Oracle Separations for the Quantum-Classical Polynomial Hierarchy

Avantika Agarwal
Institute for Quantum Computing, University of Waterloo, Canada

Shalev Ben-David
Institute for Quantum Computing, University of Waterloo, Canada

Abstract

We study the quantum-classical polynomial hierarchy, QCPH, which is the class of languages solvable by a constant number of alternating classical quantifiers followed by a quantum verifier. Our main result is that QCPH is infinite relative to a random oracle (previously, this was not even known relative to any oracle). We further prove that higher levels of PH are not contained in lower levels of QCPH relative to a random oracle; this is a strengthening of the somewhat recent result that PH is infinite relative to a random oracle (Rossman, Servedio, and Tan 2016).

The oracle separation requires lower bounding a certain type of low-depth alternating circuit with some quantum gates. To establish this, we give a new switching lemma for quantum algorithms which may be of independent interest. Our lemma says that for any d, if we apply a random restriction to a function f with quantum query complexity $\mathrm{Q}(f) \leq n^{1/3}$, the restricted function becomes exponentially close (in terms of d) to a depth-d decision tree. Our switching lemma works even in a "worst-case" sense, in that only the indices to be restricted are random; the values they are restricted to are chosen adversarially. Moreover, the switching lemma also works for polynomial degree in place of quantum query complexity.

2012 ACM Subject Classification Theory of computation → Oracles and decision trees; Theory of computation → Complexity classes; Theory of computation → Quantum complexity theory

Keywords and phrases Switching Lemma, Polynomial Hierarchy, Approximate Degree, Random Oracles, Query Complexity, Quantum Computing

Digital Object Identifier 10.4230/LIPIcs.ITCS.2026.2

Related Version *Full Version*: `https://arxiv.org/abs/2410.19062` [2]

Funding This research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), DGECR-2019-00027 and RGPIN-2019-04804.[1]

1 Introduction

In classical complexity theory, an important complexity class is the polynomial hierarchy, PH. This is a generalization of NP to higher depth: it can be written as the union $\mathsf{NP} \cup \mathsf{NP}^{\mathsf{NP}} \cup \mathsf{NP}^{\mathsf{NP}^{\mathsf{NP}}} \cup \ldots$, and corresponds to languages that can be computed using a constant number of alternating quantifiers over certificates. A problem is in PH if it can be computed in polynomial time in the presence of two computationally-unbounded provers, one of which wants to convince the verifier the input is a yes-input, and one of which wants to convince the verifier the input is a no-input, with the provers exchanging polynomially-sized public messages for a constant number of rounds. The polynomial hierarchy can therefore be viewed as the class of problems solvable by an audience member sitting in a debate between experts,

[1] Cette recherche a été financée par le Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG), DGECR-2019-00027 et RGPIN-2019-04804.

where the debate has the inefficient format of alternating between the two speakers only a constant number of times, even though the total debate time is polynomial in the input size (polynomially many alternations would result in the larger class PSPACE).

The polynomial hierarchy can be viewed as a union of different "levels", where the d-th level corresponds to debates with d alternations. The hierarchy is said to *collapse* to level d if level $d+1$ can solve no more problems than level d; in that case, all higher levels can also be simulated with d rounds of debate. It is widely believed that the polynomial hierarchy is infinite, meaning it does not collapse to any level. This is a generalization of the $\mathsf{P} \neq \mathsf{NP}$ conjecture, which is equivalent to the assertion that PH does not collapse to the 0-th level.

Since proving the polynomial hierarchy is infinite is beyond current techniques, one may ask instead for oracle separations. It turns out PH is infinite even relative to a *random* choice of oracle (with probability 1), though this result is somewhat recent.

▶ **Theorem 1** ([13]). PH *is infinite relative to a random oracle.*

We study a quantum version of the polynomial hierarchy known as the quantum-classical hierarchy (introduced in [11]). This is the class of problems solvable by a quantum audience member of a constant-round debate (with classical messages) between experts. This class, denoted QCPH, generalizes QCMA but not QMA (since the proofs received are classical rather than quantum). More formally, QCPH is the union $\bigcup_{i\in\mathbb{N}} \mathsf{QC}\Sigma_i$, where $\mathsf{QC}\Sigma_i$ is defined as the class of languages for which there is a quantum verifier V satisfying

$$x \in A_{\text{yes}} \Rightarrow \exists y_1 \forall y_2 \dots Q_i y_i \colon \Pr[V(x, y_1, y_2, \dots, y_i)] \geq 2/3$$
$$x \in A_{\text{no}} \Rightarrow \forall y_1 \exists y_2 \dots \overline{Q_i} y_i \colon \Pr[V(x, y_1, y_2, \dots, y_i)] \leq 1/3$$

for all input strings x, where the y_j represent polynomially-sized classical strings, Q_i is a quantifier, and $\overline{Q_i}$ is the opposite quantifier.

The class QCPH is interesting on its own, but another motivation for its study is the connection to quantum switching lemmas. Oracle separations for PH generally reduce to the problem of giving lower bounds on the classical circuit class AC^0, consisting of circuits of constant depth. Quantum versions of constant-depth circuits are of interest because they help model quantum devices with many qubits but few layers of gates. Lower bounds on such circuit classes are often shown using switching lemmas, which assert that certain types of functions must greatly simplify under a random restriction of their bits; these switching lemmas can therefore be useful both for the study of near-term quantum devices and for oracle separations for complexity classes such as QCPH.

1.1 Our results

Our main result generalizes Theorem 1 to the quantum setting.

▶ **Theorem 2.** *The following holds relative to a random oracle with probability* 1*. For any constant $d \geq 1$, level $d+1$ of the polynomial hierarchy, Σ_{d+1}, is not contained in level d of the quantum-classical hierarchy, $\mathsf{QC}\Pi_d$. In particular,* QCPH *is infinite, and no fixed level of the* QCPH *hierarchy contains all of* PH*.*

Previously to this work, it was not even known whether there was *any* oracle relative to which QCPH is infinite, let alone a random oracle. Previous switching lemmas for quantum algorithms, such as the one in [1], are insufficient to prove such an oracle separation.[2] We

[2] Most quantum circuit lower bounds are incomparable to ours, since they consider different models of quantum circuits. Our switching lemma is stronger than the comparable switching lemma of [1].

also note that this theorem implies Theorem 1 as a special case, since it also implies that PH is infinite relative to a random oracle.

The study of PH relative to a random oracle boils down to the study of AC^0 circuits against the uniform distribution. This, in turn, is usually done using random restrictions and the switching lemma, together with the random projection technique introduced in [13]. To prove Theorem 2, we need a new switching lemma for quantum query complexity.

▶ **Theorem 3.** *Let $N \in \mathbb{N}$ be sufficiently large, and let f be a possibly partial Boolean function on N bits with $\mathrm{Q}(f) \leq N^{1/3}$. Let $p = N^{-10/11}$ (we can choose $p \leq O(\frac{1}{Q(f)^2 N})^{4/7}$), and consider a random restriction of f which fixes each bit with probability $1-p$. Then for any $d \leq N^{1/3}$, this restriction is approximated by a decision tree of height d in the following sense.*

For an input x and a choice of unrestricted bits S, we define a restriction which sets bits in $\overline{S}$ to according to x. Then for every input x, there is an ensemble of decision trees $\{D_{x,S}\}_{x\in\{0,1\}^n, S\subseteq[N]}$, all of height at most d, with $D_{x,S}$ acting on the unrestricted input bits which are strings of length $|S|$, such that the following holds: if $S \subseteq [N]$ is chosen at random with each index $i \in [N]$ included in S independently with probability p, then

$$\forall x, y \in \{0,1\}^n \quad \Pr_S[D_{x,S}(y|_S) \neq f_{x_{\overline{S}}}(y|_S)] < e^{-d^{1/5}}.$$

Here $f_{x_{\overline{S}}}$ denotes the restriction of f to the partial assignment which fixes the bits of x outside of S.[3]

This is a type of switching lemma for quantum query complexity, though its statement can be confusing. We make a few clarifying comments. First, note that with the parameter $p = N^{-10/11}$, a function f with quantum query complexity at most $N^{1/3}$ becomes constant with high probability under a random restriction. However, the probability of the function becoming constant is merely $1 - 1/\operatorname{poly}(N)$, which is not small enough. The point of the theorem is to approximate the function f to exponentially small error (in the height of the approximating decision tree).

The random restriction in Theorem 3 is *worst-case* in the sense that while the positions to be fixed are chosen randomly, the bits to which those positions are fixed are specified by a string x, which can be chosen arbitrarily. Moreover, the resulting restricted function $f|_{x_{\overline{S}}}$ must be approximated by a decision tree even on worst-case choices of input $y|_S$ (except that the choice of the worst-case y cannot depend on the random choice of S). From such a worst-case statement, we can easily derive a more familiar average-case switching lemma.

▶ **Corollary 4.** *For sufficiently large N, let f be such that $\mathrm{Q}(f) \leq N^{1/3}$, let $p = N^{-10/11}$, and let $d \leq N^{1/3}$. If f_ρ is a random restriction in which each bit remains unfixed with probability p (and is fixed randomly to 0 or 1 otherwise), then with probability at least $1 - e^{-d^{1/10}}$, there is a decision tree of height at most d which computes f_ρ on a $1 - e^{-d^{1/10}}$ fraction of its inputs.*

Theorem 3 can be viewed as a version of Corollary 4 which works even for non-uniform distributions. We also strengthen Theorem 3 so that it works for approximate degree instead of just quantum query complexity.

▶ **Theorem 5.** *Theorem 3 still holds if the condition on f is replaced by $\widetilde{\deg}(f) \leq N^{1/3}$ instead of $\mathrm{Q}(f) \leq N^{1/3}$.*

[3] If $f_{x_{\overline{S}}}$ is undefined on input $y|_S$, we count it as equality holding in the probability, meaning that on inputs outside the domain the decision tree is allowed to output anything.

This can be viewed as establishing a random-restriction version of the Aaronson-Ambainis conjecture, which asserts that low degree polynomials can be approximated by shallow decision trees against the uniform distribution. Our version works only after a random restriction is applied to the polynomial, but it works with extremely strong parameters[4]. Another (incomparable) version of the Aaronson-Ambainis conjecture for random restrictions was given in [7].

