BQP and the Polynomial Hierarchy*

Scott Aaronson†
MIT

ABSTRACT

The relationship between BQP and PH has been an open problem since the earliest days of quantum computing. We present evidence that quantum computers can solve problems outside the entire polynomial hierarchy, by relating this question to topics in circuit complexity, pseudorandomness, and Fourier analysis.

First, we show that there exists an oracle relation problem (i.e., a problem with many valid outputs) that is solvable in BQP, but not in PH. This also yields a non-oracle relation problem that is solvable in quantum *logarithmic* time, but not in AC^0.

Second, we show that an oracle *decision* problem separating BQP from PH would follow from the *Generalized Linial-Nisan Conjecture*, which we formulate here and which is likely of independent interest. The original Linial-Nisan Conjecture (about pseudorandomness against constant-depth circuits) was recently proved by Braverman, after being open for twenty years.

Categories and Subject Descriptors

F.1.3 [**Theory of Computation**]: Computation by Abstract Devices—*Complexity Measures and Classes*

General Terms

Theory

Keywords

constant-depth circuits, Fourier analysis, Linial-Nisan Conjecture, quantum complexity classes

*We regret that, because of space limitations, we are unable to prove anything in this extended abstract. Readers interested in the proofs should stop reading now and go to www.scottaaronson.com/papers/bqpph.pdf.

†Based upon work supported by the National Science Foundation under Grant No. 0844626. Also supported by a DARPA YFA grant and the Keck Foundation.

1. INTRODUCTION

A central task of quantum computing theory is to understand how BQP—Bounded-Error Quantum Polynomial-Time, the class of all problems feasible for a quantum computer—fits in with classical complexity classes. In their original 1993 paper defining BQP, Bernstein and Vazirani [9] showed that $\mathsf{BPP} \subseteq \mathsf{BQP} \subseteq \mathsf{P}^{\#\mathsf{P}}$.[1] Informally, this says that quantum computers are at least as fast as classical probabilistic computers and no more than exponentially faster (indeed, they can be simulated using an oracle for counting). Bernstein and Vazirani also gave evidence that $\mathsf{BPP} \neq \mathsf{BQP}$, by exhibiting an oracle problem called Recursive Fourier Sampling that requires $n^{\Omega(\log n)}$ queries on a classical computer but only n queries on a quantum computer. The evidence for the power of quantum computers became dramatically stronger a year later, when Shor [23] (building on work of Simon [24]) showed that Factoring and Discrete Logarithm are in BQP. On the other hand, Bennett et al. [8] gave oracle evidence that $\mathsf{NP} \not\subset \mathsf{BQP}$, and while no one regards such evidence as decisive, today it seems extremely unlikely that quantum computers can solve NP-complete problems in polynomial time. A vast body of research, continuing to the present, has sought to map out the detailed boundary between those NP problems that are feasible for quantum computers and those that are not.

However, there is a complementary question that—despite being universally recognized as one of the "grand challenges" of the field—has had essentially zero progress over the last sixteen years:

> *Is* BQP *in* NP*? More generally, is* BQP *contained anywhere in the polynomial hierarchy* $\mathsf{PH} = \mathsf{NP} \cup \mathsf{NP}^{\mathsf{NP}} \cup \mathsf{NP}^{\mathsf{NP}^{\mathsf{NP}}} \cup \cdots$*?*

The "default" conjecture is presumably $\mathsf{BQP} \not\subset \mathsf{PH}$, since no one knows what a simulation of BQP in PH would look like. Before this work, however, there was no formal evidence for or against that conjecture. Almost all the problems for which we have quantum algorithms—including Factoring and Discrete Logarithm—are easily seen to be in $\mathsf{NP} \cap \mathsf{coNP}$.[2] One notable exception is Recursive Fourier Sampling, the problem that Bernstein and Vazirani [9] originally used to construct an oracle A relative to which $\mathsf{BPP}^A \neq$

[1]The upper bound was later improved to $\mathsf{BQP} \subseteq \mathsf{PP}$ by Adleman, DeMarrais, and Huang [2].

[2]Here we exclude BQP-complete problems such as approximating the Jones polynomial [3], which, by the very fact of being BQP-complete, seem hard to interpret as "evidence" for $\mathsf{BQP} \not\subset \mathsf{PH}$.

BQP^A. One can show, without too much difficulty, that Recursive Fourier Sampling yields oracles A relative to which $\mathsf{BQP}^A \not\subset \mathsf{NP}^A$ and indeed $\mathsf{BQP}^A \not\subset \mathsf{MA}^A$. However, while it is reasonable to conjecture that Recursive Fourier Sampling (as an oracle problem) is not in PH, it is open even to show that this problem (or any other BQP oracle problem) is not in AM! Recall that $\mathsf{AM} = \mathsf{NP}$ under plausible derandomization assumptions. Thus, until we solve the problem of constructing an oracle A such that $\mathsf{BQP}^A \not\subset \mathsf{AM}^A$, we cannot even claim to have oracle evidence (which is itself, of course, a weak form of evidence) that $\mathsf{BQP} \not\subset \mathsf{NP}$.

Before going further, we should clarify that there are two questions here: whether $\mathsf{BQP} \subseteq \mathsf{PH}$ and whether $\mathsf{PromiseBQP} \subseteq \mathsf{PromisePH}$. In the unrelativized world, it is entirely possible that quantum computers can solve promise problems outside the polynomial hierarchy, but that all *languages* in BQP are nevertheless in PH. However, for the specific purpose of constructing an oracle A such that $\mathsf{BQP}^A \not\subset \mathsf{PH}^A$, the two questions are equivalent, basically because one can "offload" a promise into the construction of the oracle A.

1.1 Motivation

There are at least four reasons why the BQP versus PH question is so interesting. At a basic level, it is both theoretically and practically important to understand what classical resources are needed to simulate quantum physics. For example, when a quantum system evolves to a given state, is there always a short classical proof that it does so? Can one estimate quantum amplitudes using approximate counting (which would imply $\mathsf{BQP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$)? If something like this were true, then while the exponential speedup of Shor's factoring algorithm might stand, quantum computing would nevertheless seem much less different from classical computing than previously thought.

Second, if $\mathsf{BQP} \not\subset \mathsf{PH}$, then many possibilities for new quantum algorithms might open up to us. One often hears the complaint that there are too few quantum algorithms, or that progress on quantum algorithms has slowed since the mid-1990s. In our opinion, the real issue here has nothing to do with quantum computing, and is simply that there are too few natural NP-intermediate problems for which there plausibly *could be* quantum algorithms! In other words, instead of focussing on Graph Isomorphism and a small number of other NP-intermediate problems, it might be fruitful to look for quantum algorithms solving completely different types of problems—problems that are not necessarily even in PH. In this paper, we will see a new example of such a quantum algorithm, which solves a problem called Fourier Checking.

Third, it is natural to ask whether the $\mathsf{P} \stackrel{?}{=} \mathsf{BQP}$ question is related to that *other* fundamental question of complexity theory, $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$. More concretely, is it possible that quantum computers could provide exponential speedups even if $\mathsf{P} = \mathsf{NP}$? If $\mathsf{BQP} \subseteq \mathsf{PH}$, then certainly the answer to that question is no (since $\mathsf{P} = \mathsf{NP} \implies \mathsf{P} = \mathsf{PH}$). Therefore, if we want evidence that quantum computing could survive a collapse of P and NP, we must also seek evidence that $\mathsf{BQP} \not\subset \mathsf{PH}$.

Fourth, a major challenge for quantum computing research is to *get better evidence that quantum computers cannot solve NP-complete problems in polynomial time.* As an example, could we show that if $\mathsf{NP} \subseteq \mathsf{BQP}$, then the polynomial hierarchy collapses? At first glance, this seems like a wild hope; certainly we have no idea at present how to prove anything of the kind. However, notice that if $\mathsf{BQP} \subseteq \mathsf{AM}$, then the desired implication would follow immediately! For in that case, $\mathsf{coNP} \subseteq \mathsf{BQP}$, hence $\mathsf{coNP} \subseteq \mathsf{AM}$, hence $\mathsf{PH} = \Sigma_2^{\mathsf{P}}$, where the last implication was shown by Boppana, Håstad, and Zachos [10].

