Randomised Composition and Small-Bias Minimax

Shalev Ben-David
University of Waterloo

Eric Blais
University of Waterloo

Mika Göös
EPFL

Gilbert Maystre
EPFL

***Abstract*—We prove[1] two results about randomised query complexity $\mathsf{R}(f)$. First, we introduce a *linearised* complexity measure LR and show that it satisfies an *inner-optimal* composition theorem: $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{R}(f)\mathsf{LR}(g))$ for all partial f and g, and moreover, LR is the largest possible measure with this property. In particular, LR can be polynomially larger than previous measures that satisfy an inner composition theorem, such as the max-conflict complexity of Gavinsky, Lee, Santha, and Sanyal (ICALP 2019).**

Our second result addresses a question of Yao (FOCS 1977). He asked if -error *expected* query complexity $\overline{\mathsf{R}}\ (f)$ admits a distributional characterisation relative to some hard input distribution. Vereshchagin (TCS 1998) answered this question affirmatively in the bounded-error case. We show that an analogous theorem *fails* in the small-bias case $= 1/2 - o(1)$.

I. Introduction

This paper is motivated by the following basic open problem in boolean function complexity theory.

Conjecture 1. $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{R}(f)\mathsf{R}(g))$ *for all total boolean functions* f, g.

Let us unpack what this conjecture is claiming. The randomised ϵ-error query complexity $\mathsf{R}_\epsilon(f)$ of a boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is defined (see [BdW02] for the classic reference) as the least number of queries a randomised algorithm (decision tree) needs to make, on the worst-case input, to the bits x_i of $x \in \{0,1\}^n$ in order to compute $f(x)$ correctly with error at most ϵ. We write $\mathsf{R} \coloneqq \mathsf{R}_{1/3}$ for the bounded-error case. For functions f and g over n and m bits, their composition $f \circ g$ is defined over nm bits by

$$(f \circ g)(x) \ \coloneqq\ f(g(x^1), \ldots, g(x^n))$$

where $x = (x^1, \ldots, x^n) \in (\{0,1\}^m)^n$. In particular, we have $\mathsf{R}(f \circ g) \leq O(\mathsf{R}(f)\mathsf{R}(g)\log \mathsf{R}(f))$ for all f, g. This holds since we can run an algorithm for f with query cost $\mathsf{R}(f)$ and whenever it queries an input bit, we can run, as a subroutine, an ϵ-error algorithm for g of cost $\mathsf{R}_\epsilon(g)$. Setting $\epsilon \ll 1/\mathsf{R}(f)$ makes sure that the errors made by the subroutines do not add up. Moreover, we have $\mathsf{R}_\epsilon(g) \leq O(\mathsf{R}(g)\log(1/\epsilon)) = O(\mathsf{R}(g)\log \mathsf{R}(f))$ by standard error reduction techniques. Conjecture 1 thus postulates that a converse inequality always holds (without the log factor).

The analogue of Conjecture 1 has been long resolved for many other well-studied complexity measures: deterministic query complexity satisfies a perfect multiplicative composition theorem, $\mathsf{D}(f \circ g) = \mathsf{D}(f)\mathsf{D}(g)$ [Sav02], quantum query complexity satisfies $\mathsf{Q}(f \circ g) = \Theta(\mathsf{Q}(f)\mathsf{Q}(g))$ [Rei11], [LMR+11], and yet more examples (degree, certificate complexity, sensitivity, rank) are discussed in [Tal13], [GSS16], [DM21]. In the randomised case, however, the conjecture has proved more delicate, exhibiting a far richer, and more surprising, structure.

[1]This is an extended abstract. For the full version of this article, please refer to [BDBGM22].

Partial counterexamples. Conjecture 1 is known to be false if we relax the requirement that f, g are total and instead consider *partial* functions (promise problems), which are undefined on some inputs x, $f(x) = *$. Indeed, works by Gavinsky, Lee, Santha, and Sanyal [GLSS19] and Ben-David and Blais [BB20b] have culminated in examples of partial functions f, g such that $\mathsf{R}(f \circ g) \leq o(\mathsf{R}(f)\mathsf{R}(g))$. Motivated by these counterexamples, we ask: *What is the best possible composition theorem one can prove for partial functions?*

A. A new composition theorem

Our first result is an *inner-optimal* composition theorem for partial functions. To state this result, we start by introducing a new *linearised* complexity measure defined for a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$ by

$$\mathsf{LR}(f) \ \coloneqq\ \min_R \max_x \frac{\mathrm{cost}(R,x)}{\mathrm{bias}_f(R,x)},$$

- where R ranges over randomised decision trees;
- x ranges over the domain of f, namely, $\mathrm{Dom}(f) \coloneqq f^{-1}(\{0,1\})$;
- $\mathrm{cost}(R,x)$ denotes the *expected* number of queries R makes on input x; and
- $\mathrm{bias}_f(R,x)$ denotes the bias R has of guessing the value $f(x)$ correctly; formally, $\mathrm{bias}_f(R,x) \coloneqq \max\{1 - 2\,\mathrm{err}_f(R,x), 0\}$ where $\mathrm{err}_f(R,x) \coloneqq \Pr_R[R(x) \neq f(x)]$. We often omit the subscript f for brevity.

This definition might seem mysterious at first sight. To get better acquainted with it, let us first note that

$$\forall f\colon \qquad \Omega(\sqrt{\mathsf{R}(f)}) \ \leq\ \mathsf{LR}(f) \ \leq\ O(\mathsf{R}(f)). \tag{1}$$

Indeed, the second inequality follows by considering a bounded-error decision tree R, with $\mathrm{cost}(R,x) \leq \mathsf{R}(f)$ and $\mathrm{bias}(R,x) \geq 1/3$. For the first inequality, if we let R be a randomised tree that achieves the minimum in the definition of $\mathsf{LR}(f)$, we can amplify the bias of R, which is possibly tiny, as follows. On input x we run $R(x)$ repeatedly until we have made a total of $\mathsf{LR}(f)^2$ queries, and then output the majority answer over all runs. We expect this simulation to run $R(x)$ for $\mathsf{LR}(f)^2/\,\mathrm{cost}(R,x) \geq 1/\mathrm{bias}(R,x)^2$ many times, which,

2575-8454/22/$31.00 ©2022 IEEE
DOI 10.1109/FOCS54457.2022.00065

by standard Chernoff bounds, is enough to amplify the bias to a constant. This shows $\mathsf{R}(f) \leq O(\mathsf{LR}(f)^2)$.

Both extremes in (1) can be realised. First, consider the n-bit *parity* function $\textsc{Xor}_n$. It is not hard to see that any randomised tree that achieves bias δ for $\textsc{Xor}_n$ needs to query all the n bits with probability at least δ, resulting in expected query cost at least δn. This shows $\mathsf{LR}(\textsc{Xor}_n) = \mathsf{R}(\textsc{Xor}_n) = n$. Second, consider the partial n-bit *gap-majority* function (here $|x|$ denotes the Hamming weight)

$$\textsc{GapMaj}_n(x) \coloneqq \begin{cases} 1 & \text{if } |x| \geq n/2 + \sqrt{n}, \\ 0 & \text{if } |x| \leq n/2 - \sqrt{n}, \\ * & \text{otherwise.} \end{cases}$$

It is well known that $\mathsf{R}(\textsc{GapMaj}_n) = \Theta(n)$. By contrast, the algorithm R that queries and outputs a uniform random bit of x has $\mathrm{cost}(R, x) = 1$ and $\mathrm{bias}(R, x) \geq \Omega(1/\sqrt{n})$, which shows $\mathsf{LR}(\textsc{GapMaj}_n) \leq O(\sqrt{n})$.

Our first main result shows that a multiplicative composition theorem holds when the inner function is measured according to LR, and moreover, our choice of LR is optimal among all inner complexity measures. Ultimately, these theorems are what lends naturalness to our definition of LR.

Theorem 1. $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{R}(f)\mathsf{LR}(g))$ *for all partial boolean functions* f, g.

Theorem 2. *Theorem 1 is optimal: If* M *is any complexity measure such that* $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{R}(f)\mathsf{M}(g))$ *for all partial* f, g*, then* $\mathsf{LR}(g) \geq \Omega(\mathsf{M}(g))$ *for all partial* g.

Additionally, LR itself satisfies a composition theorem as well.

Theorem 3. $\mathsf{LR}(f \circ g) \geq \Omega(\mathsf{LR}(f)\mathsf{LR}(g))$ *for all partial boolean functions* f, g.

B. Comparison with previous work

The randomised composition conjecture for general boolean functions was first explicitly raised in [BK16]. Several complexity measures have since been shown to satisfy an inner composition theorem, including:

1) (block-)sensitivity s, bs [ABK16],
2) randomised sabotage complexity RS [BK16],
3) randomised complexity R_δ with small-bias error $\delta \coloneqq 1/2 - 1/n^4$ [AGJ+18],
4) max-conflict complexity $\overline{\chi}$ [GLSS19] (also studied in [Li21]).

By our optimality theorem, we have $\mathsf{LR}(f) \geq \Omega(\mathsf{M}(f))$ for all $\mathsf{M} \in \{\mathsf{s}, \mathsf{bs}, \mathsf{RS}, \mathsf{R}_\delta, \overline{\chi}\}$ and all f. In fact, we can show that the largest of the above measures, namely $\overline{\chi}$, can sometimes be polynomially smaller than LR.[2]

[2]Technically, it does not seem to be known in the literature whether R_δ is always at most $\overline{\chi}$; this doesn't matter much for our purposes, as LR is larger than both and it is easy to separate LR from R_δ (for example with the $\textsc{Or}$ function).

Lemma 4. *There exists a partial* f *such that* $\mathsf{LR}(f) \geq \Omega(\overline{\chi}(f)^{1.5})$.

