A Composition Theorem for Randomized Query Complexity via Max-Conflict Complexity*

Dmitry Gavinsky
Institute of Mathematics, Czech Academy of Sciences, 115 67 Žitna 25, Praha 1, Czech Republic

Troy Lee
Centre for Quantum Software and Information, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia
troyjlee@gmail.com

Miklos Santha
CNRS, IRIF, Université de Paris, 75205 Paris, France
Centre for Quantum Technologies, National University of Singapore, Singapore 117543
MajuLab, UMI 3654, Singapore
santha@irif.fr

Swagato Sanyal
Indian Institute of Technology Kharagpur, India
swagato@cse.iitkgp.ac.in

Abstract

For any relation $f \subseteq \{0,1\}^n \times S$ and any partial Boolean function $g : \{0,1\}^m \to \{0,1,*\}$, we show that

$$\mathsf{R}_{1/3}(f \circ g^n) \in \Omega(\mathsf{R}_{4/9}(f) \cdot \sqrt{\mathsf{R}_{1/3}(g)}),$$

where $\mathsf{R}_\epsilon(\cdot)$ stands for the *bounded-error randomized query complexity* with error at most ϵ, and $f \circ g^n \subseteq (\{0,1\}^m)^n \times S$ denotes the composition of f with n instances of g.

The new composition theorem is *optimal*, at least, for the general case of relational problems: A relation f_0 and a partial Boolean function g_0 are constructed, such that $\mathsf{R}_{4/9}(f_0) \in \Theta(\sqrt{n})$, $\mathsf{R}_{1/3}(g_0) \in \Theta(n)$ and $\mathsf{R}_{1/3}(f_0 \circ g_0^n) \in \Theta(n)$.

The theorem is proved via introducing a new complexity measure, *max-conflict complexity*, denoted by $\bar{\chi}(\cdot)$. Its investigation shows that $\bar{\chi}(g) \in \Omega(\sqrt{\mathsf{R}_{1/3}(g)})$ for any partial Boolean function g and $\mathsf{R}_{1/3}(f \circ g^n) \in \Omega(\mathsf{R}_{4/9}(f) \cdot \bar{\chi}(g))$ for any relation f, which readily implies the composition statement. It is further shown that $\bar{\chi}(g)$ is always at least as large as the *sabotage complexity* of g.

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

Keywords and phrases query complexity, lower bounds

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.64

Category Track A: Algorithms, Complexity and Games

Related Version `https://arxiv.org/abs/1811.10752` [6]

Funding Part of this work was conducted while T.L. and S.S. were at the Nanyang Technological University and the Centre for Quantum Technologies, supported by the Singapore National Research Foundation under NRF RF Award No. NRF-NRFF2013-13. This work was additionally supported by the Singapore National Research Foundation, the Prime Minister's Office, Singapore and the Ministry of Education, Singapore under the Research Centres of Excellence programme under research grant R 710-000-012-135. This research was supported in part by the QuantERA ERA-NET Cofund project QuantAlgo. D.G. is partially funded by the grant 19-27871X of GA ČR.

* This paper is a merger of [5] and [12], together with some new results.

Acknowledgements We thank Rahul Jain for useful discussions. We thank Srijita Kundu and Jevgēnijs Vihrovs for their helpful comments on the manuscript. We thank Yuval Filmus for suggesting to look at the min-max version of conflict complexity, which led to the development of max-conflict complexity. We thank the anonymous reviewers for their helpful comments.

1 Introduction

For a relational problem $f \subseteq \{0,1\}^n \times S$ and a partial Boolean function $g : \{0,1\}^m \to \{0,1,*\}$, their *composition* $f \circ g^n \subseteq (\{0,1\}^m)^n \times S$ is defined as

$$f \circ g^n(x) = \begin{cases} S & \text{if } * \in \{g(x_i) \big| i \in [n]\}; \\ f(g(x_1), \dots, g(x_n)) & \text{otherwise.} \end{cases}$$

Relating the complexity of $f \circ g^n$ to the complexities of f and g is a natural research problem.

A *query algorithm* for computing h is allowed to query *individual bits* of the input x, with the goal of outputting $h(x)$ (or an element of $h(x)$ if the problem is a relational one). The *query complexity of an algorithm* is the maximum possible number of queries that it makes.

Query algorithms can be *deterministic*, *randomized* or *quantum*, where the latter two classes allow for (bounded) errors. The corresponding *query complexity of a function* – denoted, respectively, by $\mathsf{D}(h)$, $\mathsf{R}(h)$ or $\mathsf{Q}(h)$ – is the minimal query complexity of an algorithm that belongs to the corresponding class and computes h with error $1/3$[1]. Section 2 contains formal definitions of various query complexity measures.

It is easy to see that $\mathsf{D}(f \circ g^n) \le \mathsf{D}(f) \cdot \mathsf{D}(g)$. [2] For the cases of randomized and quantum query complexity the argument is slightly more subtle, though very similar conceptually; in particular, both $\mathsf{R}(f \circ g^n) \in O(\mathsf{R}(f) \cdot \mathsf{R}(g) \cdot \log n)$ and $\mathsf{Q}(f \circ g^n) \in O(\mathsf{Q}(f) \cdot \mathsf{Q}(g))$ hold. [3]

Showing a strong *lower bounds* on the query complexity of $f \circ g^n$ (preferably, matching the trivial upper bound) is often more interesting, the corresponding statements are sometimes called *composition theorems.* Such results can lead to further theoretical developments (e.g., separating complexity measures, as well as different classes in structural complexity).

For deterministic query complexity it has been shown [10, 13] that

$$\mathsf{D}(f \circ g^n) = \mathsf{D}(f) \cdot \mathsf{D}(g),$$

which means that *the trivial query algorithm for $f \circ g^n$ described above is optimal.* Similarly, for bounded-error quantum query complexity it has been shown [8, 11] that

$$\mathsf{Q}(f \circ g^n) \in \Theta(\mathsf{Q}(f) \cdot \mathsf{Q}(g)).$$

Prior to this work, the randomized query complexity of composition has remained an open problem. We partially solve it for the most general case of composition: namely, letting f be a *relational problem* and g be a *partial Boolean function.* [4]

[1] In general, $\mathsf{R}_\epsilon(h)$ (resp. $\mathsf{Q}_\epsilon(h)$) stands for the randomized (resp. quantum) query complexity with respect to error ϵ.