1.2 Our Results

This paper presents the first formal evidence for the possibility that $\mathsf{BQP} \not\subset \mathsf{PH}$. Perhaps more importantly, it places the relativized BQP versus PH question at the frontier of (classical) circuit lower bounds. The heart of the problem, we will find, is to extend Braverman's spectacular recent proof [11] of the Linial-Nisan Conjecture, in ways that would reveal a great deal of information about small-depth circuits independent of the implications for quantum computing.

We have two main contributions. First, we achieve an oracle separation between BQP and PH for the case of *relation problems*. A relation problem is simply a problem where the desired output is an n-bit string (rather than a single bit), and any string from some nonempty set S is acceptable. Relation problems arise often in theoretical computer science; one well-known example is finding a Nash equilibrium (shown to be PPAD-complete by Daskalakis et al. [13]). Within quantum computing, there is considerable precedent for studying relation problems as a warmup to the harder case of decision problems. For example, in 2004 Bar-Yossef, Jayram, and Kerenidis [4] gave a relation problem with quantum one-way communication complexity $O(\log n)$ and randomized one-way communication complexity $\Omega(\sqrt{n})$. It took several more years for Gavinsky et al. [15] to achieve the same separation for decision problems, and the proof was much more complicated.[3]

Formally, our result is as follows:

Theorem 1. *There exists an oracle A relative to which $\mathsf{FBQP}^A \not\subset \mathsf{FBPP}^{\mathsf{PH}^A}$, where FBQP and FBPP are the relation versions of BQP and BPP respectively.*[4]

Interestingly, the oracle A in Theorem 1 can be taken to be a *random* oracle. By contrast, Aaronson and Ambainis [1] have shown that, under a plausible conjecture about influences in low-degree polynomials, one cannot separate the decision classes BQP and BPP by a random oracle without also separating P from $\mathsf{P}^{\#\mathsf{P}}$ in the unrelativized world. In other words, the use of relation problems (rather than decision problems) seems essential if one wants a random oracle separation.

Underlying Theorem 1 is a new lower bound against AC^0 circuits (constant-depth circuits composed of AND, OR, and NOT gates). The close connection between AC^0 and the polynomial hierarchy that we exploit is not new. In the early 1980s, Furst-Saxe-Sipser [14] and Yao [28] noticed that, if we have a PH machine M that computes (say) the Parity

[3] Recently, Shepherd and Bremner [22] proposed a beautiful *sampling* problem that is solvable efficiently on a quantum computer but that they conjecture is hard classically. This provides yet another example where it seems easier to find evidence for the power of quantum computers once one moves away from decision problems.

[4] Confusingly, the F stands for "function"; we are simply following the standard naming convention for classes of relation problems (FP, FNP, etc).

of a 2^n-bit oracle string, then by simply reinterpreting the existential quantifiers of M as OR gates and the universal quantifiers as AND gates, we obtain an AC^0 circuit of size $2^{\mathrm{poly}(n)}$ solving the same problem. It follows that, if we can prove a $2^{\omega(\mathrm{polylog}\, n)}$ lower bound on the size of AC^0 circuits computing PARITY, we can construct an oracle A relative to which $\oplus\mathsf{P}^A \not\subset \mathsf{PH}^A$. The idea is the same for constructing an A relative to which $\mathcal{C}^A \not\subset \mathsf{PH}^A$, where $\mathcal{C}$ is any complexity class.

Indeed, the relation between PH and AC^0 is so direct that we get the following as a more-or-less immediate counterpart to Theorem 1:

THEOREM 2. *In the unrelativized world (with no oracle), there exists a relation problem solvable in* $\mathsf{BQLOGTIME}$ *but not in nonuniform* AC^0.

Here $\mathsf{BQLOGTIME}$ is the class of problems solvable by uniform logarithmic-*size* quantum circuits, with random access to the input string $x_1 \ldots x_n$. It should not be confused with the much larger class BQNC, of problems solvable by logarithmic-*depth* quantum circuits.

The relation problem that we use to separate BQP from PH, and $\mathsf{BQLOGTIME}$ from AC^0, is called FOURIER FISHING. The problem can be informally stated as follows. We are given oracle access to n Boolean functions $f_1, \ldots, f_n : \{0,1\}^n \rightarrow \{-1,1\}$, which we think of as chosen uniformly at random. The task is to output n strings, $z_1, \ldots, z_n \in \{0,1\}^n$, such that the corresponding squared Fourier coefficients $\widehat{f_1}(z_1)^2, \ldots, \widehat{f_n}(z_n)^2$ are "often much larger than average." Notice that if f_i is a random Boolean function, then each of its Fourier coefficients $\widehat{f_i}(z)$ follows a normal distribution—meaning that with overwhelming probability, a constant fraction of the Fourier coefficients will be a constant factor larger than the mean. Furthermore, it is straightforward to create a quantum algorithm that samples each z with probability proportional to $\widehat{f_i}(z)^2$, so that larger Fourier coefficients are more likely to be sampled than smaller ones.

On the other hand, computing any *specific* $\widehat{f_i}(z)$ is easily seen to be equivalent to summing 2^n bits. By well-known lower bounds on the size of AC^0 circuits computing the MAJORITY function (see Håstad [26] for example), it follows that, for any fixed z, computing $\widehat{f_i}(z)$ cannot be in PH as an oracle problem. Unfortunately, this does not directly imply any separation between BQP and PH, since the quantum algorithm does not compute $\widehat{f_i}(z)$ either: it just samples a z with probability proportional to $\widehat{f_i}(z)^2$. However, we will show that, if there exists a $\mathsf{BPP}^{\mathsf{PH}}$ machine M that even *approximately* simulates the behavior of the quantum algorithm, then one can solve MAJORITY by means of a *nondeterministic* reduction—which uses approximate counting to estimate $\Pr[M \text{ outputs } z]$, and adds a constant number of layers to the AC^0 circuit. The central difficulty is that, if M knew the specific z for which we were interested in estimating $\widehat{f_i}(z)$, then it could choose adversarially never to output that z. To solve this, we will show that we can "smuggle" a MAJORITY instance into the estimation of a *random* Fourier coefficient $\widehat{f_i}(z)$, in such a way that it is information-theoretically impossible for M to determine which z we care about.

Our second contribution is to define and study a new black-box decision problem, called FOURIER CHECKING. Informally, in this problem we are given oracle access to *two* Boolean functions $f, g : \{0,1\}^n \rightarrow \{-1,1\}$, and are promised that either

(i) f and g are both uniformly random, or

(ii) f is uniformly random, while g is extremely well correlated with f's Fourier transform over $\mathbb{Z}_2^n$ (which we call "forrelated").

The problem is to decide whether (i) or (ii) is the case.

It is not hard to show that FOURIER CHECKING is in BQP: basically, one can prepare a uniform superposition over all $x \in \{0,1\}^n$, then query f, apply a quantum Fourier transform, query g, and check whether one has recovered something close to the uniform superposition. On the other hand, being forrelated seems like an extremely "global" property of f and g: one that would not be apparent from querying *any* small number of $f(x)$ and $g(y)$ values. And thus, one might conjecture that FOURIER CHECKING (as an oracle problem) is not in PH.

In this paper, we adduce strong evidence for that conjecture. Specifically, we show that for every $k \leq 2^{n/4}$, the forrelated distribution over $\langle f, g \rangle$ pairs is $O\left(k^2/2^{n/2}\right)$-*almost k-wise independent*. By this we mean that, if one had $1/2$ prior probability that f and g were uniformly random, and $1/2$ prior probability that f and g were forrelated, then even conditioned on any k values of f and g, the posterior probability that f and g were forrelated would still be

$$\frac{1}{2} \pm O\left(\frac{k^2}{2^{n/2}}\right).$$

We conjecture that this almost k-wise independence property is enough, by itself, to imply that an oracle problem is not in PH. We call this the *Generalized Linial-Nisan Conjecture*.

