$ – it only needs to know how to map BLACK and RED entries. Once a BLUE entry is probed, $\mathcal{B}$ halts and outputs 0. In this way, we claim that $\mathcal{B}$ on input x probes the same cells as $\mathcal{A}$ on input z until it halts. Therefore its output is the same as $\mathcal{A}(z)$ unless $\mathcal{A}(z)$ probes the special cell or a good cell. Moreover, in this case, $\mathcal{B}$ outputs correctly with certainty. Thus, by Claim 6, the error of $\mathcal{B}$ is at most

$$\Pr[\mathcal{B} \text{ errs}] \leq \Pr[\mathcal{B} \text{ probes no blue cells}] \leq \Pr[\mathcal{A} \text{ doesn't probe special cell}] \leq 2\varepsilon\ .$$ ◀

2.5 Upper Bound on the Query Complexity of PtrFcn

The proof of Theorem 1 also requires a tight upper bound on the (worst-case) randomized query complexity of PTRFCN. This argument is straightforward, and similar to the analysis of Ambainis et al. [2] for their analogous pointer function.

Algorithm 1: PtrFcnSolver(x).

$S \leftarrow [m]$;
$T \leftarrow$ a random subset of $[m]$ of size $|T| = \log \frac{1}{\epsilon}$;
for *each cell* (i, j) *in a column in* T **do**
  if $\textsc{value}(x_{i,j}) = 0 \wedge \textsc{col}(x_{i,j}) \in S$ **then**
    $j^* \leftarrow \textsc{col}(x_{i,j})$;
    $i^* \leftarrow \textsc{row}_{j^*}(x_{i,j})$;
    $valid \leftarrow \textsc{True}$;
    while $|S| > 1 \wedge valid$ **do**
      $\ell \leftarrow$ any column in $S \setminus \{j^*\}$;
      if $\textsc{value}(x_{\textsc{row}_\ell(x_{i^*,j^*}),\ell}) = 0$ **then**
        $S \leftarrow S \setminus \{\ell\}$;
      else
        $valid \leftarrow \textsc{False}$;
    if $|S| = 1$ **then**
      break;
if $|S| = 1$ **then**
  fix $j \in S$. **return** 1 if (i) column j is special, (ii) there is a special cell within column j, (iii) all cells linked by the special cell have value 0, and (iv) half of linked cells point back to the special cell.
return 0

▶ **Lemma 10.** $\mathrm{R}_\varepsilon(\textsc{PtrFcn}) = O(n \log \frac{1}{\epsilon} + m)$.

Proof. The algorithm that computes the PtrFcn function is described in Algorithm 1. In this algorithm, the set S corresponds to the set of potential special columns. The query complexity of PtrFcnSolver follows from the fact that each iteration of the inner while loop either eliminates one of the columns from the set S of candidates or one of the $n \log \frac{1}{\epsilon}$ cells in the columns in T. The final check of the (lone remaining) potential special column at the end of the algorithm examines at most $n + m$ cells.

Whenever the PtrFcnSolver returns the value 1, then it in fact has observed a certificate that $\textsc{PtrFcn}(x) = 1$ so the algorithm has perfect soundness.

Conversely, suppose $\textsc{PtrFcn}(x) = 1$. Exactly half of the columns are good, so T contains such a cell with probability at least $1 - (1/2)^{\log(1/\varepsilon)} = 1 - \varepsilon$. Now, consider the for loop iteration when the first good cell (i, j) is selected. Since (i, j) is a good cell, it points back to the special cell, which in turn points to a linked cell in all columns except the special column. For any remaining $j' \neq j \in S$, $\textsc{PtrFcn}(x)$ probes the linked cell in column j', verifies the value equals 0, and removes it from S. In this way, the remaining columns in S save the special column are eliminated. Once we reduce S to a single remaining candidate, we can probe all cells in this column and all linked cells using $n + m$ queries to verify that indeed $\textsc{PtrFcn}(x) = 1$. ◀

2.6 Completing the Proof of Theorem 1

The last ingredient that we need to complete the proof of Theorem 1 is the concept of *resilient functions* [12].