Previous work has also investigated complexity measures M that admit an *outer* composition theorem, that is, $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{M}(f)\mathsf{R}(g))$ for all partial f, g. These measures include:

1) sensitivity s [GJPW18] (which was applied in [AKK16]),
2) fractional block sensitivity fbs [BDG+20],
3) noisy randomised complexity noisyR [BB20b] (also studied in [GTW21]).

In particular, noisyR is known to be *outer-optimal*: if we have $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{M}(f)\mathsf{R}(g))$ for all partial f, g, then $\mathsf{noisyR}(f) \geq \Omega(\mathsf{M}(f))$ for all partial f. Our result can be viewed as an inner analogue of this.

Finally, we mention that randomised composition has also been studied in the *super-multiplicative* regime, where we have examples of functions f, g with $\mathsf{R}(f \circ g) \geq \omega(\mathsf{R}(f)\mathsf{R}(g))$. Tight bounds exist when the outer function is identity [BB19] (building on [JKS10], [BK16]), parity [BKLS20], or majority [BGKW20], [GM21].

C. On small-bias minimax

Our second result addresses a question of Yao [Yao77]. Yao-style minimax theorems are routinely used to construct and analyse hard input distributions (including in our proof of the new composition theorem). For example, R_ϵ admits a distributional characterisation as

$$\mathsf{R}_\epsilon(f) = \max_{\mu} \min_{R \in \mathsf{R}(f,\epsilon,\mu)} \mathrm{depth}(R), \tag{2}$$

where μ ranges over distributions on $\mathrm{Dom}(f)$; the set $\mathsf{R}(f, \epsilon, \mu)$ consists of trees R with $\mathbb{E}_{x\sim\mu}[\mathrm{err}(R, x)] \leq \epsilon$; and $\mathrm{depth}(R)$ is the worst-case cost of R, that is, maximum number of queries over all inputs (and internal randomness if R is randomised). While the worst-case cost setting is perhaps what is most widely studied up to this day, Yao's original paper discussed, in fact, exclusively the expected cost setting. It is the expected cost setting that is currently undergoing a renaissance as it has proven important in the randomised composition literature surveyed above (Section I-B).

Minimax for expected cost. We define the *ϵ-error expected query complexity* and the *ϵ-error distributional expected query complexity* by

$$\overline{\mathsf{R}}_\epsilon(f) \coloneqq \min_{R \in \mathsf{R}(f,\epsilon)} \max_{x} \mathrm{cost}(R, x),$$
$$\overline{\mathsf{D}}_\epsilon(f) \coloneqq \max_{\mu} \min_{R \in \mathsf{R}(f,\epsilon,\mu)} \mathrm{cost}(R, \mu),$$

where $\mathsf{R}(f, \epsilon)$ is the set of randomised trees R such that $\mathrm{err}(R, x) \leq \epsilon$ for all inputs x; and $\mathrm{cost}(R, \mu) \coloneqq \mathbb{E}_{x\sim\mu}[\mathrm{cost}(R, x)]$ is the expected cost over μ (and internal randomness of R). We note that the set $\mathsf{R}(f, \epsilon, \mu)$ is sometimes restricted to contain only deterministic algorithms wlog (as can be done in (2)), but in the expected cost setting this may not necessarily be t†the case (see Open Problem 4); hence we allow $\mathsf{R}(f, \epsilon, \mu)$ to contain randomised trees.

Yao showed an exact distributional characterisation for zero-error algorithms, namely, $\overline{\mathsf{R}}_0(f) = \overline{\mathsf{D}}_0(f)$, and moreover, the

optimal distributional algorithm is deterministic. He asked if a similar characterisation holds in the case $\epsilon > 0$. He observed that the "easy" direction of minimax, $\overline{\mathsf{D}}_\epsilon(f) \le \overline{\mathsf{R}}_\epsilon(f)$, certainly holds (although Yao's version of this inequality had some loss in parameters as he was restricted to deterministic algorithms). Vereshchagin [Ver98] proved the "hard" direction with a modest loss in parameters; in summary,

$$\overline{\mathsf{D}}_\epsilon(f) \;\le\; \overline{\mathsf{R}}_\epsilon(f) \;\le\; 2\overline{\mathsf{D}}_{\epsilon/2}(f).$$

These bounds give a satisfying distributional characterisation in the bounded-error case. What happens in the small-bias case $\epsilon = 1/2 - o(1)$? Our second result shows that, surprisingly, the distributional characterisation fails in a particularly strong sense. We write $\dot{\delta} = (1-\delta)/2$ for short.

Theorem 5. *There is an n-bit partial function f and a bias $\delta(n) = o(1)$ such that $\overline{\mathsf{R}}_{\dot{\delta}}(f) \ge \overline{\mathsf{D}}_{\dot{\delta}}(f)^{1+\Omega(1)}$.*

This theorem says that there is no way to capture $\overline{\mathsf{R}}_\epsilon(f)$ relative to a *single* hard distribution. However, there does exist a distributional characterisation using a pair of distributions, as we explore next.

D. Discussion: How are our two results related?

Suppose we want to prove an inner composition theorem. All the previous proofs [BK16], [AGJ+18], [GLSS19] revolve around the following high-level idea. Let R be a randomised tree that on input x seeks to compute $f(g(x^1), \dots, g(x^n))$. The tree can invest different numbers of queries q_i to different components x^i, making $q = \sum_i q_i$ queries in total. If we had a complexity measure $\mathsf{M}(g)$ that allowed us to bound the bias the tree has for the i-th component $g(x^i)$ as *a linear function* of q_i—say, the bias for $g(x^i)$ is at most $q_i/\mathsf{M}(g)$—then, by *linearity of expectation*, the expected total sum of the biases for all components $g(x^1), \dots, g(x^n)$ is at most $q/\mathsf{M}(g)$. This would allow us to track the total progress R is making across all the inner functions.

What is the largest such "linearised" measure M? The most natural attempt at a definition (which the authors of this paper studied for a long time before finding the correct definition of LR) runs as follows. The measure should be such that with $q := \overline{\mathsf{R}}_{\dot{\delta}}(f)$ queries one gets bias at most $\delta \le q/\mathsf{M}(f)$. Optimising for $\mathsf{M}(f)$ this suggest the following definition (a competitor for LR)

$$\mathsf{ULR}(f) \;:=\; \min_{\delta>0} \frac{\overline{\mathsf{R}}_{\dot{\delta}}(f)}{\delta} \;=\; \min_R \max_{x,y} \frac{\mathrm{cost}(R,x)}{\mathrm{bias}(R,y)}.$$

We call it *uniform*-LR, since the tree R that achieves the minimum has an upper bound on $\mathrm{cost}(R,x)$ that is uniformly the same for all x, and similarly there is a uniform lower bound on $\mathrm{bias}(R,x)$ for all x. By contrast, the definition of $\mathsf{LR}(f)$ is *non-uniform*: a tree R that achieves the minimum for $\mathsf{LR}(f)$ has only a bound on the cost/bias *ratio*, but the individual cost and bias functions can vary wildly as a function of x.

We clearly have $\mathsf{LR}(f) \le \mathsf{ULR}(f)$ by definition. How about the converse? It is enlightening to compare the distributional characterisations of these two measures, which can be derived using the recent minimax theorem for ratios of bilinear functions [BB20a]:

$$\mathsf{LR}(f) := \min_R \max_x \frac{\mathrm{cost}(R,x)}{\mathrm{bias}(R,x)} = \max_\mu \min_R \frac{\mathrm{cost}(R,\mu)}{\mathrm{bias}(R,\mu)},$$
$$\mathsf{ULR}(f) := \min_R \max_{x,y} \frac{\mathrm{cost}(R,x)}{\mathrm{bias}(R,y)} = \max_{\mu,\nu} \min_R \frac{\mathrm{cost}(R,\mu)}{\mathrm{bias}(R,\nu)}.$$

Here, LR is captured using a single hard distribution μ such that both cost and bias are measured against it. By contrast, ULR needs a pair of distributions μ, ν, one to measure the cost, one to measure the bias. The upshot is that we are able to show that the two measures are polynomially separated.

Theorem 6. *There is an n-bit partial function f such that $\mathsf{ULR}(f) \ge \Omega(\mathsf{LR}(f)^{5/4}) \ge n^{\Omega(1)}$.*

Our optimality theorem thus implies that ULR *cannot* satisfy an inner composition theorem. This means that our attempt at finding a "linearised" measure at the start of this section missed a subtlety, namely, Yao's question: can we capture our measure relative to a single hard distribution? Our proof of the composition theorem will rely heavily on the fact that LR admits a single hard distribution. Our separation of LR and ULR is what allows us to prove the impossibility of capturing $\overline{\mathsf{R}}_\epsilon(f)$ relative to a single distribution. Indeed, Theorem 5 can be derived from Theorem 6 simply as follows.

Proof of Theorem 5. Let f be as in Theorem 6 and let R be a randomised tree witnessing $\mathsf{LR}(f)$. We may assume wlog that $\mathrm{cost}(R,x) \ge 1$ for all x. (If R places a lot of weight on a 0-cost tree, we may re-weight R without affecting the cost/bias ratio; see Lemma 9 for details.) Thus $\mathrm{bias}(R,x) \ge 1/n =: \delta$ for all x. We show the following inequalities, which would prove Theorem 5.

$$\overline{\mathsf{R}}_{\dot{\delta}}(f) \;\ge\; \delta \cdot \mathsf{ULR}(f), \tag{3}$$
$$\overline{\mathsf{D}}_{\dot{\delta}}(f) \;\le\; \delta \cdot \mathsf{LR}(f), \tag{4}$$

Indeed, (3) holds since $\mathsf{ULR}(f) \le \overline{\mathsf{R}}_{\dot{\delta}}(f)/\delta$ by the definition of ULR. For (4) consider any input distribution μ. Define R' as the randomised tree that with probability $\lambda := \delta/\mathrm{bias}(R,\mu)$ runs R, and with probability $1-\lambda$ makes no queries and outputs a random 0/1 answer. Then $\mathrm{bias}(R',\mu) = \lambda\mathrm{bias}(R,\mu) = \delta$ and $\mathrm{cost}(R',\mu) = \lambda\,\mathrm{cost}(R,\mu) = \delta\,\mathrm{cost}(R,\mu)/\mathrm{bias}(R,\mu) \le \delta\mathsf{LR}(f)$, as desired. □

E. Techniques

Composition theorem. Our first result, the inner-optimal composition theorem, is proved in Section III. As in other composition theorems for randomised algorithms, we start with a randomised algorithm R for the composition $f \circ g$ as well as hard distributions μ_0 and μ_1 for g (corresponding to distributions on $g^{-1}(0)$ and $g^{-1}(1)$), and we construct a randomised algorithm R' for f whose cost is significantly lower than that of R (we need the cost to decrease by a factor of $\mathsf{LR}(g)$). The algorithm R' will simulate R, but not every query that R makes to the large, mn-sized input to $f \circ g$ will turn into a query to the smaller, n-sized input to f that R'

has access to. Instead, R' will attempt to delay making a true query as long as possible, and instead when R makes a query (i, j) (querying position j inside copy i of an input to g), R' will return an answer that is generated according to μ_0 and μ_1, so long as these two distributions approximately agree on the answer to that query.