Without the $\pm O\left(k^2/2^{n/2}\right)$ error term, our conjecture would be equivalent[5] to a famous conjecture in circuit complexity made by Linial and Nisan [18] in 1990. Their conjecture stated that *polylogarithmic independence fools* AC^0: in other words, every probability distribution over N-bit strings that is *uniform* on every small subset of bits, is indistinguishable from the truly uniform distribution by AC^0 circuits. When we began investigating this topic a year ago, even the original Linial-Nisan Conjecture was still open. Since then, Braverman [11] (building on earlier work by Bazzi [5] and Razborov [20]) has given a beautiful proof of that conjecture. In other words, to construct an oracle relative to which $\mathsf{BQP} \not\subset \mathsf{PH}$, it now suffices to generalize Braverman's Theorem from k-wise independent distributions to almost k-wise independent ones. We believe that this is by far the most promising approach to the BQP versus PH problem.

Alas, generalizing Braverman's proof is much harder than one might have hoped. To prove the original Linial-Nisan Conjecture, Braverman showed that every AC^0 function $f : \{0,1\}^n \rightarrow \{0,1\}$ can be well-approximated, in the L_1-norm, by *low-degree sandwiching polynomials*: real polynomials $p_\ell, p_u : \mathbb{R}^n \rightarrow \mathbb{R}$, of degree $O(\mathrm{polylog}\, n)$, such that $p_\ell(x) \leq f(x) \leq p_u(x)$ for all $x \in \{0,1\}^n$. Since p_ℓ and p_u trivially have the same expectation on any k-wise independent

[5] Up to unimportant variations in the parameters

distribution that they have on the uniform distribution, one can show that f must have almost the same expectation as well. To generalize Braverman's result from k-wise independence to almost k-wise independence, we will show that it suffices to construct low-degree sandwiching polynomials that satisfy a certain additional condition. This new condition (which we call "low-fat") basically says that p_ℓ and p_u must be representable as linear combinations of *terms* (that is, products of x_i's and $(1-x_i)$'s), in such a way that the sum of the absolute values of the coefficients is bounded—thereby preventing "massive cancellations" between positive and negative terms. Unfortunately, while we know two techniques for approximating AC^0 functions by low-degree polynomials—that of Linial-Mansour-Nisan [17] and that of Razborov [19] and Smolensky [25]—neither technique provides anything like the control over coefficients that we need. To construct low-fat sandwiching polynomials, it seems necessary to reprove the LMN and Razborov-Smolensky theorems in a more "conservative," less "profligate" way. And such an advance seems likely to lead to breakthroughs in circuit complexity and computational learning theory having nothing to do with quantum computing.

Let us mention three further applications of Fourier Checking:

First, if the Generalized Linial-Nisan Conjecture holds, then just like with Fourier Fishing, we can "scale down by an exponential," to obtain a promise problem that is in $\mathsf{BQLOGTIME}$ but not in AC^0.

Second, without any assumptions, we can prove the new results that there exist oracles relative to which $\mathsf{BQP} \not\subset \mathsf{BPP}_{\mathsf{path}}$ and $\mathsf{BQP} \not\subset \mathsf{SZK}$. We can also reprove all previous oracle separations between BQP and classical complexity classes in a unified fashion.

Third, independent of its applications to complexity classes, Fourier Checking yields the largest gap between classical and quantum query complexity yet known. Its bounded-error randomized query complexity is $\Omega(N^{1/4})$ (and we conjecture $\Theta(\sqrt{N})$), while its quantum query complexity is only $O(1)$. Prior to this work, the best known gaps were of the form $N^{\Omega(1)}$ versus $O(\log N)$ (for Simon's and Shor's algorithms), or $\Theta(\log N)$ versus $O(1)$ (for the Bernstein-Vazirani problem).[6] Buhrman et al. [12] explicitly raised the open problem of finding a larger gap. We also get a *property* (namely, the property of being forrelated) that takes $N^{\Omega(1)}$ queries to test classically but only $O(1)$ queries to test quantumly.

1.3 In Defense of Oracles

This paper is concerned with finding oracles relative to which BQP outperforms classical complexity classes. As such, it is open to the usual objections: "But don't oracle results mislead us about the 'real' world? What about non-relativizing results like $\mathsf{IP} = \mathsf{PSPACE}$ [21]?"

In our view, it is most helpful to think of oracle separations, not as strange metamathematical claims, but as *lower bounds in a concrete computational model that is natural and well-motivated in its own right.* The model in question is *query complexity*, where the resource to be minimized is the number of accesses to a very long input string. When someone gives an oracle A relative to which $\mathcal{C}^A \not\subset \mathcal{D}^A$, what they really mean is simply that they have found a problem that $\mathcal{C}$ machines can solve using superpolynomially fewer queries than $\mathcal{D}$ machines. In other words, $\mathcal{C}$ has "cleared the first possible obstacle"—the query complexity obstacle—to having capabilities beyond those of $\mathcal{D}$. Of course, it could be (and sometimes is) that $\mathcal{C} \subseteq \mathcal{D}$ for other reasons, but if we do not *even* have a query complexity lower bound, then proving one is in some sense the obvious place to start.

Oracle separations have played a role in many of the central developments of both classical and quantum complexity theory. As mentioned earlier, proving query complexity lower bounds for PH machines is essentially equivalent to proving size lower bounds for AC^0 circuits—and indeed, the pioneering AC^0 lower bounds of the early 1980s were explicitly motivated by the goal of proving oracle separations for PH.[7] Within quantum computing, oracle results have played an even more decisive role: the first evidence for the power of quantum computers came from the oracle separations of Bernstein-Vazirani [9] and Simon [24], and Shor's algorithm [23] contains an oracle algorithm (for the Period-Finding problem) at its core.

Having said all that, if for some reason one still feels averse to the language of oracles, then (as mentioned before) one is free to scale everything down by an exponential, and to reinterpret a relativized separation between BQP and PH as an *un*relativized separation between $\mathsf{BQLOGTIME}$ and AC^0.

2. PRELIMINARIES

It will be convenient to consider Boolean functions of the form $f : \{0,1\}^n \rightarrow \{-1,1\}$. Throughout this paper, we let $N = 2^n$; we will often view the truth table of a Boolean function as an "input" of size N. Given a Boolean function $f : \{0,1\}^n \rightarrow \{-1,1\}$, the Fourier transform of f is defined as

$$\widehat{f}(z) := \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot z} f(x).$$

Recall Parseval's identity:

$$\sum_{x \in \{0,1\}^n} f(x)^2 = \sum_{z \in \{0,1\}^n} \widehat{f}(z)^2 = N.$$

2.1 Problems

We first define the Fourier Fishing problem, in both "distributional" and "promise" versions. In the distributional version, we are given oracle access to n Boolean functions $f_1, \ldots, f_n : \{0,1\}^n \rightarrow \{-1,1\}$, which are chosen uniformly and independently at random. The task is to output n strings, $z_1, \ldots, z_n \in \{0,1\}^n$, at least 75% of which satisfy $\left|\widehat{f}_i(z_i)\right| \geq 1$ and at least 25% of which satisfy $\left|\widehat{f}_i(z_i)\right| \geq 2$. (Note that these thresholds are not arbitrary, but were carefully chosen to produce a separation between the quantum and classical models!)

We now want a version of Fourier Fishing that removes the need to assume the f_i's are uniformly random, replacing

[6] De Beaudrap, Cleve, and Watrous [7] gave an $N^{\Omega(1)}$ versus $O(1)$ separation, but for a non-standard model of quantum query complexity where one gets to map $|x\rangle$ to $|\sigma(x)\rangle$ for a permutation σ (rather than $|x\rangle|y\rangle$ to $|x\rangle|y \oplus \sigma(x)\rangle$ as in the standard model).

[7] Yao's paper [28] was entitled "Separating the polynomial-time hierarchy by oracles"; the Furst-Saxe-Sipser paper [14] was entitled "Parity, circuits, and the polynomial time hierarchy."

it with a worst-case promise on the f_i's. Call an n-tuple $\langle f_1, \ldots, f_n \rangle$ of Boolean functions *good* if

$$\sum_{i=1}^{n} \sum_{z_i : \left|\widehat{f}_i(z_i)\right| \geq 1} \widehat{f}_i(z_i)^2 \geq 0.8Nn,$$

$$\sum_{i=1}^{n} \sum_{z_i : \left|\widehat{f}_i(z_i)\right| \geq 2} \widehat{f}_i(z_i)^2 \geq 0.26Nn.$$

(We will show in Lemma 6 that the vast majority of $\langle f_1, \ldots, f_n \rangle$ are good.) In Promise Fourier Fishing, we are given oracle access to Boolean functions $f_1, \ldots, f_n : \{0,1\}^n \rightarrow \{-1,1\}$, which are promised to be good. The task, again, is to output strings $z_1, \ldots, z_n \in \{0,1\}^n$, at least 75% of which satisfy $\left|\widehat{f}_i(z_i)\right| \geq 1$ and at least 25% of which satisfy $\left|\widehat{f}_i(z_i)\right| \geq 2$.