▶ **Definition 11.** *The function* $\phi : \{0,1\}^n \to \{0,1\}^m$ *is* t-resilient *for some* $1 \leq t < n$ *if for any set* $S \subseteq [n]$ *of* $|S| \leq t$ *coordinates and any assignment of values for the inputs* $\{x_i\}_{i \in S}$*, when the values* $\{x_i\}_{i \in [n] \setminus S}$ *are set uniformly at random then* $\phi(x)$ *is uniformly distributed in* $\{0,1\}^m$.

We use the following existence result on resilient functions that was established by Chor et al. [12].

▶ **Theorem 12** (Chor et al. [12]). *For every large enough n, there is a function $\phi : \{0,1\}^n \to \{0,1\}^m$ that is $\frac{n}{3}$-resilient and satisfies $m \geq 0.08n$.*

We use resilient functions to bound the query complexity of functions via the following lemma.

▶ **Lemma 13.** *Fix a finite set $\mathcal{X}$ of cardinality $|\mathcal{X}| = 2^\ell$ for some integer $\ell \geq 1$ and let $\phi : \{0,1\}^N \to \mathcal{X}$ be an $\frac{N}{3}$-resilient function. Then for every function $f : \mathcal{X}^m \to \{0,1\}$ and every $\epsilon \geq 0$,*

$$\mathrm{R}_\epsilon(f \circ \phi) = \Theta(N \cdot \mathrm{R}_\epsilon(f)) \qquad \text{and} \qquad \overline{\mathrm{R}}_\epsilon(f \circ \phi) = \Theta(N \cdot \overline{\mathrm{R}}_\epsilon(f)).$$

Proof. The upper bounds follow immediately from the observation that if $\mathcal{A}$ is a randomized algorithm that computes f with ϵ-error, then we can define a algorithm $\mathcal{B}$ that computes $f \circ \phi$ with the same error by simulating $\mathcal{A}$ and querying the N bits to observe the value $\phi(x)$ to return to each query.

For the lower bounds, let $\mathcal{A}$ be a randomized algorithm that computes $f \circ \phi$ with error at most ϵ. We define an algorithm $\mathcal{B}$ for computing f that simulates $\mathcal{A}$ in the following way. For the first $\frac{N}{3}$ queries to a cell, $\mathcal{B}$ answers the queries with uniformly random variables in $\{0,1\}$. On a query to the $(\frac{N}{3}+1)$-th bit of a cell, $\mathcal{B}$ queries the value v of the corresponding cell in x. It then draws a value z in $\phi^{-1}(v)$ uniformly at random among all values that agree with the $\frac{N}{3}$ bits output so far. The current query and all further queries to bits of that cell are then answered using z. Once $\mathcal{A}$ terminates, $\mathcal{B}$ returns $\mathcal{A}$'s output and terminates as well.

The correctness of $\mathcal{B}$ follows directly from the correctness of $\mathcal{A}$. Furthermore, on any input for which $\mathcal{A}$ makes q queries, $\mathcal{B}$ makes at most $q/(N/3)$ queries since $N/3$ distinct queries of $\mathcal{A}$ are required for each query that $\mathcal{B}$ eventually makes to x. Thus both the average-case and worst-case query complexities of $\mathcal{B}$ are bounded by $3/N$ times the corresponding query complexities of $\mathcal{A}$. ◀

We are now ready to complete the proof of the separation theorem.

Proof of Theorem 1. Fix $m = n = 2^\ell - 1$ for any integer $\ell \geq 1$ so that $|\Gamma| = 2^{\ell(2^\ell-1)+\ell+1}$ is a power of 2. Fix a $\frac{C}{3}$-resilient function $\phi : \{0,1\}^C \to \Gamma$ for some $C \leq 12.5 \log |\Gamma|$ and define the function $\textsc{EncFcn} = \textsc{PtrFcn} \circ \phi$. By Lemmas 13 and 10,

$$\mathrm{R}_\varepsilon(\textsc{EncFcn}) = O\big(C(n \log \tfrac{1}{\epsilon} + m)\big) = O\big(Cn \log \tfrac{1}{\epsilon}\big).$$

In particular, setting $\epsilon = \frac{1}{3}$ we obtain $\mathrm{R}(\textsc{EncFcn}) = O(Cn)$.