So far, this is the same strategy employed by several other composition theorems, including in particular that of [GLSS19]. Our innovation comes from the precise way we choose when to query the bit i versus when to return an artificially-generated query answer to the query (i, j). Specifically, in Section IV, we prove the following simulation theorem for decision trees. Suppose we are given two distributions μ_0 and μ_1, we are asked to answer online queries to the bits of a string sampled from μ_b without knowing the value of b; moreover, suppose we have access to a big red button that, when pressed, provides the value of $b \in \{0, 1\}$. Then there is a strategy to answer these online queries with perfect soundness (i.e. with distribution identical to sampling a string from μ_b) with the following guarantee: if the decision tree that is making the online queries is D, then the probability we press the button is at most $\mathrm{TV}(\mathrm{tran}(D, \mu_0), \mathrm{tran}(D, \mu_1))$ (the total variation distance between the query outputs D receives when run on μ_0 and the query outputs D receives when run on μ_1).

This simulation theorem, though somewhat technical, ends up being stronger than the simulation guarantee used by Gavinsky, Lee, Santha, and Sanyal [GLSS19] to provide their composition result for max-conflict complexity. To get a composition theorem, we need to convert this total variation distance between transcripts into a more natural measure; this can be done via some minimax arguments, and the resulting measure is LR. We note that a similarly structured argument occurred in [BB20b], but the squared-Hellinger distance between the transcripts appeared instead of the total variation distance; in that result, the authors showed that this squared-Hellinger distance between transcripts characterized $\mathsf{R}(g)$, but they failed to construct a randomised algorithm R' for f, instead constructing only a "noisy" randomised algorithm. This gave them the result $\mathsf{R}(f \circ g) = \Omega(\mathsf{noisyR}(f)\mathsf{R}(g))$. In contrast, the total variation distance allows us to get $\mathsf{R}(f)$ on the outside, at the cost of getting only $\mathsf{LR}(g)$ on the inside.

The measure LR is arguably more natural than max-conflict complexity, but the real advantage is that our composition theorem turns out to be the best possible of its type: if $\mathsf{R}(f \circ g) = \Omega(\mathsf{R}(f)\mathsf{M}(g))$ for all partial functions f and g, then $\mathsf{LR}(g) = \Omega(\mathsf{M}(g))$. To show this, we give a characterization of $\mathsf{LR}(g)$ in terms of randomised query complexity: there is a family of partial functions f_m such that for all partial functions g, we have

$$\mathsf{LR}(g) = \Theta\left(\frac{\mathsf{R}(f_m \circ g)}{\mathsf{R}(f_m)}\right),$$

where m is the input size of g. Once we have this, it clearly follows that $\mathsf{R}(f \circ g) = \Omega(\mathsf{R}(f)\mathsf{M}(g))$ implies $\mathsf{LR}(g) = \Omega(\mathsf{M}(g))$. The function family f_m turns out to be the same as the one introduced in [BB20b] (based on a family of relations introduced in [GLSS19]); the randomised query complexity $\mathsf{R}(f_m)$ was already established in that paper, so all we need is an upper bound on $\mathsf{R}(f_m \circ g)$ which uses the existence of an LR-style algorithm for g. The linear dependence on the bias which is built into the definition of $\mathsf{LR}(g)$ turns out to be precisely what is needed to upper bound $\mathsf{R}(f_m \circ g)$ (see Section VI for details).

Failure of small-bias minimax. Our second result, separation of LR and ULR, is proved in the full verison of this article [BDBGM22]. The function f that witnesses the separation $\mathsf{ULR}(f) \geq \Omega(\mathsf{LR}(f)^{5/4})$ is not hard to define. For simplicity, we denote its input length by $N := Bn$ and think of the input as being composed of $B = n^c$ blocks (for some large constant c) of n bits each. We define f as a composition of $\textsc{Maj}_B$ as an outer function, and $\textsc{Xor}_n$ as an inner function, where we are able to switch individual $\textsc{Xor}$-blocks to be easy (requiring $O(1)$ queries) or hard (requiring n queries). Moreover, we make the following promises about the input. Either

(1) all blocks are easy, and a random block has a value with bias $1/n$ towards the majority value; or
(2) $b := n^{-3/4}$ fraction of the blocks are hard, and a random block has bias $\Omega(b)$ towards the majority.

We claim that this function is easy for LR, namely, $\mathsf{LR}(f) = O(n)$. To see this, consider the algorithm R that chooses a block at random, computes it, and outputs its value. For inputs x of type (1) we have $\mathrm{cost}(R, x) = O(1)$ and $\mathrm{bias}(R, x) \geq 1/n$ so that cost/bias ratio is $O(n)$. For inputs x of type (2) we have $\mathrm{cost}(R, x) = bn + (1-b)O(1) \leq O(bn)$ and $\mathrm{bias}(R, x) = b$ so that cost/bias ratio is $O(n)$ again.

The difficult part is to show that $\mathsf{ULR}(f) \geq \Omega(n^{5/4})$. For example, the above algorithm R has ULR-style measure $\max_{x,y} \mathrm{cost}(R, x)/\mathrm{bias}(R, y) = O(bn)/(1/n) = O(n^{5/4})$, and we would like to show that this is optimal. Intuitively, it is hard to get large bias for inputs of type (1) (although query cost is small here) and it is hard to get low query cost for inputs of type (2) (although bias is relatively high here). We first argue that an algorithm that wants to keep $\mathrm{cost}(R, x)$ small uniformly for all x (even those x with high $\mathrm{bias}(R, x)$) cannot afford to solve hard blocks very often. This is formalised by picking an appropriate pair of hard distributions for f according to the minimax formulation. What remains is the following task: Show that any algorithm that does not solve hard blocks, has large cost/bias ratio relative to a *single* hard distribution, that is, show an LR-style lower bound.

To this end, we develop a suite of techniques to prove lower bounds on the cost/bias trade-off achievable by decision trees in the small-bias expected cost setting, which has not really been studied in the literature before. Consequently, we end up having to re-establish some basic facts in the expected-cost setting that have been long known in the worst-case setting. For example, we show any algorithm for $\textsc{GapMaj}_n$ (with $\sqrt{n}$ gap promise) can achieve bias at most $O(\sqrt{\mathrm{cost}/n})$ (see the full version [BDBGM22]). The proof here exploits the "$\textsc{And}$-trick" used by Sherstov [She12] to prove a lower bound on

the (worst-case) randomised communication complexity of the gap-Hamming problem. These techniques also come in handy when we separate LR from $\overline{\chi}$ for the proof of Lemma 4.

F. Open questions

The foremost open question is to resolve Conjecture 1. We can equivalently formulate it as follows.

Open Problem 1 (Conjecture 1 rephrased)**.** *Does* $\mathsf{LR}(f) = \Theta(\mathsf{R}(f))$ *for all total functions* f*?*

One intriguing open problem regarding our new-found measure LR is to show that it is lower-bounded by quantum query complexity Q. Indeed, the bias of a quantum algorithm can be amplified linearly in the query cost, so it seems sensible to conjecture this is so. However, quantum query complexity has mostly been studied in the worst-case setting, and it is unclear how one should even define quantum query complexity in expectation (in such a way that it supports linear bias amplification).

Open Problem 2. *Does it hold that* $\mathsf{LR}(f) \geq \mathsf{Q}(f)$*?*

There is a second reason to care about this question, having to do with the *composition limit* of randomised algorithms. Define $\mathsf{R}^*(f) \coloneqq \lim_{k\to\infty} \mathsf{R}(f^{\circ k})^{1/k}$; this is the limit of the k-th root of the randomised query complexity of the k-fold composition of f. Our results here imply that $\mathsf{R}^*(f) \geq \Omega(\mathsf{LR}(f))$ for all (possibly partial) functions f. Due to the composition theorem for quantum query complexity, it is also known that $\mathsf{R}^*(f) \geq \Omega(\mathsf{Q}(f))$. The above open problem asks whether one of these results dominates the other. More generally, it would be nice to characterize $\mathsf{R}^*(f)$ in terms of a simpler measure (for instance, one which is efficiently computable given the truth table of the function).

Our inner-optimal composition theorem for LR, together with the outer-optimal composition theorem for noisyR [BB20b] give a relatively satisfying picture of composition in the case of partial functions. However, we can still ask whether there remain other *incomparable* composition theorems.