Next we define a decision problem called Fourier Checking. Here we are given oracle access to two Boolean functions $f, g : \{0,1\}^n \rightarrow \{-1,1\}$. We are promised that either

(i) $\langle f, g \rangle$ was drawn from the uniform distribution $\mathcal{U}$, which sets every $f(x)$ and $g(y)$ by a fair, independent coin toss.

(ii) $\langle f, g \rangle$ was drawn from the "forrelated" distribution $\mathcal{F}$, which is defined as follows. First choose a random real vector $v = (v_x)_{x \in \{0,1\}^n} \in \mathbb{R}^N$, by drawing each entry independently from a Gaussian distribution with mean 0 and variance 1. Then set $f(x) := \mathrm{sgn}(v_x)$ and $g(x) := \mathrm{sgn}(\widehat{v}_x)$ for all x. Here $\mathrm{sgn}(\alpha) := \alpha / |\alpha|$, and $\widehat{v}$ is the Fourier transform of v over $\mathbb{Z}_2^n$:

$$\widehat{v}_y := \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot y} v_x.$$

In other words, f and g *individually* are still uniformly random, but they are no longer independent: now g is extremely well correlated with the Fourier transform of f (hence "forrelated").

The problem is to accept if $\langle f, g \rangle$ was drawn from $\mathcal{F}$, and to reject if $\langle f, g \rangle$ was drawn from $\mathcal{U}$. Note that, since $\mathcal{F}$ and $\mathcal{U}$ overlap slightly, we can only hope to succeed with overwhelming probability over the choice of $\langle f, g \rangle$, not for every $\langle f, g \rangle$ pair.

We can also define a promise-problem version of Fourier Checking. In Promise Fourier Checking, we are promised that the quantity

$$p(f,g) := \frac{1}{N^3} \left(\sum_{x,y \in \{0,1\}^n} f(x) (-1)^{x \cdot y} g(y) \right)^2$$

is either at least 0.05 or at most 0.01. The problem is to accept in the former case and reject in the latter case.

2.2 Complexity Classes

See the Complexity Zoo[8] for the definitions of standard complexity classes, such as BQP, AM, and PH. When we write $\mathcal{C}^{\mathsf{PH}}$ (i.e., a complexity class $\mathcal{C}$ with an oracle for the polynomial hierarchy), we mean $\cup_{k \geq 1} \mathcal{C}^{\Sigma_k^{\mathsf{P}}}$.

We will consider not only decision problems, but also *relation problems* (also called *function problems*). In a relation problem, the output is not a single bit but a $\mathrm{poly}(n)$-bit string y. There could be many valid y's for a given instance, and the algorithm's task is to output any one of them.

The definitions of FP and FNP (the relation versions of P and NP) are standard. We now define FBPP and FBQP, the relation versions of BPP and BQP.

Definition 3. FBPP *is the class of relations* $R \subseteq \{0,1\}^* \times \{0,1\}^*$ *for which there exists a probabilistic polynomial-time algorithm* A *that, given any input* $x \in \{0,1\}^n$*, produces an output* y *such that* $\Pr[(x,y) \in R] = 1 - o(1)$*, where the probability is over* A*'s internal randomness. (In particular, this implies that for every* x*, there exists at least one* y *such that* $(x,y) \in R$*.)* FBQP *is defined the same way, except that* A *is a quantum algorithm rather than a classical one.*

An important point about FBPP and FBQP is that, as far as we know, these classes do not admit amplification. In other words, the value of an algorithm's success probability might actually matter, not just the fact that the probability is bounded above 1/2. This is why we adopt the convention that an algorithm "succeeds" if it outputs $(x,y) \in R$ with probability $1 - o(1)$. In practice, we will give oracle problems for which the FBQP algorithm succeeds with probability $1 - 1/\exp(n)$, while any $\mathsf{FBPP}^{\mathsf{PH}}$ algorithm succeeds with probability at most (say) 0.99. How far the constant in this separation can be improved is an open problem.

Another important point is that, while $\mathsf{BPP}^{\mathsf{PH}} = \mathsf{P}^{\mathsf{PH}}$ (which follows from $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}}$), the class $\mathsf{FBPP}^{\mathsf{PH}}$ is strictly larger than $\mathsf{FP}^{\mathsf{PH}}$. To see this, consider the relation problem where we are given n, and asked to output any string of Kolmogorov complexity at least n. Clearly this problem is in FBPP: just output a random $2n$-bit string. On the other hand, just as obviously the problem is not in $\mathsf{FP}^{\mathsf{PH}}$. This is why we need to construct an oracle A such that $\mathsf{FBQP}^A \not\subset \mathsf{FBPP}^{\mathsf{PH}^A}$: because constructing an oracle A such that $\mathsf{FBQP}^A \not\subset \mathsf{FP}^{\mathsf{PH}^A}$ is trivial and not even related to quantum computing.

We now discuss some "low-level" complexity classes. AC^0 is the class of problems solvable by a nonuniform family of AND/OR/NOT circuits, with depth $O(1)$, size $\mathrm{poly}(n)$, and unbounded fanin. When we say "AC^0 circuit," we mean a constant-depth circuit of AND/OR/NOT gates, not necessarily of polynomial size. Any such circuit can be made into a *formula* (i.e., a circuit of fanout 1) with only a polynomial increase in size. The circuit has *depth* d if it consists of d alternating layers of AND and OR gates (without loss of generality, the NOT gates can all be pushed to the bottom, and we do not count them towards the depth). For example, a DNF (Disjunctive Normal Form) formula is just an AC^0 circuit of depth 2.

We will also be interested in quantum *logarithmic* time, which can be defined naturally as follows:

Definition 4. BQLOGTIME *is the class of languages* $L \subseteq \{0,1\}^*$ *that are decidable, with bounded error, by a* LOGTIME*-uniform family of quantum circuits* $\{C_n\}_n$ *such that each* C_n *has* $O(\log n)$ *gates, and can include gates that make random-access queries to the input string* $x = x_1 \ldots x_n$ *(i.e., that map* $|i\rangle |z\rangle$ *to* $|i\rangle |z \oplus x_i\rangle$ *for every* $i \in [n]$*).*

One other complexity class that arises in this paper, which is less well known than it should be, is $\mathsf{BPP}_{\mathsf{path}}$. Loosely speaking, $\mathsf{BPP}_{\mathsf{path}}$ can be defined as the class of problems that are solvable in probabilistic polynomial time, given the

[8] www.complexityzoo.com

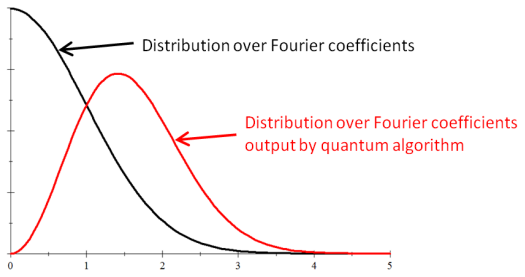

Figure 1: The Fourier coefficients of a random Boolean function follow a Gaussian distribution, with mean 0 and variance 1. However, larger Fourier coefficients are more likely to be observed by the quantum algorithm.

ability to "postselect" (that is, discard all runs of the computation that do not produce a desired result, even if such runs are the overwhelming majority).

3. QUANTUM ALGORITHMS

In this section, we show that FOURIER FISHING and FOURIER CHECKING both admit simple quantum algorithms.

3.1 Algorithm for Fourier Fishing

Here is a quantum algorithm, `FF-ALG`, that solves FOURIER FISHING with overwhelming probability in $O\left(n^2\right)$ time and n quantum queries (one to each f_i). For $i := 1$ to n, first prepare the state

$$\frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} f_i\left(x\right) |x\rangle,$$

then apply Hadamard gates to all n qubits, then measure in the computational basis and output the result as z_i.