Using Lemma 13, 9, and 8, we obtain the chain of inequalities

$$\overline{\mathrm{R}}_\varepsilon(\textsc{EncFcn}) = \Omega\big(C \cdot \overline{\mathrm{R}}_\epsilon(\textsc{PtrFcn})\big) = \Omega\big(C \cdot \overline{\mathrm{R}}_{2\epsilon}(\textsc{BlueRed})\big) = \Omega\big(Cn \cdot \overline{\mathrm{R}}_{2\varepsilon}(\textsc{GapID})\big).$$

By Lemma 7, when $\epsilon > 2^{-m} = 2^{-n}$ this implies that

$$\overline{\mathrm{R}}_\varepsilon(\textsc{EncFcn}) = \Omega\big(Cn \log \tfrac{1}{\epsilon}\big) = \Omega\big(\log \tfrac{1}{\epsilon} \cdot \mathrm{R}(\textsc{EncFcn})\big).$$

Theorem 1 is obtained by noting that $\textsc{EncFcn}$ is a function on $N = O(mn|\Gamma|) = O(n^3 \log n)$ variables. ◀

3 Strong Direct Sum Theorem

We establish Theorem 2 by proving a corresponding direct sum result in the distributional model and applying a Yao minimax principle for algorithms that err and abort with bounded probability.

We introduce the model of algorithms that can abort in Section 3.1, where we also relate this model to the average query complexity setting of randomized algorithms and establish a Yao minimax principle. In Section 3.2, we establish the main technical result, a strong direct sum theorem for distributional complexity. We complete the proof of Theorem 2 itself in Section 3.3 and the proofs of Corollaries 3 and 4 are completed in Section 3.4.

3.1 Algorithms That Can Abort

We consider randomized algorithms that are allowed to *err* and *abort*. In this setting, an algorithm outputs $\perp$ instead of giving a valid output when it chooses to abort. Let $\mathrm{D}^{\mu}_{\delta,\varepsilon}(f)$ and $\mathrm{R}_{\delta,\varepsilon}(f)$ denote the distributional and randomized query complexities of f when the algorithm aborts with probablity at most δ and errs with probability at most ε.

Randomized query complexity in the setting where algorithms can abort with constant probability δ is asymptotically equivalent to the average randomized query complexity.

▶ **Proposition 14.** *For every function $f : \{0,1\}^n \to \{0,1\}$, every $0 \le \epsilon < \frac{1}{2}$ and every $0 < \delta < 1$,*

$$\delta \cdot \mathrm{R}_{\delta,\varepsilon}(f) \le \overline{\mathrm{R}}_{\epsilon}(f) \le \tfrac{1}{1-\delta} \cdot \mathrm{R}_{\delta,(1-\delta)\epsilon}(f).$$

Proof. For the first inequality, let $\mathcal{A}$ be a randomized algorithm that computes f with ϵ error and has expected query complexity q. Let $\mathcal{B}$ be the randomized algorithm $\mathcal{B}$ that simulates $\mathcal{A}$ except that whenever $\mathcal{A}$ tries to make more than q/δ queries, it aborts. The algorithm $\mathcal{B}$ also computes f with error at most ϵ, and it has worst-case query complexity q/δ. Furthermore, by Markov's inequality, $\mathcal{B}$ aborts with probability at most δ.

For the second inequality, let $\mathcal{B}$ be a randomized algorithm with query complexity q that computes f with error probability at most $(1-\delta)\epsilon$ and abort probability at most δ. Let $\mathcal{A}$ be the randomized algorithm that simulates $\mathcal{B}$ until that algorithm does not abort, then outputs the same value. The error probability of $\mathcal{B}$ conditioned on it not aborting is at most $\frac{(1-\delta)\epsilon}{1-\delta} = \epsilon$, so the algorithm $\mathcal{A}$ also errs with probability at most ϵ, and its expected query complexity is $q(1+\delta+\delta^2+\cdots) = \frac{q}{1-\delta}$. ◀

Yao's minimax principle can be adapted for the setting of algorithms that abort as follows.

