On Statistical Query Sampling and NMR Quantum Computing

Avrim Blum* Ke Yang*

Abstract

We introduce a "Statistical Query Sampling" model, in which the goal of an algorithm is to produce an element in a hidden set $S \subseteq \{0,1\}^n$ with reasonable probability. The algorithm gains information about S through oracle calls (statistical queries), where the algorithm submits a query function $g(\cdot)$ and receives an approximation to $\Pr_{x \in S}[g(x) = 1]$. We show how this model is related to NMR quantum computing, in which only statistical properties of an ensemble of quantum systems can be measured, and in particular to the question of whether one can translate standard quantum algorithms to the NMR setting without putting all of their classical post-processing into the quantum system. Using Fourier analysis techniques developed in the related context of statistical query learning, *we prove a number of lower bounds (both information-theoretic and cryptographic) on the ability of algorithms to produces an $x \in S$, even when the set S is fairly simple. These lower bounds point out a difficulty in efficiently applying NMR quantum computing to algorithms such as Shor's and Simon's algorithm that involve significant classical post-processing. We also explicitly relate the notion of statistical query sampling to that of statistical query learning.*

1 Introduction

Recent years have witnessed the development of a number of exciting quantum algorithms: Simon's algorithm for the hidden XOR secret problem [28], Shor's algorithm for factoring and discrete logarithms [26, 27], Boneh and Lipton's algorithm for the hidden subgroup problem [4], and many generalizations and extensions [21, 11, 12, 18, 15, 17]. At the same time, work has been ongoing on various proposals for physically realizing quantum computers. Currently, one of the most promising such proposals is based on Nuclear Magnetic Resonance (NMR) [10, 7, 13, 5]. The NMR approach works by manipulating a large ensemble of quantum systems in solution. One property of the NMR method, which is the focus of this paper, is that unlike the "standard" quantum computing model, one cannot directly measure any individual quantum system in the ensemble. Instead, a measurement is limited to a single qubit, and when a measurement takes place, the device returns (an approximation to) the expected value of this measurement, over the quantum systems in the ensemble. For this reason, the model for NMR is sometimes called the "expected-value" (EV) model [6]. In contrast, the measurement in the standard quantum model yields a random sample state (which may consists of multiple bits) according to a classical probability distribution.

Given the distinction between the standard model and the EV model, the first question that arises is whether it is possible to translate algorithms working in the standard model to work in the EV model. In fact, the answer is yes. Consider any BQP algorithm [24]. Recall from the definition that a BQP algorithm solves a decision problem, and such an algorithm has a special "target" qubit to indicate acceptance. For a language L and an input x, if $x \in L$, then the measurement of the target qubit will produce a "1" with probability at least $3/4$; if $x \notin L$, the probability is at most $1/4$ when measured. Such an algorithm works naturally in the EV model, since one can simply measure the target qubit, and even with significant measurement error, use the rule that if the observed value $v \geq 1/2$, then $x \in L$, and otherwise $x \notin L$. For a search (as opposed to decision) problem, we can perform the usual reduction to a series of decision problems, solving each one by one. In fact, many researchers have used this approach [13, 24], which we call an "all-inclusive" translation.

Unfortunately, the "all-inclusive" translation can greatly increase the amount of work that must be done by the quantum system. Consider Shor's algorithm, for instance (see Appendix A). Shor's algorithm (and others like it) consists of a quantum sampling circuit Q, whose output is measured and fed into a classical extraction circuit C. For the all-inclusive translation, the classical extraction circuit C needs to be "quantumized", i.e., realized by a quantum circuit and appended to the quantum sampling circuit Q. This can cause a significant increase in the size of the quantum circuit

*Computer Science Department, Carnegie Mellon University, 5000 Forbes Ave. Pittsburgh, PA 15213. E-mail: {avrim,yangke}@cs.cmu.edu. This work is supported in part by NSF grants CCR-0105488 and NSF-ITR 0122581.

— in the case of Shor's algorithm, the entire circuitry for computing continued fractions needs to be realized in quantum — which is a rather undesirable consequence. Even in the most optimistic scenarios, quantum computers will be orders of magnitude more difficult to manufacture and maintain than classical computers, and thus we would like to put as little of the complexity as possible in the quantum system. Even more serious problems emerge when more than one sample is needed by the classical extraction circuit. For example, in Simon's algorithm, $\Omega(n)$ samples are needed for Gaussian elimination (see Appendix A). Now the all-inclusive translation needs to manufacture multiple copies of the quantum sampling circuit and then connect them together with the "quantumized" classical extraction circuit. This can cause even more blowup in the size of the quantum circuit in the EV model.

In this paper, we consider the question of whether there might be more efficient translations that apply generally to algorithms consisting of a quantum sampling circuit Q followed by a classical extraction circuit C, that work *without* having to put the classical part of the algorithm into the quantum system. Our main contributions are results that answer this question in the negative, for several natural notions of "general". We achieve these results through a connection to the notion of *statistical query learning* [22] studied in Computational Learning Theory, and in particular to a related notion that we introduce of *statistical query sampling*. Using techniques from Fourier analysis and cryptography, we show that even in cases where the distribution implied by Q is quite simple, it can be hard to use the EV model to generate a sample that can be used by C. Note that our results do not preclude the possibility of approaches tailored to specific quantum algorithms. For example, Collins [6] demonstrates a modification to Grover's algorithm that is more efficient than the all-inclusive translation (see also the discussion below). However, as pointed out by the author, his approach does not generalize to algorithms like Shor's.

1.1 Our model and results

We view the quantum sampling circuit Q as representing a hidden set $S \subseteq \{0,1\}^n$, and we view the classical post-processing as a circuit C such that $C(x) = 1$ for all $x \in S$. The goal of the translation procedure is to produce some $x \in S$. To find such an x, the algorithm has the ability to perform a "statistical query" of Q by proposing a query function (a predicate) $g : \{0,1\}^n \mapsto \{0,1\}$ and asking for $\mathbf{E}_{x\in S}[g(x)]$ up to some $1/poly$ accuracy. For example, measuring the ith qubit corresponds to the query $g(x) = x_i$. Taking the XOR of the first three qubits and then measuring the result corresponds to the query $g(x) = x_1 \oplus x_2 \oplus x_3$. The algorithm may repeat this process multiple (polynomially-many) times, with different query functions g, and in the end must (with noticeable probability) produce an $x \in S$.

Note that this task is easy to do if S is very large ($|S| \geq 2^n/poly(n)$), since a random $x \in \{0,1\}^n$ will do. It is also easy to do if S is very small ($|S| = poly(n)$). In particular, if $|S| = poly(n)$, then by asking for an accuracy of $1/(2|S|)$ one can distinguish the case that $\mathbf{E}_{x\in S}[g(x)] = 0$ from the case that $\mathbf{E}_{x\in S}[g(x)] > 0$. This allows one to walk down the bits of x, fixing bits from left to right, until a specific $x \in S$ is produced. This is the key idea of [6].

We show, however, that this task is hard in general. Specifically, we give two types of hardness results. First, we give an information-theoretic hardness result if the query algorithm is not allowed to access C. That is, the translator is allowed to use the fact that the classical extraction circuit C is polynomial in size (so the set of accepting strings cannot be totally arbitrary) but it is not allowed to examine C — it can only gain information via the queries g. Second, we give a cryptographic hardness result if we assume the translator is given C as input, but that otherwise C is an arbitrary polynomial-size circuit. We still do not know if efficient translation is possible for the specific circuit C used in Shor's algorithm.

