Quantum Property Testing‡

Harry Buhrman§ Lance Fortnow¶ Ilan Newman‖ Hein Röhrig§

November 12, 2001

Abstract

A language L has a property tester if there exists a probabilistic algorithm that given an input x only asks a small number of bits of x and distinguishes the cases as to whether x is in L and x has large Hamming distance from all y in L. We define a similar notion of quantum property testing and show that there exist languages with quantum property testers but no good classical testers. We also show there exist languages which require a large number of queries even for quantumly testing.

1 Introduction

Suppose we have a large data set, for example, a large chunk of the world-wide web or a genomic sequence. We would like to test whether the data has a certain property, but we may not have the time to even look at the entire data set or even a large portion of it.

To handle these types of problems, Rubinfeld and Sudan [RS96] and Goldreich, Goldwasser and Ron [GGR98] have developed the notion of property testing. Testable properties come in many varieties including graph properties (e.g., [GGR98, AFKS99, Fis01a, Fis01b, Alo01, GT01]), algebraic properties of functions [BLR93, RS96, EKK+00] and regular languages [ANKS99]. Ron [Ron00] gives a nice survey of this area.

In this model, the property tester has random access to the n input bits similar to the black-box oracle model. The tester can query only a small, usually some fixed constant, probabilistically-chosen set of bits of the input. Clearly we cannot determine from these small number of bits whether the input sits in some language L. However, for many languages we can distinguish the cases that the input is in L from whether the input differs from all inputs in L of the same length by some constant fraction of input bits.

Since we have seen many examples where quantum computation gives us an advantage over classical computation [BV97, Sim97, Sho97, Gro96] one may naturally ask whether using quantum computation may lead to better property testers. By using the quantum oracle-query model developed by Beals, Buhrman, Cleve, Mosca and de Wolf [BBC+98], we can easily extend the definitions of property testing to the quantum setting.

Beals, et. al. [BBC+98] have shown that for all total functions we have a polynomial relationship between the number of queries required by quantum machine and that needed by a deterministic machine. For greater separations one needs to require a promise in the input and the known examples, such as those due to Simon [Sim97] and Bernstein-Vazirani [BV97], require considerable structure in the promise. In property testing there is a natural promise of either being in the language or far from

‡Research done while all authors were visiting the NEC Research Institute.
§CWI and University of Amsterdam; partially supported by the EU fifth framework project QAIP, IST-1999-11234.
¶NEC Research Institute
‖Haifa University and NEC Research Institute

any input in the language. This promise would seem to have too little structure to give a separation but in fact we can prove that quantum property testing can greatly improve on classical testing.

We show that any subset of Hadamard codes has a quantum property tester and most subsets would require $\Theta(\log n)$ queries to test with a probabilistic tester. This shows that indeed quantum property testers are more powerful than classical testers. Moreover, we also give an example of a language where the quantum tester is exponentially more efficient.

Beals, et. al. [BBC+98] observed that any k-query quantum algorithm gives rise to a degree-$2k$ polynomial in the input bits, which gives the acceptance probability of the algorithm; thus, a quantum property tester for P gives rise to a polynomial that is on all binary inputs between 0 and 1, that is at least 2/3 on inputs with the property P and at most 1/3 on inputs far from having the property P. Szegedy [Sze99] suggested to algebraically characterize the complexity of classical testing by the minimum degree of such polynomials; however, our separation results imply that there are for example properties, for which such polynomials have constant degree, but for which the best classical tester needs $\Omega(\log n)$ queries.

Perhaps every language has a quantum property tester with a small number of queries. We show that this is not the case. We prove that for most properties of a certain size, any quantum algorithm requires $\Omega(n)$ queries. We then show that a natural property, namely, the range of a d-wise independent pseudorandom generator cannot be quantumly tested with less than $(d+1)/2$ queries for any odd $d \leq n/\log n - 1$.

2 Preliminaries

Property testing was first developed by Rubinfeld and Sudan [RS96] and Goldreich, Goldwasser and Ron [GGR98]. We will use the following formal definition of property testing from Goldreich [Gol98].