Intuitively, `FF-ALG` samples the Fourier coefficients of each f_i under a distribution that is skewed towards larger coefficients; the algorithm's behavior is illustrated pictorially in Figure 1. Recall the definition of a "good" tuple $\langle f_1, \ldots, f_n \rangle$ from Section 2.1. Assuming $\langle f_1, \ldots, f_n \rangle$ is good, it is not hard to analyze `FF-ALG`'s success probability.

LEMMA 5. *Assuming $\langle f_1, \ldots, f_n \rangle$ is good, `FF-ALG` succeeds with probability $1 - 1/\exp\left(n\right)$.*

We also have the following:

LEMMA 6. *$\langle f_1, \ldots, f_n \rangle$ is good with probability $1 - 1/\exp\left(n\right)$, if the f_i's are chosen uniformly at random.*

Combining Lemmas 5 and 6, we find that `FF-ALG` succeeds with probability $1 - 1/\exp\left(n\right)$, where the probability is over both $\langle f_1, \ldots, f_n \rangle$ and `FF-ALG`'s internal randomness.

3.2 Algorithm for Fourier Checking

We now turn to FOURIER CHECKING, the problem of deciding whether two Boolean functions f, g are independent or forrelated. Here is a quantum algorithm, `FC-ALG`, that solves FOURIER CHECKING with constant error probability using $O\left(1\right)$ queries. First prepare a uniform superposition over all $x \in \{0,1\}^n$. Then query f in superposition, apply Hadamard gates to all n qubits, query g in superposition, and apply Hadamard gates to all n qubits again, to create the state

$$\frac{1}{N^{3/2}} \sum_{x,y,z \in \{0,1\}^n} f\left(x\right) \left(-1\right)^{x \cdot y} g\left(y\right) \left(-1\right)^{y \cdot z} |z\rangle.$$

Finally, measure in the computational basis, and "accept" if and only if the outcome $|0\rangle^{\otimes n}$ is observed. If needed, repeat the whole algorithm $O\left(1\right)$ times to boost the success probability.

It is clear that the probability of observing $|0\rangle^{\otimes n}$ (in a single run of `FC-ALG`) equals

$$p\left(f, g\right) := \frac{1}{N^3} \left(\sum_{x,y \in \{0,1\}^n} f\left(x\right) \left(-1\right)^{x \cdot y} g\left(y\right) \right)^2.$$

Recall that PROMISE FOURIER CHECKING was the problem of deciding whether $p\left(f, g\right) \geq 0.05$ or $p\left(f, g\right) \leq 0.01$, promised that one of these is the case. Thus, we immediately get a quantum algorithm to solve PROMISE FOURIER CHECKING, with constant error probability, using $O\left(1\right)$ queries to f and g.

For distributional FOURIER CHECKING, we also need the following theorem, which is proved in the full version.

THEOREM 7. *If $\langle f, g \rangle$ is drawn from the uniform distribution $\mathcal{U}$, then $\mathrm{E}_{\mathcal{U}}\left[p\left(f, g\right)\right] = \frac{1}{N}$. If $\langle f, g \rangle$ is drawn from the forrelated distribution $\mathcal{F}$, then $\mathrm{E}_{\mathcal{F}}\left[p\left(f, g\right)\right] > 0.07$.*

4. FOURIER FISHING LOWER BOUND

In Section 3.1, we gave a quantum algorithm for FOURIER FISHING that made only one query to each f_i. By contrast, it is not hard to show that any classical algorithm for FOURIER FISHING requires exponentially many queries to the f_i's—or in complexity terms, that FOURIER FISHING is not in FBPP. In this section, we give a much stronger result: that FOURIER FISHING is not even in $\mathsf{FBPP}^{\mathsf{PH}}$. This result does not rely on any unproved conjectures.

4.1 Constant-Depth Circuit Lower Bounds

Our starting point will be the following AC^0 lower bound, which can be found in the book of Håstad [26] for example.

THEOREM 8 ([26]). *Any depth-d circuit that accepts all n-bit strings of Hamming weight $n/2 + 1$, and rejects all strings of Hamming weight $n/2$, has size $\exp\left(\Omega\left(n^{1/(d-1)}\right)\right)$.*

We now give a corollary of Theorem 8, which (though simple) seems to be new, and might be of independent interest. Consider the following problem, which we call ε-BIAS DETECTION. We are given a string $y = y_1 \ldots y_m \in \{0,1\}^m$, and are promised that each bit y_i is 1 with independent probability p. The task is to decide whether $p = 1/2$ or $p = 1/2 + \varepsilon$.

COROLLARY 9. *Let $\mathcal{U}\left[\varepsilon\right]$ be the distribution over $\{0,1\}^m$ where each bit is 1 with independent probability $1/2 + \varepsilon$. Then any depth-d circuit C such that*

$$\left| \Pr_{x \sim \mathcal{U}[\varepsilon]} \left[C\left(x\right)\right] - \Pr_{x \sim \mathcal{U}[0]} \left[C\left(x\right)\right] \right| = \Omega\left(1\right)$$

has size $\exp\left(\Omega\left(1/\varepsilon^{1/(d+2)}\right)\right)$.

Corollary 9 is proved in the full version. The proof proceeds exactly as one would expect: we start with an AC^0 circuit C for ε-Bias Detection, and then use many copies of C, on different random subsets of the input bits, to get an AC^0 circuit that with high probability distinguishes all strings of Hamming weight $n/2+1$ from all strings of Hamming weight $n/2$.

4.2 Secretly Biased Fourier Coefficients

In this section, we state two lemmas indicating that one can *slightly* bias one of the Fourier coefficients of a random Boolean function $f:\{0,1\}^n \rightarrow \{-1,1\}$, and yet still have f be indistinguishable from a random Boolean function (so that, in particular, an adversary has no way of knowing which Fourier coefficient was biased). These lemmas will play a key role in our reduction from ε-Bias Detection to Fourier Fishing.

Fix a string $s \in \{0,1\}^n$. Let $\mathcal{A}[s]$ be the probability distribution over functions $f:\{0,1\}^n \rightarrow \{-1,1\}$ where each $f(x)$ is 1 with independent probability $\frac{1}{2}+(-1)^{s \cdot x}\frac{1}{2\sqrt{N}}$, and let $\mathcal{B}[s]$ be the distribution where each $f(x)$ is 1 with independent probability $\frac{1}{2}-(-1)^{s \cdot x}\frac{1}{2\sqrt{N}}$. Then let $\mathcal{D}[s] = \frac{1}{2}(\mathcal{A}[s]+\mathcal{B}[s])$ (that is, an equal mixture of $\mathcal{A}[s]$ and $\mathcal{B}[s]$).

Now consider the following game involving Alice and Bob, which closely models our problem. First, Alice chooses a string $s \in \{0,1\}^n$ uniformly at random. She then draws f according to $\mathcal{D}[s]$. She keeps s secret, but sends the truth table of f to Bob. After examining the entire truth table, Bob can output any string $z \in \{0,1\}^n$ such that $\left|\widehat{f}(z)\right| \geq \beta$. Bob's goal is to avoid outputting s. The following lemma says that, regardless of Bob's strategy, if β is sufficiently greater than 1, then Bob *must* output s with probability significantly greater than $1/N$.

Lemma 10. *In the above game, we have*

$$\Pr[s=z] \geq \frac{e^{\beta}+e^{-\beta}}{2\sqrt{e}N}.$$

where the probability is over s, f, and Bob's random choices.

Lemma 10 is proved in the full version. The intuition behind it is simple: f looks to Bob almost exactly like a random Boolean function. Let V_β be the set of all $s \in \{0,1\}^n$ such that $\left|\widehat{f}(s)\right| \geq \beta$. Then when Alice biases f, of course she significantly increases the probability that $s \in V_\beta$. But the biased Fourier coefficient $\widehat{f}(s)$ still gets "lost in the noise"—that is, it is information-theoretically impossible for Bob to tell s apart from the exponentially many *other* strings z that are in V_β just by chance. Therefore, the best Bob can do is basically just to output a random element of V_β. But since $|V_\beta| \ll N$, this strategy will cause Bob to output s with probability $\gg 1/N$.