We also consider a more general setting, in which S may be large (e.g., $|S| = 2^{n-1}$), so a random string has reasonable chance of belonging to S, but the goal of the translation is to produce a string $x \in S$ with probability substantially greater than random guessing. We call this more general setting "strong SQ-sampling", and refer to the former setting as the "weak SQ-sampling". Strong SQ-sampling models situations such as Simon's algorithm, in which the quantum circuit produces a random $y \in \{0,1\}^n$ such that $y \cdot s = 0$ for the hidden secret s. In this case, a random string has probability $1/2$ of belonging to S, but we need $\Omega(n)$ correct samples in a row in order to perform Gaussian elimination. We give an information-theoretic hardness result for this problem, that holds for the specific set S used by Simon's algorithm (Theorem 2).[1]

1.2 Techniques and relation to Statistical Query learning

Our results are based on a connection to the Statistical Query (SQ) learning model, first introduced by Kearns [22] as a restricted version of the popular Probably Approximately Correct (PAC) model of Valiant [30]. In these learning models, the goal of an algorithm is to learn an approximation to a hidden function $f : \{0,1\}^n \mapsto \{0,1\}$. In the PAC model, the algorithm has access to an "example oracle", which produces a random labeled sample $\langle x, f(x)\rangle$

[1] Note, for Simon's algorithm, we no longer want to think of there existing a known classical extraction circuit. If we were given access to a circuit C such that $C(x) = 1$ iff $x \in S$ (e.g., the circuit with the hidden secret built in) then the sampling goal would be easy. See Theorem 4 for further discussion.

upon invocation. In the SQ model, however, the algorithm does not see explicit examples or their labels. Instead, the algorithm queries an "SQ-oracle" with predicates $g(x, y)$, and receives an approximation to $\Pr_x[g(x, f(x)) = 1]$. For instance, the algorithm might ask for the probability that a random example would both be positive and have its first bit set to 1 ($g(x, y) = x_1 \wedge y$).[2] The SQ model has proven to be very useful because (a) it is inherently tolerant to classification noise (this is the reason the model was developed), and (b) nearly all machine learning algorithms can be phrased as SQ algorithms. What makes the SQ model especially interesting is that one can information-theoretically prove lower bounds on the ability of SQ algorithms to learn certain classes of functions [22, 3, 20, 31, 32].

The relationship between the standard model and the EV model for quantum computation is quite similar to that between the PAC model and the SQ model in machine learning, which motivates our definition of the Statistical Query Sampling problem. In particular, the SQ sampling problem can be viewed as the SQ learning problem with two key differences: first, the goal is not to learn an approximation to f but is rather to produce a positive example, and second, the oracle for SQ sampling returns approximations to $\Pr[g(x) = 1 \mid f(x) = 1]$ rather than to $\Pr[g(x, f(x)) = 1]$ (a difference that matters when the set of positive examples is quite small).

We use techniques from Fourier analysis to prove the following lower bounds. First (Theorem 1) we show there exist simple function classes such that no algorithm, using only a polynomial number of queries of $1/poly$ accuracy, can produce a positive instance with even $1/poly$ probability. Second (Theorem 2), for the class of "negative parity" functions arising in Simon's algorithm, no algorithm using only a polynomial number of queries of $1/poly$ accuracy, can produce a nontrivial positive instance with probability more than $1/2 + 1/poly$. (Note that random guessing works with probability $1/2$). We also show that unlike the case of SQ learning, the SQ sampling problem can be computationally hard even if f is explicitly given to the algorithm, based on cryptographic assumptions (see Theorem 3).

Finally, we explicitly relate the notion of SQ sampling to that of SQ learning by proving that if a function class is "dense", meaning that a random element has non-negligible probability of being positive, then strong SQ-learnability implies strong SQ-samplability (Theorem 4). We also point out that there exists function classes that are perfectly SQ-samplable, yet not even weakly SQ-learnable.

[2] In both PAC and SQ learning models, the distribution over x need not be the uniform distribution (or even known to the learning algorithm). However, much work on SQ learning does focus on the uniform distribution, and that is the setting we are most interested in in this paper.

2 Preliminaries and Definitions

We are interested in *predicates* that map elements from a *domain* X (e.g., $\{0, 1\}^n$) to $\{0, 1\}$. For a predicate $f : X \mapsto \{0, 1\}$, an input x is a *positive input* to f if $f(x) = 1$, else it is a *negative input*. All the positive inputs to f form the *positive set* of f, denoted by S_f. A *predicate class*, often denoted by $\mathcal{C}_n$, is simply a collection of predicates over $\{0, 1\}^n$. A *predicate class family* is an infinite sequence of predicate classes $\mathcal{C} = (\mathcal{C}_1, \mathcal{C}_2, ...)$, such that $\mathcal{C}_n$ is a predicate class over $\{0, 1\}^n$.

A *parity* function $\oplus_s(x)$ is defined to be $\oplus_s(x) = s \cdot x \bmod 2$. A *negative parity* function $\neg\oplus_s(x)$ is the negation of the parity function $\oplus_s(x)$.

2.1 Statistical Query Sampling

Definition 1 (Statistical Query Sampling Oracle) *A* statistical query sampling oracle *(SQS-oracle) for a predicate f is denoted by* SQS^f. *On input (g, ξ), where $g : \{0, 1\}^n \mapsto \{-1, +1\}$ is the* query function *and $\xi \in [0, 1]$ is the* tolerance, *the oracle returns a real number y such that $|y - \mathbf{E}_{x \in S_f}[g(x)]| \leq \xi$.*

Definition 2 (SQ-Samplability) *A predicate class family $\mathcal{C}$ is* SQ-samplable at rate s in time t and tolerance ξ, *if there exists a randomized oracle machine $\mathcal{Z}$, such that for every $n > 0$ and every $f \in \mathcal{C}_n$, $\mathcal{Z}$ with access to any SQS-oracle* SQS^f, *runs in at most $t(n)$ steps, asks queries with tolerance at least ξ, and outputs an $x \in S_f$ with probability at least $s(n)$. We say $\mathcal{C}$ is* strong SQ-samplable *if for every ϵ, $\mathcal{C}$ is SQ-samplable at rate $1 - \epsilon$ in time t and tolerance ξ such that t and ξ^{-1} are polynomial in n and $1/\epsilon$. We say $\mathcal{C}$ is* weak SQ-samplable *if there exists a polynomial p, such that $\mathcal{C}$ is SQ-samplable at rate $1/p(n)$ in time and inverse tolerance polynomial in n.*

Definition 3 (Sampling Algorithms with Auxiliary Inputs) *A predicate class family $\mathcal{C}$ is* SQ-samplable with auxiliary input ϕ *if it is SQ-samplable by an algorithm $\mathcal{Z}$ which takes $\phi(f)$ as the auxiliary input, where f is the predicate being sampled.*

3 Lower Bounds Based on Fourier Analysis

We first prove two hardness results on SQ sampling, using Fourier analysis techniques developed in the context of SQ learning.

3.1 A Lower Bound on Weak SQ-Sampling

We prove that there exist very simple families of predicate classes that are not weak SQ-samplable, i.e., no efficient algorithm can produce a positive input at any non-negligible rate.

We introduce a bit more notation. We use boldface to denote a vector and index the entries of an n-dimensional vector from 0 to $(n-1)$. We use $\mathbf{x}[i]$ to denote the i-th entry of $\mathbf{x}$. $\mathbf{x}[a..b]$ indicates the sub-vector formed by the entries of $\mathbf{x}$ between the a-th and the b-th, inclusive. Let $\hat{X}_{n,p}$ be the set of all n-dimensional vectors over $\mathbb{Z}_p$ (the Galois field modulo p) whose last $n-1$ entries are not all-zero, i.e.,

$$\hat{X}_{n,p} = \{\mathbf{x} \in \mathbb{Z}_p^n \mid \mathbf{x}[1..n-1] \neq (0,0,...,0)\}. \quad (1)$$

It is easy to see that $|\hat{X}_{n,p}| = p^n - p$.

Definition 4 (Booleanized Linear Functions) *A booleanized linear function over $\hat{X}_{n,p}$ with parameter $\mathbf{a}$ is denoted by $L_{\mathbf{a}}$ and defined as*

$$L_{\mathbf{a}}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{a}\cdot\mathbf{x} = 1 \pmod p \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

We say $L_{\mathbf{a}}$ is normalized *if* $\mathbf{a}[0] = 1$. *The* normalized booleanized linear function class, *denoted by $\mathcal{L}_{n,p}$, consists of all normalized booleanized linear functions over $\hat{X}_{n,p}$. In other words,*

$$\mathcal{L}_{n,p} = \{L_{\mathbf{a}} \mid \mathbf{a} \in \mathbb{Z}_p^n,\ \mathbf{a}[0] = 1\} \quad (3)$$

Theorem 1 *If a sampling algorithm for the normalized booleanized linear function class $\mathcal{L}_{n,p}$ makes less than $p^{n/4}$ queries, each of tolerance $1/p^{n/3}$, then the probability it produces a positive input $\mathbf{x} \in \hat{X}_{n,p}$ is at most $1/p + 1/p^{n/13}$.*

Notice that the requirement $\mathbf{x} \in \hat{X}_{n,p}$ is simply to rule out the trivial positive input $100\ldots0$, and we could have equivalently just modified the definition of a "booleanized linear function" so that this specific example is made negative. Also, notice that if we choose p to be much greater than n, say picking p to be an n-bit prime number, then $1/p + 1/p^{n/13}$ is exponentially small, while the size of the problem is still polynomial in n. Furthermore, if a completely random $\mathbf{x}$ is picked, the probability it is a positive input is $1/p$. Thus even exponentially many queries may only help the sampling by an exponentially small margin.