[2] To compute $f \circ g^n$, one can simulate an optimal query algorithm for f, serving every query of this algorithm by running an optimal query algorithm for g.

[3] The multiplicative factor of $\log n$ in the case of $\mathsf{R}(h)$ is due to the need to reduce the error in computing each instance of g to $O(1/n)$; in the quantum case this can be handled in a more elegant, "lossless" way.

[4] Letting g be a relation seems to result in a rather awkward definition of $f \circ g^n$. Letting g be a non-Boolean promise function doesn't seem to lead to any interesting development (the original version of this work [5] has demonstrated the same composition result for an arbitrary partial g; switching to the Boolean case in the current version has allowed somewhat clearer presentation).

▶ **Theorem 1.** *For any relation $f \subseteq \{0,1\}^n \times S$ and any partial Boolean function $g : \{0,1\}^m \to \{0,1,*\}$,*

$$\mathsf{R}_{1/3}(f \circ g^n) \in \Omega\Big(\mathsf{R}_{4/9}(f) \cdot \sqrt{\mathsf{R}_{1/3}(g)}\Big).$$

Note that the above lower bound *does not match the trivial upper bound*, so its optimality has to be addressed separately. That is done via constructing an example where the above bound is tight: in other words, while some *incomparable* lower bounds on $\mathsf{R}_{1/3}(f \circ g^n)$ are conceivable, the statement of Theorem 1 is *a strongest possible in general.* [5]

▶ **Theorem 2.** *There exists a relation $f_0 \subseteq \{0,1\}^n \times \{0,1\}^n$ and a partial Boolean function $g_0 : \{0,1\}^n \to \{0,1,*\}$, such that*

$$\mathsf{R}_{4/9}(f_0) \in \Theta\big(\sqrt{n}\big),\ \mathsf{R}_{1/3}(g_0) \in \Theta(n) \text{ and } \mathsf{R}_{1/3}(f_0 \circ g_0^n) \in \Theta(n).$$

Our approach

We introduce a new complexity measure of Boolean functions, the *max-conflict complexity*, denoted by $\bar{\chi}(g)$. We show that $\bar{\chi}(g)$ is a quadratically tight lower bound on randomized query complexity of a (partial) function g.

▶ **Theorem 3.** *For any partial Boolean function $g : \{0,1\}^m \to \{0,1,*\}$,*

$$\bar{\chi}(g) \in \Omega\Big(\sqrt{\mathsf{R}_{1/3}(g)}\Big).$$

The main technical ingredient of this work is the following composition statement for the max-conflict complexity.

▶ **Theorem 4.** *For any relation $f \subseteq \{0,1\}^n \times S$ and any partial Boolean function $g : \{0,1\}^m \to \{0,1,*\}$,*

$$\mathsf{R}_{1/3}(f \circ g^n) \in \Omega\big(\mathsf{R}_{4/9}(f) \cdot \bar{\chi}(g)\big).$$

Combining Theorem 3 with Theorem 4 implies Theorem 1.

Previous work

In the special case of f being a partial function and g being a total one, a significant progress has been made by Ben-David and Kothari [4], who showed recently that

$$R_{1/3}(f \circ g^n) \in \Omega\left(R_{1/3}(f) \cdot \sqrt{\frac{R_0(g)}{\log R_0(g)}}\right). \tag{1}$$

To prove the above statement, the authors have introduced and investigated a new complexity measure of Boolean functions, *sabotage complexity*, denoted by $\mathrm{RS}(g)$. This notion has a very natural definition and is of independent interest. In this work we show that max-conflict complexity is always lower-bounded by the sabotage complexity of the same function.

▶ **Theorem 5.** *For any partial Boolean function $g : \{0,1\}^m \to \{0,1,*\}$,*

$$\bar{\chi}(g) \geq \mathrm{RS}(g).$$

Theorem 5 along with Theorem 4 imply (1).

[5] The following construction also witnesses the possibility of $\mathsf{R}(f \circ g^n) \in O(\mathsf{R}(g))$ when $\mathsf{R}(f) \in \Omega\big(\sqrt{n}\big)$ – in other words, *it is not, in general, true, that composition with a "hard" relation makes a Boolean function harder for randomized query algorithms.*

1.1 Proof Technique

At a high level, the proof of Theorem 4 follows the structure of the proof by Anshu et al. [2] and Ben-David and Kothari [4]. We show that for every probability distribution η over the input space $\{0,1\}^n$ of f, there exists a deterministic query algorithm $\mathcal{A}$ that makes $O(\mathsf{R}_{1/3}(f \circ g^n)/\sqrt{\mathsf{R}_{1/3}(g)})$ queries in the worst case, and computes f with high probability, $\Pr_{z\sim\eta}[(z, \mathcal{A}(z)) \in f] \geq 5/9$. By the minimax principle (Fact 4) this implies Theorem 4.

We do this by using a query algorithm for $f \circ g^n$ to construct a query algorithm for f. We define a sampling procedure that for any $z \in \{0,1\}^n$ samples $x = (x_1, \ldots, x_n)$ such that $(z, s) \in f$ if and only if $(x, s) \in f \circ g^n$. This procedure is defined in terms of $\mathcal{Q}$, which is a probability distribution over pairs of distributions (μ_0, μ_1), where μ_0 is supported on $g^{-1}(0)$ and μ_1 is supported on $g^{-1}(1)$. We define a distribution γ_η over $(\{0,1\}^m)^n$ in terms of this sampling process as follows:

1. Sample $z = (z_1, \ldots, z_n)$ from $\{0,1\}^n$ according to η.
2. Independently sample $(\mu_0^{(i)}, \mu_1^{(i)})$ from $\mathcal{Q}$ for $i = 1, \ldots, n$.
3. Sample $x_i = (x_i^{(1)}, \ldots, x_i^{(m)})$ according to $\mu_{z_i}^{(i)}$ for $i = 1, \ldots, n$. Return $x = (x_1, \ldots, x_n)$.

Notice that steps (1) and (2) are independent and the order in which they are performed does not matter. For future reference, for a fixed z let $\gamma_z(\mathcal{Q})$ be the probability distribution defined by the last two steps.