▶ **Lemma 15.** *For any $\alpha, \beta > 0$ such that $\alpha + \beta \le 1$, we have*

$$\max_{\mu} \mathrm{D}^{\mu}_{\delta/\alpha,\varepsilon/\beta}(f) \le \mathrm{R}_{\delta,\varepsilon}(f) \le \max_{\mu} \mathrm{D}^{\mu}_{\alpha\delta,\beta\varepsilon}(f).$$

Proof. We handle the initial inequality (i.e., the *easy direction*) first. Fix a q-query randomized algorithm $\mathcal{A}$ achieving $\mathrm{R}_{\delta,\varepsilon}(f)$. By the guarantee of $\mathcal{A}$, we have that for any input x, $\mathcal{A}$ aborts with probability at most δ and errs with probabiltiy at most ε. Let $\mathbf{1}_{\delta}(x)$ and $\mathbf{1}_{\varepsilon}(x)$ be indicator variables for the events that $\mathcal{A}$ aborts on x and $\mathcal{A}$ errs on x respectively. Then, we have $\mathrm{E}_R[\mathbf{1}_{\delta}(x)] \le \delta$ and similarly $\mathrm{E}_R[\mathbf{1}_{\varepsilon}(x)] \le \varepsilon$ when the expectation is taken over the randomness R of the algorithm $\mathcal{A}$. Next, fix any input distribution μ and let $X \sim \mu$. It follows that

$$\mathop{\mathrm{E}}_{R}\left[\mathop{\mathrm{E}}_{X}[\mathbf{1}_{\delta}(X)]\right] = \mathop{\mathrm{E}}_{X}\left[\mathop{\mathrm{E}}_{R}[\mathbf{1}_{\delta}(X)]\right] \le \delta \quad \text{and} \quad \mathop{\mathrm{E}}_{R}\left[\mathop{\mathrm{E}}_{X}[\mathbf{1}_{\varepsilon}(X)]\right] = \mathop{\mathrm{E}}_{X}\left[\mathop{\mathrm{E}}_{R}[\mathbf{1}_{\varepsilon}(X)]\right] \le \varepsilon.$$

Using Markov's inequality twice, we have

$$\Pr_R\left[\mathrm{E}_X[\mathbf{1}_\delta(X)] > \delta/\alpha\right] < \alpha \qquad \text{and} \qquad \Pr_R\left[\mathrm{E}_X[\mathbf{1}_\varepsilon(X)] > \varepsilon/\beta\right] < \beta.$$

By a union bound, there exists a setting of the random string R such that both $\mathrm{E}[\mathbf{1}_\delta(X)] \le \delta/\alpha$ and $\mathrm{E}[\mathbf{1}_\varepsilon(X)] \le \varepsilon/\beta$. Fixing this R gives a q-query deterministic algorithm that aborts with probability at most δ/α and errs with probability at most ε/β, hence $\mathrm{D}^\mu_{\delta/\alpha,\varepsilon/\beta}(f) \le \mathrm{R}_{\delta,\varepsilon}(f)$.

For the second inequality, let $c := \max_\mu \mathrm{D}^\mu_{\alpha\delta,\beta\varepsilon}(f)$. Consider a two-player, zero-sum game where player 1 selects a c-query deterministic algortihm $\mathcal{A}$ for f, player 2 selects an input x, and player 1 is paid $-\varepsilon$ if $\mathcal{A}(x)$ aborts, $-\delta$ if $\mathcal{A}(x)$ errs, and 0 otherwise. Note that each mixed strategy for player 1 corresponds to a randomized algorithm and each mixed strategy for player 2 corresponds to an input distribution μ. By our choice of c, it follows that for any mixed strategy for player 2, player 1 can obtain payoff $-\varepsilon(\alpha\delta) - \delta(\beta\varepsilon) \ge -\varepsilon\delta$. By the minimax theorem, it follows that there is a mixed strategy for player 1 (i.e., a c-query randomized algorithm $\mathcal{A}$) that provides the same payoff for every choice of player 2. Finally, note that $\mathcal{A}$ aborts with probability at most δ and errs with probability at most ε; otherwise, the payoff would be less than $-\varepsilon\delta \le -\varepsilon\delta(\alpha+\beta)$. We've shown a c-query randomized algorithm that aborts w/probability at most δ and errs w/probability at most ε, hence $\mathrm{R}_{\delta,\varepsilon}(f) \le c = \max_\mu \mathrm{D}^\mu_{\alpha\delta,\beta\varepsilon}(f)$. ◀

3.2 Strong Direct Sum for Distributional Complexity

We prove a slightly more precise variant of Lemma 5.