Definition 2.1 *Let S be a finite set, and P a subset of functions mapping S to $\{0,1\}$. A* property tester *for P is a probabilistic oracle machine M, which given a distance parameter $\epsilon > 0$ and oracle access to an arbitrary function $f : S \to \{0,1\}$, satisfies the following conditions:*

1. *the tester accepts f if it is in P:*

$$\text{if } f \in P \text{ then } \Pr(M^f(\epsilon) = 1) \geq \frac{2}{3}$$

2. *the tester rejects f if it is far from P:*

$$\text{if } |\{x \in S : f(x) \neq g(x)\}| > \epsilon \cdot |S|, \text{ for every } g \in P, \text{ then } \Pr(M^f(\epsilon) = 1) \leq \frac{1}{3}$$

Definition 2.2 *The complexity of the tester is the number of oracle queries it makes: A property P has a (ϵ, q)-*tester *if there is a tester for P that makes at most q oracle queries for distance parameter ϵ.*

We often consider a language $L \subseteq \{0,1\}^*$ as the family of properties $\{P_n\}$ with P_n the characteristic functions of the length-n strings from L, and analyze the query complexity $q = q(\epsilon, n)$ asymptotically for large n.

To define quantum property testing we simply modify Definition 2.1 by allowing M to be a quantum oracle machine. We need to be careful to make sure our oracle queries are unitary operations. If $|f(x)| = |g(y)|$ for all $x, y \in S$ and $f, g \in P$, we use the oracle-query model by Beals, Buhrman, Cleve, Mosca and de Wolf [BBC+98]: we define the unitary transformation O_f that maps the basis state $|x, y, z\rangle$ to $|x, y \oplus f(x), z\rangle$ where $|x| = \lceil \log |S| \rceil$, $|y| = |f(x)|$ and $\oplus$ denotes bitwise exclusive or. In case there are x, y, f, g so that $|f(x)| \neq |g(y)|$, we define O_f as mapping $|x, l, y, z\rangle$ to $|x, l + |f(x)| \mod k, y \oplus 0^{k-|f(x)|} f(x), z\rangle$ where $k = \max\{|f(x)| : f \in P \text{ and } x \in S\}$, $|x| = \lceil \log |S| \rceil$, $|l| = \lceil \log k \rceil$, and $|y| = k$.

We recommend the book of Nielsen and Chuang [NC00] for background information on quantum computing and the surveys of Ron [Ron00] and Goldreich [Gol98] for a background on property testing.

3 Separating Quantum and Classical Property Testing

We show that there exist languages with $(\epsilon, \mathrm{O}(1))$ quantum property testers that do not have $(\epsilon, \mathrm{O}(1))$ classical testers.

Theorem 3.1 *There is a languages L that is ϵ-testable by a quantum test with $O(1/\epsilon)$ number of queries but for which any probabilistic $1/3$-test requires $\Omega(\log n)$ queries.*

We use Hadamard codes to provide examples for Theorem 3.1:

Definition 3.2 *The* Hadamard code *of $y \in \{0,1\}^{\log n}$ is $x = h(y) \in \{0,1\}^n$ such that $x_i = y \cdot i$ where $y \cdot i$ denotes the inner product of two vectors $y, i \in \mathbb{F}_2^{\log n}$.*

Note: the Hadamard mapping $h : \{0,1\}^{\log n} \to \{0,1\}^n$ is one-to-one. Bernstein and Vazirani [BV97] showed that a quantum computer can extract y with one query to an oracle for (the bits of) x, whereas a classical probabilistic procedure needs $\Omega(\log n)$ queries. Based on this separation for a decision problem we construct for $A \subseteq \{0,1\}^{\log n}$ the property $P_A \subseteq \{0,1\}^n$,

$$P_A := \{x : \exists y \in A \text{ s.t. } x = h(y)\}.$$

Theorem 3.1 follows from the following two lemmas.

Lemma 3.3 *For every A, P_A can be $(\epsilon, O(1/\epsilon))$-quantumly tested. Furthermore, the test has one-sided error.*

Lemma 3.4 *For most A of size $|A| = n/2$, P_A requires $\Omega(\log n)$ queries for a probabilistic $1/3$-test (even for testers with two-sided error.)*

Before we prove Lemma 3.3 we note that for any A, P_A can be tested by an (even nonadaptive) one-sided algorithm with $O(1/\epsilon + \log n)$; hence, the result of Lemma 3.4 is tight. An $O(1/\epsilon \log n)$-test follows from Observation 3.5 below. The slightly more efficient test is the following: First we query x_{2^i}, $i = 0, \ldots, \log n$. Note that if $x = h(y)$ then $y_i = x_{2^i}$ for $i = 0, \ldots, \log n$. Thus a candidate y for $x = h(y)$ is found. If $y \notin A$ then x is rejected. Then for $k = \mathrm{O}(1/\epsilon)$ times independently a random index $i \in \{1, \ldots, n\}$ is chosen and if $x_i \neq y \cdot i$, for the candidate y that was found before, then x is rejected. Otherwise, x is accepted. Clearly if x is rejected then $x \notin P_A$. It is easily verified that if x has Hamming distance more than ϵn from every z in P_A then with constant probability x is rejected.