Proof: Our proof strategy is similar to that used by Kearns [22] and Blum et. al. [3] in the context of SQ learning. We describe an "adversarial" SQS-oracle $\widetilde{\mathrm{SQS}}$ that does not commit to any particular predicate at the beginning. Rather, the oracle maintains a "candidate predicate set" P, which initially includes all predicates in the class $\mathcal{L}_{n,p}$ (a total p^{n-1} of them). Each time the algorithm $\mathcal{Z}$ makes a query, $\widetilde{\mathrm{SQS}}$ replies with an answer that yields very little information. Some predicates in the candidate set P might not be consistent with the answer and will be removed from set P. After all the queries are finished, $\widetilde{\mathrm{SQS}}$ then commits to a random predicate remaining in P. We shall prove that each query only removes a small fraction of the predicates from P. Thus if $\mathcal{Z}$ does not make enough number of queries, there would be enough predicates left in P such that no element can be positive with high probability.

For a query function $g : \hat{X}_{n,p} \mapsto \{-1,+1\}$, we say that a subset $S \subseteq \{0,1\}^n$ is a *ξ-independent subset* for g, if $|\mathbf{E}_{x\in S}[g(x)] - \mathbf{E}_{x\in\hat{X}_{n,p}}[g(x)]| \leq \xi$, and we say a predicate f is *ξ-independent* from g, if its positive set S_f is a ξ-independent set for g. Intuitively, if a predicate f is ξ-independent from g, then the query (g,ξ) reveals almost no information about f, since $\widetilde{\mathrm{SQS}}$ can reply with $\mathbf{E}_{x\in\hat{X}_{n,p}}[g(x)]$ instead, which is completely independent from f.

We describe the behavior of our SQS-oracle $\widetilde{\mathrm{SQS}}$ in more detail. On query g, $\widetilde{\mathrm{SQS}}$ replies with $\mathbf{E}_{x\in\hat{X}_{n,p}}[g(x)]$, and removes all predicates that are not ξ-independent from g from the candidate set P. We assume that all queries have tolerance $\xi = p^{-n/3}$. We shall prove that for any query g, there are at most $p^{2n/3+2}$ predicates not $p^{-n/3}$-independent from g. This proof is by a Fourier analysis technique and is given as Lemma 5 in Appendix D. Thus, if less than $p^{n/4}$ queries are made, the candidate set still contains at least $p^{n-1}(1-p^{-n/12-3})$ parity functions.

Now consider the domain $\hat{X}_{n,p}$. It is not hard to see that every $x \in \hat{X}_{n,p}$ is positive for only p^{n-2} predicates. So, if the oracle commits to a random predicate out of the set of $p^{n-1}(1-p^{-n/12-3})$, the probability that x is positive is at most $1/p + 1/p^{n/13}$. ■

3.2 A Lower Bound on Sampling Negative Parity Predicates

We prove that a class of negative parity functions is not SQ-samplable in polynomial time at any rate non-negligibly higher than $1/2$.

Theorem 2 *Let $X_n = \{0,1\}^n \backslash \{0^n\}$ and $\mathcal{C}_n$ be the class of negative parity functions over X_n. If a sampling algorithm for $\mathcal{C}_n$ makes less than $2^{n/4}$ queries, each of tolerance $2^{-n/4}$, then the probability it produces a positive input is at most $\frac{1}{2} + \frac{1}{2^{n/4-2}}$.*

Before proving the theorem, we point out how this result relates to the translation of Simon's algorithm to the NMR model. In Simon's algorithm, the quantum sampling circuit produces a random $y \in \{0,1\}^n$ such that $y \cdot s = 0$, where s is the "hidden" secret (see Appendix A). Thus the hidden set corresponds exactly to the negative parity function $\neg\oplus_s$. In the algorithm, the quantum sampling circuit is invoked $\Theta(n)$ times and produces $\Theta(n)$ samples for Gaussian elimination. Notice that $y = 0^n$ is useless. Therefore, a translation of the quantum sampling circuit will produce an SQ-sampling algorithm $\mathcal{Z}$ to be executed $\Theta(n)$ times and to produce $\Theta(n)$ positive samples in $X_n = \{0,1\}^n \backslash \{0^n\}$. However, Theorem 2 implies that it is not possible to sample efficiently at any rate non-negligibly higher than $1/2$ (notice that a random $x \in X_n$ is positive with probability almost $1/2$). This result suggests that it appears necessary to manufacture $\Theta(n)$ copies of the quantum sampling circuit and run these copies together in the NMR model.

Proof sketch: The proof strategy is similar to that of Theorem 1. We assume that each query has tolerance $\xi = 1/2^{n/4}$. We construct an SQS-oracle that on query function g, replies with $\mathbf{E}_{x\in\{0,1\}^n}[g(x)]$, and remove all predicates that are not ξ-independent from g from the candidate set P (here the definition of "ξ-independent" naturally changes to $|\mathbf{E}_{x\in S}[g(x)] - \mathbf{E}_{x\in\{0,1\}^n}[g(x)]| \le \xi$). We shall prove in Lemma 7 (in Appendix D) that for any query g, there are at most $2^{n/2+2}$ predicates not $2^{-n/4}$-independent from g. Thus, if less than $2^{n/4}$ queries are made, the candidate set still contains at least $2^n - 2^{3n/4+2} - 1$ parity functions.

Now consider the domain $X_n = \{0,1\}^n \backslash \{0^n\}$. It is not hard to see that every $x \in X_n$ is positive for 2^{n-1} negative parity functions. Now if a random parity function is chosen from a set of size $2^n - 2^{3n/4+2} - 1$, the probability that x is positive is at most

$$\frac{2^{n-1}}{2^n - 2^{3n/4+2} - 1} \le \frac{1}{2} + \frac{1}{2^{n/4-2}}.$$

This is true for any $x \in X_n$. Therefore, whatever $\mathcal{Z}$ outputs, the probability that it is positive is at most $\frac{1}{2} + \frac{1}{2^{n/4-2}}$. ■

4 A Cryptographic Lower Bound

We next prove a cryptographic lower bound. Assuming that one-way functions exist, we show that there exist predicate class families that are not weak SQ-samplable, even if the sampling algorithm is given the complete description of the predicate as the auxiliary input. The technique we use here is somewhat similar to that of Angluin and Kharitonov [1], who used signature schemes to prove that membership queries do not help to learn DNF.

We briefly describe the ideas behind our proof. We will use a digital signature scheme secure against adaptive chosen message attack [14], which exists if one-way functions exist [25]. Let the predicate be the signature verification function $\mathsf{ver}_{vk}(m, s)$, which returns 1 if s is a valid signature to message m with respect to the verification key vk. The security of the signature scheme states that no "breaker" $\mathcal{B}$, given access to a signing oracle, can produce a *new* valid signature it has not yet seen. We want to argue that this implies no sampling algorithm $\mathcal{Z}$, given access to a SQ-sampling oracle, can produce *any* valid signature. We will show that if such an algorithm $\mathcal{Z}$ exists, we can construct a "breaker" $\mathcal{B}$ as follows. The breaker will have access to a signing oracle OSign that signs any message given to it as input, and runs $\mathcal{Z}$ as a subroutine. The only non-trivial part for $\mathcal{B}$ is to simulate an SQS-oracle used by $\mathcal{Z}$ without revealing to $\mathcal{Z}$ any information about which signatures it has already seen (so that $\mathcal{Z}$ is not biased towards producing an already-seen signature). Upon a query (g, ξ) from $\mathcal{Z}$, $\mathcal{B}$ will produce a number of random messages, ask the signing oracle to sign them, and use these samples to estimate $\mathbf{E}_{x\in S_f}[g(x)]$. Next, $\mathcal{B}$ "randomizes" this estimate by adding an artificial noise to it. With properly chosen parameters, this "randomized" estimate is still a valid answer with very high probability, and yet almost independent from the messages $\mathcal{B}$ produces. Finally, $\mathcal{Z}$ produces a positive input, which is a message/signature pair (m', s'). The distribution of the this pair (m', s') is also almost independent from the messages $\mathcal{B}$ produces, and if $\mathcal{Z}$ only makes polynomially many queries, then only polynomially many messages will be produced by $\mathcal{B}$. Therefore the probability that m' is one of the messages produced by $\mathcal{B}$ is very small, and so $\mathcal{B}$ breaks the digital signature scheme with reasonably high probability.