Open Problem 3. *Are there multiplicative composition theorems, stating that* $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{M}_1(f)\mathsf{M}_2(g))$ *for all partial* f, g*, that can sometimes prove better lower bounds than* $\Omega(\max\{\mathsf{R}(f)\mathsf{LR}(g), \mathsf{noisyR}(f)\mathsf{R}(g)\})$*?*

Regarding the failure of the distributational characterization of $\overline{\mathsf{R}}_\epsilon$ in the low bias regime (Theorem 5), one may wonder whether the definition of $\overline{\mathsf{D}}_\epsilon$ should really involve randomized decision trees instead of deterministic ones. As hinted in Section I-C, while considering deterministic trees is the natural choice in the bounded error regime, we feel it might not be in the regime where $\epsilon \approx 1/2$. Indeed, while a randomised decision tree can get cost arbitrarily close to zero for ϵ approaching $1/2$ (by taking an appropriate mixture with the zero-query tree), a deterministic one will get stuck at making one query and thus cost 1. Deciding whether the two versions are equivalent (up to constant factors and additive terms) is our last open question.

Open Problem 4. *Define* $\overline{\mathsf{D}}^\star_\epsilon(f)$ *for a boolean function* f *with*

$$\overline{\mathsf{D}}^\star_\epsilon(f) \coloneqq \max_\mu \min_{D\in\mathsf{D}(f,\epsilon,\mu)} \mathrm{cost}(D,\mu)$$

where $\mathsf{D}(f,\epsilon,\mu)$ *is the set of all deterministic decision trees solving* f *with error at most* ϵ *relative to inputs sampled from* μ*. For any partial* f *and* ϵ*, do we have* $\overline{\mathsf{D}}^\star_\epsilon(f) \leq O(\overline{\mathsf{D}}_\epsilon(f)+1)$*?*

II. Preliminaries

A. Query complexity notation

Fix a natural number $n \in \mathbb{N}$. A total boolean function is a function $f\colon \{0,1\}^n \to \{0,1\}$. We will consider several generalizations of total boolean functions: first, there are *partial* boolean functions, which are defined on a domain which is a subset of $\{0,1\}^n$. We use $\mathrm{Dom}(f) \subseteq \{0,1\}^n$ to denote the domain of such a function. A further way to generalize boolean functions is to expand the input and output alphabets; that is, for finite sets Σ_I and Σ_O, we can consider functions $f\colon \mathrm{Dom}(f) \to \Sigma_O$ with $\mathrm{Dom}(f) \subseteq \Sigma_I^n$, which take in input strings over the alphabet Σ_I and output a symbol in Σ_O.

A still further way to generalize such functions is to consider *relations* instead of partial functions. A relation is a subset of $\Sigma_I^n \times \Sigma_O$, or alternatively, it is a function that maps Σ_I^n to a subset of Σ_O. Any partial function can be viewed as a (total) relation, where on an input x which is not in the domain of the partial function, the corresponding relation relates all output symbols to x (meaning that if x is the input, any output symbol is considered valid).

Given a boolean function f (or, more generally, a relation), we will denote its *deterministic query complexity* by $\mathsf{D}(f)$. This is the minimum height of a *decision tree* D which correctly computes $f(x)$ on any $x \in \mathrm{Dom}(f)$; in other words, it is the minimum number of worst-case adaptive queries required by a deterministic algorithm computing f. For a formal definition, see [BdW02].

In this work we will mostly be dealing with randomised algorithms rather than deterministic ones, so let us more carefully define those. A *randomised query algorithm* or *randomised decision tree* will be a probability distribution over deterministic decision trees. Such deterministic decision trees will have internal nodes labeled by $[n] \coloneqq \{1,2,\ldots,n\}$ (representing the index of the input to query), arcs labeled by Σ_I (representing the symbol we might see after querying an index), and leaves labeled by Σ_O (representing output symbols to return at the end of the algorithm). We will assume that no internal node shares a label with an ancestor, meaning that a deterministic algorithm does not query the same index twice.

For such a randomised algorithm R and for an input $x \in \Sigma_I^n$, we denote by $R(x)$ the random variable we get by sampling a deterministic tree D from R, and returning $D(x)$ (the label of the leaf of D reached after starting from the root and taking the path determined by x). For a function f, we write $\mathrm{err}_f(R,x) \coloneqq \Pr_R[R(x) \neq f(x)]$ (or $\Pr_R[R(x) \notin f(x)]$ if f

is a relation), and we write $\mathrm{bias}^{\pm}_f(R,x) \coloneqq 1 - 2\,\mathrm{err}_f(R,x)$, $\mathrm{bias}_f(R,x) \coloneqq \max\{\mathrm{bias}^{\pm}_f(R,x), 0\}$; we omit the subscript f when it is clear from context.

For a deterministic tree D, let $\mathrm{cost}(D,x)$ be the number of queries D makes on input x; this is the height of the leaf of D that is reached when D is run on x. For a randomised algorithm R, we then define $\mathrm{cost}(R,x) \coloneqq \mathbb{E}_{D\sim R}[\mathrm{cost}(D,x)]$ (this is the expected number of queries R makes when run on x).

We extend both of the above to distributions μ over Σ_I^n instead of just inputs x; that is, define

$$\begin{aligned}\mathrm{bias}^{\pm}_f(R,\mu) &\coloneqq \mathbb{E}_{x\sim\mu}[\mathrm{bias}^{\pm}_f(R,x)]\\ &= \mathbb{E}_{x\sim\mu}\mathbb{E}_{D\sim R}[\mathrm{bias}^{\pm}_f(D,x)]\end{aligned}$$

$$\mathrm{cost}(R,\mu) \coloneqq \mathbb{E}_{x\sim\mu}[\mathrm{cost}(R,x)] = \mathbb{E}_{x\sim\mu}\mathbb{E}_{D\sim R}[\mathrm{cost}(D,x)],$$

with $\mathrm{bias}_f(R,\mu) \coloneqq \max\{\mathrm{bias}^{\pm}_f(R,\mu), 0\}$. We also define $\mathrm{tran}(R,\mu)$ to be the random variable we get by sampling a decision tree D from R, a string x from μ, and returning the pair (D,ℓ), where ℓ is the leaf of D reached when D is run on x. Intuitively, $\mathrm{tran}(R,\mu)$ is the "transcript" when R is run on an input sampled from μ, and such a transcript records all information that an agent running R knows about the input x at the end of the algorithm. We will use $\mathrm{TV}(\mu,\nu) \coloneqq \frac{1}{2}\sum_{x\in\mathcal{X}} |\mu[x] - \nu[x]|$ to denote the total variation distance between distributions μ and ν over set $\mathcal{X}$. Most often, we will employ it with respect to the transcript of R on two different distributions as a way to quantify the extent to which R can tell these distributions apart.

We say that a randomised algorithm R computes f to error ϵ if $\mathrm{err}_f(R,x) \le \epsilon$ for all $x \in \mathrm{Dom}(f)$. We then let $\overline{\mathsf{R}}_\epsilon(f)$ the minimum possible value of $\max_x \mathrm{cost}(R,x)$ over randomised algorithms R satisfying $\mathrm{err}_f(R,x) \le \epsilon$ for all $x \in \mathrm{Dom}(f)$. We also use $\mathsf{R}_\epsilon(f)$ to denote the minimum number T such that there is a randomised algorithm R with $\mathrm{err}_f(R,x) \le \epsilon$ for all $x \in \mathrm{Dom}(f)$ such that all decision trees in the support of R have height at most T. The difference between $\mathsf{R}_\epsilon(f)$ and $\overline{\mathsf{R}}_\epsilon(f)$ is that the former measures the worst-case cost of an algorithm computing f to error ϵ (maximizing over both the input string and the internal randomness), while the latter measures the expected worst-case cost of the algorithm computing f to error ϵ (this still maximizes over the input strings x, but takes an expectation over the internal randomness of the algorithm R).

It is easy to see that $\overline{\mathsf{R}}_\epsilon(f) \le \mathsf{R}_\epsilon(f)$ for all f. The other direction also holds if we tolerate a constant-factor loss, as well as an additive constant loss in ϵ; to see this, note that if we cut off the $\overline{\mathsf{R}}_\epsilon(f)$ algorithm after it makes 10 times more queries than it is expected to, then the probability of reaching such a cutoff is at most $1/10$ by Markov's inequality, and hence the error probability of the algorithm increases by at most $1/10$; this converts an $\overline{\mathsf{R}}_\epsilon(f)$ algorithm into a $\mathsf{R}_\epsilon(f)$ algorithm.

Standard error reduction techniques imply that for a boolean function f, $\mathsf{R}_\epsilon(f)$ is related to $\mathsf{R}_{\epsilon'}(f)$ by a constant factor that depends only on ϵ and ϵ', so long as both are in $(0,1/2)$. For this reason, the value of ϵ does not matter when ϵ is a constant in $(0,1/2)$ (so long as we ignore constant factors and so long as the function is boolean), so we omit ϵ when $\epsilon = 1/3$. The same error reduction property holds for $\overline{\mathsf{R}}_\epsilon(f)$. Combined with the Markov inequality argument above, both $\mathsf{R}(f)$ and $\overline{\mathsf{R}}(f)$ are the same measure (up to constant factors) for a boolean function and for constant values of ϵ.

We warn that these equivalences break if f is not boolean (especially if f is a relation) or if the value of ϵ is not constant; in particular, when $\epsilon = 1/n$ or when $\epsilon = 1/2 - 1/n$, the values of $\mathsf{R}_\epsilon(f)$ and $\overline{\mathsf{R}}_\epsilon(f)$ may differ by more than a constant factor.

B. *Linearised* R

For a (possibly partial) boolean function f on n bits, we define

$$\mathsf{LR}(f) \coloneqq \min_R \max_x \frac{\mathrm{cost}(R,x)}{\mathrm{bias}(R,x)}.$$

Here R ranges over randomised decision trees and x ranges over the domain of f, and we treat $0/0$ as ∞.

We call this measure *linearised randomised query complexity*. The name comes from the linear dependence on the bias achieved by the algorithm. Note that if we wanted to amplify bias γ to constant bias, we would, in general, have to repeat the algorithm $\Theta(1/\gamma^2)$ times to do so. In some sense, then, the measure $\mathsf{R}(f)$ charges $1/\gamma^2$ for an algorithm that achieves bias γ instead of achieving constant bias. The measure $\mathsf{LR}(f)$, in contrast, charges only $1/\gamma$ for such an algorithm, so it can be up to quadratically smaller than $\mathsf{R}(f)$.