Now γ_η is simply a probability distribution over $(\{0,1\}^m)^n$. Thus by the minimax principle (Fact 4 below), there is a deterministic query algorithm $\mathcal{A}'$ of worst-case complexity at most $\mathsf{R}_{1/3}(f \circ g^n)$ such that $\Pr_{x\sim\gamma_\eta}[(x, \mathcal{A}'(x)) \in f \circ g^n] \geq 2/3$. We first use $\mathcal{A}'$ to construct a randomized query algorithm T for f with bounded *expected* query complexity and error at most $1/3$. The final algorithm $\mathcal{A}$ will be a truncation of T which has bounded worst-case complexity and error at most $4/9$.

On input z, the algorithm T seeks to sample a string x from $\gamma_z(\mathcal{Q})$, and run $\mathcal{A}'$ on x. Put another way, $\gamma_z(\mathcal{Q})$ induces a probability distribution over the leaves of $\mathcal{A}'$, and the goal of T is to sample a leaf of $\mathcal{A}'$ according to this distribution. Since for each $s \in S$, $(x, s) \in f \circ g^n$ if and only if $(z, s) \in f$, and $\Pr_{x\sim\gamma_\eta}[(x, \mathcal{A}'(x)) \in f \circ g^n] \geq 2/3$, we have that $\Pr_{z\sim\eta}[(z, T(z)) \in f] \geq 2/3$. Thus T meets the accuracy requirement.

The catch, of course, is to specify how T samples from $\gamma_z(\mathcal{Q})$ *without making too many queries to* z. To sample x_i from $\mu_{z_i}^{(i)}$ seems to require knowledge of z_i, and thus T would have to query all of z.

To bypass this problem, we remember that $\mathcal{A}'$, being an efficient algorithm, will query only a few bits of x. This allows us to sample x bit by bit as and when they are queried by $\mathcal{A}'$. To see this more clearly, consider a run of T where the pairs of distributions $(\mu_0^{(1)}, \mu_1^{(1)}), \ldots, (\mu_0^{(n)}, \mu_1^{(n)})$ were chosen in step (2) of the sampling procedure. Suppose that T is trying to simulate $\mathcal{A}'$ at a vertex v where $x_i^{(j)}$ is queried. To respond to this query, T will sample $x_i^{(j)}$ from its marginal distribution according to $\mu_{z_i}^{(i)}$ conditioned on the event $x \in v$. Let the following be the marginal distributions of $x_i^{(j)}$ for the two possible values of z_i.

	$\Pr_{x_i\sim\mu_{z_i}^{(i)}}[x_i^{(j)} = 0 \mid x \in v]$	$\Pr_{x_i\sim\mu_{z_i}^{(i)}}[x_i^{(j)} = 1 \mid x \in v]$
$z_i = 0$	p_0	$1 - p_0$
$z_i = 1$	p_1	$1 - p_1$

Without loss of generality, assume that $p_0 \leq p_1$. T answers the query by the procedure BITSAMPLER given in Algorithm 1. Note that the bit returned by BITSAMPLER has the desired distribution. The step in which BITSAMPLER returns the bit depends on the value

Algorithm 1: BITSAMPLER (suppose $p_0 \leq p_1$).

```
1  Sample $r \sim [0,1]$ uniformly at random.
2  if $r < p_0$ then
3      return 0.
4
5  else if $r > p_1$ then
6      return 1.
7
8  else
9      query $z_i$.
10     if $r \leq p_{z_i}$ then
11         return 0.
12     else
13         return 1.
```

of r sampled in step 1. In particular, z_i is queried if and only if $r \in [p_0, p_1]$, and the bit is returned in step 11 or 13. Such a query to z_i contributes to the query complexity of T. Thus the probability that T makes a query when the underlying simulation of $\mathcal{A}'$ is at vertex v is $(p_1 - p_0)$. We refer to this quantity as $\Delta(v)$. It plays an important role in our analysis (in particular, in the proof of Theorem 6 that can be found in [6]).

Our sampling procedure and the tools we use to bound its cost is reminiscent of work of Barak et al. [3] in communication complexity. They look at a communication analog of our setting where two players are trying to sample a leaf in a communication protocol while communicating as little as possible.

1.1.1 Conflict complexity and max-conflict complexity

Bounding the query complexity of T naturally suggests the quantities that we define in this work: the conflict complexity $\chi(g)$ and the max-conflict complexity $\bar{\chi}(g)$ of a partial Boolean function g. A formal definition can be found in Section 4; here we give the high-level idea and motivation behind these quantities.

Forget about T for a moment and just consider a deterministic query algorithm $\mathcal{B}$ computing the partial function $g \subseteq \{0,1\}^m \times \{0,1\}$. Let μ_0, μ_1 be distributions with support on $g^{-1}(0), g^{-1}(1)$, respectively. For each vertex $v \in \mathcal{B}$ let $p_0(v)$ (respectively $p_1(v)$) be the probability that the answer to the query at v is 0 on input $x \sim \mu_0$ (respectively $x \sim \mu_1$), conditioned on x reaching v. Now we can imagine a process $\mathcal{P}(\mathcal{B}, \mu_0, \mu_1)$ that runs BITSAMPLER on the tree $\mathcal{B}$: $\mathcal{P}(\mathcal{B}, \mu_0, \mu_1)$ begins at the root, and at a vertex v in $\mathcal{B}$ it uniformly chooses a random real number $r \in [0,1]$. If $r < \min\{p_0(v), p_1(v)\}$ then the query is "answered" 0 and it moves to the left child. If $r > \max\{p_0(v), p_1(v)\}$ then the query is "answered" 1 and it moves to the right child. If $r \in [\min\{p_0(v), p_1(v)\}, \max\{p_0(v), p_1(v)\}]$ then the process halts. The conflict complexity $\chi(\mathcal{B}, (\mu_0, \mu_1))$ is the expected number of vertices this process visits before halting. The conflict complexity of g is defined to be

$$\chi(g) = \max_{(\mu_0,\mu_1)} \min_{T} \chi(T, (\mu_0, \mu_1)) \enspace ,$$

where the minimum is taken over trees T that compute g. For *max-conflict complexity* we enlarge the set over which we maximize. Let $\mathcal{Q}$ be a distribution over pairs of distributions (μ_0, μ_1), where $\text{supp}(\mu_0) \subseteq g^{-1}(0), \text{supp}(\mu_1) \subseteq g^{-1}(1)$ for each pair (μ_0, μ_1) in the support of $\mathcal{Q}$. Let $\chi(\mathcal{B}, \mathcal{Q}) = \mathsf{E}_{(\mu_0,\mu_1)\sim\mathcal{Q}}\left[\chi(\mathcal{B}, (\mu_0, \mu_1))\right]$. The max-conflict complexity $\bar{\chi}(g)$ is defined as

$$\bar{\chi}(g) = \max_{\mathcal{Q}} \min_{T} \chi(T, \mathcal{Q}) \ ,$$

where the minimum is taken over trees T that compute g. Clearly, the max-conflict complexity is at least as large as the conflict complexity.