Although we don't explicitly show it, Theorem 5 can be strengthened further, so that it works with a measure known in the literature as (the square root of) "critical fractional block sensitivity", which lower bounds approximate degree [4]. This measure also lower bounds the positive-weight quantum adversary bound, so our switching lemma also works for that measure.

Finally, we give an application of our switching lemmas to yet another type of oracle separation: we show that the "QCMA hierarchy," that is,

$$\mathsf{QCMA} \cup \mathsf{QCMA}^{\mathsf{QCMA}} \cup \mathsf{QCMA}^{\mathsf{QCMA}^{\mathsf{QCMA}}} \cup \dots$$

is also infinite relative to a random oracle. Explicitly, defining $\mathsf{QCMAH}_1 = \mathsf{QCMA}$ and $\mathsf{QCMAH}_{d+1} = \mathsf{QCMA}^{\mathsf{QCMAH}_d}$ for $d \geq 1$, we have the following result.

▶ **Theorem 6.** *The following holds relative to a random oracle with probability* 1*. For any constant* $d \geq 1$*,* QCMAH_{d+1} *is not contained in* QCMAH_d*. Moreover, no fixed level* QCMAH_d *contains all of* PH*.*

Proving lower bounds on QCMAH relative to an oracle amounts to proving lower bounds on constant-depth circuits which have alternating layers of polynomial-fanin AND gates, and of "gates" consisting of quantum query algorithms which make few quantum queries. This demonstrates that our techniques can be useful for proving lower bounds on shallow quantum circuit classes.

1.2 Our techniques

Random oracle separations. To prove our oracle separation for QCPH relative to a random oracle, it suffices to show that depth d AC^0 circuits with "quantum query complexity gates" at the bottom layer cannot compute some function in computable in depth $d+2$ classical AC^0, against the uniform distribution. The quantum query complexity gates means that at the bottom of the d alternating layers of AND and OR gates lie Boolean functions which can each be computed in polylog N quantum queries. (For the QCMAH separation, we need to allow quantum query complexity gates in the middle of the circuit as well, not just the bottom.) This circuit separation implies the random oracle separation through a standard technique sometimes called "slow diagonalization" (see full version).

To construct a function which is computable in depth $d+2$ classically but not in depth d quantumly (against the uniform distribution), we use the construction of [13], which separated classical AC^0 circuits of different depths. Their proof used random projections, which we also employ. We must modify their construction to account for the extra layer of quantum gates, which we will need to "strip away" in the analysis using a random projection and our quantum switching lemma.

[4] Note that Aaronson-Ambainis conjecture is about bounded real-valued functions rather than Boolean functions; our switching lemma also works in that setting.

In order to establish a depth-hierarchy theorem for circuits, the restrictions used need to ensure that an AND-OR tree retains structure (as in [12] and [13]) while a smaller-depth circuit simplifies. Thus the sequence of restrictions/projections used needs to alternate between heavily biased towards 1 and heavily biased towards 0 (so that not all AND/OR gates are set to constant). In the quantum switching lemma of [1], the quantum query algorithm becomes close to a DNF with respect to the uniform distribution, when the restriction is also sampled uniformly. We will however need to sample a projection from an underlying distribution different from the one with respect to which we compare the closeness of the resulting quantum query algorithm and DNF (because of the alternating choice of distributions described earlier). This mismatch between the two distributions requires us to use our "worst-case" type of quantum switching lemma.

Our separation for the QCMAH hierarchy works similarly, but requires carefully changing the parameters to ensure the structure is maintained even in the presence of intermediate quantum query gates. See Section 6 for details.

Quantum switching lemma. The proof of our quantum switching lemma relies on an *adaptive* application of the quantum hybrid method, which may be of independent interest. Given a quantum algorithm Q acting on a string x, a standard hybrid argument of [6] says there will only be a small set of positions in the string x that the algorithm Q "looks at"; the output of Q can only be sensitive to a change in a bit of x at one of those few positions. Call those the heavy positions of x. Now, if some of those heavy positions of x are indeed changed, then not only can the output of Q change, but even the set of heavy positions of the input can change.

This poses a problem for us: we would like to restrict the function to the bits of x, except at a few random positions. If some of those random positions are heavy (with respect to x), then the quantum algorithm can still depend on them in a nontrivial manner. We could try to mimic this by a classical algorithm which queries those few non-fixed heavy bits, but the problem is that this is not sufficient to fix the output of Q: the output of the algorithm Q may now depend on new heavy bits. (It is not clear if a classical decision tree can find these new heavy bits, since a quantum algorithm may try to query the rest of the unfixed bits in superposition.)

We get around this problem by applying the hybrid method iteratively, in an adaptive manner. Beginning with a string x, we find its heavy bits, and randomly choose a few of them to leave unfixed; we replace those bits with values from a different string y. This gives us a hybrid string $x^{(1)}$ which mostly contains bits of x, but where a few heavy positions were replaced by bits of y. We then iterate this process: we find the heavy bits of $x^{(1)}$ which are not heavy bits of x, sample a few of those positions at random, and replace them with more bits from y, resulting in $x^{(2)}$. We continue this way and terminate when there are no new heavy bits.

In each round, there is a constant (or better) probability of having no new heavy bits which are unfixed, since the number of heavy bits is small and the probability of leaving a bit unfixed is also small. This means the number of rounds cannot be too large except with exponentially small probability. Once the process terminates, the set of all heavy bits encountered along the way cannot be too large (except with exponentially small probability), and can be used to compute the randomly-restricted function on most inputs.

Polynomial lower bound. Since our quantum switching lemma relies fundamentally on the hybrid method, which talks about "where the quantum algorithm queried the string", it may seem surprising that we can generalize our result to a switching lemma for polynomials as well.

To do this, we rely on a property shown in [4] (expanding on earlier work by [14]) which says that approximate degree is lower bounded by fractional block sensitivity. Fractional block sensitivity, in turn, is equal to fractional certificate complexity, a measure which assigns to each bit of each input a non-negative weight: a way of saying "how much did the polynomial look at position i when given string x as input".

In fact, we will need a stronger version of the result of [4]; we adapt their method to prove this. Our result is the following theorem, which may be of independent interest.

▶ **Theorem 7.** *Let $f\colon \{0,1\}^n \to [0,1]$ be a real-valued function which can be expressed as a polynomial of degree at most d. Then there is an assignment of weights $\{c_{x,i}\}_{x\in\{0,1\}^n, i\in[n]}$ to inputs x and positions $i \in [n]$ such that for all $x, y \in \{0,1\}^n$, we have*

$$\sum_{i:x_i\neq y_i} c_{x,i} \geq |f(x) - f(y)|, \qquad \sum_{i=1}^{n} c_{x,i} \leq \frac{\pi^2}{4} d^2.$$

The proof is similar to the one in [4]; it uses strong duality of linear programming to convert fractional certificates to fractional block sensitivity, and uses composition of f with a version of promise-OR to convert fractional block sensitivity to regular block sensitivity; the latter can be turned into sensitivity using a projection, and results in approximation theory can be used to relate sensitivity to polynomial degree.

2 Preliminaries

▶ **Definition 8** (Projection)**.** *Given a function $f : \{0,1\}^{n\times l} \to \mathrm{R}$, and a restriction ρ in $\{0,1,*\}^{n\times l}$, the projected function $\mathsf{proj}_\rho f : \{0,1\}^n \to \mathrm{R}$ is defined as $\mathsf{proj}_\rho f(x) = f(y)$ where*

$$y_{i,j} = \begin{cases} x_i & \text{if } \rho(i,j) = * \\ \rho(i,j) & \text{otherwise} \end{cases}$$

So the projection operator maps all unrestricted variables in a given block of size l, to the same new variable. If ρ is a random restriction, then proj_ρ is a random projection.

2.1 Quantum-Classical Polynomial Hierarchy

▶ **Definition 9** ($\mathsf{QC\Sigma}_i$)**.** *Let $A = (A_{\text{yes}}, A_{\text{no}})$ be a promise problem. We say that A is in $\mathsf{QC\Sigma}_i(c,s)$ for poly-time computable functions $c, s : \mathbb{N} \mapsto [0,1]$ if there exists a poly-bounded function $p : \mathbb{N} \mapsto \mathbb{N}$ and a poly-time uniform family of quantum circuits $\{V_n\}_{n\in\mathbb{N}}$ such that for every n-bit input x, V_n takes in classical proofs $y_1 \in \{0,1\}^{p(n)}, \ldots, y_i \in \{0,1\}^{p(n)}$ and outputs a single qubit, such that:*

- *Completeness: $x \in A_{\text{yes}} \Rightarrow \exists y_1 \forall y_2 \ldots Q_i y_i$ s.t. $\Pr[V_n$ accepts $(y_1, \ldots, y_i)] \geq c$.*
- *Soundness: $x \in A_{\text{no}} \Rightarrow \forall y_1 \exists y_2 \ldots \overline{Q}_i y_i$ s.t. $\Pr[V_n$ accepts $(y_1, \ldots, y_i)] \leq s$.*

Here, Q_i equals $\exists$ when i is odd and equals $\forall$ otherwise and $\overline{Q}_i$ is the complementary quantifier to Q_i. Define

$$\mathsf{QC\Sigma}_i := \bigcup_{c-s\in\Omega(1/\operatorname{poly}(n))} \mathsf{QC\Sigma}_i(c,s).$$

Note that the first level of this hierarchy corresponds to QCMA. The complement of the i^{th} level of the hierarchy, $\mathsf{QC\Sigma}_i$, is the class $\mathsf{QC\Pi}_i$ defined next.