Formally, a *signature scheme* SIG is a triple $(\mathsf{sig\_gen}, \mathsf{sig\_sign}, \mathsf{sig\_verify})$ of algorithms, the first two being probabilistic, and all running in polynomial time. $\mathsf{sig\_gen}$ takes as input 1^n and outputs a signing/verification key pair (sk, vk). $\mathsf{sig\_sign}$ takes a message m and a signing key sk as input and outputs a signature s for m. WLOG we assume that both m and s are n-bits long. $\mathsf{sig\_verify}$ takes a message m, a verification key vk, and a candidate signature s' for m as input and returns the bit $b = 1$ if s' is a valid signature for m for the corresponding verification key vk, and otherwise returns the bit $b = 0$. Naturally, if $s = \mathsf{sig\_sign}(sk, m)$, then $\mathsf{sig\_verify}(vk, m, s) = 1$. In an adaptive chosen message attack [14], an adversary ("breaker") $\mathcal{B}$ is given vk, where $(sk, vk) \leftarrow \mathsf{sig\_gen}(1^n)$, and tries to forge signatures with respect to vk. The breaker $\mathcal{B}$ is allowed to query a signing oracle OSign_{vk}, which signs any message with respect to vk, on messages of its choice. It succeeds in existential forgery if after this it can output a pair (m, s), where $\mathsf{sig\_verify}(vk, m, s) = 1$, but m was not one of the messages signed by the signature oracle. A signature scheme SIG is existentially unforgeable against adaptive chosen message attacks if there is no forging algorithm $\mathcal{B}$ that runs in time polynomial in n and succeeds with probability $1/poly(n)$. Such schemes exist if one-way functions exist [25].

Theorem 3 *Let* $\mathsf{SIG} = (\mathsf{sig\_gen}, \mathsf{sig\_sign}, \mathsf{sig\_verify})$ *be a digital signature scheme secure against adaptive chosen* *message attack. Then the predicate class family $\mathcal{C}_n = \{\mathsf{ver}_{vk}\}$ is not weakly SQ-samplable, even if the sampling algorithm is given vk as the auxiliary input. Here ver_{vk} is defined to be $\mathsf{ver}_{vk}(m,s) = \mathsf{sig\_verify}(vk,m,s)$, where $(sk,vk) \leftarrow \mathsf{sig\_gen}(1^n)$, and $m,s \in \{0,1\}^n$.*

Proof: Assume to the contrary that there exists an algorithm $\mathcal{Z}$ that weak SQ-samples the function class $\mathcal{C}_n = \{\mathsf{ver}_{vk}\}$. More precisely, we assume that $\mathcal{Z}$ produces a positive input with probability ϵ by making q queries, where both $1/\epsilon$ and q are bounded by a polynomial in n. We shall construct a polynomial-time algorithm $\mathcal{B}$ that breaks the signature scheme SIG with probability $\epsilon/2$, causing a contradiction.

We now describe the behavior of $\mathcal{B}$. $\mathcal{B}$ has access to a signing oracle OSign_{vk} and interacts with the sampling algorithm $\mathcal{Z}$ as the SQS-oracle. When $\mathcal{Z}$ makes a query (g,ξ), $\mathcal{B}$ does the following. First, $\mathcal{B}$ computes $\xi_0 = \frac{\xi\cdot\epsilon}{10q}$ and $M = \frac{2\ln(10q/\epsilon)}{\xi_0^2}$. Then $\mathcal{B}$ draws M random messages $m_1, m_2, ..., m_M \in \{0,1\}^n$, and asks the signing oracle to sign all of them. Assume the signatures are $s_1, s_2, ..., s_M$. Next, $\mathcal{B}$ uses these message/signature pairs to estimate the expected value of g by computing $x = \frac{1}{M}\sum_{k=1}^{M} g(m_k, s_k)$. Then $\mathcal{B}$ "randomizes" x by drawing a y uniformly randomly from the interval $[x - \frac{\xi}{2}, x + \frac{\xi}{2}]$, and sending y to $\mathcal{Z}$ as the answer to the query (g,ξ). $\mathcal{B}$ also maintains a "history set" set H of all the messages it has generated, which is initially $\emptyset$. After a query from $\mathcal{Z}$ is answered, $\mathcal{B}$ adds the messages $m_1, m_2, ..., m_M$ to set H.

After all the q queries are made, $\mathcal{Z}$ produces a pair (m', s'). If $\mathsf{ver}_{vk}(m', s') = 1$ and $m' \notin H$, then $\mathcal{B}$ outputs (m', s') and successfully forges a signature. Otherwise $\mathcal{B}$ aborts and announces failure.

It is clear that $\mathcal{B}$ runs in polynomial time. Intuitively, we can show that after the randomization, with high probability the sample (m', s') produced by $\mathcal{Z}$ is almost independent from the history set H. Therefore, with high probability, $m' \notin H$, and so $\mathcal{B}$ will succeed. More precisely, we prove that $\mathcal{B}$ will succeed with probabilit at least $\epsilon/2$.

We use S_{vk} to denote the positive set for predicate ver_{vk}. In other words, S_{vk} consists of valid message/signature pairs with respect to the verification key vk.

Claim 1 *For a query function g, if we define $\sigma = \mathbf{E}_{(m,s)\in S_{vk}}[g(m,s)]$, then with probability at least $1-\epsilon/5q$, we have $|x - \sigma| \le \xi_0$ (all quantities are as defined in the proof sketch of Theorem 3).*

Proof: This is due to a straightforward application of the Hoeffding Bound. Each sample (m_k, s_k) is an independent random element from S_{vk} and thus $\mathbf{E}_{(m,s)\in S_{vk}}[g(m,s) = 1] = \sigma$. So the expected value of x is σ. Now, the probability that M independent samples yields an average below $\sigma - \xi_0$ is at most $e^{-M\xi_0^2/2}$ (notice that the range of g is $\{-1, +1\}$). Also the probability that the average is above $\sigma + \xi_0$ is at most $e^{-M\xi_0^2/2}$. Therefore with probability at least $1 - 2e^{-M\xi_0^2/2} \ge 1 - \epsilon/5q$, we have $|x - \sigma| \le \xi_0$. ■

We fix a set consisting of M message/signature pairs generated by $\mathcal{B}$ in response to a query (g,ξ), and denote this by U: $U = \{(m_k, s_k)\}_{k=1}^{M}$. We call this set a *sample set*. We say U is *typical*, if the average $g(m_k, s_k)$ is indeed ξ_0-close to σ. By Claim 1, at most $\epsilon/5q$ fraction of the sample sets are not typical.

Notice that a typical sample set will yield an average that is ξ_0-close to σ. This is a much higher accuracy than required by the $\mathcal{Z}$, which has a tolerance of ξ. However, $\mathcal{B}$ needs this accuracy to perform the randomization.

Claim 2 *If U is a typical set, then the answer from $\mathcal{B}$ for this query is valid.*

Proof: Notice that if U is typical, then the average x is ξ_0-close to the true value σ. After the randomization, it is $(\xi_0 + \xi/2)$-close to σ. This is less than ξ. ■

We consider the distribution of the answer produced by $\mathcal{B}$ for a particular query (g,ξ). We denote this distribution by D_U, where U is the sample set used by $\mathcal{B}$.