Proof of Lemma 3.3. P_A can be checked with $\mathrm{O}(1/\epsilon)$ queries on a quantum computer: The test is similar to the test above except that y can be found in $O(1)$ queries: k times query for random i, j values x_i, x_j, and $x_{i \oplus j}$. If $x_i \oplus x_j \neq x_{i \oplus j}$ reject. $k = \mathrm{O}(1/\epsilon)$ is sufficient to detect an input x that is ϵn-far from being a Hadamard codeword with high probability. Now run the Bernstein-Vazirani algorithm to obtain y. Accept if and only if $y \in A$. Obviously, if $x \in P_A$, the given procedure accepts, and if x is far from any $x' \in P_A$, then it is either far from being a Hadamard codeword (which is detected in the first step) or it is close to a Hadamard codeword $h(y')$ for an $y' \notin A$; note that in this case x is far from any $h(y)$, $y \in A$ as two distinct Hadamard codewords are of Hamming distance $n/2$. Thus, in this case the second part of the tester succeeds with high probability in finding y' and rejects because $y' \notin A$. We note also that this algorithm has one-sided error. □

Proof of Lemma 3.4. The lower bound makes use of the Yao principle [Yao77]: let D be an arbitrary probability distribution on positive and negative inputs (i.e., inputs that either belong to P_A or are ϵn-far from P_A). Then if every deterministic algorithm that makes at most q queries, errs with probability at least $1/8$ (with respect to input chosen according to D,) then q is a lower bound on the number of queries of any randomized algorithm for testing P_A with error probability bounded by $1/8$.

D will be the uniform distribution over Hadamard codewords of length n, namely, generated by choosing $y \in \{0,1\}^{\log n}$ uniformly at random and setting $x = h(y)$. Note that for any $A \subset \{0,1\}^{\log n}$, D is concentrated on positive and negative inputs as required, as two Hadamard codewords are of Hamming distance $n/2$ apart.

The lower bound will be established by a counting argument. We show that for a fixed tester that makes $q \leq (\log n)/2$ queries, the probability (over random choices of A) that the the algorithm errs on at most a $1/8$-fraction of the inputs is bounded from above by $1/(10T)$ where T is the number of such algorithms. By the union bound it follows that for most properties there is no such algorithm.

Indeed, let $A \subseteq \{0,1\}^{\log n}$ be chosen by picking independently each $i \in \{0,1\}^{\log n}$ to be in A with probability $1/2$; this will not necessarily result in a set A of size $n/2$ but we can condition on the event that $|A| = n/2$ and will not loose much. Let $\mathcal{T}$ be any fixed deterministic decision tree performing at most q queries in every branch. Then let $c(\mathcal{T}) := \{y|\mathcal{T}(h(y)) = \text{accept}\}$ and let $\mu(\mathcal{T}) := |c(\mathcal{T})|/n$, i.e., $\mu(\mathcal{T})$ is the fraction of inputs that $\mathcal{T}$ accepts. Assume first that $\mu(\mathcal{T}) \leq 1/2$. Since for a random y we have $\Pr_y[\mathcal{T}(h(y)) = \text{accept}] = \mu(\mathcal{T}) \leq 1/2$, it follows by a Chernoff-type bound that $\Pr_A[|A \cap c(\mathcal{T})| \geq 3/4|A|] \leq 2^{-n/8}$. However, if $|A \cap c(\mathcal{T})| < 3/4|A|$ then $\mathcal{T}$ will be wrong on at least $1/4$ of the positive inputs which is at least $n/8$ of all inputs. Hence with probability at most $2^{-n/8}$, $\mathcal{T}$ will be correct on at least $7/8$ of the inputs. If $\mu(\mathcal{T}) > 1/2$ the same reasoning shows that with probability of at most $1 - 2^{-n/8}$ it will err on at least an $1/4$-fraction of the negative inputs. Hence in total, for every fixed $\mathcal{T}$, $\Pr_A[\mathcal{T}$ is correct on at least $7/8$ of the inputs$] \leq 2^{-n/8}$.