▶ **Definition 10** ($\mathsf{QC\Pi}_i$). *Let $A = (A_{\text{yes}}, A_{\text{no}})$ be a promise problem. We say that $A \in \mathsf{QC\Pi}_i(c,s)$ for poly-time computable functions $c, s : \mathbb{N} \mapsto [0,1]$ if there exists a polynomially bounded function $p : \mathbb{N} \mapsto \mathbb{N}$ and a poly-time uniform family of quantum circuits $\{V_n\}_{n\in\mathbb{N}}$ such that for every n-bit input x, V_n takes in classical proofs $y_1 \in \{0,1\}^{p(n)}, \ldots, y_i \in \{0,1\}^{p(n)}$ and outputs a single qubit, such that:*

- *Completeness: $x \in A_{\text{yes}} \Rightarrow \forall y_1 \exists y_2 \ldots Q_i y_i$ s.t. $\Pr[V_n$ accepts $(y_1, \ldots, y_i)] \geq c$.*
- *Soundness: $x \in A_{\text{no}} \Rightarrow \exists y_1 \forall y_2 \ldots \overline{Q}_i y_i$ s.t. s.t. $\Pr[V_n$ accepts $(y_1, \ldots, y_i)] \leq s$.*

Here, Q_i equals $\forall$ when i is odd and equals $\exists$ otherwise, and $\overline{Q}_i$ is the complementary quantifier to Q_i. Define

$$\mathsf{QC\Pi}_i := \bigcup_{c-s\in\Omega(1/\operatorname{poly}(n))} \mathsf{QC\Pi}_i(c,s).$$

Now the corresponding quantum-classical polynomial hierarchy is defined as follows.

▶ **Definition 11** (Quantum-Classical Polynomial Hierarchy (QCPH)).

$$\mathsf{QCPH} = \bigcup_{i\in\mathbb{N}} \mathsf{QC\Sigma}_i = \bigcup_{i\in\mathbb{N}} \mathsf{QC\Pi}_i.$$

See the full version for a survey of the previous work on quantum polynomial hierarchies.

▶ **Definition 12** ($\mathsf{QAC}^0_i(s)$ circuits). *A circuit family $\{C_n\}_{n=1}^{\infty}$ is in $\mathsf{QAC}^0_i(s)$ for some function $s : \mathbb{N} \to \mathbb{N}$ if it satisfies the following:*

1. *C_n has i alternating layers of $\vee$ and $\wedge$ gates, followed by a layer of quantum query circuits of size $\operatorname{polylog}(s(n))$ $\forall n \in \mathbb{N}$*
2. *size(C_n) $\leq s(n)$ $\forall n \in \mathbb{N}$ where size of the circuit is the number of $\vee$, $\wedge$ gates plus the number of quantum query circuits in the bottom layer*

Note that in the above definition, if the quantum query circuits are not computing total functions, then we treat each gate in the circuit as computing a partial function, whose output is evaluated in the following manner. For an input x to the circuit, a $\vee$ gate outputs 1 if at least one of its inputs is 1 regardless of the arbitrary 0/1 responses of the input partial functions whose promise is violated. It outputs 0 if all of its inputs are 0 regardless of the arbitrary 0/1 responses of the input partial functions whose promise is violated. Otherwise, the output of the $\vee$ gate is undefined. The evaluation of the output of $\wedge$ gate is done in a similar manner. This method of evaluation gives the same output as the one if we evaluate the output of the circuit in a manner consistent with Definition 9 (or Definition 10). That is, a $\vee$ gate corresponds to an $\exists$ quantifier and a $\wedge$ gate corresponds to a $\forall$ quantifier. We show this formally in the full version.

2.2 Query complexity

In query complexity, an algorithm (classical or quantum) is given black-box access to a string $x \in \{0,1\}^N$, and is supposed to compute some boolean function $f(x)$ by querying x as few times as possible. We have included the basic definitions of query complexity in the full version. Interested readers can refer to [8] for a detailed introduction to query complexity. Let $N = 2^n$. In the classical query model, an algorithm queries $i \in \{0,1\}^n$ and receives x_i, which we will denote by $O_x(i) = x_i$. In the quantum query model, an algorithm queries $\sum_{i\in\{0,1\}^n} \alpha_i|i\rangle$ and receives $\sum_{i\in\{0,1\}^n} \alpha_i(-1)^{x_i}|i\rangle$, which we will denote by $O_x(\sum_{i\in\{0,1\}^n} \alpha_i|i\rangle) = \sum_{i\in\{0,1\}^n} \alpha_i(-1)^{x_i}|i\rangle$.

▶ **Definition 13** (Query Magnitudes [6]). *Given a quantum query algorithm Q computing a (partial) function $f : \{0,1\}^N \to \{0,1,\perp\}$ (where $N = 2^n$) with T queries, define query magnitude at time step t, denoted $m_i^t(x)$, for $t \in [T]$ and $i \in \{0,1\}^n$ as follows:*

$$Q(x) = U_{T+1}O_xU_T \dots O_xU_1|0^n\rangle|0^w\rangle$$
$$m_i^t(x) = \Pr[\textit{Measuring query register of } U_{t+1}O_xU_t \dots O_xU_1|0^n\rangle|0^w\rangle \textit{ gives } i]$$
$$m_i(x) = \sum_{t=1}^{T} m_i^t(x)$$

We call $m_i(x)$ as the query magnitude of Q for bit i on input x. Since $\sum_{i\in\{0,1\}^n} m_i^t(x) = 1$ for all $t \in [T]$, we know that $\sum_{i\in\{0,1\}^n} m_i(x) \leq T$ (it need not be exactly equal since the quantum algorithm might make less than T queries on some inputs x).

▶ **Definition 14** (Partial Assignment). *A partial assignment p of length N is any string from the set $\{0,1,*\}^N$. A string $x \in \{0,1,*\}^N$ is consistent with p if $p_i = x_i$ whenever $p_i \neq *$.*

▶ **Definition 15** (Certificate). *Given a (partial) function $f : \{0,1\}^N \to \{0,1,\perp\}$, a partial assignment $p \in \{0,1,*\}^N$ is a b-certificate for f if $f(x) = f(y) = b$ for all $x, y \in f^{-1}(\{0,1\})$ which are consistent with p. The length of this certificate p is the number of bits in p not equal to $*$.*

▶ **Definition 16** (Certificate Complexity). *Given a (partial) function $f : \{0,1\}^N \to \{0,1,\perp\}$, the certificate complexity of f at input $x \in \mathsf{dom}(f)$ is defined as $\mathrm{C}(f,x) = \min_p \mathsf{len}(p)$ where the minimization is over all partial assignments p consistent with x, which are also $f(x)$-certificates for f. Then the certificate complexity $\mathrm{C}(f) = \max_{x\in\mathsf{dom}(f)} \mathrm{C}(f,x)$.*

▶ **Definition 17** (Approximate Degree). *Given a (partial) function $f : \{0,1\}^N \to \{0,1,\perp\}$, the approximate degree of f, denoted $\widetilde{\mathsf{deg}}(f)$, is the minimum degree of a multi-linear polynomial p which approximate f to error within additive error $1/3$ for all input $x \in \mathsf{dom}(f)$ and $p(x) \in [0,1]$ for all $x \in \{0,1\}^N$.*

▶ **Lemma 18** ([5]). *If a (partial) function $f : \{0,1\}^N \to \{0,1,\perp\}$ is computable by a quantum algorithm Q making T queries, we have that there is a multilinear polynomial p of degree at most $2T$ such that $p(x)$ is equal to the probability that Q outputs 1 on input x. Therefore p approximates f to error within additive error $1/3$ for all input $x \in \mathsf{dom}(f)$ and $p(x) \in [0,1]$ for all $x \in \{0,1\}^N$ and $\widetilde{\mathsf{deg}}(f) \leq 2T$.*

2.3 QCMA Hierarchy

In this section, we define the QCMA hierarchy (called QCMAH). It is a hierarchy of classes, where the i^{th} level, called QCMAH_i is defined as follows:

$$\mathsf{QCMAH}_1 := \mathsf{QCMA}$$
$$\mathsf{QCMAH}_i := \mathsf{QCMA}^{\mathsf{QCMAH}_{i-1}}$$
$$\mathsf{QCMAH} := \cup_{i=1}^{\infty}\mathsf{QCMAH}_i$$

Note that QCMA is a class of promise problems, thus when querying a QCMAH_i oracle, any QCMA algorithm V is making queries to a promise oracle. Note that the QCMA *algorithm* V is different from the QCMA *verifier* (which takes a single proof as input and outputs 0/1), here V instead refers to the OR of all these 0/1 outputs of the verifier for every possible proof. If V makes a query on an input which violates the promise of the QCMAH_i oracle, then the

oracle can return an arbitrary 0/1 response. Therefore we need to be careful about when the behaviour of V is well-defined. In this paper, we follow the definition of well-defined behaviour proposed in [1] (note however that there can be many other reasonable definitions of well-defined behaviour, though we do not describe them here). The behaviour of V is well-defined, if the output of V remains the same regardless of the arbitrary 0/1 responses of the QCMAH_i oracle on inputs violating the promise. In particular for a yes input, the QCMA verifier might accept only one proof for some choice of oracle responses outside the promise, and it might accept (say) half of the proofs for some other choice of oracle responses, but the behaviour remains well-defined as long as it accepts at least one proof for every choice of oracle responses.

In the rest of the paper, we will denote as QCMAH_i verifier V as an i-tuple of QCMA verifiers $V = \langle V_1, \ldots, V_i \rangle$ where V_1 corresponds to the base QCMA verifier and the remaining $i-1$ verifiers form the $(i-1)$-tuple for the verifier corresponding to the QCMAH_{i-1} oracle.

3 Polynomials and fractional block sensitivity

In this section, we establish a relationship between fractional block sensitivity and degree for real-valued functions. We will need this relationship to establish a switching lemma for approximate degree (the tools of this section are not necessary for the quantum switching lemma, which may instead rely on only the hybrid argument of [6], but we opt to prove the stronger switching lemma for polynomials).

3.1 Definitions for real-valued functions

When studying the degree of bounded real-valued functions, most of the literature uses the convention that the functions have signature $\{\pm 1\}^n \to [-1, 1]$, known as the ± 1 basis. Switching between $\{0, 1\}$ and $\{\pm 1\}$ does not affect most measures: we can interpret -1 as 1 and $+1$ as 0, and we can plug in $1 - 2x_i$ in each variable to convert a $\{0, 1\}$ variable to a $\{\pm 1\}$ variable, or plug in $(1 - x_i)/2$ to go in the reverse direction. This does not affect the degree. We will use the $\{\pm 1\}$ basis in this section.