Claim 3 *If both U_0 and U_1 are typical sets, then the statistical distance between D_{U_0} and D_{U_1} is at most $\epsilon/5q$.*

Proof: We use x_0 and x_1 to denote the averages obtained from U_0 and U_1, respectively. If both U_0 and U_1 are typical, we have $|x_0 - \sigma| \le \xi_0$ and $|x_1 - \sigma| \le \xi_0$. Thus we have $|x_0 - x_1| \le 2\xi_0$. Notice that D_{U_0} is a uniform distribution over the interval of length ξ centered at x_0, and D_{U_1} a uniform distribution of same length centered at x_1. The claim follows from Lemma 4. ■

Notice the history set H consists of q sample sets. We say a history set H is *typical*, if all its sample sets are typical. Then at most $\epsilon/5$ fraction of the history sets are not typical. We denote the distribution of all answers produced by $\mathcal{B}$ using history set H by T_H.

Claim 4 *If both H_0 and H_1 are typical, then the statistical distance between T_{H_0} and T_{H_1} is at most $\epsilon/5$.*

Proof: This directly follow the sub-additivity of statistical distance (see Appendix C). ■

Now we fix an arbitrary typical set $\tilde{H}$ and denote its corresponding distribution of the answers by $\tilde{T}$. Then we know the distribution from any typical set is at most $\epsilon/5$ away from $\tilde{T}$.

The only information $\mathcal{Z}$ receives from $\mathcal{B}$ is represented by the distribution of the answers produced by $\mathcal{B}$, which is in turn determined by the history set $\mathcal{B}$ uses. Thus, the distribution of the pair (m', s') is completely determined by the history set H, and we denote this distribution by O_H. We know that if H is typical, then $\Pr_{(m,s)\in O_H}[\mathsf{ver}_{vk}(m,s) = 1] \geq \epsilon$. We fix the distribution $\tilde{O}$ that corresponds to the history set $\tilde{H}$. Then we have

$$\Pr_{(m,s)\in\tilde{O}}[\mathsf{ver}_{vk}(m,s) = 1] \geq \epsilon. \quad (4)$$

Furthermore, we know that for any typical history set H, its corresponding distribution of O_H is $\epsilon/5$-close to $\tilde{O}$.

Consider a new experiment (a new execution of the breaker $\mathcal{B}$) that is identical to the original one, except when $\mathcal{Z}$ outputs a pair (m', s'), it does so according to the fixed distribution $\tilde{O}$.

Claim 5 *Let $\hat{M}$ be the maximum size of the sample sets in $\tilde{H}$. Then the probability of the new experiment is at least $\epsilon - \hat{M} \cdot q/2^n$.*

Proof: Notice that the output of $\mathcal{Z}$ is independent from the history set H. Moreover, the history set contains at most $\hat{M} \cdot q$ messages. So the probability that a particular m is in H is at most $\hat{M} \cdot q/2^n$. This fact, along with (4), proves the claim. ■

Now putting things together, with probability at most $\epsilon/5$, the history set H is not typical; if H is typical, the difference between the probabilities of the two experiments is at most $\epsilon/5$; the probability of success of the new experiment is at least $\epsilon - \hat{M} \cdot q/2^n$. Therefore the probability of success of the original experiment is at least (for n large enough) $\epsilon - \hat{M} \cdot q/2^n - \epsilon/5 - \epsilon/5 > \epsilon/2$.

This finishes the proof. ■

5 SQ sampling and SQ learning

We now point out relationships between our SQ sampling model and the SQ learning model of Kearns [22]. We begin with definitions of SQ learning. (In these definitions, we assume learning is with respect to the uniform distribution over examples.)

Definition 5 (Statistical Query Learning Oracle) *A* statistical query learning oracle *(SQL-oracle) for a predicate f is denoted by SQL^f. On an input (g, ξ), where $g : \{0,1\}^n \times \{0,1\} \mapsto \{-1,+1\}$ is the* query function *and $\xi \in [0,1]$ is the* tolerance*, the oracle returns a real number y such that $|y - \mathbf{E}_{x\in\{0,1\}^n}[g(x, f(x))]| \leq \xi$.*

Definition 6 (Strong SQ-Learnability) *A predicate class family $\mathcal{C}$ is* Strong SQ-learnable *if there exists a randomized oracle machine $\mathcal{Z}$, such that for every $n > 0$, every $f \in \mathcal{C}_n$ and for every $\epsilon > 0$, $\delta > 0$, $\mathcal{Z}$ with access to any SQL-oracle SQL^f outputs a hypothesis $\hat{f}$ such that $\Pr_{x\in\{0,1\}^n}[\hat{f}(x) = f(x)] \geq 1 - \epsilon$ with probability at least $1 - \delta$, and furthermore, both the running time of $\mathcal{Z}$ and the inverse of the tolerance of each query made by it are bounded by a polynomial in n, $1/\epsilon$ and $1/\delta$. Here ϵ is called the* accuracy *and δ the* confidence.

Definition 7 (Weak SQ-Learnability) *A predicate class family $\mathcal{C}$ is* weak SQ-learnable *if there exists a randomized oracle machines $\mathcal{Z}$ and a polynomial $p(\cdot)$, such that for every n and for every $f \in \mathcal{C}_n$, $\mathcal{Z}$ with access to any SQL-oracle SQL^f, outputs a hypothesis $\hat{f}$ such that $\Pr_{x\in\{0,1\}^n}[\hat{f}(x) = f(x)] \geq 1/2 + 1/p(n)$, and furthermore, both the running time of $\mathcal{Z}$ and the inverse of the tolerance of each query made by $\mathcal{Z}$ are bounded by a polynomial in n.*

The first observation to make is that a predicate class can be strongly SQ-learnable and yet not even weakly SQ-samplable. In particular, any class with a sufficiently low density of positive examples can be trivially learned by producing the "all zero" hypothesis. (Formally, if we wish be correct even for values of ϵ that are exponentially small, it suffices to have the density less than $1/2^{n/2}$ so that if necessary we can use the SQL oracle to identify all positive examples.) In the other direction, a class can be strongly SQ-samplable and yet not even weakly SQ-learnable. Indeed, the family of negative parity functions taken over the domain $\{0,1\}^n$ is trivially SQ-samplable (because $f(0^n) = 1$ for any such f), but such functions are not even weakly SQ-learnable [22]. It is interesting to compare this to Theorem 2, since the predicate class families in these two theorems are very similar (one can think of the difference either as removing 0^n from the domain, or simply as changing the values of the functions at this one point), yet they have completely different characterization in terms of SQ-samplability.

However, we show there is a relationship between these notions when the set of positive examples is sufficiently dense.

5.1 SQ-learnability sometimes implies SQ-samplability

We prove that under certain circumstances, SQ-learnability implies SQ-samplability.

Definition 8 (Density of Predicates) *The* density *of a predicate $f : \{0,1\}^n \mapsto \{0,1\}$, denoted by $\rho(f)$, is the fraction of its inputs that are positive. In other words, $\rho(f) = \Pr_{x\in\{0,1\}^n}[f(x) = 1]$.*

Definition 9 (Dense Predicates) *A predicate class family $\mathcal{C}$ is* dense *if there exists a polynomial $p(\cdot)$ such that for every n and for every $f \in \mathcal{C}_n$, $\rho(f) \geq 1/p(n)$.*

Theorem 4 *If a dense predicate class family is strong SQ-learnable, then it is also strong SQ-samplable with the auxiliary input ρ.*

Proof: Let $\mathcal{Z}$ be the algorithm that strongly SQ-learns dense predicate family $\mathcal{C}$. We construct a new algorithm A that strong SQ-samples $\mathcal{C}$ using the density ρ of the predicate f as auxiliary input. A runs a copy of $\mathcal{Z}$, whose accuracy and confidence are set to be $\epsilon = \rho \cdot \epsilon'/4\ln(\frac{4}{\epsilon'})$ and $\delta = \epsilon'/4$, and simulates the SQL-oracle used by $\mathcal{Z}$. We shall prove that A produces a positive input with probability at least $1-\epsilon'$.

We now describe the behavior of A. A works in two phases. In this first phase, it simulates the SQL-oracle SQL^f. When $\mathcal{Z}$ submits a query (g,ξ) to A, A does the following.

1. Set $M = \frac{9\ln(2q/\delta)}{2\xi^2}$, draw M independent samples $x_1, x_2, \ldots, x_M$ from $\{0,1\}^n$, and compute
$$s = \frac{1}{M}\sum_{i=1}^{M} g(x_i, 0).$$
2. Construct two query functions $g_0(x) = g(x,0)$ and $g_1(x) = g(x,1)$. Submit queries $(g_0, \xi/3)$ and $(g_1, \xi/3)$ to the SQS-oracle SQS^f and receive y_0 and y_1 as answers.
3. Compute $y = s + (y_1 - y_0)\cdot\rho$ and send y to $\mathcal{Z}$ as the answer to the query (g,ξ).