Now, let us bound from above the number of algorithms that make at most q queries. As an algorithm may be adaptive, it can be defined by $2^q - 1$ query positions (all queries for all branches) and a Boolean function $f : \{0,1\}^q \to \{\text{accept}, \text{reject}\}$ of the decision made by the algorithm for the possible answers. Hence, there are at most $T \leq (2n)^{2^q}$ such algorithms. However, for $q < (\log n)/2$, we have $T \cdot 2^{-n/8} = \mathrm{o}(1)$, which shows that for most A as above, any ϵ-test that queries at most $(\log n)/2$ many queries has error probability of at least $1/8$. Standard amplification techniques then imply that any algorithm that queries $c \log n$ many queries (for some constant c) has error at least $1/3$. □

Observation 3.5 *Let $P \subseteq \{0,1\}^n$ be a property with $|P| = s$ then P can be ϵ-tested by a one-sided algorithm using $O((\log s)/\epsilon)$ many queries.*

We omit the proof from this draft.

4 An Exponential Separation

In this section, we show that a quantum computer can be exponentially more efficient in testing certain properties than a classical computer.

Theorem 4.1 *There is a languages L that is $(\epsilon, \log n \log\log n)$-quantumly testable for any $\epsilon = \Omega(1)$. However, any probabilistic $1/8$-test for L requires $n^{\Omega(1)}$ queries.*

The language that we provide is inspired by Simon's problem [Sim97] and our quantum testing algorithm makes use of Brassard and Høyer's algorithm for Simon's problem [BH97]. Simon's problem is to find $s \in \{0,1\}^n \setminus \{0^n\}$ from an function-query oracle for some $f : \{0,1\}^n \to \{0,1\}^n$, such that $f(x) = f(y) \Leftrightarrow x = y \oplus s$. Simon proved that classically, $\Omega(2^{n/2})$ queries are required on average to find s, and gave a quantum algorithm for determining s with an expected number of queries that is polynomial in n; Brassard and Høyer improved the algorithm to worst-case polynomial time. Their algorithm produces in each run a z with $z \cdot s = 0$ that is linearly independent to all previously computed such zs. Essentially, our quantum tester uses this subroutine to try to extract information about s until it fails repeatedly. Recently Høyer [Høy01] analyzed this approach in group-theoretic terms, obtaining an alternative proof to Theorem 4.3.

In the following, let $N = 2^n$ denote the length of the binary string encoding a function $f : \{0,1\}^n \to \{0,1\}$. For $x \in \{0,1\}^n$ let $x[j]$ be the jth bit of x, i.e., $x = x[1]\dots x[n]$; the inner product of $x, y \in \{0,1\}^n$ as vectors in $\mathbb{F}_2^n$ is $x \cdot y = \sum_{j=1}^n x[j]y[j] \mod 2$. We define

$$L := \{f \in \{0,1\}^N : \exists s \in \{0,1\}^n \setminus \{0^n\}\ \forall x \in \{0,1\}^n\ f(x) = f(x \oplus s)\}$$

Theorem 4.1 follows from the following two theorems.

Theorem 4.2 *Any classical 1/8-tester for L must make $\Omega(\sqrt{N})$ queries (even when allowing two-sided error.)*

Theorem 4.3 *There is a quantum property tester for L making $O(\log N \log\log N)$ queries. Moreover, this quantum property tester makes all its queries nonadaptively.*

Proof of Theorem 4.2. We again apply the Yao principle [Yao77] as in the proof of Lemma 3.4: we construct two distributions, P and U, on positive and negative (at least $N/8$-far) inputs, respectively, such that any deterministic (adaptive) decision tree $\mathcal{T}$ has error $1/2 - o(1)$ when trying to distinguish whether an input is chosen from U or P. Indeed, we will show a stronger statement: Let $\mathcal{T}$ be any deterministic decision tree. Let v be a vertex of $\mathcal{T}$. Let $\Pr_P(v)$ and $\Pr_U(v)$ be the probability the an input chosen according to P and U, respectively, is consistent with v. We will show that for any vertex v of $\mathcal{T}$ $|\Pr_P(v) - \Pr_U(v)| = o(1)$; hence, $\mathcal{T}$ has error $1/2 - o(1)$.

The distribution P is defined as follows: We first chose $s \in \{0,1\}^n$ at random. This defines a matching M_s of $\{0,1\}^n$ by matching x with $x \oplus s$. Now a function f_s is defined by choosing for each matched pair independently $f_s(x) = f_s(x \oplus s) = 1$ with probability 1/2 and $f_s(x) = f_s(x \oplus s) = 0$ with probability 1/2. Clearly, this defines a distribution that is concentrated on positive inputs. Note that it might be that by choosing different s's we end up choosing the same function, however, this will be considered different events in the probability space. Namely, the atomic events in P really are the pairs (s, f_s) as described above.