▶ **Lemma 16.** *For every function $f : \{0,1\}^n \to \{0,1\}$, every distribution μ on $\{0,1\}^n$, and every $0 \le \delta, \epsilon \le \frac{1}{4}$,*

$$\mathrm{D}^{\mu^k}_{\delta,\epsilon}(f^k) = \Omega\left(k \cdot \mathrm{D}^\mu_{\frac{1}{10}+4\delta+4\epsilon,\frac{48\epsilon}{k}}(f)\right).$$

Proof. Let $\mathcal{A}$ be a deterministic algorithm with query complexity q that computes f^k with error probability at most ϵ and abort probability at most δ when the input $x = (x^{(1)},\dots,x^{(k)})$ is drawn from μ^k. Then conditioned on $\mathcal{A}$ not aborting, it outputs the correct value of f^k with probability at least $1 - \frac{\epsilon}{1-\delta} \ge 1 - 2\epsilon$ and

$$\begin{aligned} 1 - 2\epsilon &\le \Pr_{x\sim\mu^k}\left[\mathcal{A}(x) = f^k(x) \;\middle|\; \mathcal{A}(x) \ne \perp\right] \\ &= \prod_{i\le k} \Pr_{x\sim\mu^k}\left[\mathcal{A}(x)_i = f(x^{(i)}) \;\middle|\; \mathcal{A}(x)_{<i} = f^k(x)_{<i}, \mathcal{A}(x) \ne \perp\right]. \end{aligned}$$

This implies that at least $\frac{2}{3}k$ indices $i \in [k]$ satisfy

$$\Pr_{x\sim\mu^k}\left[\mathcal{A}(x)_i \ne f(x^{(i)}) \;\middle|\; \mathcal{A}(x)_{<i} = f^k(x)_{<i}, \mathcal{A}(x) \ne \perp\right] \le \frac{12\epsilon}{k}, \tag{6}$$

otherwise the product in the product in the previous inequality would be less than $(1 - 12\epsilon/k)^{k/3} \le e^{-4\epsilon} < 1 - 2\epsilon$, contradicting the lower bound on this product.

For each $i \le k$, let $q_i(x)$ denote the number of queries that $\mathcal{A}$ makes to $x^{(i)}$ on input x. The query complexity of $\mathcal{A}$ guarantees that for each input x, $\sum_{i\le k} q_i(x) \le q$. Therefore, $\sum_{i\le k} \mathrm{E}_{x\sim\mu^k}[q_i(x)] \le q$ and at least $\frac{2}{3}k$ indices $i \in [k]$ satisfy

$$\mathrm{E}_{x\sim\mu^k}[q_i(x)] \le \frac{3q}{k}. \tag{7}$$

Thus, some index $i^* \in [k]$ satisfies both (6) and (7). Fix such an index i^*. For inputs $y \in \mu^k$ and $x \in \mu$, write $y^{(i^* \leftarrow x)} := (y^{(1)}, \ldots, y^{(i^*-1)}, x, y^{(i^*+1)}, \ldots, y^{(k)})$ to be the input obtained by replacing $y^{(i^*)}$ with x in y. With this notation, the two conditions (6) and (7) satisfied by i^* can be rewritten as

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\Pr_{x\sim\mu}\left[\mathcal{A}(y^{(i^*\leftarrow x)})_{i^*} \neq f(x) \;\middle|\; \mathcal{A}(y^{(i^*\leftarrow x)})_{<i^*} = f^k(y^{(i^*\leftarrow x)})_{<i^*}, \mathcal{A}(y^{(i^*\leftarrow x)}) \neq \perp\right]\right] \leq \frac{12\epsilon}{k}$$

and

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\mathop{\mathrm{E}}_{x\sim\mu}\left[q_{i^*}(y^{(i^*\leftarrow x)})\right]\right] \leq \frac{3q}{k}.$$