Sensitivity. The sensitivity of a real-valued function can be defined as follows. For a block $B \subseteq [n]$, define

$$\mathrm{s}(f, x, B) \coloneqq \frac{|f(x) - f(x^B)|}{2},$$

where the notation x^B refers to the string x with the block $B \subseteq [n]$ of bits flipped. Note that we divide by 2 because f can take values from $[-1, 1]$ instead of $[0, 1]$. This is the sensitivity of a specific block B to a specific input $x \in \{\pm 1\}^n$, with respect to the function f; it is a value between 0 and 1. We next define the sensitivity of f at x to be $s(f, x)$ which is the total sensitivity of all the bits of x. We then use it to define sensitivity of f.

$$\mathrm{s}(f, x) \coloneqq \sum_{i=1}^{n} \mathrm{s}(f, x, \{i\})$$
$$\mathrm{s}(f) \coloneqq \max_{x \in \{\pm 1\}^n} \mathrm{s}(f, x)$$

For a Boolean function, this definition matches the usual definition of sensitivity $\mathrm{s}(f)$.

Block sensitivity. We define block sensitivity analogously: $\mathrm{bs}(f,x)$ will be the maximum possible value of $\sum_{j=1}^{k} \mathrm{s}(f,x,B_j)$ over choices of disjoint blocks $B_j \subseteq [n]$, and $\mathrm{bs}(f)$ will be the maximum possible value of $\mathrm{bs}(f,x)$ over choices of inputs $x \in \{\pm 1\}^n$.

Fractional block sensitivity. We define $\mathrm{fbs}(f,x)$ to be the maximum possible value of the sum $\sum_{B\subseteq[n]} w_B \,\mathrm{s}(f,x,B)$, subject to the constraints $w_B \geq 0$ for all $B \subseteq [n]$ and $\sum_{B\ni i} w_B \leq 1$ for all $i \in [n]$. (The weights w_B represent fractions of a block, and the total weight of all blocks containing a bit i must be at most 1, making the blocks "fractionally disjoint".) As usual, we define $\mathrm{fbs}(f) \coloneqq \max_{x\in\{\pm1\}^n} \mathrm{fbs}(f,x)$.

Fractional certificate complexity. The definition of $\mathrm{fbs}(f,x)$ is the optimal value of a linear maximization program in terms of the real variables w_B. The dual of this linear program can is as follows. The variables are $c_i \geq 0$, with the constraint $\sum_{i\in B} c_i \geq \mathrm{s}(f,x,B)$ for all $B \subseteq [n]$. The objective is to minimize $\sum_{i\in[n]} c_i$. This can be interpreted as finding a fractional certificate, with c_i representing the fraction with which bit i is used in the certificate, such that any block B which, when flipped, changes the value of the function by $\mathrm{s}(f,x,B)$ must be "detected" by the certificate in the sense that the certificate puts total weight at least $\mathrm{s}(f,x,B)$ on the bits of B.

Fractional certificate complexity is equal to fractional block sensitivity, so we will not give it new notation. However, we denoting by $c_{x,i}$ the fractional certificate for x, we get that for all x, y,

$$\sum_{i:x_i\neq y_i} c_{x,i} \geq \frac{|f(x)-f(y)|}{2}, \qquad \sum_{i=1}^{n} c_{x,i} \leq \mathrm{fbs}(f).$$

The factor of 2 comes from the use of $[-1,1]$ outputs for f; if f takes outputs in $[0,1]$ instead (for example, if we consider the function $f(x) = (1-g(x))/2$ where g takes values in $[-1,1]$ we are comparing the differences $|f(x)-f(y)|$ to the value of $\mathrm{fbs}(g)$), we do not need to divide by 2.

3.2 Relationships to degree

Our results will rely on the following theorem by [9], which states that sensitivity lower bounds the squared degree even for a real-valued function. This theorem follows from results in approximation theory (though it is possible to get a version which is weaker by a constant factor using a proof analogous to the one for the discrete case).

▶ **Theorem 19** ([9]). *For any $f\colon \{\pm1\}^n \to [-1,1]$, we have $\mathrm{s}(f) \leq \deg(f)^2$.*

With this theorem in hand, we show in Corollary 20 that block sensitivity also lower bounds the squared degree (proven in full version).

▶ **Corollary 20.** *For any $f\colon \{\pm1\}^n \to [-1,1]$, we have $\mathrm{bs}(f) \leq \deg(f)^2$.*

Finally, we show in Theorem 21 that fractional block sensitivity lower bounds the squared degree (proven in full version).

▶ **Theorem 21.** *For any $f\colon \{\pm1\}^n \to [-1,1]$, we have $\mathrm{fbs}(f) \leq \frac{\pi^2}{4}\deg(f)^2$.*

Theorem 7 follows.

4 Random restrictions for polynomials

In this section, we show that functions which have low fractional block sensitivity (and therefore, low approximate degree and low quantum query complexity) become *close* to a small-depth decision tree with high probability, after applying a sufficiently strong random restriction. This resolves an open problem in [1], asking whether functions with low approximate degree become close to a DNF on applying a random restriction. We first define a notion of $f(x)$-certificates for a function $F : \{0,1\}^N \to \{0,1,\perp\}$, which is approximated to error $1/3$ by a real-valued function f.

▶ **Definition 22** ($f(x)$-certificate). *Let $F : \{0,1\}^N \to \{0,1,\perp\}$ be a partial function which is approximated to error 1/3 by a real-valued function f. Then a pair (K,x) for $K \subseteq [N]$ is a 1-certificate for F on input $x \in \{0,1\}^N$ if for all $y \in \{0,1\}^N$ such that $y_i = x_i$ for all $i \in K$, $f(y) > 1/2$. In particular, the bits of x in K certify that $F(x) \neq 0$. Similarly, a pair (K,x) for $K \subseteq [N]$ is a 0-certificate for f on input $x \in \{0,1\}^N$ if for all $y \in \{0,1\}^N$ such that $y_i = x_i$ for all $i \in K$, $f(y) \leq 1/2$.*

A random restriction result was shown by [1] for quantum query algorithms, which they use to show that efficient QMA-query algorithms become *close* to a DNF in expectation over the choice of a suitably strong random restriction. We reprove this result (with minor change in parameters) for functions with low fractional block sensitivity in Lemma 23 (proof in full version). The proof is analogous to that of [1], with query magnitudes of a quantum algorithm replaced by the weight assignment for fractional certificate complexity.

▶ **Lemma 23** (Analogue of Theorem 60 of [1]). *Let $f : \{0,1\}^N \to \{0,1,\perp\}$ be a partial function which is approximated to error 1/3 by a real-valued function $\tilde{f}$ of fractional block-sensitivity F. Pick a set $S \subseteq [N]$ where each index $i \in [N]$ is put in S independently with probability p. Choose an arbitrary $k \in \mathbb{N}$ and $x \in \{0,1\}^N$. Define $\tau = \frac{2p^{3/4}F}{k}$ and $K = \{i \in S : c_{x,i} > \tau\}$, $c_{x,i}$ is the fractional certificate of $\tilde{f}$ on index i when the input is x. Then with probability at least $1 - 2e^{-k/6}$ over choice of S, $|K| \leq k$, $|S| \leq 2pN$ and for all $y \in \{0,1\}^N$ with $\{i \in [N] : x_i \neq y_i\} \subseteq S \setminus K$, we have:*

$$|\tilde{f}(x) - \tilde{f}(y)| \leq \frac{4p^{7/4}FN}{k}$$

Note that for Lemma 23, we have that $|K| \geq 1$ with probability at most $\frac{k}{2}p^{1/4}$ by a union bound. In particular, if $k = \text{polylog}(N)$ and p is sufficiently small (say $1/N^{3/4}$), then with reasonably high probability over the choice of S (say $1/N^{2/3}$), none of the indices i such that $c_{x,i} > \tau$ are included in S. Therefore, the restricted function becomes a constant function in this case.

▶ **Theorem 24.** *Let $f : \{0,1\}^N \to \{0,1,\perp\}$ be a partial function which is approximated to error 1/3 by a real-valued function $\tilde{f}$ of fractional block sensitivity F. Pick a set $S \subseteq [N]$ where each index $i \in [N]$ is put in S independently with probability $p \leq (\frac{k}{48FN})^{4/7}$, where $k \in \mathbb{N}$. Then*

$$\forall x, y \in \{0,1\}^N \Pr_S[g(y_{|S}) = f_\rho(y_{|S})] \geq 1 - (2 + \frac{k}{6})e^{-k/6}$$

where $y_{|S}$ is the string y restricted to indices in S, ρ is a p-random restriction defined from x and S as follows:

$$\rho(i) = \begin{cases} * & \text{if } i \in S \\ x_i & \text{otherwise} \end{cases}$$

and g is a width-k^2 DNF dependent on ρ (defined explicitly in the proof). Note that if $f_\rho(y_{|S}) = \perp$, then g is allowed to output 0 or 1 arbitrarily.