The algorithm A enters the second phase when $\mathcal{Z}$ produces a hypothesis $\hat{f}$. Then A repeats the following procedure. It draws a random $x \in \{0,1\}^n$, and check if $\hat{f}(x) = 1$. If so it stops and output x; otherwise it continues. The procedure is repeated $\ln\left(\frac{1}{\delta}\right)/\rho$ times and if A still hasn't stopped, it produces a random $x \in \{0,1\}^n$ and outputs it.

It is clear that A runs in polynomial time. Now, we prove that A produces a positive sample with probability at least $1-\epsilon'$.

First, we prove that with probability at least $1-\delta$, all answers provided by A are valid in the first phase. Consider an average s as an approximation of $\mathbf{E}_{x\in\{0,1\}^n}[g(x,0)]$. We say s is "bad", if $|s - \mathbf{E}_{x\in\{0,1\}^n}[g(x,0)]| > \xi/3$. Then a simple application of the Hoeffding Bound (see Appendix B) proves that the probability that s is bad is at most δ/q.

Next, notice that
$$g(x, f(x)) = g(x,0) + [g(x,1) - g(x,0)]\cdot f(x).$$
Therefore we have
$$\begin{aligned}\mathbf{E}_{x\in\{0,1\}^n}[g(x,f(x))] &= \mathbf{E}_{x\in\{0,1\}^n}[g(x,0)] + \\ &\quad \mathbf{E}_{x\in\{0,1\}^n}[(g(x,1)-g(x,0))\cdot f(x)] \\ &= \mathbf{E}_{x\in\{0,1\}^n}[g(x,0)] + \\ &\quad \left(\mathbf{E}_{x\in S_f}[g(x,1)] - \mathbf{E}_{x\in S_f}[g(x,0)]\right)\cdot\rho\end{aligned}$$

Therefore, if s is not bad, then the y computed by A is a valid reply to query (g,ξ). Since $\mathcal{Z}$ makes a total of q queries, with probability at least $1-\delta$, all the replies by A are valid and $\mathcal{Z}$ should perform well.

Next, consider the second phase of A. With probability at least $1-\delta$, $\mathcal{Z}$ should produce a hypothesis $\hat{f}$ that agrees with f with probability at least $1-\epsilon$. Let us assume th $\mathcal{Z}$ does produce such a $\hat{f}$. Now since a ρ fraction of the inputs are positive, the probability that A doesn't draw a positive input in $\ln\left(\frac{1}{\delta}\right)/\rho$ rounds is at most δ. The probability that $\hat{f}$ makes a mistake in any of the rounds is at most $\ln\left(\frac{1}{\delta}\right)\cdot\epsilon/\rho$. If $\hat{f}$ doesn't make any mistakes and at least one positive input is drawn, then A will correctly output it.

Putting everything together, we know that with probability at least $1 - 3\delta - \ln\left(\frac{1}{\delta}\right)\cdot\epsilon/\rho = 1-\epsilon'$, A will output a positive input. ∎

We remark that it appears necessary for the SQ-sampling algorithm to have the density ρ as an auxiliary input. One difference between SQ-sampling and the SQ-learning is the *resolution*. In the reply of an SQS-oracle, the underline distribution is uniform over the "hidden set" S_f; for an SQL-oracle, the distirbution is uniform over the entire set $\{0,1\}^n$. Therefore, a sampling algorithm needs to know the size of S_f in order to perform the simulation (more precisely, in step 3 of the first phase).

It is interesting to compare this result to Theorem 3, which shows a predicate class family that is perfectly SQ-learnable, but not even weakly SQ-samplable. Nevertheless, there is no contradiction since the predicate class family in Theorem 3 is not dense.

Acknowledgements

We would like to thank David Collins for bringing this problem to our attention, and Bob Griffiths and David Collins for helpful discussions.

References

[1] D. Angluin and M. Kharitonov. When won't membership queries help? In *STOC 1991*, pp. 444–454, 1991.

[2] E. Bernstein and U. Vazirani. Quantum complexity theory. In *SIAM J. Comp.*, 26(5);1411–1473, also available at *LANL e-print* `quant-ph/9701019`.

[3] A. Blum, M. Furst, J. Jackson, M. Kearns, Y. Mansour, and S. Rudich. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In *STOC 1994*, pp. 253–262, 1994.

[4] D. Boneh and R. Lipton. Quantum cryptanalysis of hidden linear forms. In *Crypto '95*, LNCS 963, pp. 424–437, 1995.

[5] I. Chuang, L. Vandersypen, X. Zhou, D. Leung, and S. Lloyd. Experimental realization of a quantum algorithm. In *Nature*, 393:143–146, 1998.

[6] D. Collins. Modified Grover's algorithm for an expectation-value quantum computer. In *Phys. Rev. A.*, **65**, 052321, 2002.

[7] D. G. Cory, A. F. Fahmy, and T. F. Havel. Ensemble quantum computing by nuclear magnetic resonance spectroscopy. In *Proc. Natl. Acad. Sci.* 94:1634–1639, 1997.

[8] D. Deutsch. Quantum theory, the Church-Turing Principle and the universal quantum computer. In *Proc. R. Soc. Lond. A*, 400:97–117, 1985.

[9] D. Deutsch. Quantum computational networks. In *Proc. R. Soc. Lond. A*, 425:73, 1989.

[10] D. P. DiVincenzo. Two-bit gates are universal for quantum computation. In *Phys. Rev. A*, 51(2):1015–1022, 1995.

[11] M. Ettinger and P. Høyer. On quantum algorithms for non-commutative hidden subgroups. In *STACS'99*, also available at *LANL e-print* `quant-ph/9807029`.

[12] M. Ettinger, P. Høyer, and E. Knill. Hidden subgroup states are almost orthogonal. In *LANL e-print* `quant-ph/9901034`, 1999.

[13] N. Gershenfeld and I. L. Chuang. Bulk spin resonance quantum computation. In *Science*, 275:350, 1997.

[14] S. Goldwasser, S. Micali and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. In *SIAM J. Comput.*, 17:281–308, 1988.

[15] M. Grigni, L. J. Schulman, M. Vazirani, U. V. Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. In *STOC 2001*, pp. 68–74, 2001.

[16] L. Hales, S. Hallgren. Quantum Fourier sampling simplified. In *STOC 1999*, pp. 330–338, 1999.

[17] S. Hallgren. Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. In *STOC 2002*, pp.653–658, 2002.

[18] S. Hallgren, A. Russell, A. Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *STOC 2000*, pp. 627–635, 2000.

[19] W. Hoeffding. Probability inequalities for sums of bounded random variables. In *Journal of the American Statistical Association*, 58:13–30, 1963.

[20] J. Jackson. On the efficiency of noise-tolerant PAC algorithms derived from statistical queries. In *COLT 2000*, 2000.

[21] R. Jozsa. Quantum algorithms and the Fourier transform. In *LANL e-print* `quant-ph/9707033`, 1997.

[22] M. Kearns. Efficient noise-tolerant learning from statistical queries. In *STOC 1993*, pp. 392–401, 1993.

[23] M. Kearns and U. V. Vazirani. An introduction to computational learning theory. MIT Press, 1994.

[24] M. Nielsen and I. Chuang. Quantum Computation and Quantum Information. *Cambridge University Press*, 2000.

[25] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd ACM Symposium on the Theory of Computing*, pp. 387–394, 1990.

[26] P. Shor. Algorithms for quantum computation: discrete logarithms and factorization. In *FOCS '94*, pp.124–134, 1994.

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

[28] D. Simon. On the power of quantum computation. In *FOCS'94*, pp. 116–123, 1994. Journal version available at *SIAM J. Comp.*, 26(5):1474–1483, 1997.

[29] S.Vadhan. A study of statistical zero-knowledge proofs. *Ph.D. thesis*, MIT, 2000.

[30] L. Valiant. A theory of the learnable. In *Communications of the ACM*, 27(11): 1134–1142, 1984.