Now let U be the uniform distribution over all functions, namely, we select the function by choosing for each x independently $f(x) = 1$ with probability 1/2 and 0 with probability 1/2. Since every function has a non zero probability, U is not supported exclusively on the negative instances. However, as we proceed to show, a function chosen according to U is $N/8$-far from having the property with very high probability, and hence U will be a good approximation to the desired distribution:

Definition 4.4 *For $f : \{0,1\}^n \to \{0,1\}$ and $s \in \{0,1\}^n$ we define $n_s := |\{x : f(x) = f(x \oplus s)\}|$.*

Lemma 4.5 *Let f be chosen according to U. Then $\Pr_U[\exists s \in \{0,1\}^n : n_s \geq N/8] \leq \exp(-\Omega(N))$.*

Proof. Let f be chosen according to U and $s \in \{0,1\}^n$. By a Chernoff bound we obtain $\Pr_U[n_s \geq N/8] \leq \exp(-\Omega(N))$. Together with the union bound over all s's this yields $\Pr_U[\exists s \in \{0,1\}^n : n_s \geq N/8] \leq 2^n \cdot \exp(-\Omega(N)) \leq \exp(-\Omega(N))$. □

In particular, a direct consequence of Lemma 4.5 is that with probability $1 - \exp(-\Omega(N))$ an input chosen according to U will be $N/8$-far from having the property.

From the definition of U, we immediately obtain the following:

Lemma 4.6 *Let $\mathcal{T}$ be any fixed deterministic decision tree and let v be a vertex of depth d in $\mathcal{T}$. Then* $\Pr_U[f \text{ is consistent with the path to } v] = 2^{-d}$.

We now want to derive a similar bound as in the last lemma for functions chosen according to P. For this we need the following definition for the event that after d queries, nothing has been learned about the hidden s:

Definition 4.7 *Let $\mathcal{T}$ be a deterministic decision tree and u a vertex in $\mathcal{T}$ at depth d. We denote the path from the root of $\mathcal{T}$ to u by* $\mathrm{path}(u)$. *Any vertex v in $\mathcal{T}$ defines a query position $x_v \in \{0,1\}^n$. For $f = f_s$ chosen according to P, we denote by B_u the event $B_u := \{(s, f_s) : s \neq x_v \oplus x_w$ for all $v, w \in \mathrm{path}(u)\}$.*

Lemma 4.8 *Let v be a vertex of depth d in a decision tree $\mathcal{T}$. Then* $\Pr_P[B_v] \geq 1 - \binom{d-1}{2}/N$

Proof. B_v does not occur if for some v, w on the path to v we have $s = x_v \oplus x_w$. As there are $d-1$ such vertices, there are at most $\binom{d-1}{2}$ pairs. Each of these pair excludes exactly one s and there are N possible s's. □

Lemma 4.9 *Let v be a vertex of depth d in a decision tree $\mathcal{T}$ and let f be chosen according to P. Then* $\Pr_P[f \text{ is consistent with } v | B_v] = 2^{-d}$.

Proof. By the definition of P, f gets independently random values on vertices that are not matched. But if B_v occurs, then no two vertices along the path to v are matched and hence the claim follows. □

Now we can complete the proof of the theorem: assume that $\mathcal{T}$ is a deterministic decision tree of depth $d = \mathrm{o}(\sqrt{N})$ and let v be any leaf of $\mathcal{T}$. Then by Lemmas 4.8 and 4.9, we get that $\Pr_P[f \text{ is consistent with } v] = (1 - \mathrm{o}(1))2^{-d}$. On the other hand, let U' be the distribution on negative inputs defined by U conditioned on the event that the input is at least $N/8$-far from the property. Then by Lemmas 4.5 and 4.6 we get that $\Pr_{U'}[f \text{ is consistent with } v] = (1 - \mathrm{o}(1))2^{-d}$ and hence $\mathcal{T}$ has only $\mathrm{o}(1)$ bias of being right on every leaf which directly imply that its error probability is $1/2 - \mathrm{o}(1)$. □

Proof of Theorem 4.3. We give a quantum algorithm making $\mathrm{O}(\log N \log\log N)$ queries to the quantum oracle for input $f \in \{0,1\}^N$. We will show that it accepts with probability 1 if $f \in L$ and rejects with high probability if the Hamming distance between f and every $g \in L$ is at least ϵN. Our algorithm, given in Figure 1, consists of a classical main program and a quantum subroutine Q adapted from Brassard and Høyer's algorithm for Simon's problem [BH97, Section 4]. The quantum gates used are the 2^n-dimensional Hadamard transform H_{2^n}, which applies $\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}/\sqrt{2}$ individually to each of n qubits, the quantum oracle query O_f, and classical reversible operations run in quantum superposition.