To motivate the max-conflict complexity, note that the query complexity of T is the number of times step 9 in BITSAMPLER is executed, i.e. when the random number $r \in [p_0, p_1]$. In the definition of T we will choose $\mathcal{Q}$ to achieve the optimal value in the definition of $\bar{\chi}(g)$. Then intuitively one expects that for each i, T queries z_i only after $\mathcal{A}'$ makes about $\bar{\chi}(g)$ queries into x_i. By means of a direct sum theorem for max-conflict complexity we make this intuition rigorous and prove that the expected query complexity of T is at most $\mathsf{R}_{1/3}(f \circ g^n)/\bar{\chi}(g)$. We refer the reader to [6] for a formal proof.

1.1.2 $\bar{\chi}(g)$ and $\mathsf{R}(g)$

Note that applying Theorem 4 with the outer function $f(z) = z_1$ shows that $\mathsf{R}_{1/3}(g) \in \Omega(\bar{\chi}(g))$. We complete the proof of Theorem 1 by showing that max-conflict complexity is a quadratically tight lower bound on randomized query complexity, even for partial functions g. In fact, we show the stronger result that this is true even for the conflict complexity.

▶ **Theorem 6.** *For any partial Boolean function $g \subseteq \{0,1\}^m \times \{0,1\}$,*

$$\chi(g) \in \Omega\Big(\sqrt{\mathsf{R}_{1/3}(g)}\Big).$$

A proof of Theorem 6 can be found in [6]. At a high level, our proof is reminiscent of the result of [3] on compressing communication protocols in that both look at a random sampling process to navigate a tree, and relate the probability of this process needing to query or communicate at a node to the amount of information that is learned at the node.

To prove $\mathsf{R}(g) \in O(\chi(g)^2)$, we again resort to the minimax principle; we show that for each probability distribution μ over the valid inputs to g, there is an accurate and efficient distributional query algorithm for g. For $b \in \{0,1\}$, let μ_b be the distribution obtained by conditioning μ on the event $g(x) = b$. By the definition of $\chi(g)$, there is a query algorithm $\mathcal{B}$ such that the following is true: if its queries are served by BITSAMPLER, step 9 is executed within expected $\chi(\mathcal{B}, \mu_0, \mu_1) \le \chi(g)$ queries. Note that at a vertex v which queries i, the probability that step 9 is executed is $\Delta(v) = |\Pr_{\mu_0}[x_i = 0 \mid x \text{ at } v] - \Pr_{\mu_1}[x_i = 0 \mid x \text{ at } v]|$. This roughly implies that for a typical vertex v of $\mathcal{B}$, $\Delta(v)$ is at least about $\frac{1}{\chi(g)}$. By a technical claim this implies that the query outcome at v carries about $\frac{1}{\chi(g)^2}$ bits of information about $g(x)$. Using the *chain rule of mutual information*, we can show that the mutual information between $g(x)$ and the outcomes of first $O(\chi(g))^2$ queries by $\mathcal{B}$ is $\Omega(1)$. This enables us to conclude that we can infer the value of $g(x)$ with success probability $1/2 + \Omega(1)$ from the transcript of $\mathcal{B}$ restricted to the first $O(\chi(g)^2)$ queries. The distributional algorithm of g for μ is simply the algorithm $\mathcal{B}$ terminated after $O(\chi(g)^2)$ queries.

1.1.3 $\bar{\chi}(g)$ and $\mathrm{RS}(g)$

To see why $\bar{\chi}(g) \ge \mathrm{RS}(g)$, we first give an alternative characterization of $\mathrm{RS}(g)$. For a deterministic tree T computing g and strings x, y such that $g(x) \ne g(y)$, let $\text{sep}_T(x, y)$ be the depth of the node v in T such that x and y both reach v yet $x_{q(v)} \ne y_{q(v)}$, where $q(v)$

is the index queried at v. Let $\mathcal{T}$ be a zero-error randomized protocol for g, i.e. $\mathcal{T}$ is a probability distribution supported on deterministic trees that compute g. Then we have (for a proof see [6])

$$\mathrm{RS}(g) = \min_{\mathcal{T}} \max_{\substack{x,y \\ g(x)\neq g(y)}} \mathsf{E}_{T\sim\mathcal{T}}[\mathrm{sep}_T(x,y)] \ .$$

By von Neumann's minimax theorem [14], this is equal to

$$\mathrm{RS}(g) = \max_p \min_T \mathsf{E}_{(x,y)\sim p}[\mathrm{sep}_T(x,y)] \ .$$

Here, the max is taken over distributions p on pairs (x,y) where $g(x)\neq g(y)$, and the min is taken over deterministic trees T computing g.

We have seen that the definition of $\bar{\chi}(g)$ is

$$\bar{\chi}(g) = \max_{\mathcal{Q}} \min_T \mathsf{E}_{(\mu_0,\mu_1)\sim\mathcal{Q}} \left[\chi(T,(\mu_0,\mu_1))\right] \ ,$$

where $\mathcal{Q}$ is a distribution over pairs (μ_0,μ_1) and T is a deterministic tree computing g. When (μ_0,μ_1) are taken to be singleton distributions, i.e. μ_0 puts all its weight on a single x with $g(x)=0$, and μ_1 puts all its weight on a single y with $g(y)=1$, it can be shown that $\chi(T,(\mu_0,\mu_1)) = \mathrm{sep}_T(x,y)$ (see [6] for details). Thus $\bar{\chi}(g)$ is at least as large as the sabotage complexity of g as $\mathcal{Q}$ is allowed to be a distribution over general (μ_0,μ_1), not just singleton distributions.