[31] K. Yang. On learning correlated functions using statistical query. In *ALT'01*, LNAI 2225, pp. 59–76, 2001. Full version available at *ECCC* TR01-098.

[32] K. Yang. New lower bounds for statistical query learning. In *COLT 2002*, LNAI 2375, pp. 229–243, 2002. Full version available at *ECCC* TR02-060.

[33] A. Yao. Quantum Circuit Complexity. In *FOCS'93*, pp. 351–361, 1993.

A Shor's Algorithm and Simon's Algorithm

We briefly summarize Shor's algorithm for factoring and Simon's algorithm for the hidden XOR-secret problem.

A.1 Shor's Algorithm for Factoring

Standard number theory reduces factoring N to finding the order of a random element a modulo N, i.e., $r > 0$ such that $a^r \equiv 1 \pmod N$ but $a^s \not\equiv 1 \pmod N$ for any $0 < s < r$. Suppose $2^{n-1} < N \leq 2^n$. Shor's algorithm uses $2n$ qubits, separated into two n-qubit registers. Initially the state is initialized to $|\,\phi_0\rangle = |\,0^n\rangle|\,0^n\rangle$. By applying the Fourier transformation followed by modular exponentiation, this state is converted to $|\,\phi_1\rangle = \frac{1}{2^{n/2}}\sum_x |\,x\rangle|\,a^x \mod N\rangle$. Then one measures the second register and discard it, leading to a state $|\,\phi_2\rangle = \sum_t |\,t\cdot r + c\rangle$ for some random $c \in [r]$, where t ranges from 0 to $\lfloor(2^n - 1 - c)/r\rfloor$ (we ignore the scalar factor). Finally, one applies the inverse Fourier transform to the first register followed by a measurement. The distribution of the measurement result is approximately uniform over $\{[t \cdot 2^n/r] : 0 \le t \le \lfloor (2^n - 1 - c)/r \rfloor\}$. One can then solve r from one instance of $[t \cdot 2^n/r]$ using continued fraction.

A.2 Simon's Problem and Algorithm

A function $f : \{0,1\}^n \mapsto \{0,1\}^n$ is given as an oracle, with the promise that there exists an $s \in \{0,1\}^n$ (known as the "hidden secret") such that $f(x) = f(y)$ iff $x \oplus y = s$. Notice that if $s = 0^n$, then f is a permutation, and otherwise f is a 2-to-1 function. The problem is to tell if $s = 0^n$.

Simon's algorithm works as follows. One starts with $2n$ qubits, separated into two n-qubit registers. Originally one initializes the state to $|\phi_0\rangle = |0^n\rangle|0^n\rangle$. Next, one applies the Hadamard operator to the first register and then the oracle operator $|x\rangle|y\rangle \mapsto |x\rangle|f(x) \oplus y\rangle$. The state becomes $|\phi_1\rangle = \frac{1}{2^{n/2}}\sum_x |x\rangle|f(x)\rangle$. Next, the second register is measured and discarded. If $s = 0^n$, then the measurement result is $|\phi_2\rangle = |x\rangle$ for a random $x \in \{0,1\}^n$. If $s \ne 0^n$, then the measurement is $|\phi_2'\rangle = \frac{1}{\sqrt{2}}(|x\rangle + |x \oplus s\rangle)$ for a random x. Next, a Hadamard operator is applied to the first register. In the case $s = 0^n$, the result is $|\phi_3\rangle = |y\rangle$ for a random y; in the case $s \ne 0^n$, the result is $|\phi_3'\rangle = |y\rangle$ for a random y such that $y \cdot s = 0$. Finally one measures the first register and obtains y. Repeating the experiment $O(n)$ times, one can solve for s by using Gaussian elimination and distinguish the case $s = 0^n$ from the case $s \ne 0^n$.

B The Hoeffding Bound

We state the Hoeffding Bound, a classical result in estimating tail probabilities.

Lemma 1 (Hoeffding Bound [19]) *Let $k = (p - \epsilon)n$, where ϵ is a real number between 0 and $1/2$, and p is a real number between 0 and 1. We have*

$$\sum_{j=0}^{k} \binom{n}{j} p^j (1-p)^{m-j} \le e^{-2n\epsilon^2} \tag{5}$$

■

C Statistical Distance

We define the statistical distance and state some of its properties. The definitions and the results are standard. A good reference to the statistical distance is Vadhan's thesis [29].

Definition 10 (Statistical Distance) *The* statistical distance *between two probability distributions A and B, denoted as $\mathsf{SD}(A,B)$, is defined to be*

$$\mathsf{SD}(A,B) = \frac{1}{2}\sum_x |A(x) - B(x)| \tag{6}$$

where the summation is taken over the support of A and B. If $\mathsf{SD}(A,B) \le \epsilon$, we say A is ϵ-close *to B.*

This definition can be easily extended to the continuous case with the summation being replaced by integral and the distributions replaced by density functions.

Lemma 2 *Let $T(x)$ be a probabilistic event with x as input. Let A and B be two distributions. We have*

$$\left| \Pr_{x \in A}[T(x)] - \Pr_{x \in B}[T(x)] \right| \le \mathsf{SD}(A,B) \tag{7}$$

■

Lemma 3 (Sub-additivity) *Let A_1, A_2, B_1, B_2 be distributions, then we have*

$$\mathsf{SD}(A_1B_1, A_2B_2) \le \mathsf{SD}(A_1, A_2) + \mathsf{SD}(B_1, B_2) \tag{8}$$

where AB denotes the tensor product *of the distributions A and B, i.e., $AB(a,b) = A(a) \cdot B(b)$.* ■

Lemma 4 *Let D_1 be a uniform distribution over an interval $[a, a+l]$ and D_2 a uniform distributions over $[b, b+l]$. Then $\mathsf{SD}(D_1, D_2)$ is at most $|a-b|/l$.*

Proof: Notice that both D_1 and D_2 are uniform distributions of same length, and thus their density functions have value $1/l$ over their supports and 0 elsewhere. Consider the absolute difference between the two density functions, $|D_1(x) - D_2(x)|$. The size of its support is at most $2|a-b|$. Thus $\mathsf{SD}(D_1, D_2) \le |a-b|/l$. ■

D Proofs

Lemma 5 *Let $\hat{X}_{n,p}$ be the domain defined in (1) and $\mathcal{L}_{n,p}$ be the class of normalized booleanized linear functions over $\hat{X}_{n,p}$. For any query function $g : \hat{X}_{n,p} \mapsto \{0,1\}$, there are at most $p^{2n/3+2}$ predicates in $\mathcal{L}_{n,p}$ that are not $1/p^{n/3}$-independent from g.*

For the proof we will need:

Lemma 6 ([31]) *Let $\Omega = \{f_i\}$ be a set of function of range $\{-1,+1\}$ and d be its cardinality. If $\langle f_i, f_j\rangle = \lambda$ for all $i \ne j$, then the set $\{\tilde{f}_i\}$ forms an orthonormal basis for the linear space spanned by Ω, where*

$$\tilde{f}_i(x) = \frac{1}{\sqrt{1-\lambda}} f_i(x) - \frac{1}{d} \cdot \left(\frac{1}{\sqrt{1-\lambda}} - \frac{1}{\sqrt{1+(d-1)\lambda}} \right) \cdot \sum_{j=1}^{d} f_j(x) \tag{9}$$

■

Proof of Lemma 5: We first slightly modify the class $\mathcal{L}_{n,p}$ so that its range becomes $\{-1,+1\}$. We define $\tilde{L}_{\mathbf{a}}(\mathbf{x}) = 2 \cdot L_{\mathbf{a}}(\mathbf{x}) - 1$. It is not hard to see that each of the p^{n-1} normalized booleanized linear functions maps a $1/p$ fraction of the elements in $\hat{X}_{n,p}$ to $+1$, and a straightforward but tedious analysis (see [31] for a detailed account) shows that any two normalized booleanized linear functions agree at exactly $(p^2-2p+2)p^{n-2}-p$ places in $\hat{X}_{n,p}$. We define an inner product between functions over $\hat{X}_{n,p}$ as

$$\langle f, g\rangle = \frac{1}{p^n - p}\sum_{x \in \hat{X}_{n,p}} f(x)g(x), \qquad (10)$$

With this inner product, any query function has norm 1, and any pair of distinct functions $\tilde{L}_{\mathbf{a}}$ and $\tilde{L}_{\mathbf{b}}$ have the same inner product. This will allow us to "extract" an orthonormal basis from the class $\mathcal{L}_{n,p}$ using Lemma 6.