Proof. Choose arbitrary $x, y \in \{0,1\}^N$, and choose S as described in the theorem statement to define ρ. Suppose f is approximated to error 1/3 by a real-valued function $\tilde{f}$ of fractional block sensitivity F. Let C be the set of all 1-certificates for $\tilde{f}$ after applying the restriction ρ. Then the DNF g is defined as follows (this is the same DNF that [1] show closeness to):

$$g(y) = \bigvee_{\substack{(K_{x'},x')\in C \\ |K_{x'}|\leq k^2}} \bigwedge_{i\in K_{x'}} y_i = x'_i$$

Note that if $f_\rho(y_{|S}) = 0$, then g outputs 0 because it will not find a 1-certificate in $y_{|S}$. In addition, we do not care about the output of g when $f_\rho(y_{|S}) = \perp$. So now we only worry about the case when $f_\rho(y_{|S}) = 1$. Define $h(x) = \{i \in [N] : c_{x,i} > \tau\}$, where $\tau = \frac{2p^{3/4}F}{k}$ and $c_{x,i}$ is the fractional weight of $\tilde{f}$ on index i for input x. We now think of sampling S in stages (instead of all at once), and we start by sampling S_0 for indices in $h(x)$. Note that the set S is still sampled by putting each bit i in S independently with probability p, we only adopt this viewpoint of stages to analyze the sampling. In particular, since each bit is included in S independently, we can analyze the random restriction by looking at a subset of bits at a time and seeing whether they were included in S or not, without affecting the inclusion/exclusion of other bits, which we can analyze in the next step. Recall, we start by sampling S_0 for indices in $h(x)$. Define $A_0 = \emptyset$, $A_1 = \{i \in S_0 : x_i \neq y_i\}$ and x^{A_i} as the string x with bits in A_i flipped. Then S is sampled further in stage j as follows: S_j is obtained by sampling indices in $h(x^{A_j}) \setminus \cup_{l\leq j-1} h(x^{A_l})$, and $A_{j+1} = \{i \in \cup_{l\leq j} S_l : x_i \neq y_i\}$. Note that $\forall\ j\ |h(x^{A_j})| \leq \frac{F}{\tau}$, and therefore by a union bound, $|S_j| \geq 1$ with probability at most $p\frac{F}{\tau} = \frac{k}{2}p^{1/4} < \frac{1}{e}$ over choice of S_j (we assume that k is sufficiently small, say $k < N^{1/8}$). If $|S_j| = 0$, then we stop the stage-wise analysis and make a decision for all the input bits we haven't analyzed yet to get the full set S. Therefore, the probability that we reach stage $\frac{k}{6}$ of the sampling to sample $S_{\frac{k}{6}}$ is at most $e^{-k/6}$, since the sampling in each stage is performed independently of the previous stages. Note that the decision of whether or not to put any given bit in S is independent of the other bits, by stages of the sampling we only change the order in which we analyze this decision. Further, in each stage of sampling, by Lemma 23, $|S_j| > k$ with probability at most $e^{-k/6}$, thus the probability that any of S_j has size more than k for $j < k/6$ is at most $\frac{k}{6}e^{-k/6}$ by union bound. Therefore, with probability at least $1 - (\frac{k}{6}+1)e^{-k/6}$, $A_{k/6} = A_{k/6-1}$ and $|\cup_{l\leq k/6-1} S_l| \leq \frac{k}{6}k$ (since $|S_j| \leq k$ for all $j \leq k/6-1$). Now we sample the remaining indices of S, and by Lemma 23, $|S| > 2pN$ with probability at most $e^{-k/6}$. So now we assume that $A_{k/6} = A_{k/6-1}$, $|\cup_{l\leq k/6-1} S_l| \leq \frac{k}{6}k$ and $|S| \leq 2pN$, which happens with probability at least $1 - (\frac{k}{6}+2)e^{-k/6}$. For convenience, we now set $j = k/6 - 1$. Define $y' \in \{0,1\}^N$ as follows:

$$y'_i = \begin{cases} y_i & \text{if } i \in S \\ x_i & \text{otherwise} \end{cases}$$

Therefore $y'_{|S} = y_{|S}$ and $f(y') = f_\rho(y_{|S})$. Let $K = h(x^{A_j}) \cap S$. Note that $K \subseteq \cup_{l\leq j} S_l$, because S_j was sampled from bits in $h(x^{A_j}) \setminus \cup_{l\leq j-1} h(x^{A_l})$ and the bits of $h(x^{A_j})$ in $\cup_{l\leq j-1} h(x^{A_l})$ were sampled in $\cup_{l\leq j-1} S_l$. So $|K| \leq \frac{k}{6}k$ because we assume $|\cup_{l\leq j} S_l| \leq \frac{k}{6}k$. From Lemma 23, $|\tilde{f}(x^{A_j}) - \tilde{f}(z)| \leq 1/12$ for all z which differ from x^{A_j} only on indices in $S \setminus K$. In particular, y' differs from x^{A_j} only on bits in $S \setminus K$, because x^{A_j} agrees with y on all bits in $\cup_{l\leq j} S_l$ and K is a subset of these bits. Therefore $|\tilde{f}(x^{A_j}) - \tilde{f}(y')| \leq 1/12$. Therefore, if $f(y') = 1$, then $\tilde{f}(z) > 1/2$ for all z which differ from x^{A_j} only on bits in $S \setminus K$. Thus, (K, x^{A_j}) is a 1-certificate of size at most $\frac{k^2}{6}$ for $\tilde{f}$ when bits outside of S are fixed to those of x. Finally,

$y_{|S}$ is consistent with this certificate, thus $g(y_{|S}) = 1$. Since our assumption holds true with probability at least $1 - (\frac{k}{6} + 2)e^{-k/6}$, therefore, for every $x, y \in \{0,1\}^N$, the statement in the theorem holds true. ◀

The argument above from Theorem 24 actually shows that a given 1-input of f is consistent with a small 1-certificate (of size k^2) with high probability over the choice of unrestricted bits. A symmetric argument can be made for 0-certificates as well. Thus the theorem can be restated as follows:

▶ **Corollary 25.** *Let $f : \{0,1\}^N \to \{0,1,\perp\}$ be a partial function which is approximated to error 1/3 by a real-valued function $\tilde{f}$ of fractional block sensitivity F. Pick a set $S \subseteq [N]$ where each index $i \in [N]$ is put in S independently with probability $p \leq (\frac{k}{48FN})^{4/7}$, where $k \in \mathbb{N}$. Then*

$$\forall x, y \in \{0,1\}^N \Pr_S[C_{y_{|S}}(f_\rho) \leq k^2] \geq 1 - (2 + \frac{k}{6})e^{-k/6}$$

where $y_{|S}$ is the string y restricted to indices in S, ρ is a p-random restriction defined from x and S as follows:

$$\rho(i) = \begin{cases} * & \text{if } i \in S \\ x_i & \text{otherwise} \end{cases}$$

In particular, since we now have small certificates (with high probability) for the restricted function, we can construct a decision tree of small-depth which is correct with high probability over the choice of random restriction and a uniformly random input. We show this in Corollary 26 (proof in the full version).

▶ **Corollary 26.** *Let $f : \{0,1\}^N \to \{0,1,\perp\}$ be a partial function which is approximated to error 1/3 by a real-valued function $\tilde{f}$ of fractional block sensitivity F. Pick a set $S \subseteq [N]$ where each index $i \in [N]$ is put in S independently with probability $p \leq (\frac{k}{48FN})^{4/7}$, where $k \in \mathbb{N}$. Then*

$$\forall x, y \in \{0,1\}^N \Pr_S[g(y_{|S}) = f_\rho(y_{|S})] \geq 1 - (2 + \frac{k}{6})e^{-k/6}$$

where $y_{|S}$ is the string y restricted to indices in S, ρ is a p-random restriction defined from x and S as follows:

$$\rho(i) = \begin{cases} * & \text{if } i \in S \\ x_i & \text{otherwise} \end{cases}$$

and g is a decision tree of depth-k^4 (described explicitly in the proof), dependent on ρ. Note that if $f_\rho(y_{|S}) = \perp$, then g is allowed to output 0 or 1 arbitrarily.

As a corollary of Theorem 24, we can also conclude that functions which have low quantum query complexity also become close to a decision tree of small depth, after applying a random restriction (using Lemma 18).

▶ **Theorem 27.** *Theorem 24 also holds for partial functions $f : \{0,1\}^N \to \{0,1,\perp\}$ of quantum query complexity T, when we set $p \leq (\frac{k}{48\pi^2 T^2 N})^{4/7}$ to obtain a width-k^2 DNF.*

▶ **Theorem 28.** *Corollary 26 also holds for partial functions $f : \{0,1\}^N \to \{0,1,\perp\}$ of quantum query complexity T, when we set $p \leq (\frac{k}{48\pi^2 T^2 N})^{4/7}$ to obtain a depth-k^4 decision tree.*

We discuss in the full version why our proof for the switching lemma from Theorem 27 does not directly extend to QMA-query algorithms, unlike Theorem 65 of [1].

5 Random projections for quantum query algorithms

We now consider the effect of random projections on quantum query algorithms, and we start by showing the following theorem on block random restrictions (where we restrict an entire block of bits at a time, instead of restricting individual bits) for quantum query algorithms. This proof is essentially the same as Theorem 24, see full version for details.

▶ **Theorem 29.** *Let $f : \{0,1\}^{N\times l} \to \{0,1,\perp\}$ be a partial function computable by a quantum query algorithm making T queries to the input. Pick a set $S \subseteq [N]$ where each index $i \in [N]$ is put in S independently with probability $p = (\frac{k}{192^2T^2N})^{4/7}$, where $k \in \mathbb{N}$. Then*

$$\forall x, y \in \{0,1\}^{N\times l} \Pr_S[g(y_{|S}) = f_\rho(y_{|S})] \geq 1 - (2 + \frac{k}{6})e^{-k/6}$$

where $y_{|S}$ is the string y restricted to blocks in S, ρ is a p-block-random restriction defined from x and S as follows:

$$\rho(i,j) = \begin{cases} * & \text{if } i \in S \\ x_{ij} & \text{otherwise} \end{cases}$$

and g is a width-lk^2 DNF dependent on ρ (defined explicitly in the proof). Note that if $f_\rho(y_{|S}) = \perp$, then g is allowed to output 0 or 1 arbitrarily.

We now state in Lemma 30 an average case version of Theorem 29, with respect to two (potentially distinct) block-product distributions from which the strings x and y are sampled (proof in full version). An analogous proof for random restrictions where both the distributions are uniform will recover Corollary 4. Note below that $\{*_{1/2}, 1_{1/2}\}^l \setminus \{1\}^l$ denotes the product distribution where each bit is set independently to $*$ or 1 with probability 1/2, conditioned on not setting every bit to 1.