A minimax theorem for ratios such as [BB20a] (Theorem 2.18) can show that

$$\mathsf{LR}(f) = \max_\mu \min_D \frac{\mathrm{cost}(D,\mu)}{\mathrm{bias}(D,\mu)}, \tag{5}$$

where D ranges over deterministic decision trees and μ ranges over probability distributions over $\mathrm{Dom}(f)$.

It is not hard to see that the maximizing distribution μ above will place equal weight on 0 and 1 inputs. This is because otherwise, we could take D to be a decision tree that makes 0 queries, and then $\mathrm{cost}(D,\mu)$ would be 0 while $\mathrm{bias}(D,\mu)$ would be positive.

If μ is balanced over 0 and 1 inputs, we may express it as $\mu \coloneqq \mu_0/2 + \mu_1/2$ and it is not hard to show that for the best possible choice of leaf labels for an unlabeled decision tree D, we have

$$\mathrm{bias}(D,\mu)^{\pm} = \mathrm{bias}(D,\mu) = \mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1)).$$

This follows, for example, from [BB20a] (Lemma 3.9); to see this intuitively, recall that $\mathrm{tran}(D,\mu)$ is the random variable for the leaf of D reached when D is run on μ, and note that the best choice of leaf label if D reaches a leaf ℓ is 0 if the probability of D reaching ℓ is higher when run on μ_0 than on μ_1, and it is 1 otherwise. Therefore, the bias for the best choice of leaf labels is the sum, over leaves ℓ of D, of $2\max\{\Pr_{\mu_0}[\ell], \Pr_{\mu_1}[\ell]\} - 1$, which is easily seen to be the total variation distance between the two distributions over leaves.

Given (**??**), we can also write

$$\mathsf{LR}(f) = \max_{\mu_0,\mu_1} \min_{D} \frac{\mathrm{cost}(D, \frac{\mu_0+\mu_1}{2})}{\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))},$$

where μ_0 ranges over probability distributions with support $f^{-1}(0)$ and μ_1 ranges over probability distributions with support $f^{-1}(1)$. Observe that neither the top nor the bottom depend on the leaf labels of D, so we can now assume D is an unlabeled decision tree if we wish. Note also that $\mathrm{cost}(D,\mu)$ is linear in the second argument, so we can write

$$\mathsf{LR}(f) = \max_{\mu_0,\mu_1} \min_{D} \frac{\mathrm{cost}(D,\mu_0) + \mathrm{cost}(D,\mu_1)}{2\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))}.$$

We clearly have

$$\mathsf{LR}(f) \geq \max_{\mu_0,\mu_1} \min_{D} \frac{\min\{\mathrm{cost}(D,\mu_0), \mathrm{cost}(D,\mu_1)\}}{\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))}.$$

Lemma 7. *For any fixed μ_0 and μ_1, we have*

$$\min_{D} \frac{\mathrm{cost}(D,\mu_1)}{\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))} \leq 6 \min_{D} \frac{\mathrm{cost}(D,\mu_0)}{\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))}.$$

Proof. See the full version of this article [BDBGM22]. □

Corollary 8.

$$\mathsf{LR}(f) \geq \max_{\mu_0,\mu_1} \min_{D} \frac{\min\{\mathrm{cost}(D,\mu_0), \mathrm{cost}(D,\mu_1)\}}{\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))} \text{ and}$$
$$\mathsf{LR}(f) \leq 6 \max_{\mu_0,\mu_1} \min_{D} \frac{\min\{\mathrm{cost}(D,\mu_0), \mathrm{cost}(D,\mu_1)\}}{\mathrm{TV}(\mathrm{tran}(D,\mu_0), \mathrm{tran}(D,\mu_1))}$$

One useful property of LR complexity is that up to a multiplicative factor of 2, we can consider only randomised decision trees that always query at least one bit of their input.

Lemma 9. *For every non-constant partial function g, there is a randomised decision tree A that always queries at least one bit of g's input and satisfies, for every x,*

$$\frac{\mathrm{cost}(A,x)}{\mathrm{bias}(A,x)} \leq 2 \cdot \mathsf{LR}(g).$$

Proof. See the full version of this article [BDBGM22]. □

As a corollary, we obtain a universal lower bound on the LR complexity of every non-constant function.

Corollary 10. *For every non-constant partial function g, $\mathsf{LR}(g) \geq \frac{1}{2}$.*

Proof. By Lemma 9, there exists a randomised decision tree A that always queries at least one bit of its input and satisfies $\mathrm{cost}(A,x)/\mathrm{bias}(A,x) \leq 2 \cdot \mathsf{LR}(g)$ for all x in the domain of g. But since A always makes at least one query, $\mathrm{cost}(A,x) \geq 1$. And by definition, $\mathrm{bias}(A,x) \leq 1$, so the cost-bias ratio of A is always bounded below by 1. □

III. An overview of the composition theorem

In the following sections, we prove our inner-optimal composition theorem, Theorems 1 and 2 restated below, along with related results.

Theorem 1. $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{R}(f)\mathsf{LR}(g))$ *for all partial boolean functions f, g.*

Theorem 2. *Theorem 1 is optimal: If M is any complexity measure such that $\mathsf{R}(f \circ g) \geq \Omega(\mathsf{R}(f)\mathsf{M}(g))$ for all partial f, g, then $\mathsf{LR}(g) \geq \Omega(\mathsf{M}(g))$ for all partial g.*

The heart of the proof of Theorem 1 is a simulation theorem showing that for any two distributions μ_0 and μ_1 and any decision tree T, it is possible to simulate T on inputs drawn from μ_b for some initially unknown $b \in \{0,1\}$ while querying the actual value of b with probability bounded by the total variation distance between the two distributions μ_0 and μ_1. This result, Theorem 11, is established in Section IV.

In Section V, we use the simulation theorem to complete the proof of the main composition theorem, Theorem 13, a slightly more general version of Theorem 1. We also use the simulation theorem to establish the perfect composition for LR complexity, Theorem 3, in this section.

The proof of Theorem 2 is completed in Section VI. Finally, in Section VII, we establish the separation between LR complexity and max-conflict complexity of Lemma 4.

IV. Decision tree simulation theorem

An *online decision tree simulator* is a randomised algorithm that is given two distributions μ_0 and μ_1 on inputs $\{0,1\}^n$, oracle access to a bit $b \in \{0,1\}$, and a stream of queries $i_1, \ldots, i_k \in [n]$ that represent the queries made by a decision tree T that is not known to the algorithm. The goal of an online decision tree simulator is to answer the queries according to the distribution μ_b while querying the value of b itself with as small probability as possible. We think of this protocol as having a big red button that gives b, and it tries to pretend to have a sample from μ_b without pressing the button for as long as possible.

Theorem 11. *There exists an online decision tree simulator that simulates the queries of T on μ_b while querying the value of b with probability* $\mathrm{TV}\big(\mathrm{tran}(T,\mu_0), \mathrm{tran}(T,\mu_1)\big)$.

The algorithm that satisfies the theorem is stated below. In the algorithm, $x \in \{0,*,1\}^n$ is a partially defined boolean string: the coordinates labelled with $*$ are undefined. Given a string $x \in \{0,*,1\}^n$, an index $i \in [n]$, and a value $a \in \{0,1\}$, the notation $x^{(i \leftarrow a)}$ denotes the string y which equals x on all coordinates except i, where it takes the value $y_i = a$.

Note that each vertex in a decision tree T corresponds to the partial string $x \in \{0,1,*\}^n$ of the values revealed on the path to that vertex in T. Our main task is to show that each vertex in T (including each leaf) is reached with probability $\mu_b(x)$ in the algorithm and that the probability that we reach x *and* don't reveal b along the way is $\mu_{\min}(x)$.

Algorithm 1: OnlineQuerySimulator(μ_0, μ_1)

for *all* $x \in \{0,*,1\}^n$ **do**
  $\mu_{\min}(x) \leftarrow \min\{\mu_0(x), \mu_1(x)\}$;
$x \leftarrow *^n$;
$b \leftarrow *$;

while *more queries remain* **do**
  $i \leftarrow$ NextQuery;
  $u \leftarrow \mu_{\min}(x^{(i\leftarrow 0)}) + \mu_{\min}(x^{(i\leftarrow 1)})$;
  if $b = *$ **then**
    With probability $1 - u/\mu_{\min}(x)$, query the value of b;
  if $b = *$ **then**
    $x_i \leftarrow \mathrm{Ber}\big(\mu_{\min}(x^{(i\leftarrow 1)})/u\big)$;
  else
    $x_i \leftarrow \mathrm{Ber}\left(\frac{\mu_b(x^{(i\leftarrow 1)}) - \mu_{\min}(x^{(i\leftarrow 1)})}{\mu_b(x) - u}\right)$;

Lemma 12. *For every $x \in \{0,1,*\}^n$, when we run the* OnlineQuerySimulator*, then*

1) *We reach the vertex x with probability $\mu_b(x)$, and*
2) *We reach the vertex x* and *don't query the value b on the way to x with probability $\mu_{\min}(x)$.*

Proof. We prove the claim by induction on the number of defined coordinates on x. The base case corresponds to $x = *^n$, which trivially satisfies both conditions of the claim.

Consider now any $x \neq *^n$. Let z be the parent of x in the decision tree T, and let i denote the coordinate where $z_i = *$ and $x_i \neq *$. Define also y to be x's sibling in T. Let us assume that $x_i = 1$. (The case where $x_i = 0$ is essentially identical.)