Now we fix a query function g and relate predicates that are not ξ-independent from g to the Fourier coefficients of g. Consider a booleanized linear function $L_{\mathbf{a}}$, and we denote its positive set by S. We have that $|S| = p^{n-1}-1$. Suppose g maps a elements in $\hat{X}_{n,p}$ to $+1$, and b elements in S to $+1$. Then if $L_{\mathbf{a}}$ is not ξ-independent from g, we have

$$\left|\frac{2a - p^n + p}{p^n - p} - \frac{2b - p^{n-1}+1}{p^{n-1}-1}\right| > \xi, \qquad (11)$$

or $|a - bp| > \frac{p^n - p}{2}\xi$. We write $b = a/p + \delta$, and we have $|\delta| \geq \frac{p^{n-1}-1}{2}\xi$.

Next we compute the inner product of g and $\tilde{L}_{\mathbf{a}}$. Straightforward computation shows that

$$\begin{aligned}\langle g, \tilde{L}_{\mathbf{a}}\rangle &= 2\cdot\left(\frac{2b - a + (p-1)(p^{n-1}-1)}{p^n - p}\right) - 1 \\ &= \left(1 - \frac{2a}{p^n - p}\right)\left(1 - \frac{2}{p}\right) + \frac{4\delta}{p^n - p}\end{aligned}$$

On the other hand, the inner product of g with an *average* over booleanized linear functions is

$$\begin{aligned}\frac{1}{p^{n-1}}\sum_{\mathbf{b}[0]=1}\langle g, \tilde{L}_{\mathbf{b}}\rangle &= \frac{1}{p^{n-1}(p^n-p)}\sum_{\mathbf{b}[0]=1}\sum_{x\in\hat{X}_{n,p}} g(x)\tilde{f}_{\mathbf{b}}(x) \\ &= \frac{1}{p^{n-1}(p^n-p)}\sum_{x\in\hat{X}_{n,p}} g(x)\sum_{\mathbf{b}[0]=1}\tilde{f}_{\mathbf{b}}(x) \\ &= \left(1 - \frac{2a}{p^n-p}\right)\left(1 - \frac{2}{p}\right)\end{aligned}$$

Now we apply Lemma 6, setting $d = p^{n-1}$ and $\lambda = \frac{(p^2-4p+4)p^{n-2}-p}{p^n-p}$. We will obtain an orthonormal basis, which we denote by $\{\hat{L}_{\mathbf{b}}\}$.

Putting things together, we can compute that Fourier coefficient of g over the component $\hat{L}_{\mathbf{a}}$.

$$\begin{aligned}\langle g, \hat{L}_{\mathbf{a}}\rangle &= \frac{1}{\sqrt{1-\lambda}}\langle g, \tilde{L}_{\mathbf{a}}\rangle - \\ &\quad \left(\frac{1}{\sqrt{1-\lambda}} - \frac{1}{\sqrt{1+(d-1)\lambda}}\right)\cdot\frac{1}{d}\sum_{\mathbf{b}[0]=1}\langle g, \tilde{L}_{\mathbf{b}}\rangle \\ &= \frac{1}{\sqrt{1-\lambda}}\cdot\left[\left(1-\frac{2}{p}\right)\cdot\left(1 - \frac{2a}{p^n-p}\right) + \frac{4\delta}{p^n-p}\right] - \\ &\quad \left(\frac{1}{\sqrt{1-\lambda}} - \frac{1}{\sqrt{1+(d-1)\lambda}}\right)\cdot\left(1-\frac{2}{p}\right)\cdot\left(1-\frac{2a}{p^n-p}\right) \\ &= \frac{1}{\sqrt{1+(d-1)\lambda}}\left(1-\frac{2}{p}\right)\cdot\left(1-\frac{2a}{p^n-p}\right) + \\ &\quad \frac{1}{\sqrt{1-\lambda}}\cdot\frac{4\delta}{p^n-p} \\ &= \frac{1}{p^{(n-1)/2}}\left(1 - \frac{2a}{p^n-p}\right) + \\ &\quad \frac{2\delta}{\sqrt{p}(p^{n-1}-1)}\cdot\sqrt{\frac{1-1/p^{n-1}}{1-4/p}} \\ &\geq \frac{2\delta}{\sqrt{p}(p^{n-1}-1)} - \frac{1}{p^{(n-1)/2}}\end{aligned}$$

Now we substitute in $\xi = 1/p^{n/3}$, and we have

$$|\langle g, \hat{L}_{\mathbf{a}}\rangle| \geq \frac{\xi}{\sqrt{p}} - \frac{1}{p^{(n-1)/2}} \geq \frac{1}{p^{n/3+1}} \qquad (12)$$

Thus g can have at most $p^{2n/3+2}$ such Fourier coefficients, and so there can be at most $p^{2n/3+2}$ predicates that are not $1/p^{n/3}$-independent from g. ■

Lemma 7 *Let $X_n = \{0,1\}^n \backslash \{0^n\}$ and $\mathcal{C}_n$ be the class of negative parity functions over X_n. For any query function $g : \{0,1\}^n \mapsto \{-1,+1\}$, there are at most $2^{n/2+2}$ predicates in $\mathcal{C}_n$ that are not $2^{-n/4}$-independent from g.*

Proof: We fix a negative parity function f. Let a denote the number of $x \in \{0,1\}^n$ such that $g(x) = 1$, and let b denote the number of $x \in S_f$ such that $g(x) = 1$. Notice that since all parity functions are balanced, we have $|S_f| = 2^{n-1} - 1$ (since $f(0^n) = 1$ but $0^n \notin S_f$). Then if f is not ξ-independent from g, we have

$$\left|\frac{2b - 2^{n-1}+1}{2^{n-1}-1} - \frac{2a - 2^n}{2^n}\right| > \xi \qquad (13)$$

or

$$\left|\frac{a-2b}{2^{n-1}-1}\right| > \xi - \frac{a}{2^{n-1}(2^{n-1}-1)} > \xi - \frac{1}{2^{n-1}-1} \quad (14)$$

Next we perform Fourier analysis. We first define an inner product of real functions over $\{0,1\}^n$:

$$\langle f, g\rangle = \frac{1}{2^n} \sum_{x\in\{0,1\}^n} f(x)g(x). \tag{15}$$

We define a set of "modified parity functions" as $\tilde{\oplus}_s(x) = (-1)^{s\cdot x}$, which map elements in $\{0,1\}^n$ to $\{-1,+1\}$. It is clear that the set of all parity functions $\{\tilde{\oplus}_s(x)\}_s$ form an orthonormal basis, and $\tilde{\oplus}_s(x) = 1 - 2\neg \oplus_s (x)$. If a parity function $\neg \oplus_s (x)$ is not ξ-independent from g, then (13) holds (by setting $f = \neg\oplus_s$). Let $t = g(0^n)$. Within the subset where $\tilde{\oplus}_s(x) = -1$, which includes 0^n and the positive set of $\neg\oplus_s$, g maps $b+t$ inputs to $+1$. Outside this subset, g maps $a-b-t$ inputs to $+1$, and $2^{n-1} - a + b + t$ input to -1. Thus, we can compute the Fourier coefficient of g on $\tilde{\oplus}_s$.

$$\begin{aligned}
\langle\tilde{\oplus}_s, g\rangle &= 1 - 2\cdot \Pr_{x\in\{0,1\}^n}[\tilde{\oplus}_s(x) = g(x)] \\
&= 1 - 2\cdot\left(\frac{a-b-t}{2^n} + \frac{2^{n-1} - a + b + t}{2^n}\right) \\
&= \frac{2a - 4b - 4t}{2^n}
\end{aligned}$$

Substituting in (14), we have

$$|\langle\tilde{\oplus}_s, g\rangle| > \xi - 6/2^n. \tag{16}$$

However, notice that the query function $g(x)$ has norm 1 and thus it can have at most $1/(\xi - 6/2^n)^2$ Fourier coefficients such that (16) holds. Now plugging in $\xi = 2^{-n/4}$, we have $1/(\xi - 6/2^n)^2 \leq 2^{n/2+2}$, and the Lemma is proved.

■