▶ **Lemma 30.** *Let $D_1 = \otimes_{i=1}^N D_{1i}$ and $D_2 = \otimes_{i=1}^N D_{2i}$ be two block-product distributions on $\{0,1\}^{N\times l}$. Let ρ be a p-block-random restriction with underlying distribution D_1, where each block is sampled from $\{*_{1/2}, 1_{1/2}\}^l \setminus \{1\}^l$ with probability p and from the corresponding block of D_1 otherwise. Let D be the distribution induced by D_2 on the indices set to $*$ by ρ. Let $f : \{0,1\}^{N\times l} \to \{0,1,\perp\}$ be a partial function computable by a quantum query algorithm making T queries to the input. Set $p = (\frac{k}{192^2T^2N})^{4/7}$, then there exists a DNF g of width-lk^2 such that*

$$\mathbb{E}_\rho[\Pr_{z\sim D}[f_\rho(z) \neq g(z)]] \leq (\frac{k}{6} + 2)e^{-k/6}$$

where g is allowed to answer 0 or 1 arbitrarily if $f_\rho(z) = \perp$.

In particular, on applying $\mathsf{proj}_{\rho_{\mathsf{init}}}$ to a quantum query algorithm, the resulting function is close to a DNF of width k^2.

▶ **Corollary 31.** *Let $D_1 = \otimes_{i=1}^N D_{1i}$ be a block-product distribution on $\{0,1\}^{N\times l}$ and $D_2 = \otimes_{i=1}^N D_{2i}$ be a product distribution on $\{0,1\}^N$. Let ρ be a p-block-random restriction with underlying distribution D_1, where each block is sampled from $\{*_{1/2}, 1_{1/2}\}^l \setminus \{1\}^l$ with probability p and from the corresponding block of D_1 otherwise. Let D be the distribution induced by D_2 on the indices set to $*$ by proj_ρ. Let $f : \{0,1\}^{N\times l} \to \{0,1,\perp\}$ be a partial function computable by a quantum query algorithm making T queries to the input. Set $p = (\frac{k}{192^2T^2N})^{4/7}$, then there exists a DNF g of width-k^2 such that*

$$\mathbb{E}_\rho[\Pr_{z\sim D}[\mathsf{proj}_\rho(f)(z) \neq g(z)]] \leq (\frac{k}{6} + 2)e^{-k/6}$$

where g is allowed to answer 0 or 1 arbitrarily if $\mathsf{proj}_\rho(f)(z) = \perp$.

Proof. Consider the DNF h of width-lk^2 obtained from Lemma 30, and define g to be the DNF of width-k^2 obtained as a projection of h, i.e., $g = \mathsf{proj}(h)$, so every variable in h from a given block is mapped to the same variable in g. Then the claim follows from Lemma 30. ◀

5.1 Random projections for AC^0 circuits

In our proof we rely heavily on the notation as well as results from [13], therefore in this section we mention all the notation and results we use. Note that we reference the arXiv version of the paper in the rest of the discussion.

▶ **Definition 32** (SIPSER_d functions [13]). *The SIPSER_d function is a depth d read-once monotone formula with n variables, with alternating layers of AND and OR gates. The bottom layer (adjacent to input variables) consists of AND gates, so the root is an OR gate if d is even and AND gate otherwise. All the gates at a particular depth have the same fan-in, which is denoted by w_i for gates of depth i. So the bottom fan-in is w_{d-1} and the top fan-in is w_0. The parameters t_k correspond to the bias of the distribution for the random projection for depth k.*

$$
\begin{aligned}
w_{d-1} &:= m \\
w &:= \lfloor m2^m / \log e \rfloor \\
q &:= \sqrt{p} = 2^{-m/2} = \Theta\left(\sqrt{\frac{\log w}{w}}\right) \\
w_i &:= w \text{ for } 1 \le k \le d-2 \\
w_0 &:= \min_{i \in \mathbb{N}} \{(1-t_1)^{qi} \le \frac{1}{2}\} = 2^m \ln(2) \cdot (1 \pm o_m(1)) \\
\lambda &:= \frac{(\log w)^{3/2}}{w^{5/4}} \\
t_{d-1} &:= \frac{p-\lambda}{q} = \Theta\left(\sqrt{\frac{\log w}{w}}\right) \\
t_{k-1} &:= \frac{(1-t_k)^{qw} - \lambda}{q} = \Theta\left(\sqrt{\frac{\log w}{w}}\right) \text{ for } 2 \le k \le d-1 \\
n &= \frac{1 \pm o_m(1)}{\log e} \cdot \left(\frac{m2^m}{\log e}\right)^{d-1}
\end{aligned}
$$

Next we describe the addressing scheme used for the gates and input variables of SIPSER_d (which we will also use for SIPSER'_d). Let $A_0 = \{\mathsf{output}\}$, and for $1 \le k \le d$, let $A_k = A_{k-1} \times [w_{k-1}]$. An element of A_k specifies the address of a gate at depth k; so $A_d = \{\mathsf{output}\} \times [w_0] \times \cdots \times [w_{d-1}]$ is the set of addresses of the input variables. For some string $\tau \in \{0,1,*\}^{A_k} = \{0,1,*\}^{A_{k-1} \times [w_{k-1}]}$, use τ_a for $a \in A_{k-1}$ to denote the a^{th} block of τ of length w_{k-1}.

The symbol $\{0_p, 1_{1-p}\}^n$ denotes the random bit string of length n where each bit is sampled independently, and is 0 with probability p and 1 with probability $1-p$. The symbol $\{0_p, 1_{1-p}\}^n \setminus \{x\}$ denotes the product distribution conditioned on not outputting the string x. For our setting, x will be either 0^n or 1^n. The notation $x = a \pm b$ is shorthand for $x \in [a-b, a+b]$.

We modify the first random projection from that of [13], to replace the bottom layer of quantum circuits in QAC^0 circuits by DNFs. Our subsequent random projections are the same as that of [13]. Since the random projections are defined adaptively based on

the outcome of the previous random projection, they define the notion of lift of a random restriction, which tells us the value taken by each gate in the bottom layer of the circuit to which the corresponding random projection has been applied, and this is used to decide how to sample the next random restriction.

▶ **Definition 33** (Lift of a restriction, Definition 7 from [13]). *Let $2 \leq k \leq d$ and $\tau \in \{0,1,*\}^{A_k}$. Assume that the gates of SIPSER_d (or SIPSER'_d) at depth $k-1$ are $\wedge$ gates (otherwise the roles of 0 and 1 below are reversed). The lift of τ is the string $\hat{\tau} \in \{0,1,*\}^{A_{k-1}}$ defined as follows: for each $a \in A_{k-1}$, the coordinate $\hat{\tau}_a$ of $\hat{\tau}$ is*

$$\hat{\tau}_a = \begin{cases} 0 & \text{if } \tau_{a,i} = 0 \text{ for any } i \in [w_{k-1}] \\ 1 & \text{if } \tau_a = \{1\}^{w_{k-1}} \\ * & \text{if } \tau_a \in \{*,1\}^{w_{k-1}} \setminus \{1^{w_{k-1}}\}. \end{cases}$$

To show that the SIPSER_d function retains structure after applying a random projection, they define the notion of a typical restriction. Roughly, on applying a random projection corresponding to a typical restriction to SIPSER_d, the depth of the formula is reduced by 1, and the bottom layer of $\wedge$ gates of SIPSER_d takes on values in $\{0,1,*\}$ such that the bottom fan-in of the projected formula is approximately qw. In addition, the fan-in of the layers above remains roughly the same. More generally, after applying a series of random projections to SIPSER_d, all of which correspond to typical restrictions, applying the next random projection corresponding to a typical restriction ensures that the resulting formula has depth reduced by 1, bottom fan-in approximately qw and fan-ins of all other layers remain roughly the same. [13] show that each of their random restrictions is typical with high probability, assuming that the previous restriction is typical. We will show that the first random restriction which we redefine, is also typical with high probability, and therefore all the subsequent random restrictions remain typical using the results of [13]. Note that in the discussion above and the following definition, we talk about projections corresponding to restrictions which are typical, [13] however, talk about projections corresponding to restrictions for which the lift is typical. So though Definition 34 is identical to theirs, it is stated slightly differently here.

▶ **Definition 34** (Typical random restriction, Definition 14 of [13]). *Let $\tau \in \{0,1,*\}^{A_k}$ where $3 \leq k \leq d$. The restriction τ is typical if*

1. *(Bottom fan-in after projection $\simeq qw$) For all $a \in A_{k-2}$*

$$|\hat{\tau}_a^{-1}(*)| = qw \pm w^{\beta(k-1,d)} \quad \text{where} \quad \beta(k,d) := \frac{1}{3} + \frac{d-k-1}{12d}$$

2. *(Preserves rest of the structure) For all $a \in A_{k-3}$*

$$w_{k-3} - w^{4/5} \leq |(\hat{\hat{\tau}}_a)^{-1}(*)| \leq w_{k-3}$$

These conditions also imply that all the gates in A_{k-3} remain undetermined. This is because suppose τ is applied to a layer of $\wedge$ gates. Then by condition 1, the only values that $\vee$ gates in A_{k-2} can get are $*$ or 1 (so $\wedge$ gates in A_{k-3} have inputs from $*$ and 1). By condition 2, $\wedge$ gates in A_{k-3} have at least one $*$ input, and since none of their inputs is 0, they remain undetermined.

Finally, [13] show that if an OR function (or its restriction) is close to unbiased under some input distribution where each bit of the input is independent and identically distributed, then it has a small correlation with CNFs of small width under this distribution.

▶ **Lemma 35** (Proposition 11.1 of [13]). *Let $C : \{0,1\}^n \to \{0,1\}$ be a CNF of width-r and $\tau \in \{0,1,*\}^n$. Let* OR *be the $\vee$ function on n bits and $\mathbf{Y} \leftarrow \{0_{1-p}, 1_p\}^n$ for $p \in [0,1]$, then*

$$\Pr_{\mathbf{Y}}[(\mathsf{OR}_\tau(\mathbf{Y}) \neq C(\mathbf{Y})] \geq \mathsf{bias}(\mathsf{OR}_\tau, \mathbf{Y}) - rp$$

In order to get an average case depth-hierarchy theorem for AC^0 circuits, [13] show three properties for their sequence of random projections:

1. The sequence of random projections completes to the uniform distribution.
2. The target function SIPSER_d remains "hard" to compute after applying the sequence of random projections.
3. The approximating circuit C trying to compute SIPSER_d "simplifies" greatly after applying the sequence of random projections.