The following technical lemma captures the operation of the quantum subroutine Q. Essential proofs to this and later lemmas are included in the appendix.

Main Program
for $k = 0$ to $n-1$ **do**
  $l \leftarrow 0$
  repeat
    $z \leftarrow Q(z_1, \dots, z_k)$
    $l \leftarrow l + 1$
  until $z \neq 0$ or $l > 2(\log n)/\epsilon^2$
  if $z = 0$ **then**
    accept
  else
    $z_{k+1} \leftarrow z$
reject

Subroutine Q
input: $z_1, \dots, z_k \in \{0,1\}^n$
output: $z \in \{0,1\}^n$
quantum workspace: $\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z}$
(where $\mathcal{X}$ is n qubits $\mathcal{X} = \mathcal{X}_1 \otimes \cdots \otimes \mathcal{X}_n$, $\mathcal{X}_i = \mathbb{C}^2$,
$\mathcal{Y} = \mathbb{C}^2$ is one qubit, and
$\mathcal{Z}$ is k qubits $\mathcal{Z} = \mathcal{Z}_1 \otimes \cdots \otimes \mathcal{Z}_k$, $\mathcal{Z}_j = \mathbb{C}^2$)
initialize the workspace to $|0^n\rangle|0\rangle|0^k\rangle$
apply H_{2^n} to $\mathcal{X}$
apply O_f to $\mathcal{X} \otimes \mathcal{Y}$
apply H_{2^n} to $\mathcal{X}$
for $j = 1$ to k **do**
  $i \leftarrow \min\{i : z_j[i] = 1\}$
  apply CNOT with control $\mathcal{X}_i$ and target $\mathcal{Z}_j$
  apply $|x\rangle \mapsto |x \oplus z_j\rangle$ to $\mathcal{X}$ conditional on $\mathcal{Z}_j$
  apply H_2 to $\mathcal{Z}_j$
return measurement of $\mathcal{X}$

Figure 1: The Quantum Property Tester

Lemma 4.10 *When Q is passed k linearly independent vectors $z_1, \dots, z_k$ so that all $i_j := \min\{i : z_j[i] = 1\}$ are distinct for $1 \leq j \leq k$, then the state before the measurement is*

$$|\psi\rangle := \frac{\sqrt{2^k}}{N} \sum_{x \in \{0,1\}^n} \sum_{\substack{y \in \{0,1\}^n \\ y[i_j]=0\ \forall j \leq k}} (-1)^{x \cdot y} |y\rangle |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_k\rangle.$$

As an immediate consequence, we can establish the invariant that in the main program $\{z_1, \dots, z_k\}$ always is linearly independent with $i_j = \min\{i : z_j[i] = 1\}$ distinct for $1 \leq j \leq k$; moreover, if $f \in L$, then just as in Simon's algorithm, a nonzero z is orthogonal to the hidden s:

Lemma 4.11 *If measuring the first register, $\mathcal{X}$, yields a nonzero value z, then*

1. *$\{z_1, \dots, z_k, z\}$ is linearly independent,*
2. *$\min\{i : z[i] = 1\}$ is distinct from i_j for $1 \leq j \leq k$, and*
3. *if $f \in L$, then $z \cdot s = 0$ for any s such that $f(x) = f(x \oplus s)$ for all x.*

Next, we want to assess the probability of obtaining $z = 0$ in the main loop. We let P_0 denote the projection operator mapping $|0\rangle|y\rangle|z\rangle \mapsto |0\rangle|y\rangle|z\rangle$ and $|x\rangle|y\rangle|z\rangle \mapsto 0$ for $x \neq 0$; hence, $\|P_0|\psi\rangle\|^2$ is the probability of obtaining 0 when measuring subspace $\mathcal{X}$ of the quantum register in state $|\psi\rangle$. We can characterize the probability for outcome $z = 0$ in terms of the following definition and lemma:

Definition 4.12 *For $c \in \{0,1\}^k$ and $z_1, \dots, z_k \in \{0,1\}^n$ we define $D_c := \{x \in \{0,1\}^n : x \cdot z_1 = c[1], \dots, x \cdot z_k = c[k]\}$.*

Lemma 4.13 *Let $|\psi\rangle$ be the state before the measurement in Q, when Q is passed k linearly independent vectors $z_1, \dots, z_k$ so that all $i_j := \min\{i : z_j[i] = 1\}$ are distinct for $1 \leq j \leq k$.*

1. *$\|P_0|\psi\rangle\|^2 = 1$ if and only if for every $c \in \{0,1\}^k$, f is constant when restricted to D_c.*

2. If $\|P_0|\psi\rangle\|^2 \geq 1-\epsilon^2/2$, then f differs in at most ϵN points from some function g that is constant when restricted to D_c for every $c \in \{0,1\}^k$.