By the induction hypothesis, the probability that we reach z and don't query the value b is $\mu_{\min}(z)$. With probability $\big(\mu_{\min}(x) + \mu_{\min}(y)\big)/\mu_{\min}(z)$, we don't query the value of b while processing the query i either. And when this occurs the algorithm next reaches x with probability $\mu_{\min}(x)/\big(\mu_{\min}(x) + \mu_{\min}(y)\big)$. So the overall probability that we reach x without querying b along the way is

$$\mu_{\min}(z) \cdot \frac{\mu_{\min}(x) + \mu_{\min}(y)}{\mu_{\min}(z)} \cdot \frac{\mu_{\min}(x)}{\mu_{\min}(x) + \mu_{\min}(y)} = \mu_{\min}(x).$$

Next, by the induction hypothesis again the probability that we query the value of b either on the way to z or while processing the query i is

$$\begin{aligned}&(\mu_b(z) - \mu_{\min}(z)) + \mu_{\min}(z) \cdot \left(1 - \frac{\mu_{\min}(x) + \mu_{\min}(y)}{\mu_{\min}(z)}\right)\\ &\quad = \mu_b(z) - \big(\mu_{\min}(x) + \mu_{\min}(y)\big).\end{aligned}$$

Then the probability we output x conditioned on having revealed b is

$$\frac{\mu_b(x) - \mu_{\min}(x)}{\mu_b(z) - (\mu_{\min}(x) + \mu_{\min}(y))},$$

so that the overall probability that we reach x and reveal b along the way is $\mu_b(x) - \mu_{\min}(x)$. Therefore, the overall probability that we reach x is $\mu_b(x)$. □

The proof of Theorem 11 is now essentially complete, as it just requires combining the lemma with a simple identity on total variation distance.

Proof of Theorem 11. Lemma 12 implies that the OracleQuerySimulator indeed reaches each leaf with the correct probability $\mu_b(x)$. And the probability that it queries the value of b is $1 - \sum_{\ell \in T} \min\{\mu_0(\ell), \mu_1(\ell)\}$, which is the total variation distance between $\mathrm{tran}(T, \mu_0)$ and $\mathrm{tran}(T, \mu_1)$. □

V. Composition Theorems

The inner-optimal composition theorem, Theorem 1, is established in Section V-A. In fact, we establish a slight generalization of that theorem, stated below in Theorem 13. Then the perfect composition theorem for LR complexity, Theorem 3, is established in Section V-B.

A. Composition for randomised query complexity

For a boolean string $y \in \{0,1\}^n$ and a pair of distributions μ_0, μ_1, we define $y \circ (\mu_0, \mu_1)$ to be the product distribution $\bigotimes_{i=1}^n \mu_{y_i}$. In particular, if μ_0 and μ_1 are hard distributions for the 0- and 1-inputs of g respectively, and if y is an input to f, then $y \circ (\mu_0, \mu_1)$ will give a distribution over the inputs to the composed $f \circ g$ (all of which correspond to the same f-input y).

We prove the following composition theorem, which is a slightly more general version of Theorem 1.

Theorem 13. *Let Σ_I and Σ_O be finite alphabets, and let $n, m \in \mathbb{N}$. Let $f \subseteq \{0,1\}^n \times \Sigma_O$ be a (possibly partial) relation on n bits, and let g: $\mathrm{Dom}(g) \to \{0,1\}$ be a (possibly partial) boolean function, with $\mathrm{Dom}(g) \subseteq \Sigma_I^m$. Let $\epsilon \in [0, 1/2)$. Then*

$$\overline{\mathsf{R}}_\epsilon(f \circ g) \geq \overline{\mathsf{R}}_\epsilon(f)\mathsf{LR}(g)/6.$$

Proof. Let μ_0 and μ_1 be distributions over the 0-inputs and 1-inputs to g, respectively, that maximize the expression in the right-hand side of Corollary 8. Let Π be the online decision tree simulator from Theorem 11. Let R be a randomised algorithm that computes $f \circ g$ to error ϵ using $\overline{\mathsf{R}}_\epsilon(f \circ g)$ expected queries. We describe a randomised algorithm R' for computing f on worst-case inputs.

Given input $y \in \{0,1\}^n$, the algorithm R' will instantiate n copies of Π, which we denote $\Pi_1, \Pi_2, \ldots, \Pi_n$, one for each bit of the input; if protocol Π_i presses the button, it gets y_i (and this causes R' to make a real query to the real input). Each of these copies of Π will assume the distributions to be simulated are μ_0 and μ_1. Then R' will run R, and whenever R makes a query (i, j) (corresponding to querying bit j inside of the i-th copy of g), the algorithm R' will ask Π_i to give an answer to query j, and it will use that answer to determine the next query of R.

Note that since the protocols Π_i are guaranteed to be sound, the outcome of the simulation of R made by R' is precisely the same (in distribution) as the outcome of running R on an input sampled from $y \circ (\mu_0, \mu_1)$. Therefore, by the correctness guarantee of R, the output will be a valid output for $f(y)$

except with error probability ϵ. It remains to show that for each $y \in \mathrm{Dom}(f)$, the expected number of real queries R' makes when run on y is at most $6\overline{\mathsf{R}}_\epsilon(f \circ g)/\mathsf{LR}(g)$.

Fix any $y \in \mathrm{Dom}(f)$. Now, when R' is run on y, let T be the expected number of fake queries it makes; in other words, let $T = \mathrm{cost}(R, y \circ (\mu_0, \mu_1)) \le \overline{\mathsf{R}}_\epsilon(f \circ g)$. For each i, let T_i be the expected number of queries to Π_i that R' makes when run on y, so that $T_1 + T_2 + \cdots + T_n = T$. Let p_i the overall probability that Π_i presses the button when R' runs on y; the sum $q = p_1 + p_2 + \cdots + p_n$ is therefore the expected number of real queries made by R' on y. We would like to show that $q \le 6T/\mathsf{LR}(g)$, or equivalently, $T/q \ge \mathsf{LR}(g)/6$.

Since $T/q = (T_1 + \cdots + T_n)/(p_1 + \cdots + p_n)$, there must be some i such that $T/q \ge T_i/p_i$. It will therefore suffice to show that $T_i/p_i \ge \mathsf{LR}(g)/6$ for all $i \in [n]$. Fix such i, and recall that T_i is the number of (fake) queries R' makes to Π_i when run on y, and p_i is the probability that Π_i presses the button when R' is run on y. Consider the algorithm $R_{y,i}$ which takes in an input x in $\mathrm{Dom}(g)$, generates $n-1$ additional fake inputs to g from the distributions μ_{y_ℓ} for $\ell \ne i$, places the real input x as the i-th input among the n inputs to g, and runs R on this tuple (treating it as an input to $f \circ g$). Note that when $R_{y,i}$ is run on an input from μ_{y_i}, its behavior is exactly the same as the behavior of R when run on $y \circ (\mu_0, \mu_1)$; therefore, it makes T_i expected queries. Consider running $R_{y,i}$ with query answers generated by Π instead of by making real queries; then when Π uses the hidden bit y_i and simulates the distributions μ_0, μ_1, the behavior of $R_{y,i}$ is the same as when we run it on μ_{y_i}, and hence the expected number of queries it makes to Π is T_i and the probability that Π presses the button is exactly p_i.

Now, by Theorem 11, we know that Π presses the button with probability $\mathrm{TV}(\mathrm{tran}(D, \mu_0), \mathrm{tran}(D, \mu_1))$ when simulating a deterministic decision tree D. For a random decision tree such as the one given by $R_{y,i}$, the probability p_i of the button being pressed will be the mixture of the values $\mathrm{TV}(\mathrm{tran}(D, \mu_0), \mathrm{tran}(D, \mu_1))$ for the deterministic decision trees D in the support of $R_{y,i}$. Also, the expected number of queries T_i that $R_{y,i}$ makes is a matching mixture of the expected number of queries made by the decision trees D in the support of $R_{y,i}$; the latter is $\mathrm{cost}(D, \mu_{y_i})$. Hence to lower bound T_i/p_i, it will suffice to lower bound $\frac{\mathrm{cost}(D, \mu_{y_i})}{\mathrm{TV}(\mathrm{tran}(D, \mu_0), \mathrm{tran}(D, \mu_1))}$ for all deterministic decision trees D acting on inputs in $\mathrm{Dom}(g)$. We now write

$$\begin{aligned}
&\frac{\mathrm{cost}(D, \mu_{y_i})}{\mathrm{TV}(\mathrm{tran}(D, \mu_0), \mathrm{tran}(D, \mu_1))} \\
&\quad \ge \frac{\min\{\mathrm{cost}(D, \mu_0), \mathrm{cost}(D, \mu_1)\}}{\mathrm{TV}(\mathrm{tran}(D, \mu_0), \mathrm{tran}(D, \mu_1))} \\
&\quad \ge \mathsf{LR}(g)/6
\end{aligned}$$

(using Corollary 8). The desired result follows. □

B. Composition for LR complexity

Theorem 3. *$\mathsf{LR}(f \circ g) \ge \Omega(\mathsf{LR}(f)\mathsf{LR}(g))$ for all partial boolean functions f, g.*

We actually prove the more explicit result $\mathsf{LR}(f \circ g) \ge \mathsf{LR}(f)\mathsf{LR}(g)/6$.

Proof. The proof is similar to that of Theorem 13. We fix hard distributions μ_0 and μ_1 for $\mathsf{LR}(g)$, and we fix a randomised algorithm R for $f \circ g$ such that $\max_z \mathrm{cost}(R, z)/\mathrm{bias}(R, z) \le \mathsf{LR}(f \circ g)$. We then define a randomised algorithm R' for f; this time, unlike in the proof of Theorem 13, we want R' to solve f in the $\mathsf{LR}(f)$ sense instead of being a randomised algorithm that solves f to error ϵ. We define R' as before: on input $y \in \mathrm{Dom}(f)$, R' instantiates n protocols Π_i, one for each bit of y; it instantiates each with the distributions (μ_0, μ_1), and gives Π_i the hidden bit y_i if it presses the button. Then R' will run R, and whenever R makes a query (i, j) (to the bit j inside the i-th input to g), R' will ask Π_i for bit j.