The correctness of $\mathcal{A}$ also guarantees that

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\Pr_{x\sim\mu}\left[\mathcal{A}(y^{(i^*\leftarrow x)}) = \perp\right]\right] \leq \delta$$

and

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\Pr_{x\sim\mu}\left[\mathcal{A}(y^{(i^*\leftarrow x)})_{<i^*} \neq f^k(y^{(i^*\leftarrow x)})_{<i^*}) \;\middle|\; \mathcal{A}(y^{(i^*\leftarrow x)}) \neq \perp\right]\right] \leq \epsilon.$$

Therefore, by Markov's inequality, there exists an input $z \in \{0,1\}^{n\times k}$ such that

$$\begin{aligned}
\Pr_{x\sim\mu}\left[\mathcal{A}(z^{(i^*\leftarrow x)}) = \perp\right] &\leq 4\delta,\\
\Pr_{x\sim\mu}\left[\mathcal{A}(z^{(i^*\leftarrow x)})_{<i^*} \neq f^k(z^{(i^*\leftarrow x)})_{<i^*} \;\middle|\; \mathcal{A}(z^{(i^*\leftarrow x)}) \neq \perp\right] &\leq 4\epsilon,\\
\Pr_{x\sim\mu}\left[\mathcal{A}(z^{(i^*\leftarrow x)})_{i^*} \neq f(x) \;\middle|\; \mathcal{A}(z^{(i^*\leftarrow x)})_{<i^*} = f^k(z^{(i^*\leftarrow x)}), \mathcal{A}(z^{(i^*\leftarrow x)}) \neq \perp\right] &\leq \frac{48\epsilon}{k}, \text{ and}\\
\mathop{\mathrm{E}}_{x\sim\mu}\left[q_{i^*}(z^{(i^*\leftarrow x)})\right] &\leq \frac{12q}{k}.
\end{aligned}$$

Let $\mathcal{A}'$ be the deterministic algorithm that computes $f(x)$ by simulating $\mathcal{A}$ on the input $z^{(i^*\leftarrow x)}$ with two additions:

1. If $\mathcal{A}$ attempts to query more than $\frac{120q}{k}$ bits of x, $\mathcal{A}'$ aborts, and
2. When $\mathcal{A}$ terminates, the algorithm $\mathcal{A}'$ first verifies that the output generated by $\mathcal{A}$ satisfies $\mathcal{A}(z^{(i^*\leftarrow x)})_{\leq i^*} = f^k(z^{(i^*\leftarrow x)})$. If so $\mathcal{A}'$ returns the value $\mathcal{A}(z^{(i^*\leftarrow x)})_{i^*}$; if not, $\mathcal{A}'$ aborts.

The algorithm $\mathcal{A}'$ has query complexity at most $\frac{120q}{k}$ and, by the conditions satisfied by z, it aborts with probability at most $\frac{1}{10} + 4\delta + 4\epsilon$ and errs with probability at most $\frac{48\epsilon}{k}$ when $x \sim \mu$. ◀

3.3 Proof of Theorem 2

We now complete the proof of Theorem 2. Fix $\delta = \frac{1}{40}$. By Proposition 14 and the second inequality of Lemma 15,

$$\overline{\mathrm{R}}_{\frac{96\epsilon}{k}}(f) \leq 2\,\mathrm{R}_{\frac{1}{2},\frac{48\epsilon}{k}}(f) \leq 2\,\mathrm{R}_{\frac{1}{5}+4\delta+4\epsilon,\frac{48\epsilon}{k}}(f) \leq 2\max_{\mu}\mathrm{D}^{\mu}_{\frac{1}{10}+2\delta+2\epsilon,\frac{24\epsilon}{k}}(f).$$

Let μ^* denote a distribution where the maximum is attained. By Lemma 16,

$$\mathrm{D}^{\mu^*}_{\frac{1}{10}+2\delta+2\epsilon,\frac{24\epsilon}{k}}(f) = O\left(\frac{1}{k}\cdot \mathrm{D}^{(\mu^*)^k}_{\frac{\delta}{2},\frac{\epsilon}{2}}(f^k)\right).$$

Using the first inequality of Lemma 15 we then obtain

$$\mathrm{D}^{(\mu^*)^k}_{\frac{\delta}{2},\frac{\epsilon}{2}}(f^k) \le \max_{\nu} \mathrm{D}^{\nu}_{\frac{\delta}{2},\frac{\epsilon}{2}}(f^k) \le \mathrm{R}_{\delta,\epsilon}(f^k).$$