We need to relate these two cases to membership in L and bound the number of repetitions needed to distinguish between the two cases. This is achieved by the following two lemmas.

Lemma 4.14 *Let k be the minimum number of linearly independent vectors $z_1, \ldots, z_k$ so that for each $c \in \{0,1\}^k$, f is constant when restricted to D_c. Then $f \in L$ if and only if $k < n$.*

Lemma 4.15 *Let $0 < q < 1$, and $|\phi_1\rangle, \ldots, |\phi_m\rangle$ be quantum states satisfying $\|P_0|\phi_j\rangle\|^2 < 1-\delta$ for $1 \leq j \leq m$. If $m = \log q/\log(1-\delta) = \Theta(-\log q/\delta)$, then with probability at most q measuring the $\mathcal{X}$ register of $|\phi_1\rangle, \ldots, |\phi_m\rangle$ will yield m times outcome 0.*

Proof. $\Pr\left[m \text{ times } 0 \,\middle|\, \forall j : \|P_0|\phi_j\rangle\|^2 < 1-\delta\right] < (1-\delta)^m = (1-\delta)^{\log q/\log(1-\delta)} = q.$ □

Now all the ingredients for wrapping up the argument are at hand; first consider $f \in L$. Let $S := \{s : f(x) = f(x \oplus s)\ \forall x\}$ be the set of all "Simon promises" of f, and $S^\perp := \{z : z \cdot s = 0\ \forall s \in S\}$ the vectors that are orthogonal to all such promises. By Lemma 4.11 the nonzero z computed by the algorithm lie in $S^\perp$ and are linearly independent, therefore after $\dim S^\perp$ rounds of the main loop, we measure $z = 0$ with certainty. Since $f \in L$, $\dim S > 0$ and thus $\dim S^\perp < n$.

If f is ϵn-far from being in L, then by Lemma 4.14 f is ϵn-far from being close to a function for which a $k < n$ and $z_1, \ldots, z_k$ exist so that f is constant when restricted to D_c for any of the $c \in \{0,1\}^k$. Therefore, by Lemma 4.13 case 2, for all $k < n$, $\|P_0|\psi\rangle\|^2 < 1-\epsilon^2/2$. Thus, Lemma 4.15 guarantees that we accept with probability at most $1/3$ if we let $q = 1/(3n)$ and thus $m \leq 2(\log n)/\epsilon^2$. □

5 Quantum Lower Bounds

In this section we prove that not every language has a quantum property tester.

Theorem 5.1 *Most properties containing $2^{n/20}$ elements of $\{0,1\}^n$ require quantum property testers using $\Omega(n)$ queries.*

Proof. Fix n, a small ϵ, and a quantum algorithm A making $q := n/400$ queries. Pick a property P as a random subset of $\{0,1\}^n$ of size $2^{n/20}$. Let $P_\epsilon := \{y : d(x,y) < \epsilon n \text{ for some } x \in P\}$; using $\sum_{k=0}^{\epsilon n}\binom{n}{k} \leq 2^{H(\epsilon)n}$ where $H(\epsilon) = -\epsilon\log\epsilon - (1-\epsilon)\log(1-\epsilon)$, we obtain $|P_\epsilon| \leq 2^{(1/20+H(\epsilon))n}$. In order for A to test properties of size $2^{n/20}$, it needs to reject with high probability on at least $2^n - 2^{(1/20+H(\epsilon))n}$ inputs; but then, the probability that A accepts with high probability on a random $x \in \{0,1\}^n$ is bounded by $2^{(1/20+H(\epsilon))n}/2^n$ and therefore the probability that A accepts with high probability on $|P|$ random inputs is bounded by $2^{-(1-1/20-H(\epsilon))n|P|} = 2^{-2^{n/20+\Theta(\log n)}}$.

We would like to sum this success probability over all algorithms using the union bound to argue that for most properties no algorithm can succeed. However, there is an uncountable number of possible quantum algorithms with arbitrary quantum transitions. But by Beals, Buhrman, Cleve, Mosca, and de Wolf [BBC+98], the acceptance probability of A can be written as a multilinear polynomial of degree at most $2q$ where the n variables are the bits of the input; using results of Bennett, Bernstein, Brassard, and Vazirani [BBBV97] and Solovay and Yao [SY96], any quantum algorithm can be approximated by another algorithm such that the coefficients of the polynomials describing the accepting probability are integers of absolute value less than $2^{n^{O(1)}}$ over some fixed denominator. There are less than $2^{nH(2q/n)}$ degree $2q$ polynomials in n variables, thus we can limit ourselves to $2^{n^{O(1)}2^{nH(2q/n)}} \leq 2^{2^{n/20\cdot 91/100+\Theta(\log n)}}$ algorithms.