Note that by the soundness of the protocols Π_i, we have $\mathrm{bias}(R', y) = \mathrm{bias}(R, y \circ (\mu_0, \mu_1))$. We will next show that $\mathrm{cost}(R', y) \le 6\,\mathrm{cost}(R, y \circ (\mu_0, \mu_1))/\mathsf{LR}(g)$; This way, we will have

$$\begin{aligned}
\mathsf{LR}(f) &= \max_y \frac{\mathrm{cost}(R', y)}{\mathrm{bias}(R', y)} \\
&\le \frac{6}{\mathsf{LR}(g)} \max_y \frac{\mathrm{cost}(R, y \circ (\mu_0, \mu_1))}{\mathrm{bias}(R, y \circ (\mu_0, \mu_1))} \\
&\le \frac{6}{\mathsf{LR}(g)} \max_z \frac{\mathrm{cost}(R, z)}{\mathrm{bias}(R, z)} \\
&\le \frac{6\mathsf{LR}(f \circ g)}{\mathsf{LR}(g)}.
\end{aligned}$$

Fix any $y \in \mathrm{Dom}(f)$; it remains to show that $\mathrm{cost}(R, y \circ (\mu_0, \mu_1))/\mathrm{cost}(R', y) \ge \mathsf{LR}(g)/6$. For every $i \in [n]$, let T_i be the expected number of queries R makes to the i-th input on $y \circ (\mu_0, \mu_1)$, and let p_i be the probability that R' queries the i-th bit when run on input y. Let $T = T_1 + \cdots + T_n$, and let $q = p_1 + \cdots + p_n$. We wish to show $T/q \ge \mathsf{LR}(g)/6$. This precise statement was shown in the proof of Theorem 13, which completes this proof as well. □

VI. Optimality of the composition theorem

We complete the proof of Theorem 2 in this section.

Theorem 2. *Theorem 1 is optimal: If M is any complexity measure such that $\mathsf{R}(f \circ g) \ge \Omega(\mathsf{R}(f)\mathsf{M}(g))$ for all partial f, g, then $\mathsf{LR}(g) \ge \Omega(\mathsf{M}(g))$ for all partial g.*

The proof of Theorem 2 is obtained by a characterization of LR complexity in terms of the complexity of functions composed with the *approximate index* partial function $\textsc{ApproxIndex}_k : \{0,1\}^k \times \{0,1,2\}^{2^k} \to \{0,1,*\}$ where $\textsc{ApproxIndex}_k(a, y) = y_a$ if if $y_a \in \{0,1\}$, $y_b = y_a$ for all $|b - a| \le \frac{k}{2} - 2\sqrt{k \log k}$ and $y_b = 2$ for all other b and $\textsc{ApproxIndex}_k(a, y) = *$ else. The randomised query complexity of the approximate index function is as follows.

Lemma 14 ([BB20b, Lemma 27])**.** $\mathsf{R}(\textsc{ApproxIndex}_k) = \Theta\left(\sqrt{k \log k}\right)$.

The key to the proof of Theorem 2 is the following characterization of LR complexity in terms of composition with the approximate index function.

Lemma 15. *For every partial boolean function $g : \Sigma^m \to \{0,1,*\}$, when $k \in \mathbb{N}$ satisfies $\frac{k}{\log k} \geq (36m)^2$ then*

$$\mathsf{LR}(g) = \Theta\left(\frac{\mathsf{R}(\textsc{ApproxIndex}_k \circ g)}{\mathsf{R}(\textsc{ApproxIndex}_k)}\right).$$

Proof. The lemma trivially holds when g is a constant function. For the rest of the proof, fix g to be any non-constant partial function. Theorem 1 implies the upper bound

$$\mathsf{LR}(g) = O\left(\frac{\mathsf{R}(\textsc{ApproxIndex}_k \circ g)}{\mathsf{R}(\textsc{ApproxIndex}_k)}\right).$$

The goal of the remainder of the proof is to establish a matching lower bound by showing that

$$\overline{\mathsf{R}}(\textsc{ApproxIndex}_k \circ g) = O\left(\sqrt{k \log k} \cdot \mathsf{LR}(g)\right).$$

This bound suffices to complete the proof because $\overline{\mathsf{R}}(f) = \Theta(\mathsf{R}(f))$ for every partial function f.

Let R denote a randomised algorithm that satisfies

$$\mathrm{cost}(R, x) \leq 2 \cdot \mathsf{LR}(g) \cdot \mathrm{bias}(R, x)$$

for all x in the domain of g and always queries at least one bit of its input. Such an algorithm is guaranteed to exist by Lemma 9. We define a new randomised algorithm A that proceeds as follows: it runs the algorithm R sequentially on the first instances $x_1, x_2, \ldots, x_\ell$ of g which correspond to the initial address bits of the input to $\textsc{ApproxIndex}_k$. It continues this process until the total number of queries made to the underlying inputs exceeds $36\sqrt{k \log k} \cdot \mathsf{LR}(g)$. By the choice of k and the trivial bound $\mathsf{LR}(g) \leq m$, this process terminates when R has computed the first ℓ instances of g with some biases $b_1, \ldots, b_\ell$ for some $\ell \leq k$. The algorithm A then guesses the value of the remaining $k - \ell$ bits of the address. It finally computes the value of g on the instance corresponding to the address obtained with error probability at most $\frac{1}{9}$ and returns that value.

Let $c_1, \ldots, c_\ell$ denote the query cost incurred by R when running on the ℓ computed instances of g. The random variables $(X_i)_{i \leq k}$ defined by $X_i = \sum_{j \leq i} c_j - \mathrm{cost}(R, x_j)$ form a discrete-time martingale and ℓ is the stopping time of this martingale. By the optional stopping theorem, $\mathrm{E}[X_\ell] = 0$. So $\mathrm{E}[\sum_{i \leq \ell} c_i] = \sum_{i \leq \ell} \mathrm{cost}(R, x_i)$. By Markov's inequality, the probability that the total cost exceeds 6 times the expected cost on the same inputs is at most $1/6$; let us consider from now on only the case when this does not occur. In this case,

$$\sum_{i=1}^{\ell} \mathrm{cost}(R, x_i) \geq \frac{1}{6} \sum_{i=1}^{\ell} c_i \geq 6\sqrt{k \log k} \cdot \mathsf{LR}(g).$$

By our choice of R, the biases $\beta_1, \ldots, \beta_\ell$ on the values $g(x_1), \ldots, g(x_\ell)$ satisfy $\sum_{i=1}^{\ell} \beta_i \geq 3\sqrt{k \log k}$ and so if we let $b \in \{0,1\}^k$ denote the address computed by the algorithm, we observe that

$$\mathrm{E}\big[|b - a|\big] = \sum_{i=1}^{k} \Pr[b_i \neq g(x_i)] \leq \frac{k}{2} - 3\sqrt{k \log k}.$$

Furthermore, each of the k events $b_i \neq g(x_i)$ are independent. So by Hoeffding's bound the probability that more than $\frac{k}{2} - 2\sqrt{k \log k}$ of these events occur is at most $e^{-2 \log^2 k}$, which is less than $\frac{1}{9}$ when $k \geq 3$. When this event does not occur, the address b computed by the algorithm satisfies $x_b = x_a$. Since A lastly computes $g(x_b)$ with error at most $\frac{1}{9}$, in total it computes $\textsc{ApproxIndex}_k \circ g$ with error at most $\frac{1}{3}$.

It remains to show that the expected query cost of the algorithm A satisfies the desired bound. The first round of the algorithm uses at most $36\sqrt{k \log k} \cdot \mathsf{LR}(g)$ queries plus the number of queries of the instance of R run on x_ℓ. In expectation, this additional number of queries is at most $\mathrm{cost}(R, x_\ell) \leq \mathsf{LR}(g)$. And then computing $g(x_b)$ requires another $\mathsf{R}(g) \leq m < \sqrt{k}$ queries. So the overall expected query complexity of A is at most $(36\sqrt{k \log k} + 1) \cdot \mathsf{LR}(g) + \sqrt{k}$. By Corollary 10, $\mathsf{LR}(g) \geq \frac{1}{2}$ for every non-constant function g so this query complexity is bounded above by $O\big(\sqrt{k \log k} \cdot \mathsf{LR}(g)\big)$, as required. □

The proof of Theorem 2 now follows easily from Lemma 15.

Proof of Theorem 2. Let M be a measure that satisfies the condition of the theorem. Then, choosing f to be the $\textsc{ApproxIndex}_k$ function for a large enough value of k and applying Lemma 15, we obtain

$$M(g) = O\left(\frac{\mathsf{R}(\textsc{ApproxIndex}_k \circ g)}{\mathsf{R}(\textsc{ApproxIndex}_k)}\right) = O\left(\mathsf{LR}(g)\right). \quad \square$$

VII. Separation from $\overline{\chi}$

In this section, we exhibit a polynomial separation between LR and $\overline{\chi}$, the max conflict complexity introduced by Gavinsky, Lee, Santha and Sanyal in [GLSS19] (see Section VII-A for a formal definition of $\overline{\chi}$).