We first re-establish Property 1 after modifying the initial random projection to be applied. Then we establish Property 2 for our modified SIPSER'_d function, by showing that the effect of the modified initial random projection on SIPSER'_d is roughly the same as the effect of the initial random projection of [13] on SIPSER_d. Since our subsequent random projections are the same as theirs, we can conclude that SIPSER'_d retains structure under this sequence of random projections. Finally, we use our random projection result for quantum query algorithm to show that the bottom layer of a QAC^0 circuit simplifies after applying the initial random projection. Then we can use the results of [13] to conclude that the resulting AC^0 circuit also simplifies after applying the subsequent random projections.

5.2 Random projections for QAC^0 circuits

We start by showing an analogue of [10] for QCPH in Proposition 36 (proof in full version), to use a circuit lower bound for QAC^0 to get oracle separation result for QCPH. Note that [1] show a similar analogue for the QMA-hierarchy.

▶ **Proposition 36** (Analogue of [10]). *Let $L \subseteq \{0\}^*$ be some language decided by a $\mathsf{QC}\Sigma_i$ verifier V with oracle access to O, and which has size at most $p(n)$ for inputs of length n. Then for every $n \in \mathbb{N}$, there is a circuit C of size at most $2^{\mathrm{poly}(n)}$ and depth $i+1$, where each layer upto depth i is a layer of AND or OR gates, and depth $i+1$ contains $p(n)$ sized quantum circuits with query complexity at most $p(n)$, such that $\forall\ x \in \{0,1\}^n$*

$$V^O(x) = C(O_{[\leq p(n)]})$$

where $O_{[\leq p(n)]}$ denotes the concatenation of bits of O on all strings of length at most $p(n)$.

We now describe the modification of SIPSER_d function (used in [13]) that we show the QAC^0 lower bound for. The modified function which we use will be called SIPSER'_d and is exactly the same as SIPSER_d defined previously, except for the bottom two fan-ins. The number of variables for SIPSER'_d will be denoted N.

$$\begin{aligned} w &:= \lfloor m 2^m / \log e \rfloor \\ w_{d-2} &:= qwN^{5/7} \\ q' &:= 1/N^{5/7} \\ x &:= \frac{1}{w_{d-2} w^{1/4}} \end{aligned}$$

$$p_1 := x + q'(\frac{p-\lambda}{q})$$
$$w_{d-1} := -\log_2(p_1) = \text{polylog}(N)$$
$$N = \frac{q}{q'}.\frac{n}{m}\log_2(\frac{1}{p_1})$$

Note that N and n are polynomially related. We also redefine the first random projection that is applied to SIPSER'_d, in order to ensure that SIPSER'_d retains structure, while the quantum query algorithms become simple enough to be replaced by DNFs.

▶ **Definition 37** (Restriction for initial random projection). *The underlying random restriction ρ_{init} on $\{0,1,*\}^{A_{d-1}\times[w_{d-1}]}$ is defined as follows: independently for each $a \in A_{d-1}$*

$$\rho(a) \leftarrow \begin{cases} \{1\}^{w_{d-1}} & \text{with probability } x \\ \{*_{1/2}, 1_{1/2}\}^{w_{d-1}} \setminus \{1^{w_{d-1}}\} & \text{with probability } q' \\ \{0_{1/2}, 1_{1/2}\}^{w_{d-1}} \setminus \{1^{w_{d-1}}\} & \text{with probability } 1-x-q' \end{cases}$$

We will call this distribution on random restrictions as $\mathcal{R}_{\mathsf{init}}$.

We first show in Lemma 38 (proof in full version) that the projection defined using $\rho_{\mathsf{init}} \leftarrow \mathcal{R}_{\mathsf{init}}$ on composition with the subsequent random projections of [13] completes to the uniform distribution on $\{0,1\}^N$.

▶ **Lemma 38** (Analogue of Lemma 8.2 of [13]). *Let $\rho \leftarrow \mathcal{R}_{\mathsf{init}}$ and $\mathbf{Y} \leftarrow \{0_{1-t_{d-1}}, 1_{t_{d-1}}\}^{A_{d-1}}$, and let the string $\mathbf{X} \in \{0,1\}^N$ be defined as follows:*

$$\mathbf{X}_{a,i} = \begin{cases} \mathbf{Y}_a & \text{if } \rho_{a,i} = * \\ \rho_{a,i} & \text{otherwise} \end{cases}$$

Then each bit of $\mathbf{X}$ is independent and distributed uniformly at random.

The subsequent $(d-2)$ random projections applied to the SIPSER'_d function are the same as those of [13] (described formally in the full version). It was shown (in the proof of Proposition 8.4) in [13] that the subsequent (d-2) random projections after ρ_{init} (along with a suitably chosen distribution for the unrestricted variables) complete to the distribution $\{0_{1-t_{d-1}}, 1_{t_{d-1}}\}^{(\widehat{\rho_{\mathsf{init}}})^{-1}(*)}$. We state this formally in the full version. Therefore, we can conclude that the overall random projection completes to the uniform distribution, using Lemma 38.

For ease of writing, [13] use $\mathbf{\Psi}(f)$ as notation for the resulting function after applying all the random projections sampled to a function f. We define this formally in the full version.

▶ **Corollary 39** (Proposition 8.1 of [13]). *Consider $C : \{0,1\}^N \to \{0,1\}$ be a circuit computing SIPSER'_d defined on N variables. Let $\mathbf{X} \leftarrow \{0_{1/2}, 1_{1/2}\}^N$ and $\mathbf{Y} \leftarrow \{0_{1-t_1}, 1_{t_1}\}^{w_0}$ (we assume that d is even). If $\mathbf{\Psi}$ is sampled according to the sequence of projections,*

$$\Pr_{\mathbf{X}}[\mathsf{SIPSER}'_d(\mathbf{X}) \neq C(\mathbf{X})] = \Pr_{\mathbf{\Psi},\mathbf{Y}}[(\mathbf{\Psi}(\mathsf{SIPSER}'_d))(\mathbf{Y}) \neq (\mathbf{\Psi}(C))(\mathbf{Y})]$$

We now show that ρ_{init} as sampled above, results in a typical ρ with high probability, in Lemma 40 and Lemma 41 (proof in full version).

▶ **Lemma 40** (Analogue of Lemma 10.3 of [13]). *Let $\tau = \hat{\rho}$ so that $\tau \in \{0,1,*\}^{A_{d-1}}$. Then*

$$\Pr[|\tau_\alpha^{-1}(*)| = qw \pm w^{1/3}] \geq 1 - e^{-\tilde{\Omega}(w^{1/6})} \qquad \forall\ \alpha \in A_{d-2}$$

▶ **Lemma 41** (Analogue of Lemma 10.4 of [13]). *Let $\tau = \hat{\rho}$ so that $\tau \in \{0,1,*\}^{A_{d-1}}$. Then*

$$\Pr[w - w^{4/5} \leq |\hat{\tau}_\alpha^{-1}(*)| \leq w] \geq 1 - e^{-\Omega(w^{4/5})} \qquad \forall\ \alpha \in A_{d-3}$$

▶ **Lemma 42** (Lemma 10.6 and Lemma 10.8 of [13]). *For $2 \leq k \leq d-1$, let $\tau \in \{0,1,*\}^{A_{k+1}}$ be typical. Then $\rho \leftarrow \mathcal{R}(\hat{\tau})$ is typical with probability at least $1 - e^{-\tilde{\Omega}(w^{1/6})}$.*

On application of the overall sequence of random projections $\mathbf{\Psi}$, the function SIPSER'_d becomes an OR of fan-in at most w_0 if d is even, and an AND of fan-in at most w_0 otherwise. We will now assume that SIPSER'_d becomes an OR of fan-in at most w_0, the case for AND is analogous. Then we sample the first (d-2) restrictions and assume they are all typical, which happens with probability at least $1 - de^{-\tilde{\Omega}(w^{1/6})}$ using Lemma 40, Lemma 41 and Lemma 42 and a union bound. We know from Definition 34 that the top OR gate of the function after applying the first (d-2) projections is undetermined and has fan-in at least $w_0 - w^{4/5}$. [13] then show that in expectation over the final random projection, the bias of the projected function is close to 1/2 under the distribution $\{0_{1-t_1}, 1_{t_1}\}^{w_0}$.

▶ **Lemma 43** (Proposition 10.13 of [13]). *Let $\mathbf{Y} \leftarrow \{0_{1-t_1}, 1_{t_1}\}^{w_0}$ and $\mathbf{\Psi}(\mathsf{SIPSER}'_d)$ be the random projection of SIPSER'_d when $\mathbf{\Psi}$ is sampled according to the sequence of projections. Then*

$$\mathop{\mathbb{E}}_{\mathbf{\Psi}}[\mathsf{bias}(\mathbf{\Psi}(\mathsf{SIPSER}'_d), \mathbf{Y})] \geq \frac{1}{2} - \tilde{O}(w^{-1/12})$$

Now we state and use the projection switching lemma of [13] (statement taken from their paper), to obtain our final circuit lower bound.

▶ **Theorem 44** (Proposition 9.2 of [13]). *Let $2 \leq k \leq d-1$ and $F : \{0,1\}^{A_k} \to \{0,1\}$ be a DNF/CNF of width r. Then for all $\tau \in \{0,1,*\}^{A_k}$ and $s \geq 1$,*

$$\Pr_{\rho \leftarrow \mathcal{R}(\tau)}[\mathsf{DT}(\mathsf{proj}_\rho F) > s] = (O(re^{rt_k/(1-t_k)} w^{-1/4}))^s$$

We want to use Theorem 44 (d-2) times on the circuit obtained after applying the first random projection and replacing the layer of quantum circuits in a QAC^0_{d-2} circuit with a small width DNF/CNF as per Corollary 31, to conclude that this circuit becomes a decision tree with high probability. Assuming that $\rho_{\mathsf{init}} = \tau$, we sample $d-2$ random restrictions same as that of [13], and use $\mathbf{\Psi}_\tau$ to denote the composition of the corresponding random projections. We define this formally in the full version.