Formalizing the above intuition involves a surprisingly simple Bayesian calculation. Basically, all we need to do is fix f together with Bob's output z (which we know belongs to V_β), and then calculate how likely f was to be drawn from $\mathcal{D}[z]$ rather than $\mathcal{D}[z']$ for some $z' \neq z$.

To see the connection to our problem, think of Bob as a putative AC^0 circuit for Fourier Fishing, and Alice as an AC^0 reduction that uses Bob to solve ε-Bias Detection. Bob might *try* not to cooperate, but by choosing s randomly, Alice can force even an adversarial Bob to give her useful information about the magnitude of $\widehat{f}(s)$.

Now let $\mathcal{D} = \mathrm{E}_s[\mathcal{D}[s]]$ (that is, an equal mixture of all the $\mathcal{D}[s]$'s). Then we prove in the full version that $\mathcal{D}$ is extremely close in variation distance to $\mathcal{U}$, the uniform distribution over all Boolean functions $f:\{0,1\}^n \rightarrow \{-1,1\}$.

Lemma 11. $\|\mathcal{D}-\mathcal{U}\| = O(1/\sqrt{N})$.

An immediate corollary of Lemma 11 is that, if a Fourier Fishing algorithm succeeds with probability p on $\langle f_1,\ldots,f_n\rangle$ drawn from $\mathcal{U}^n$, then it also succeeds with probability at least $p-\|\mathcal{D}^n-\mathcal{U}^n\| \geq p-O(n/\sqrt{N})$ on $\langle f_1,\ldots,f_n\rangle$ drawn from $\mathcal{D}^n$.

4.3 Putting It All Together

Using the results of Sections 4.1 and 4.2, we can give a lower bound on the constant-depth circuit complexity of Fourier Fishing.

Theorem 12. *Any depth-d circuit that solves the* Fourier Fishing *problem, with probability at least* 0.99 *over $f_1,\ldots,f_n$ chosen uniformly at random, has size* $\exp\left(\Omega\left(N^{1/(2d+8)}\right)\right)$.

Combining Theorem 12 with standard diagonalization tricks, we can also give an oracle separation (in fact, a *random* oracle separation) between the complexity classes FBQP and $\mathsf{FBPP}^{\mathsf{PH}}$.

Theorem 13. $\mathsf{FBQP}^A \not\subset \mathsf{FBPP}^{\mathsf{PH}^A}$ *with probability* 1 *for a random oracle A.*

If we "scale down by an exponential," then we can eliminate the need for the oracle A, and get a relation problem that is solvable in quantum *logarithmic* time but not in AC^0.

Theorem 14. *There is a relation in* $\mathsf{BQLOGTIME} \setminus \mathsf{AC}^0$.

Theorems 12, 13, and 14 are proved in the full version.

5. FOURIER CHECKING LOWER BOUND

Section 4 settled the relativized BQP versus PH question, if we are willing to talk about relation problems. Ultimately, though, we also care about decision problems. So in this section we consider the Fourier Checking problem, of deciding whether two Boolean functions f,g are independent or forrelated. In Section 3.2, we saw that Fourier Checking has quantum query complexity $O(1)$. What is its classical query complexity?[9]

It is not hard to give a classical algorithm that solves Fourier Checking using $O(\sqrt{N}) = O(2^{n/2})$ queries. In the next section, we will show that Fourier Checking has a property called *almost k-wise independence*, which immediately implies a lower bound of $\Omega(\sqrt[4]{N}) = \Omega(2^{n/4})$ on its randomized query complexity[10] (as well as exponential lower bounds on its MA, $\mathsf{BPP}_{\mathsf{path}}$, and SZK query complexities). Indeed, we conjecture that almost k-wise independence is enough to imply that Fourier Checking is not in PH. We discuss the status of that conjecture in Section 6.

[9]So long as we consider the distributional version of Fourier Checking, the deterministic and randomized query complexities are the same (by Yao's principle).

[10]With some more work, we believe that the randomized query complexity lower bound can be improved to $\Omega(\sqrt{N})$, though we have not pursued that here.

5.1 Almost k-Wise Independence

Let $X = x_1 \dots x_N \in \{0,1\}^N$ be a string. Then a *literal* is an expression of the form x_i or $1 - x_i$, and a *k-term* is a product of k literals (each involving a different x_i), which is 1 if the literals all take on prescribed values and 0 otherwise. We can give an analogous definition for strings over the alphabet $\{-1, 1\}$; we simply let the literals have the form $\frac{1 \pm x_i}{2}$.

Let $\mathcal{U}$ be the uniform distribution over N-bit strings. The following definition will play a major role in this work.

DEFINITION 15. *A distribution $\mathcal{D}$ over N-bit strings is ε-almost k-wise independent if for every k-term C,*

$$1 - \varepsilon \leq \frac{\Pr_{X \sim \mathcal{D}}[C(X)]}{\Pr_{X \sim \mathcal{U}}[C(X)]} \leq 1 + \varepsilon.$$

(Note that $\Pr_{X \sim \mathcal{U}}[C(X)]$ is just 2^{-k}.)

In other words, there should be no assignment to any k input bits, such that conditioning on that assignment gives us much information about whether X was drawn from $\mathcal{D}$ or $\mathcal{U}$.

Now let $\mathcal{F}$ be the forrelated distribution over pairs of Boolean functions $f, g : \{0,1\}^n \rightarrow \{-1,1\}$, where we first choose a vector $v = (v_x)_{x \in \{0,1\}^n} \in \mathbb{R}^N$ of independent $\mathcal{N}(0,1)$ Gaussians, then set $f(x) := \operatorname{sgn}(v_x)$ and $g(x) := \operatorname{sgn}(\widehat{v}_x)$ for all $x \in \{0,1\}^n$.

THEOREM 16. *$\mathcal{F}$ is $O(k^2/\sqrt{N})$-almost k-wise independent for all $k \leq \sqrt[4]{N}$.*

We prove Theorem 16 in the full version. To do so, we first compute the measure of the Gaussian distribution on an affine subspace of $\mathbb{R}^{2N}$ defined by $K + L$ equations of the form $v_{x_i} = a_i$ and $\widehat{v}_{y_j} = b_j$. This computation is somewhat involved, but is enormously aided by rotational invariance and other nice properties of the Gaussian distribution. We then perform a discretization step, to deal with the fact that the functions f and g are Boolean.

5.2 Oracle Separation Results

The following lemma says that *any* almost k-wise independent distribution is indistinguishable from the uniform distribution by $\mathsf{BPP}_{\mathsf{path}}$ or SZK machines.

LEMMA 17. *Suppose a probability distribution $\mathcal{D}$ over oracle strings is $1/t(n)$-almost $\operatorname{poly}(n)$-wise independent, for some superpolynomial function t. Then no $\mathsf{BPP}_{\mathsf{path}}$ machine or SZK protocol can distinguish $\mathcal{D}$ from the uniform distribution $\mathcal{U}$ with non-negligible bias.*

We can combine Theorem 16 and Lemma 17 with standard diagonalization tricks, to obtain an oracle relative to which $\mathsf{BQP} \not\subset \mathsf{BPP}_{\mathsf{path}}$ and $\mathsf{BQP} \not\subset \mathsf{SZK}$.

THEOREM 18. *There exists an oracle A relative to which $\mathsf{BQP}^A \not\subset \mathsf{BPP}_{\mathsf{path}}^A$ and $\mathsf{BQP}^A \not\subset \mathsf{SZK}^A$.*

Since $\mathsf{BPP} \subseteq \mathsf{MA} \subseteq \mathsf{BPP}_{\mathsf{path}}$, Theorem 18 supersedes the previous results that there exist oracles A relative to which $\mathsf{BPP}^A \neq \mathsf{BQP}^A$ [9] and $\mathsf{BQP}^A \not\subset \mathsf{MA}^A$ [27].

Lemma 17 and Theorem 18 are proved in the full version.

6. THE GENERALIZED LN CONJECTURE

In 1990, Linial and Nisan [18] conjectured that "polylogarithmic independence fools AC^0"—or loosely speaking, that every probability distribution $\mathcal{D}$ over n-bit strings that is uniform on all small subsets of bits, is *indistinguishable* from the uniform distribution by polynomial-size, constant-depth circuits. We now state a variant of the Linial-Nisan Conjecture, not with the best parameters but with weaker, easier-to-understand parameters that suffice for our application.