2 Preliminaries

Let $g \subseteq \{0,1\}^m \times \{0,1\}$ be a partial Boolean function. For $b\in\{0,1\}$, $g^{-1}(b)$ is defined to be the set of strings x in $\{0,1\}^m$ for which $(x,b)\in g$ and $(x,\bar{b})\notin g$. We refer to $g^{-1}(0)\cup g^{-1}(1)$ as the set of valid inputs to g. We assume that for all strings $y\notin g^{-1}(0)\cup g^{-1}(1)$, both $(y,0)$ and $(y,1)$ are in g. For a string $x\in g^{-1}(0)\cup g^{-1}(1)$, $g(x)$ refers to the unique bit b such that $(x,b)\in g$. All the probability distributions μ over the domain of a partial Boolean function g in this paper are assumed to have support on $g^{-1}(0)\cup g^{-1}(1)$. Thus $g(x)$ is well-defined for any x in the support of μ.

Let S be any set. Let $h\subseteq\{0,1\}^k\times S$ be any relation. Consider query algorithms $\mathcal{A}$ that accept a string $x\in\{0,1\}^k$ as input, query various bits of x, and produce an element of S as output. We denote the output by $\mathcal{A}(x)$.

▶ **Definition 1** (Deterministic query complexity). *A deterministic query algorithm $\mathcal{A}$ is said to compute h if $(x,\mathcal{A}(x))\in h$ for all $x\in\{0,1\}^k$. The deterministic query complexity $\mathsf{D}(h)$ of h is the minimum over all deterministic query algorithms $\mathcal{A}$ computing h of the maximum number of queries made by $\mathcal{A}$ over $x\in\{0,1\}^k$.*

▶ **Definition 2** (Bounded-error randomized query complexity). *Let $\epsilon\in[0,1/2)$. We say that a randomized query algorithm $\mathcal{A}$ computes h with error ϵ if $\Pr[(x,\mathcal{A}(x))\in h]\geq 1-\epsilon$ for all $x\in\{0,1\}^k$. The bounded-error randomized query complexity $\mathsf{R}_\epsilon(h)$ of h is the minimum over all randomized query algorithms $\mathcal{A}$ computing h with error ϵ of the maximum number of queries made by $\mathcal{A}$ over all $x\in\{0,1\}^k$ and the internal randomness of $\mathcal{A}$.*

▶ **Definition 3** (Distributional query complexity). *Let μ a distribution on the input space $\{0,1\}^k$ of h, and $\epsilon\in[0,1/2)$. We say that a deterministic query algorithm $\mathcal{A}$ computes h with distributional error ϵ on μ if $\Pr_{x\sim\mu}[(x,\mathcal{A}(x))\in h]\geq 1-\epsilon$. The distributional query complexity $\mathsf{D}^\mu_\epsilon(h)$ of h is the minimum over deterministic algorithms $\mathcal{A}$ computing h with distributional error ϵ on μ of the maximum over $x\in\{0,1\}^k$ of the number of queries made by $\mathcal{A}$ on x.*

We will use the minimax principle in our proofs to go between distributional and randomized query complexity.

▶ **Fact 4** (Minimax principle). *For any integer $k > 0$, set S, and relation $h \subseteq \{0,1\}^k \times \mathcal{S}$,*

$$\mathsf{R}_\epsilon(h) = \max_\mu \mathsf{D}_\epsilon^\mu(h).$$

A proof of Fact 4 can be found in [6].

Let μ be a probability distribution over $\{0,1\}^k$. We use $\text{supp}(\mu)$ to denote the support of μ. By $x \sim \mu$ we mean that x is a random string drawn from μ. Let $C \subseteq \{0,1\}^k$ be an arbitrary set such that $\Pr_{x\sim\mu}[x \in C] = \sum_{y\in C} \mu(y) > 0$. Then $\mu \mid C$ is defined to be the probability distribution obtained by conditioning μ on the event that the sampled string belongs to C, i.e.,

$$(\mu \mid C)(x) = \begin{cases} 0 & \text{if } x \notin C \\ \frac{\mu(x)}{\sum_{y\in C}\mu(y)} & \text{if } x \in C \end{cases}$$

For a distribution $\mathcal{Q}$ over pairs of distributions (μ_0, μ_1), let $\text{supp}_0(\mathcal{Q}) = \cup\{\text{supp}(\mu_0) : \exists \mu_1, (\mu_0, \mu_1) \in \text{supp}(\mathcal{Q})\}$. Similarly let $\text{supp}_1(\mathcal{Q}) = \cup\{\text{supp}(\mu_1) : \exists \mu_0, (\mu_0, \mu_1) \in \text{supp}(\mathcal{Q})\}$. We say that $\mathcal{Q}$ is *consistent* if $\text{supp}_0(\mathcal{Q})$ and $\text{supp}_1(\mathcal{Q})$ are disjoint sets. We say that $\mathcal{Q}$ is consistent with a (partial) function g if $\text{supp}_0(\mathcal{Q}) \subseteq g^{-1}(0)$ and $\text{supp}_1(\mathcal{Q}) \subseteq g^{-1}(1)$.

▶ **Definition 5** (Subcube, co-dimension). *A subset* $\mathsf{C} \subseteq \{0,1\}^m$ *is called a subcube if there exists a set* $S \subseteq \{1,\ldots,m\}$ *of indices and an* assignment function $\sigma : S \to \{0,1\}$ *such that* $\mathsf{C} = \{x \in \{0,1\}^m : \forall i \in S, x_i = \sigma(i)\}$*. The co-dimension* $\mathsf{codim}(C)$ *of C is defined to be $|S|$.*

Now we define the composition of a relation and a partial Boolean function.

▶ **Definition 6** (Composition of a relation and a partial Boolean function). *Let $f \subseteq \{0,1\}^n \times S$ and $g \subseteq \{0,1\}^m \times \{0,1\}$ be a relation and a partial Boolean function respectively. The composed relation $f \circ g^n \subseteq (\{0,1\}^m)^n \times S$ is defined as follows: For $x = (x^{(1)}, \ldots, x^{(n)}) \in (\{0,1\}^m)^n$ and $s \in S$, $(x, s) \in f \circ g^n$ if and only if one of the following holds:*
- *$x_i \notin g^{-1}(0) \cup g^{-1}(1)$ for some $i \in \{1,\ldots,n\}$.*
- *$x_i \in g^{-1}(0) \cup g^{-1}(1)$ for each $i \in \{1,\ldots,n\}$ and $((g(x_1),\ldots,g(x_n)), s) \in f$.*

We will often view a deterministic query algorithm as a binary decision tree. In each vertex v of the tree, an input variable is queried. Depending on the outcome of the query, the computation goes to a child of v. The child of v corresponding to outcome b of the query is denoted by v_b.