Lemma 4. *There exists a partial f such that $\mathsf{LR}(f) \geq \Omega(\overline{\chi}(f)^{1.5})$.*

Proof. The function f we build takes input of size $n^2 + \sqrt{n}$ with format $(x_1, \ldots, x_{n^2}, a_1, \ldots, a_{\sqrt{n}})$. The function value is given as the parity of $\textsc{GapMaj}(x)$ and $\textsc{Xor}(a)$, i.e.:

$$f(x_1, \ldots, x_{n^2}, a_1, \ldots, a_{\sqrt{n}}) = \textsc{GapMaj}^{n^2}_{n^{-1/2}}(x_1, \ldots, x_{n^2}) \oplus \textsc{Xor}_{\sqrt{n}}(a_1, \ldots, a_{\sqrt{n}})$$

$\textsc{GapMaj}^{n^2}_{n^{-1/2}}(x)$ is the majority function on n^2 bits with promise that $|x| \notin [n^2/2 - n^{3/2}, n^2/2 + n^{3/2}]$ so that returning the value of a random index holds bias at least $n^{-1/2}$. Thus, f is a partial function whose domain is constrained by the gap majority instance. Lemma 16 shows that $\mathsf{LR}(f) \geq \Omega\left(n^{3/4}\right)$ and Lemma 17 that $\overline{\chi}(f) \leq O(n^{1/2})$, as desired. □

Lemma 16. $\mathsf{LR}(F) \geq \Omega\left(n^{3/4}\right)$

Proof. See the full version of this article [BDBGM22]. □

A. An upper bound for $\overline{\chi}$

We recall here the definition of max conflict complexity (but see [GLSS19] for an in-depth treatment of the measure). Let f be a fixed boolean function, μ^0, μ^1 a pair of distribution over $f^{-1}(0)$ and $f^{-1}(1)$ respectively and D a deterministic decision tree solving f. For each node v in D, we let $\mu^0|_v, \mu^1|v$ be the distributions conditioned on reaching v and $q(v)$ be the index queried at node v. Furthermore, we associate to each $v \in \mathcal{N}(D)$ a number $R^D_\mu(v)$ inductively. If v is the root of D, we let $R^D_\mu(v) = 1$ and if v is the child of w which is reached when the query answer to $q(w)$ is $b \in \{0,1\}$:

$$R^D_\mu(v) = R^D_\mu(w) \cdot \min\left\{ \Pr_{x\sim\mu^0|_w}[x_{q(w)} = b], \Pr_{x\sim\mu^1|_w}[x_{q(w)} = b] \right\}$$

Finally, we define $\Delta^D_\mu(v)$ for each $v \in \mathcal{N}(D)$ with:

$$\Delta^D_\mu(v) \coloneqq \left|\Pr_{x\sim\mu^0|_w}[x_{q(v)} = 0] - \Pr_{x\sim\mu^1|_w}[x_{q(v)} = 0]\right|$$

$R^D_\mu(v)$ can be interpreted as the probability of reaching node v in a random walk that starts at the root and with probability $\min\{\Pr_{x\sim\mu^0|_v}[x_i = 0], \Pr_{x\sim\mu^1|_v}[x_i = 0]\}$ moves left, with probability $\min\{\Pr_{x\sim\mu^0|_v}[x_i = 1], \Pr_{x\sim\mu^1|_v}[x_i = 1]\}$ moves right and with remaining probability $\Delta^D_\mu(v)$ stops. As such, it holds that $\sum_{v\in\mathcal{N}(D)} \Delta^D_\mu(v) R^D_\mu(v) = 1$ and that for any partition Γ of $\{0,1\}^n$ we have $\sum_{\gamma\in\Gamma} R^D_\mu(\gamma) \leq 1$. The max conflict complexity $\overline{\chi}(f)$ is defined as:

$$\overline{\chi}(f) \coloneqq \max_{\mathcal{Q}} \min_{D\in\mathsf{D}(f)} \mathbb{E}_{\mu\sim\mathcal{Q}} \left[\sum\nolimits_{v\in\mathcal{N}(D)} |v| \Delta^D_\mu(v) R^D_\mu(v) \right]$$

Where $\mathcal{Q}$ ranges over distributions of pairs of distributions over $f^{-1}(0)$ and $f^{-1}(1)$ and $\mathsf{D}(f)$ is the set of all decision tree solving f correctly.

Lemma 17. $\overline{\chi}(F) \leq O(n^{1/2})$

Proof. See the full version of this article [BDBGM22]. □

Acknowledgements: We thank anonymous FOCS reviewers for their comments.

References

[ABK16] Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proceedings of the 48th Symposium on Theory of Computing (STOC)*, pages 863–876. ACM, 2016. doi:10.1145/2897518.2897644.

[AGJ+18] Anurag Anshu, Dmitry Gavinsky, Rahul Jain, Srijita Kundu, Troy Lee, Priyanka Mukhopadhyay, Miklos Santha, and Swagato Sanyal. A composition theorem for randomized query complexity. In *Proceedings of the 37th Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 93, pages 10:1–10:13, 2018.doi:10.4230/LIPIcs.FSTTCS.2017.10.

[AKK16] Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly optimal separations between communication (or query) complexity and partitions. In *Proceedings of the 31st Computational Complexity Conference (CCC)*, pages 4:1–4:14. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.CCC.2016.4.

[BB19] Eric Blais and Joshua Brody. Optimal separation and strong direct sum for randomized query complexity. In *Proceedings of the 34th Computational Complexity Conference (CCC)*, pages 29:1–29:17. Schloss Dagstuhl, 2019. doi:10.4230/LIPIcs.CCC.2019.29.

[BB20a] Shalev Ben-David and Eric Blais. A new minimax theorem for randomized algorithms. In *Proceedings of the 61st Symposium on Foundations of Computer Science (FOCS)*. IEEE, nov 2020. doi:10.1109/focs46700.2020.00045.

[BB20b] Shalev Ben-David and Eric Blais. A tight composition theorem for the randomized query complexity of partial functions. In *Proceedings of the 61st Symposium on Foundations of Computer Science (FOCS)*. IEEE, nov 2020. doi:10.1109/focs46700.2020.00031.

[BDBGM22] Shalev Ben-David, Eric Blais, Mika Göös, and Gilbert Maystre. Randomised composition and small-bias minimax. 2022. URL: https://eccc.weizmann.ac.il/report/2022/112/.

[BDG+20] Andrew Bassilakis, Andrew Drucker, Mika Göös, Lunjia Hu, Weiyun Ma, and Li-Yang Tan. The power of many samples in query complexity. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 168, pages 9:1–9:18. Schloss Dagstuhl, 2020. doi:10.4230/LIPIcs.ICALP.2020.9.

[BdW02] Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002. Complexity and Logic. doi:10.1016/S0304-3975(01)00144-X.

[BGKW20] Shalev Ben-David, Mika Göös, Robin Kothari, and Thomas Watson. When is amplification necessary for composition in randomized query complexity? In *Proceedings of the 24h International Conference on Randomization and Computation (RANDOM)*, volume 176, pages 28:1–28:16. Schloss Dagstuhl, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.28.

[BK16] Shalev Ben-David and Robin Kothari. Randomized query complexity of sabotaged and composed functions. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 55,

pages 60:1–60:14, 2016. doi:10.4230/LIPIcs.ICALP.2016.60.

[BKLS20] Joshua Brody, Jae Tak Kim, Peem Lerdputtipongporn, and Hariharan Srinivasulu. A strong XOR lemma for randomized query complexity, 2020. arXiv:2007.05580.

[DM21] Yogesh Dahiya and Meena Mahajan. On (simple) decision tree rank. In *Proceedings of the 41st oundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 213, pages 15:1–15:16. Schloss Dagstuhl, 2021. doi:10.4230/LIPIcs.FSTTCS.2021.15.

[GJPW18] Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication versus partition number. *ACM Transactions on Computation Theory*, 10(1), 2018. doi:10.1145/3170711.

[GLSS19] Dmitry Gavinsky, Troy Lee, Miklos Santha, and Swagato Sanyal. A composition theorem for randomized query complexity via max-conflict complexity. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 132, pages 64:1–64:13, 2019. doi:10.4230/LIPIcs.ICALP.2019.64.

[GM21] Mika Göös and Gilbert Maystre. A majority lemma for randomised query complexity. In *Proceedings of the 36th Computational Complexity Conference (CCC)*, volume 200, pages 18:1–18:15. Schloss Dagstuhl, 2021. doi:10.4230/LIPIcs.CCC.2021.18.

[GSS16] Justin Gilmer, Michael Saks, and Srikanth Srinivasan. Composition limits and separating examples for some boolean function complexity measures. *Combinatorica*, 36(3):265–311, 2016. doi:10.1007/s00493-014-3189-x.

[GTW21] Uma Girish, Avishay Tal, and Kewen Wu. Fourier Growth of Parity Decision Trees. In *Proceedings of the 36th Computational Complexity Conference (CCC)*, volume 200, pages 39:1–39:36. Schloss Dagstuhl, 2021. doi:10.4230/LIPIcs.CCC.2021.39.

[JKS10] Rahul Jain, Hartmut Klauck, and Miklos Santha. Optimal direct sum results for deterministic and randomized decision tree complexity. *Information Processing Letters*, 110(20):893–897, 2010. doi:10.1016/j.ipl.2010.07.020.

[Li21] Yaqiao Li. Conflict complexity is lower bounded by block sensitivity. *Theoretical Computer Science*, 856:169–172, feb 2021. doi:10.1016/j.tcs.2020.12.038.

[LMR+11] Troy Lee, Rajat Mittal, Ben Reichardt, Robert Spalek, and Mario Szegedy. Quantum query complexity of state conversion. In *Proceedings of the 52nd Symposium on Foundations of Computer Science (FOCS)*, pages 344–353. IEEE, 2011. doi:10.1109/FOCS.2011.75.

[Rei11] Ben Reichardt. Reflections for quantum query algorithms. In *Proceedings of the 22nd Symposium on Discrete Algorithms (SODA)*, pages 560–569, 2011.

[Sav02] Petr Savický. On determinism versus unambiquous nondeterminism for decision trees. Technical Report TR02-009, Electronic Colloquium on Computational Complexity (ECCC), 2002. URL: http://eccc.hpi-web.de/report/2002/009/.

[She12] Alexander A. Sherstov. The communication complexity of gap hamming distance. *Theory of Computing*, 8(8):197–208, 2012. URL: http://www.theoryofcomputing.org/articles/v008a008, doi:10.4086/toc.2012.v008a008.

[Tal13] Avishay Tal. Properties and applications of boolean function composition. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 441–454, 2013. doi:10.1145/2422436.2422485.

[Ver98] Nikolai Vereshchagin. Randomized boolean decision trees: Several remarks. *Theoretical Computer Science*, 207(2):329–342, nov 1998. doi:10.1016/s0304-3975(98)00071-1.

[Yao77] Andrew Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Symposium on Foundations of Computer Science (FOCS)*, pages 222–227, Oct 1977. doi:10.1109/SFCS.1977.24.