CONJECTURE 19 (LINIAL-NISAN OR LN CONJECTURE). *Let $\mathcal{D}$ be an $n^{\Omega(1)}$-wise independent distribution over $\{0,1\}^n$, and let $f : \{0,1\}^n \rightarrow \{0,1\}$ be computed by an AC^0 circuit of size $2^{n^{o(1)}}$ and depth $O(1)$. Then*

$$\left| \Pr_{x \sim \mathcal{D}}[f(x)] - \Pr_{x \sim \mathcal{U}}[f(x)] \right| = o(1).$$

After seventeen years of almost no progress, in 2007 Bazzi [5] finally proved the LN Conjecture for the special case of depth-2 circuits (also called DNF formulas). Bazzi's proof was about 50 pages, but it was dramatically simplified a year later, when Razborov [20] discovered a 3-page proof. Then in 2009, Braverman [11] gave a breakthrough proof of the full LN Conjecture.

THEOREM 20 (BRAVERMAN'S THEOREM [11]). *Let $f : \{0,1\}^n \rightarrow \{0,1\}$ be computed by an AC^0 circuit of size S and depth d, and let $\mathcal{D}$ be a $\left(\log \frac{S}{\varepsilon}\right)^{7d^2}$-wise independent distribution over $\{0,1\}^n$. Then for all sufficiently large S,*

$$\left| \Pr_{x \sim \mathcal{D}}[f(x)] - \Pr_{x \sim \mathcal{U}}[f(x)] \right| \leq \varepsilon.$$

We conjecture a modest-seeming extension of Braverman's Theorem, which says (informally) that *almost* k-wise independent distributions fool AC^0 as well.

CONJECTURE 21 (GLN CONJECTURE). *Let $\mathcal{D}$ be a $1/n^{\Omega(1)}$-almost $n^{\Omega(1)}$-wise independent distribution over $\{0,1\}^n$, and let $f : \{0,1\}^n \rightarrow \{0,1\}$ be computed by an AC^0 circuit of size $2^{n^{o(1)}}$ and depth $O(1)$. Then*

$$\left| \Pr_{x \sim \mathcal{D}}[f(x)] - \Pr_{x \sim \mathcal{U}}[f(x)] \right| = o(1).$$

By the usual correspondence between AC^0 and PH, the GLN Conjecture immediately implies the following counterpart of Lemma 17:

> *Suppose a probability distribution $\mathcal{D}$ over oracle strings is $1/t(n)$-almost $\operatorname{poly}(n)$-wise independent, for some superpolynomial function t. Then no PH machine can distinguish $\mathcal{D}$ from the uniform distribution $\mathcal{U}$ with non-negligible bias.*

And thus we get the following implication:

THEOREM 22. *Assuming the GLN Conjecture, there exists an oracle A relative to which $\mathsf{BQP}^A \not\subset \mathsf{PH}^A$.*

The proof is the same as that of Theorem 18; the only difference is that the GLN Conjecture now plays the role of Lemma 17. Likewise:

THEOREM 23. *Assuming the GLN Conjecture for the special case of depth-2 circuits (i.e., DNF formulas), there exists an oracle A relative to which $\mathsf{BQP}^A \not\subset \mathsf{AM}^A$.*

As a side note, it is conceivable that one could prove $\Pr_{x\sim\mathcal{D}}[\varphi(x)] - \Pr_{x\sim\mathcal{U}}[\varphi(x)] = o(1)$ for every almost k-wise independent distribution $\mathcal{D}$ and small *CNF* formula φ, without getting the same result for DNF formulas (or vice versa). However, since BQP is closed under complement, even such an asymmetric result would imply an oracle A relative to which $\mathsf{BQP}^A \not\subset \mathsf{AM}^A$.

If the GLN Conjecture holds, then we can also "scale down by an exponential," to obtain an *unrelativized* decision problem that is solvable in BQLOGTIME but not in AC^0.

THEOREM 24. *Assuming the GLN Conjecture, there exists a promise problem in* $\mathsf{BQLOGTIME} \setminus \mathsf{AC}^0$.

6.1 Low-Fat Polynomials

Given that the GLN Conjecture would have such remarkable implications for quantum complexity theory, the question arises of how we can go about proving it. As we are indebted to Louay Bazzi for pointing out to us, the GLN Conjecture is *equivalent* to the following conjecture, about approximating AC^0 functions by low-degree polynomials.

CONJECTURE 25 (LOW-FAT SANDWICH CONJECTURE). *For every function $f : \{0,1\}^n \rightarrow \{0,1\}$ computable by an AC^0 circuit, there exist polynomials $p_\ell, p_u : \mathbb{R}^n \rightarrow \mathbb{R}$ of degree $k = n^{o(1)}$ that satisfy the following three conditions.*

(i) ***Sandwiching:*** *$p_\ell(x) \leq f(x) \leq p_u(x)$ for all x.*

(ii) ***L_1-Approximation:*** *$\mathrm{E}_{x\sim\mathcal{U}}[p_u(x) - p_\ell(x)] = o(1)$.*

(iii) ***Low-Fat:*** *$p_\ell(x)$ and $p_u(x)$ can be written as linear combinations of terms, $p_\ell(x) = \sum_C \alpha_C C(x)$ and $p_u(x) = \sum_C \beta_C C(x)$ respectively, such that $\sum_C |\alpha_C| \, 2^{-|C|} = n^{o(1)}$ and $\sum_C |\beta_C| \, 2^{-|C|} = n^{o(1)}$. (Here a term is a product of literals of the form x_i and $1 - x_i$.)*

If we take out condition (iii), then Conjecture 25 becomes equivalent to the *original* LN Conjecture (see Bazzi [5] for a proof). And indeed, all progress so far on "Linial-Nisan problems" has crucially relied on this connection with polynomials. Bazzi [5] and Razborov [20] proved the depth-2 case of the LN Conjecture by constructing low-degree, approximating, sandwiching polynomials for every DNF, while Braverman [11] proved the full LN Conjecture by constructing such polynomials for every AC^0 circuit.[11] Given this history, proving Conjecture 25 would seem like the "obvious" approach to proving the GLN Conjecture.

It is easy to prove one direction of the equivalence: that to prove the GLN Conjecture, it suffices to construct low-fat sandwiching polynomials for every AC^0 circuit. The other direction—that the GLN Conjecture implies Conjecture 25, and hence, there is no loss of generality in working with polynomials instead of probability distributions—follows from a linear programming duality calculation.

7. DISCUSSION

We now take a step back, and use our results to address some conceptual questions about the relativized BQP versus PH question, the GLN Conjecture, and what makes them so difficult.

The first question is an obvious one. Complexity theorists have known for decades how to prove constant-depth circuit lower bounds, and how to use those lower bounds to give oracles A relative to which (for example) $\mathsf{PP}^A \not\subset \mathsf{PH}^A$ and $\oplus\mathsf{P}^A \not\subset \mathsf{PH}^A$. So why should it be so much harder to give an A relative to which $\mathsf{BQP}^A \not\subset \mathsf{PH}^A$? What makes this AC^0 lower bound different from all other AC^0 lower bounds?

The answer seems to be that, while we have powerful techniques for proving that a function f is not in AC^0, *all of those techniques, in one way or another, involve arguing that f is not approximated by a low-degree polynomial.* The Razborov-Smolensky technique [19, 25] argues this explicitly, while even the random restriction technique [14, 28, 26] argues it "implicitly," as shown by Linial, Mansour, and Nisan [17]. And this is a problem, if f is also computed by an efficient quantum algorithm. For Beals et al. [6] proved the following in 1998:

LEMMA 26 ([6]). *Suppose a quantum algorithm Q makes T queries to a Boolean input $X \in \{0,1\}^N$. Then Q's acceptance probability is a real multilinear polynomial $p(X)$, of degree at most $2T$.*

In other words, if a function f is in BQP, then for that very reason, f *has* a low-degree approximating polynomial! As an example, we already saw that the following polynomial p, of degree 4, successfully distinguishes the forrelated distribution $\mathcal{F}$ from the uniform distribution $\mathcal{U}$:

$$p(f,g) := \frac{1}{N^3}\left(\sum_{x,y\in\{0,1\}^n} f(x)(-1)^{x\cdot y} g(y)\right)^2. \qquad (1)$$

Therefore, we cannot hope to prove a lower bound for FOURIER CHECKING, by any argument that would also imply that such a p cannot exist.