Combining these inequalities and applying Proposition 14 once more yields

$$\overline{\mathrm{R}}_{\frac{96\epsilon}{k}}(f) \le O\big(\tfrac{1}{k} \cdot \mathrm{R}_{\delta,\epsilon}(f^k)\big) \le O\big(\tfrac{1}{k} \cdot \overline{\mathrm{R}}_{\epsilon}(f^k)\big).$$

Theorem 2 follows from the identity $\overline{\mathrm{R}}_{\frac{\epsilon}{k}}(f) = \Theta\big(\overline{\mathrm{R}}_{\frac{96\epsilon}{k}}(f)\big)$ obtained from the standard success amplification trick. ◀

3.4 Proof of Corollaries 3 and 4

Corollary 3 is obtained as a direct consequence of Theorems 1 and 2.

Proof of Corollary 3. The upper bound is via the universal bound (3). For the matching lower bound, let $f : \{0,1\}^n \to \{0,1\}$ be a function that satisfies the condition of Theorem 1. By Theorem 2, the randomized communication complexity of f^k satisfies

$$R(f^k) \ge \overline{\mathrm{R}}(f^k) = \Omega\big(k \cdot \overline{\mathrm{R}}_{\frac{1}{3k}}(f)\big)$$

By Theorem 1,

$$\overline{\mathrm{R}}_{\frac{1}{3k}}(f) = \Omega\big(\mathrm{R}(f) \cdot \log k\big)$$

as long as $k \le 2^{(\frac{n}{\log n})^{1/3}}$. Combining those inequalities yields $\mathrm{R}(f^k) = \Omega\big(k \log k \cdot \mathrm{R}(f)\big)$, as we wanted to show. ◀

The proof of Corollary 4 uses the following randomized query-to-communication lifting theorem of Göös, Pitassi, and Watson [17].

▶ **Theorem 17** (Göös, Pitassi, Watson). *Define* $\textsc{Ind}_m : [m] \times \{0,1\}^m \to \{0,1\}$ *to be the index function mapping* (x, y) *to* y_x *and fix* $m = n^{256}$*. For every* $f : \{0,1\}^n \to \{0,1\}$*,*

$$\mathrm{R}^{\mathrm{cc}}(f \circ \textsc{Ind}_m) = \mathrm{R}(f) \cdot \Theta(\log n)$$

and

$$\mathrm{R}^{\mathrm{cc}}(f^k \circ \textsc{Ind}_m) = \mathrm{R}(f^k) \cdot \Theta(\log n).$$

▶ Remark 18. The statement of Theorem 17 in [17] only mentions the first identity explicitly. However, as discussed in their Section II, the theorem statement holds for functions with any finite range.[2] Therefore, the theorem holds for the function f^k as well as f.

Proof of Corollary 4. By Corollary 3, there exists a function $f : \{0,1\}^n \to \{0,1\}$ which satisfies $\mathrm{R}(f^k) = \Theta(k \log k \cdot \mathrm{R}(f))$. Combining this result with Theorem 17, we obtain

$$\begin{aligned}
\mathrm{R}^{\mathrm{cc}}\big((f \circ \textsc{Ind}_m)^k\big) &= \mathrm{R}^{\mathrm{cc}}\big(f^k \circ \textsc{Ind}_m\big) \\
&= \mathrm{R}(f^k) \cdot \Theta(\log n) \\
&= \Theta(k \log k \cdot \mathrm{R}(f) \cdot \log n) \\
&= \Theta(k \log k) \cdot \mathrm{R}^{\mathrm{cc}}(f \circ \textsc{Ind}_m).
\end{aligned}$$ ◀

[2] In fact, their theorem also holds in even more general settings such as when f is a partial function or a relation, for example.