The set of inputs that lead the computation of a decision tree to a certain vertex is a subcube. We will use the same symbol (e.g. v) to refer to a vertex as well as the subcube associated with it.

The execution of a decision tree terminates at some leaf. If the tree computes some relation $h \subseteq \{0,1\}^k \times S$, the leaves are labelled by elements of S, and the tree outputs the label of the leaf at which it terminates. We will also consider decision tree with unlabelled leaves (see Section 4).

3 Conflict Complexity

In this section, we define the conflict complexity and max-conflict complexity of a partial Boolean function g on m bits. For this, we will need to introduce some notation related to a deterministic decision tree T. For a node $v \in T$, let $\pi(v) = \perp$ if v is the root and $\pi(v)$ be the parent of v otherwise. Let $q(v)$ be the index that is queried at v in T, and let $d_T(v)$ be the number of vertices on the unique path in T from the root to v. The depth of the root is 1.

Now fix a partial function $g \subseteq \{0,1\}^m \times \{0,1\}$ and probability distributions μ_0, μ_1 over $g^{-1}(0), g^{-1}(1)$, respectively. Let T be a tree that computes g. For a node $v \in T$ let $p_0(v) = \Pr_{\mu_0}[x_{q(v)} = 0 | x \text{ at } v]$ and $p_1(v) = \Pr_{\mu_1}[x_{q(v)} = 0 | x \text{ at } v]$, and

$$R(v) = \begin{cases} 1 & \text{if } v \text{ is the root} \\ R(\pi(v)) \cdot \min\{\Pr_{\mu_0}[x \to v | x \text{ at } \pi(v)], \Pr_{\mu_1}[x \to v | x \text{ at } \pi(v)]\} & \text{otherwise} \end{cases} .$$

Also define

$$\Delta(v) = |p_0(v) - p_1(v)| \ .$$

To gather intuition about these quantities, imagine a random walk on T that begins at the root. At a node v, this walk moves to the left child with probability $\min\{p_0(v), p_1(v)\}$, and it moves to the right child with probability $1 - \max\{p_0(v), p_1(v)\}$. With the remaining probability, $\Delta(v)$, it terminates at v. Note that for any tree T computing g we have $\sum_{v \in T} \Delta(v) R(v) = 1$. This is because the walk always terminates before it reaches a leaf of T. In particular, this means that $\sum_{v \in T} d_T(v) \Delta(v) R(v)$ – the expected number of steps the walk takes before it terminates – is always at most the depth of the tree T.

▶ **Definition 7** (Conflict complexity and max-conflict complexity). *Let g be a partial function. For distributions μ_0, μ_1 with $\mathrm{supp}(\mu_b) \subseteq g^{-1}(b)$ for $b \in \{0,1\}$, and a deterministic decision tree T computing g, define*

$$\chi(T, (\mu_0, \mu_1)) = \sum_{v \in T} d_T(v) \Delta(v) R(v) \ .$$

The conflict complexity of g is

$$\chi(g) = \max_{\mu_0, \mu_1} \min_T \chi(T, (\mu_0, \mu_1)) \ ,$$

where the maximum is over all pairs of distributions (μ_0, μ_1) supported on $g^{-1}(0)$ and $g^{-1}(1)$ respectively, and the minimum is taken over all deterministic trees T computing g. For $\mathcal{Q}$ a distribution over pairs satisfying $\mathrm{supp}_b(\mathcal{Q}) \subseteq g^{-1}(b)$ for $b \in \{0,1\}$, and T a deterministic tree computing g, let $\chi(T, \mathcal{Q}) = \mathsf{E}_{(\mu_0,\mu_1) \sim \mathcal{Q}}[\chi(T, (\mu_0, \mu_1))]$. Finally, the max-conflict complexity of g is

$$\bar{\chi}(g) = \max_{\mathcal{Q}} \min_T \chi(T, \mathcal{Q}) \ ,$$

where the maximum is taken over $\mathcal{Q}$ with $\mathrm{supp}_b(\mathcal{Q}) \subseteq g^{-1}(b)$ for $b \in \{0,1\}$, and the minimum is taken over deterministic trees T computing g.

We can extend the definition of conflict complexity and max-conflict complexity to more general query processes that do not necessarily compute a function. We first need the notion of FULL.

▶ **Definition 8.** *For a deterministic tree T and pair of distributions (μ_0, μ_1) with disjoint support, we say that $(T, (\mu_0, \mu_1))$ is* FULL *if $\sum_{v \in T} \Delta(v) R(v) = 1$, i.e. if the random walk described above terminates with probability 1. We say that $(T, \mathcal{Q})$ is* FULL *if $(T, (\mu_0, \mu_1))$ is* FULL *for each $(\mu_0, \mu_1) \in \mathrm{supp}(\mathcal{Q})$.*

▶ **Definition 9.** *For a deterministic tree T and pair of distributions (μ_0, μ_1) such that $(T, (\mu_0, \mu_1))$ is* FULL*, define $\chi(T, (\mu_0, \mu_1)) = \sum_{v \in T} d_T(v) \Delta(v) R(v)$. For a distribution $\mathcal{Q}$ such that $(T, \mathcal{Q})$ is* FULL*, define $\chi(T, \mathcal{Q}) = \mathsf{E}_{(\mu_0,\mu_1) \sim \mathcal{Q}}[\chi(T, (\mu_0, \mu_1))]$.*

3.1 Comparison with other query measures

Li [9] shows that the conflict complexity of a total Boolean function g is at least the block sensitivity of g. As mentioned in Section 1.1.3, in this work we show that the max-conflict complexity of a (partial) function g is at least as large as the sabotage complexity of g. For a total Boolean function g, Ben-David and Kothari [4] show that the sabotage complexity of g is at least as large as the fractional block sensitivity of g [1, 13, 7], which in turn is at least as large as the block sensitivity. They also show examples where the sabotage complexity is much larger than the partition bound, quantum query complexity and approximate polynomial degree, thus the same holds for max-conflict complexity as well.

▶ **Theorem 7.** *Let $g \subseteq \{0,1\}^m \times \{0,1\}$ be a partial function. Then $\bar{\chi}(g) \geq \mathrm{RS}(g)$.*

A proof of Theorem 7 can be found in [6].