This brings us to a second question. If

(i) every known technique for proving $f \notin \mathsf{AC}^0$ involves showing that f is not approximated by a low-degree polynomial, but

(ii) every function f with low quantum query complexity *is* approximated by a low-degree polynomial,

does that mean there is no hope of solving the relativized BQP versus PH problem using polynomial-based techniques?

We believe the answer is no. The essential point here is that an AC^0 function can be approximated by different *kinds* of low-degree polynomials. Furthermore, to show that $f \notin \mathsf{AC}^0$, it suffices to show that f is not approximated by a low-degree polynomial in *any one* of these senses. For example, even though the PARITY function has degree 1 over the finite field $\mathbb{F}_2$, Razborov and Smolensky showed that over other fields (such as $\mathbb{F}_3$), any degree-$o(\sqrt{n})$ polynomial disagrees with PARITY on a large fraction of inputs—and that is enough to imply that PARITY$\notin \mathsf{AC}^0$. In other words, we simply need to find a *type* of polynomial approximation that works for AC^0 circuits, but does not work for FOURIER CHECKING. If true, Conjecture 25 (the Low-Fat Sandwich Conjecture) provides exactly such a type of approximation.

But this raises another question: what is the significance of the "low-fat" requirement in Conjecture 25? Why, of

[11] Strictly speaking, Braverman constructed approximating polynomials with slightly different (though still sufficient) properties. We know from Bazzi [5] that it must be possible to get sandwiching polynomials as well.

all things, do we want our approximating polynomial p to be expressible as a linear combination of terms, $p(x) = \sum_C \alpha_C C(x)$, such that $\sum_C |\alpha_C| 2^{-|C|} = n^{o(1)}$?

The answer takes us to the heart of what an oracle separation between BQP and PH would have to accomplish. Notice that, although the polynomial p from equation (1) solved the FOURIER CHECKING problem, it did so only by *cancelling massive numbers of positive and negative terms,* then representing the answer by the tiny residue left over. Not coincidentally, this sort of cancellation is a central feature of quantum algorithms. By contrast, we show that, if a polynomial p does *not* involve such massive cancellations, but is instead more "conservative" and "reasonable" (like the polynomials that arise from classical decision trees), then p cannot distinguish almost k-wise independent distributions from the uniform distribution, and therefore cannot solve FOURIER CHECKING. If Conjecture 25 holds, then every small-depth circuit can be approximated, not just by any low-degree polynomial, but by a "reasonable" low-degree polynomial—one with a bound on the coefficients that prevents massive cancellations. This would prove that FOURIER CHECKING has no small constant-depth circuits, and hence that there exists an oracle separating BQP from PH.

8. OPEN PROBLEMS

First, of course, prove the GLN Conjecture, or prove the existence of an oracle A relative to which $\mathsf{BQP}^A \not\subset \mathsf{PH}^A$ by some other means. A natural first step would be to prove the GLN Conjecture for the special case of DNFs: as shown in Theorem 23, this would imply an oracle A relative to which $\mathsf{BQP}^A \not\subset \mathsf{AM}^A$. We have offered a $200 prize for the PH case and a $100 prize for the AM case.[12]

Second, it would be of interest to prove the GLN Conjecture for classes of functions weaker than (or incomparable with) DNFs: for example, monotone DNFs, read-once formulas, and read-k-times formulas.

Third, can we give an example of a Boolean function $f : \{0,1\}^n \to \{-1,1\}$ that is well-approximated by a low-degree polynomial, but *not* by a low-degree low-fat polynomial?

Fourth, can we give an oracle relative to which $\mathsf{BQP} \not\subset \mathsf{IP}$?

Fifth, what else does the GLN Conjecture imply?

Sixth, how much can we say about the BQP versus PH question in the unrelativized world? As one concrete challenge, can we find a nontrivial way to "realize" the FOURIER CHECKING oracle (in other words, an explicit computational problem that is solvable using FOURIER CHECKING)?

Seventh, how far can the gap between the success probabilities of FBQP and $\mathsf{FBPP}^{\mathsf{PH}}$ algorithms be improved?

9. ACKNOWLEDGMENTS

I thank Louay Bazzi for reformulating the GLN Conjecture as the Low-Fat Sandwich Conjecture, Andy Drucker, Lance Fortnow, and Sasha Razborov for helpful discussions, and an anonymous reviewer for comments.

10. REFERENCES

[1] S. Aaronson and A. Ambainis. The need for structure in quantum speedups. arXiv:0911.0996, 2009.

[2] L. Adleman, J. DeMarrais, and M.-D. Huang. Quantum computability. *SIAM J. Comput.*, 26(5):1524–1540, 1997.

[3] D. Aharonov, V. Jones, and Z. Landau. A polynomial quantum algorithm for approximating the Jones polynomial. In *Proc. ACM STOC*, p. 427–436, 2006.

[4] Z. Bar-Yossef, T. S. Jayram, and I. Kerenidis. Exponential separation of quantum and classical one-way communication complexity. *SIAM J. Comput.*, 38(1):366–384, 2008.

[5] L. Bazzi. Polylogarithmic independence can fool DNF formulas. In *Proc. IEEE FOCS*, p. 63–73, 2007.

[6] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001.

[7] J. N. de Beaudrap, R. Cleve, and J. Watrous. Sharp quantum versus classical query complexity separations. *Algorithmica*, 34(4):449–461, 2002.

[8] C. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997.

[9] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM J. Comput.*, 26(5):1411–1473, 1997.

[10] R. B. Boppana, J. Håstad, and S. Zachos. Does co-NP have short interactive proofs? *Inf. Proc. Lett.*, 25:127–132, 1987.

[11] M. Braverman. Poly-logarithmic independence fools AC^0 circuits. In *Proc. IEEE Complexity*, p. 3–8, 2009.

[12] H. Buhrman, L. Fortnow, I. Newman, and H. Röhrig. Quantum property testing. *SIAM J. Comput.*, 37(5):1387–1400, 2008.

[13] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *Commun. ACM*, 52(2):89–97, 2009.

[14] M. Furst, J. B. Saxe, and M. Sipser. Parity, circuits, and the polynomial time hierarchy. *Math. Systems Theory*, 17:13–27, 1984.

[15] D. Gavinsky, J. Kempe, I. Kerenidis, R. Raz, and R. de Wolf. Exponential separation for one-way quantum communication complexity, with applications to cryptography. *SIAM J. Comput.*, 38(5):1695–1708, 2008.

[16] A. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31:1501–1526, 2002.

[17] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993.

[18] N. Linial and N. Nisan. Approximate inclusion-exclusion. *Combinatorica*, 10(4):349–365, 1990.

[19] A. A. Razborov. Lower bounds for the size of circuits of bounded depth with basis $\{\&, \oplus\}$. *Mathematicheskie Zametki*, 41(4):598–607, 1987.

[20] A. A. Razborov. A simple proof of Bazzi's theorem. *ACM Trans. Comput. Theory*, 1(1), 2009.

[21] A. Shamir. IP=PSPACE. *J. ACM*, 39(4):869–877, 1992.

[22] D. Shepherd and M. J. Bremner. Temporally unstructured quantum computation. *Proc. Roy. Soc. London*, A465(2105):1413–1439, 2009.

[23] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[24] D. Simon. On the power of quantum computation. In *Proc. IEEE FOCS*, p. 116–123, 1994.

[25] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proc. ACM STOC*, p. 77–82, 1987.

[26] J. Håstad. *Computational Limitations for Small Depth Circuits*. MIT Press, 1987.

[27] J. Watrous. Succinct quantum proofs for properties of finite groups. In *Proc. IEEE FOCS*, p. 537–546, 2000.

[28] A. C-C. Yao. Separating the polynomial-time hierarchy by oracles. In *Proc. IEEE FOCS*, p. 1–10, 1985.

[12] See http://scottaaronson.com/blog/?p=381