▶ **Corollary 45** (Analogue of Proposition 9.13 of [13]). *For any constant $d \geq 2$, let $C : \{0,1\}^{A_{d-1}} \to \{0,1\}$ be a depth-d circuit which has $\wedge$ (or $\vee$) as output gate, with bottom fan-in $w^{1/5}$ and size $S \leq 2^{w^{1/5}}$. Then for any $\tau \in \{0,1,*\}^{A_d}$*

$$\Pr_{\mathbf{\Psi}_\tau}[\mathbf{\Psi}_\tau(C) \text{ is a CNF (or DNF) of width at most } w^{1/5}] \geq 1 - e^{-\Omega(w^{1/5})}$$

Proof. Apply Theorem 44 (d-2) times on C, with $r = s = w^{1/5}$, along with a union bound on the number of gates in the bottom layer (at most S) for each application of Theorem 44. ◀

Finally, we can combine all the above ingredients to show the desired circuit lower bound for QAC^0 circuits in Theorem 46 (see full version for proof).

▶ **Theorem 46.** *For any even constant $d \geq 4$, let SIPSER'_d be defined on N variables. Let $C : \{0,1\}^N \to \{0,1\}$ be any QAC^0_{d-2} circuit of size $S = \mathsf{quasipoly}(N) < 2^{N^{\frac{1}{6(d-1)}}}$, which has $\wedge$ as its output gate. Then for a uniformly random input $\mathbf{X}$, we have*

$$\Pr_{\mathbf{X}}[\mathsf{SIPSER}'_d(\mathbf{X}) \neq C(\mathbf{X})] \geq \frac{1}{2} - \frac{1}{N^{\Omega(1/d)}}$$

An analogous proof follows for the case when $d \geq 3$ is odd, by replacing OR with AND (and $\wedge$ with $\vee$) and flipping the values 0 and 1 in the relevant distributions. Note also that the function SIPSER'_d has bottom fan-in polylog(N). Therefore, using Theorem 46, for every $d \geq 2$, we can conclude that relative to a random oracle O, $\Sigma^O_d \not\subset \mathsf{QC}\Pi^O_{d-1}$ when d is odd, and $\Pi^O_d \not\subset \mathsf{QC}\Sigma^O_{d-1}$ when d is even. In particular, for every $d \geq 2$ and relative to a random oracle O, we have that $\Sigma^O_d \not\subset \mathsf{QC}\Pi^O_{d-1}$. In addition, we can conclude that relative to a random oracle, QCPH is infinite (proof in full version).

▶ **Theorem 47.** *With probability 1, a random oracle O satisfies $\Sigma^O_d \not\subset \mathsf{QC}\Pi^O_{d-1}$ for odd $d \geq 3$.*

▶ **Corollary 48.** *With probability 1, a random oracle O satisfies $\Pi^O_d \not\subset \mathsf{QC}\Sigma^O_{d-1}$ for even $d \geq 2$.*

Proof. Similar to Theorem 47, using the analogue of Theorem 46 for SIPSER'_d when d is odd. ◀

▶ **Corollary 49.** *With probability 1, a random oracle O satisfies $\mathsf{QC}\Sigma^O_{d+1} \not\subset \mathsf{QC}\Pi^O_d$ for all $d \geq 1$, and therefore QCPH is infinite relative to O.*

Proof. Follows from Theorem 47, Corollary 48 and collapse theorem for QCPH [3]. ◀

We discuss in the full version how our switching lemma for quantum query algorithms (Theorem 27) compares to the switching lemma (Theorem 65) from [1], and why the latter is not sufficient to establish our oracle separation result.

6 Oracle Separation for QCMA Hierarchy

In this section, we show that there exists an oracle such that the QCMA hierarchy (called QCMAH) is infinite, and no fixed level of QCMA hierarchy contains all of PH. We first redefine the SIPSER function, which we will call SIPSER''. The function SIPSER''_d will be an AND-OR tree of depth $2d+2$ (with AND gates at the bottom), and we will show that it can not be computed by a circuit corresponding to a QCMAH_d machine. The exact parameters for SIPSER''_d and some properties of the function are given in the full version.

We will apply $2d-1$ random projections. For $1 \leq i \leq d$, we sample random restrictions $\rho_{i,1} \in \{0,1,*\}^{A_{2i+2}}$ which acts on quantum query algorithms and $\rho_{i,2} \in \{0,1,*\}^{A_{2i+1}}$ which acts on DNFs. These are defined formally and some properties of the projections are shown in the full version.

A proof very similar to Proposition 8.4 of [13] (stated in full version) and Lemma 38 can then be used to conclude that this sequence of random projections sampled completes to the uniform distribution. More strongly, the proof of Proposition 8.4 of [13] tells us that for $1 \leq i \leq d$ projections upto $\rho_{i,2}$ complete to the t_{2i+1} biased product distribution and projections upto $\rho_{i,1}$ complete to the t_{2i+2} biased product distribution. This fact will be useful later to be able to apply the projection switching lemma for quantum algorithms repeatedly.

In order to show that the SIPSER''_d formula retains structure after applying each random projection, we modify the definition of typical restrictions (Definition 34) slightly, to account for the new parameters. We state the formal definition in the full version. We then need to show that the sequence of projections applied is typical. It follows from Lemma 40 and Lemma 41 that $\rho_{d,1}$ is typical with probability at least $1 - e^{-\Omega(w^{1/6})}$. It follows from Lemma 42 that $\rho_{i,2}$ is typical, for $1 \leq i \leq d$ (note that we have changed the fan-ins of the function and hence the definition of typical restriction, but the proof for Lemma 42 works to establish this as well). We then show (proof in the full version) that $\rho_{i,1}$ is typical with high probability, for $1 \leq i \leq d-1$. This proof is very similar to that of Lemma 42 and can be adapted to show typicality for $\rho_{i,2}$ as well.

Finally, we show an analogue of [10] for the QCMA hierarchy in Proposition 50 (proof in full version), to use a circuit lower bound to get an oracle separation result.

▶ **Proposition 50** (Analogue of [10]). *Let $L \subseteq \{0\}^*$ be some language decided by a QCMAH_i verifier $V = \langle V_1, \ldots, V_i \rangle$ with oracle access to O, and where each V_j has size at most $p(n)$ for inputs of length n. Let $q_i(n) = p(p(\ldots p(n)\ldots)$ where p is composed i times. Then for every $n \in \mathbb{N}$, there is a circuit C of size at most $2^{\mathrm{poly}(n)}$ and depth $2i$, where the top layer is an OR gate and the layers alternate between OR gates of width $2^{q_i(n)}$, and quantum circuits of size $q_i(n)$ with query complexity at most $q_i(n)$ such that $\forall\ x \in \{0,1\}^n$*

$$V^O(x) = C(O_{[\leq q_i(n)]})$$

where $O_{[\leq q_i(n)]}$ denotes the concatenation of bits of O on all strings of length at most $q_i(n)$.

The proof for the depth-hierarchy theorem now follows very similarly to the proof for QCPH (or QAC^0 circuits) in Theorem 46. We noted earlier that the sequence of random projections completes to an appropriately biased product distribution at every level of the AND-OR tree, hence we can apply the projection switching lemma for quantum algorithms which works for arbitrary inner and outer product distributions. This switching lemma and the projection switching lemma for DNFs Theorem 44 by [13] are applied in an alternating manner to get a depth-hierarchy theorem (the projection switching lemma for quantum algorithms is applied for $\rho_{i,1}$ and the projection switching lemma for DNFs is applied for $\rho_{i,2}$). Then oracle separations analogous to the case for QCPH follow for QCMAH using a standard diagonalization argument. Note however that now we will get an oracle separation between Σ_{2d+1} and QCMAH_d relative to a random oracle O (whereas for QCPH we got an oracle separation between Σ_{d+1} and $\mathsf{QC\Pi}_d$).

References

1. Scott Aaronson, DeVon Ingram, and William Kretschmer. The acrobatics of BQP. In *37th Computational Complexity Conference, CCC*, 2022. doi:10.4230/LIPIcs.CCC.2022.20.
2. Avantika Agarwal and Shalev Ben-David. Oracle separations for the quantum-classical polynomial hierarchy. *CoRR*, abs/2410.19062, 2024. doi:10.48550/arXiv.2410.19062.
3. Avantika Agarwal, Sevag Gharibian, Venkata Koppula, and Dorian Rudolph. Quantum Polynomial Hierarchies: Karp-Lipton, Error Reduction, and Lower Bounds. In *49th International Symposium on Mathematical Foundations of Computer Science (MFCS 2024)*, 2024. doi:10.4230/LIPIcs.MFCS.2024.7.
4. Anurag Anshu, Shalev Ben-David, and Srijita Kundu. On query-to-communication lifting for adversary bounds. In *36th Computational Complexity Conference, CCC*, 2021. doi:10.4230/LIPIcs.CCC.2021.30.
5. Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. doi:10.1145/502090.502097.

6 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, October 1997. doi:10.1137/s0097539796300933.

7 Sreejata Kishor Bhattacharya. Aaronson-ambainis conjecture is true for random restrictions, 2024. doi:10.48550/arXiv.2402.13952.

8 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theor. Comput. Sci.*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

9 Yuval Filmus, Hamed Hatami, Nathan Keller, and Noam Lifshitz. On the sum of the l1 influences of bounded functions, 2015. arXiv:1404.3396.

10 Merrick Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical systems theory*, 17(1):13–27, 1984. doi:10.1007/BF01744431.

11 Sevag Gharibian, Miklos Santha, Jamie Sikora, Aarthi Sundaram, and Justin Yirka. Quantum generalizations of the polynomial hierarchy with applications to QMA(2). *Comput. Complex.*, 31(2):13, 2022. doi:10.1007/S00037-022-00231-8.

12 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20. ACM, 1986. doi:10.1145/12130.12132.

13 Johan Håstad, Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. *J. ACM*, 64(5):35:1–35:27, 2017. doi:10.1145/3095799.

14 Avishay Tal. Properties and applications of boolean function composition. In *Innovations in Theoretical Computer Science, ITCS*, pages 441–454. ACM, 2013. doi:10.1145/2422436.2422485.