4 Query Process

We now come to the most important definition of the paper, that of the query process $\mathcal{P}(\mathcal{B}, \mathcal{Q})$. Let $t > 0$ be any integer and $\mathcal{B}$ be any deterministic query algorithm that runs on inputs in $(\{0,1\}^m)^t$. Let $x = (x_i^{(j)})_{\substack{i=1,\ldots,t \\ j=1,\ldots,m}}$ be a generic input to $\mathcal{B}$, and let x_i stand for $(x_i^{(j)})_{j=1,\ldots,m}$. For a vertex v of $\mathcal{B}$, let $v^{(i)}$ denote the subcube in v corresponding to x_i, i.e., $v = v^{(1)} \times \ldots \times v^{(t)}$. Recall from Section 2 that v_b stands for the child of v corresponding to the query outcome being b, for $b \in \{0,1\}$.

The query process $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ runs on an input $z \in \{0,1\}^t$ and uses the BITSAMPLER (Algorithm 1) routine to simulate the queries of $\mathcal{B}$ to x when it can. This process is the heart of how we will transform an algorithm for $f \circ g^n$ into a query efficient algorithm for f.

▶ **Definition 10** (Query process $\mathcal{P}(\mathcal{B}, \mathcal{Q})$). *Let $\mathcal{B}$ be a decision tree that runs on inputs $(\{0,1\}^m)^t$. Let $\mathcal{Q}$ be a consistent probability distribution over pairs of distributions (μ_0, μ_1). The query process $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ is run on an input $z \in \{0,1\}^t$ and is defined by Algorithm 2.*

A few comments about Definition 10. First, we think of $\mathcal{B}$ and $\mathcal{P}$ as query procedures that query input variables and terminate. In particular, they do not have to produce outputs, i.e. their leaves do not have to be labeled. Also note that in Algorithm 2 the segment from line 9 to line 19 corresponds to the BITSAMPLER procedure in Algorithm 1. Queries to the input bits z_i are made in line 15, which corresponds to step 9 of BITSAMPLER.

We now present an important structural result about $\mathcal{P}(\mathcal{B}, \mathcal{Q})$. In particular, this formally proves that the procedure BITSAMPLER given in Algorithm 1 samples the bits from the right distribution.

▶ **Theorem 8.** *Let $\mathcal{B}$ be a deterministic decision tree running on inputs from $(\{0,1\}^m)^t$, and let v be a vertex in $\mathcal{B}$. Let $A_z(v, \mathcal{Q})$ be the event that $\mathcal{P}(\mathcal{B}, \mathcal{Q})$, when run on z, reaches node v. Let $B_z(v, \mathcal{Q})$ be the event that for a random input x sampled from $\gamma_z(\mathcal{Q})$, the computation of $\mathcal{B}$ reaches v. Then for every $z \in \{0,1\}^t$ and each vertex v of $\mathcal{B}$,*

$$\Pr[A_z(v, \mathcal{Q})] = \Pr[B_z(v, \mathcal{Q})] \ .$$

A proof of Theorem 8 is given in [6].

We will be interested in the number of queries $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ is able to simulate before making a query to z_i. To this end, let the random variable $\mathcal{N}_i(\mathcal{B}, z, \mathcal{Q})$ stand for the value of the variable N_i in Algorithm 2 after the termination of $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ on input z. Note that $\mathcal{N}_i$ depends on the randomness in the choices of r (step 9) and also on the randomness in $\mathcal{Q}$ in the choice of distributions $(\mu_0^{(k)}, \mu_1^{(k)})$ (step 4).

Algorithm 2: $\mathcal{P}(\mathcal{B}, \mathcal{Q})$.

```
Input: $z = (z_1, \dots, z_t) \in \{0,1\}^t$.
for $1 \leq k \leq t$ do
    $\mathsf{QUERY}_k \leftarrow 0$.                    // Indicates if $z_k$ is queried.
    $\mathsf{N}_k \leftarrow 0$.          // Counts references to $x_k$ till $z_k$ is queried.
    Sample $(\mu_0^{(k)}, \mu_1^{(k)})$ from $\mathcal{Q}$.
$v \leftarrow$Root of $\mathcal{B}$                       // Corresponds to $(\{0,1\}^m)^t$.
while $v$ is not a leaf of $\mathcal{B}$ do
    Let $q(v) = (i, j)$, the $j^{th}$ coordinate of $x_i$
    if $\mathsf{QUERY}_i = 0$ then
        Sample a fresh real number $r \sim [0,1]$ uniformly at random.
        if $r < \min_b \Pr_{x_i \sim \mu_b^{(i)}}[x_i^{(j)} = 0 \mid x_i \in v^{(i)}]$ then
            $v \leftarrow v_0$.
        else if $r > \max_b \Pr_{x_i \sim \mu_b^{(i)}}[x_i^{(j)} = 0 \mid x_i \in v^{(i)}]$ then
            $v \leftarrow v_1$.
        else
            Query $z_i$. $\mathsf{QUERY}_i \leftarrow 1$.
            if $r \leq \Pr_{x_i \sim \mu_{z_i}^{(i)}}[x_i^{(j)} = 0 \mid x_i \in v^{(i)}]$ then
                $v \leftarrow v_0$.
            else
                $v \leftarrow v_1$.
        $\mathsf{N}_i \leftarrow \mathsf{N}_i + 1$.
    else
        $b \leftarrow \begin{cases} 1 & \text{with probability } \Pr_{x_i \sim \mu_{z_i}^{(i)}}[x_i^{(j)} = 1 \mid x_i \in v^{(i)}] \\ 0 & \text{with probability } \Pr_{x_i \sim \mu_{z_i}^{(i)}}[x_i^{(j)} = 0 \mid x_i \in v^{(i)}] \end{cases}$
        $v \leftarrow v_b$
```

4.1 Relating $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ to max-conflict complexity

A key to our composition theorem will be relating the number of simulated queries made by $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ to max-conflict complexity, which we do in this section. Let $\mathcal{B}$ be a query algorithm taking inputs from $\{0,1\}^m$. In this case, $\mathcal{N}_1(\mathcal{B}, 1, \mathcal{Q}) = \mathcal{N}_1(\mathcal{B}, 0, \mathcal{Q})$. This is because the behavior of $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ on input 0 is exactly the same as the behavior on input 1 before a query to z is made, and after z is queried the value of N_i does not change.