References

1. Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proceedings 48th Annual ACM Symposium on Theory of Computing*, pages 863–876, 2016.
2. Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in query complexity based on pointer functions. *Journal of the ACM*, 64(5):32, 2017.
3. Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly optimal separations between communication (or query) complexity and partitions. In *Proceedings 31st Annual Conference on Computational Complexity*, page 4, 2016.
4. Anurag Anshu, Aleksandrs Belovs, Shalev Ben-David, Mika Göös, Rahul Jain, Robin Kothari, Troy Lee, and Miklos Santha. Separations in communication complexity using cheat sheets and information complexity. In *Proceedings 57th Annual IEEE Symposium on Foundations of Computer Science*, pages 555–564, 2016.
5. Ziv Bar-Yossef, Thathachar 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.
6. Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to Compress Interactive Communication. *SIAM Journal on Computing*, 42(3):1327–1363, 2013. doi:10.1137/100811969.
7. Yosi Ben-Asher and Ilan Newman. Decision Trees with AND, OR Queries. In *Proceedings 10th Annual Structure in Complexity Theory Conference*, pages 74–81, 1995. doi:10.1109/SCT.1995.514729.
8. Shalev Ben-David and Robin Kothari. Randomized Query Complexity of Sabotaged and Composed Functions. *Theory of Computing*, 14(1):1–27, 2018. doi:10.4086/toc.2018.v014a005.
9. Mark Braverman and Anup Rao. Information Equals Amortized Communication. *IEEE Transactions on Information Theory*, 60(10):6058–6069, 2014.
10. Harry Buhrman, Ilan Newman, Hein Röhrig, and Ronald de Wolf. Robust Polynomials and Quantum Algorithms. *Theory Comput. Syst.*, 40(4):379–395, 2007. doi:10.1007/s00224-006-1313-z.
11. Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Chi-Chih Yao. Informational Complexity and the Direct Sum Problem for Simultaneous Message Complexity. In *Proceedings 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 270–278, 2001. doi:10.1109/SFCS.2001.959901.
12. Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The Bit Extraction Problem or t-Resilient Functions. In *Proceedings 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 396–407, 1985. doi:10.1109/SFCS.1985.55.
13. Andrew Drucker. Improved direct product theorems for randomized query complexity. *Computational Complexity*, 21(2):197–244, 2012.
14. Tomás Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized Communication Complexity. *SIAM Journal on Computing*, 24(4):736–750, 1995. doi:10.1137/S0097539792235864.
15. Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication. In *Proceedings 55th Annual IEEE Symposium on Foundations of Computer Science*, pages 176–185, 2014.
16. Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings 56th Annual IEEE Symposium on Foundations of Computer Science*, pages 1077–1088, 2015.
17. Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-Communication Lifting for BPP. In *Proceedings 58th Annual IEEE Symposium on Foundations of Computer Science*, 2017.
18. Russell Impagliazzo, Ran Raz, and Avi Wigderson. A Direct Product Theorem. In *Proceedings 9th Annual Structure in Complexity Theory Conference*, pages 88–96, 1994. doi:10.1109/SCT.1994.315814.

19 Rahul Jain, Hartmut Klauck, and Miklos Santha. Optimal direct sum results for deterministic and randomized decision tree complexity. *Inf. Process. Lett.*, 110(20):893–897, 2010. doi:10.1016/j.ipl.2010.07.020.

20 Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5(3):191–204, 1995.

21 Marco Molinaro, David P. Woodruff, and Grigory Yaroslavtsev. Beating the Direct Sum Theorem in Communication Complexity with Implications for Sketching. In *Proceedings 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1738–1756, 2013. doi:10.1137/1.9781611973105.125.

22 Marco Molinaro, David P Woodruff, and Grigory Yaroslavtsev. Amplification of one-way information complexity via codes and noise sensitivity. In *Proceedings 42nd Annual International Colloquium on Automata, Languages, and Programming*, pages 960–972. Springer, 2015.

23 Sagnik Mukhopadhyay and Swagato Sanyal. Towards Better Separation between Deterministic and Randomized Query Complexity. In *Proceedings 35th Annual Foundations of Software Technology and Theoretical Computer Science*, pages 206–220, 2015.

24 Noam Nisan, Steven Rudich, and Michael E. Saks. Products and Help Bits in Decision Trees. *SIAM Journal on Computing*, 28(3):1035–1050, 1999. doi:10.1137/S0097539795282444.

25 Ronen Shaltiel. Towards proving strong direct product theorems. *Computational Complexity*, 12(1-2):1–22, 2003.

26 Alexander Sherstov. The Power of Asymmetry in Constant-Depth Circuits. *SIAM Journal on Computing*, 47(6):2362–2434, 2018. doi:10.1137/16M1064477.