▷ Claim 11. Let $\mathcal{B}$ be an algorithm taking inputs from $\{0,1\}^m$. Then $(\mathcal{B}, \mathcal{Q})$ is FULL if and only if $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ queries z with probability 1. If $(\mathcal{B}, \mathcal{Q})$ is FULL then

$$\chi(\mathcal{B}, \mathcal{Q}) = \mathsf{E}[\mathcal{N}_1(T, 1, \mathcal{Q})]$$

Proof. Note that until z is queried, $\mathcal{P}(\mathcal{B}, (\mu_0, \mu_1))$ exactly executes the random walk described in Section 3, and querying z in $\mathcal{P}(\mathcal{B}, (\mu_0, \mu_1))$ corresponds to this random walk terminating. The first part of the claim then follows as $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ queries z with probability 1 if and only if $\mathcal{P}(\mathcal{B}, (\mu_0, \mu_1))$ queries z with probability 1 for every $(\mu_0, \mu_1) \in \mathrm{supp}(\mathcal{Q})$.

Also because $\mathcal{P}(\mathcal{B}, (\mu_0, \mu_1))$ exactly executes the random walk described in Section 3 we see that $\chi(\mathcal{B}, (\mu_0, \mu_1)) = \mathsf{E}[\mathcal{N}_1(T, 1, (\mu_0, \mu_1))]$. The second part of the claim follows by taking the expectation of this equality over $(\mu_0, \mu_1) \sim \mathcal{Q}$. ◁

The correspondence of claim 11 prompts us to define FULL in a more general setting.

▶ **Definition 12** (FULL). *Let $\mathcal{B}$ be a query algorithm taking inputs from $(\{0,1\}^m)^t$. The pair $(\mathcal{B}, \mathcal{Q})$ is said to be* FULL *if for every $z \in \{0,1\}^t$ it holds that $\mathcal{P}(\mathcal{B}, \mathcal{Q})$ queries z_i with probability 1, for every $i = 1, \dots, t$.*

5 The Composition Theorem

A proof of Theorem 4 is given in [6].

6 Tightness: $\mathsf{R}_{1/3}(f \circ g^n) \in O\left(\mathsf{R}_{4/9}(f) \cdot \sqrt{\mathsf{R}_{1/3}(g)}\right)$ is possible

In this section we prove Theorem 2. We construct a relation $f_0 \subseteq \{0,1\}^n \times \{0,1\}^n$ (i.e., $\mathcal{S} = \{0,1\}^n$) and a promise function $g_0 \subseteq \{0,1\}^n \times \{0,1\}$ (i.e., $m = n$), such that $\mathsf{R}_{4/9}(f_0) \in \Theta(\sqrt{n})$, $\mathsf{R}_{1/3}(g_0) \in \Theta(n)$ and $\mathsf{R}_{1/3}(f_0 \circ g_0^n) \in \Theta(n)$.

For strings $x = (x_1, \dots, x_n), z = (z_1, \dots, x_n)$ in $\{0,1\}^n$, let $x \oplus z$ be the string $(x_1 \oplus z_1, \dots, x_n \oplus z_n)$ obtained by taking their bitwise XOR. Let $|x|$ stand for the *Hamming weight* $|\{i \in [n] : x_i = 1\}|$ of x. We define f_0 as follows:

$$f_0(z) \overset{\text{def}}{=} \left\{(a, z) \in \{0,1\}^n \times \{0,1\}^n \middle| |a \oplus z| \le \frac{n}{2} - \sqrt{n}\right\}$$

Now we define g_0 by specifying $g_0^{-1}(0)$ and $g_0^{-1}(1)$.

$$g_0^{-1}(0) \overset{\text{def}}{=} \left\{(x, 0) \middle| x \in \{0,1\}^n, |x| \le n/2 - \sqrt{n}\right\},$$
$$g_0^{-1}(1) \overset{\text{def}}{=} \left\{(x, 1) \middle| x \in \{0,1\}^n, |x| \ge n/2 + \sqrt{n}\right\}.$$

We now determine the randomized query complexities of f_0, g_0 and $f_0 \circ g_0^n$.

▷ Claim 13.
(i) $\mathsf{R}_{4/9}(f_0) \in \Omega(\sqrt{n})$.
(ii) $\mathsf{R}_{1/3}(g_0) \in \Omega(n)$.
(iii) $\mathsf{R}_{\varepsilon}(f_0 \circ g_0^n) \in O\left(n \cdot \sqrt{\log(1/\varepsilon)}\right)$.

A proof of Claim 13 can be found in [6]. Theorem 2 follows from Theorem 4 and Claim 13 with ε set to 1/3.

References

1 Scott Aaronson. Quantum certificate complexity. *Journal of Computer and System Sciences*, 74(3):313–322, 2008.

2 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 *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, pages 10:1–10:13, 2017.

3 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to Compress Interactive Communication. *SIAM J. Comput.*, 42(3):1327–1363, 2013. doi:10.1137/100811969.

4 Shalev Ben-David and Robin Kothari. Randomized Query Complexity of Sabotaged and Composed Functions. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 60:1–60:14, 2016.

5 Dmitry Gavinsky, Troy Lee, and Miklos Santha. On the randomised query complexity of composition. Technical report, arXiv, 2018. `arXiv:1801.02226`.

6 Dmitry Gavinsky, Troy Lee, Miklos Santha, and Swagato Sanyal. A composition theorem for randomized query complexity via max conflict complexity. *CoRR*, abs/1811.10752, 2018. `arXiv:1811.10752`.

7 Justin Gilmer, Michael Saks, and Srikanth Srinivasan. Composition limits and separating examples for some Boolean function complexity measures. *Combinatorica*, 36(3):265–311, 2016.

8 Peter Høyer, Troy Lee, and Robert Spalek. Negative weights make adversaries stronger. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 526–535, 2007.

9 Yaqiao Li. Conflict complexity is lower bounded by block sensitivity. Technical report, arXiv, 2018. `arXiv:1810.08873`.

10 Ashley Montanaro. A composition theorem for decision tree complexity. *Chicago J. Theor. Comput. Sci.*, 2014, 2014.

11 Ben Reichardt. Reflections for quantum query algorithms. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 560–569, 2011.

12 Swagato Sanyal. A composition theorem via conflict complexity. Technical report, arXiv, 2018. `arXiv:1801.03285`.

13 Avishay Tal. Properties and applications of boolean function composition. In *Innovations in Theoretical Computer Science, ITCS '13*, pages 441–454, 2013.

14 John von Neumann. Zur Theorie der Gesellschaftsspiele. *Math. Ann.*, 100:295–320, 1928.