Thus, by the union bound, for most properties of size $2^{n/20}$, no quantum algorithm with q queries will be a tester for it. □

We also give an explicit natural property that requires a large number of quantum queries to test.

Theorem 5.2 *The range of a d-wise independent pseudorandom generator requires $(d+1)/2$ quantum queries to test for any odd $d \leq n/\log n - 1$.*

We will make use of the following lemma:

Lemma 5.3 (see [ABI86]) *Suppose $n = 2^k - 1$ and $d = 2t + 1 \leq n$. Then there exists a uniform probability space Ω of size $2(n+1)^t$ and d-wise independent random variables $\xi_1, \ldots, \xi_n$ over Ω each of which takes the values 0 and 1 with probability $1/2$.*

The proof of Lemma 5.3 is constructive and the construction uniform in n; for given n and d, consider the language P of bit strings $\xi(z) := \xi_1(z) \ldots \xi_n(z)$ for all events $z \in \Omega = \{1, \ldots, 2(n+1)^t\}$. Classically, deciding membership in P takes more than d queries: for all d positions $i_1, \ldots, i_d$ and all strings $v_1 \ldots v_d \in \{0,1\}^d$ there is an z such that $\xi_{i_1}(z) \ldots \xi_{i_d}(z) = v_1 \ldots v_d$. On the other hand, $\lfloor \log |\Omega| \rfloor + 1 = \mathrm{O}(d \log n)$ queries are always sufficient.

Proof of Theorem 5.2. A quantum computer deciding membership for $x \in \{0,1\}^n$ in $P := \{\xi(z) : z \in \Omega\}$ with T queries gives rise to a degree $2T$ approximating (multilinear n-variable) polynomial $p(x) = p(x_1, \ldots, x_n)$ (see [BBC+98].) We show that there must be high-degree monomials in p by comparing the expectation of $p(x)$ for randomly chosen $x \in \{0,1\}^n$ with the expectation of $p(x)$ for randomly chosen $x \in P$.

For uniformly distributed $x \in \{0,1\}^n$, we have $\mathrm{E}[p(x)|x \in P] \geq 2/3$ and $\mathrm{E}[p(x)|x \notin P] \leq 1/3$. Since of $|P| = \mathrm{o}(2^n)$, $\mathrm{E}[p(x)] \leq 1/3 + \mathrm{o}(1)$ and thus $\Delta := \mathrm{E}[p(x)|x \in P] - \mathrm{E}[p(x)] \geq 1/3 - \mathrm{o}(1)$. Considering $p(x) = \sum_i \alpha_i m_i(x)$ as a linear combination of n-variable multilinear monomials m_i, we have by the linearity of expectation $\mathrm{E}[p(x_1, \ldots, x_n)] = \sum_i \alpha_i \mathrm{E}[m_i(x_1, \ldots, x_n)]$. Because of the d-wise independence of the bits of any $x \in P$, for any m_i of degree at most d holds $\mathrm{E}[m_i(x)] = \mathrm{E}[m_i(x)|x \in P]$. Since $\Delta > 0$, p must comprise monomials of degree greater than d. Hence, the number of queries T is greater than $d/2$.

This proof extends in a straightforward manner to the case of testing the property P: let again $P_\epsilon := \{y : d(x,y) < \epsilon n \text{ for some } x \in P\}$. Then $|P_\epsilon| \leq 2^{H(\epsilon)n}|P| = \mathrm{O}(2^{H(\epsilon)n + d\log n})$, so $\mathrm{E}[p(x)] = |P_\epsilon|/2^n \, \mathrm{E}[p(x)|x \in P_\epsilon] + (1 - |P_\epsilon|)/2^n \, \mathrm{E}[p(x)|x \notin P_\epsilon] \leq 1/3 + \mathrm{o}(1)$ for any $d = n/\log n - \omega(1/\log n)$ and any ϵ with $H(\epsilon) = 1 - \omega(1/n)$. □

6 Further Research

Our paper opens the door to the world of quantum property testing. Several interesting problems remain including

- Can one get the greatest possible separation of quantum and classical property testing, i.e., is there a language that requires $\Omega(n)$ classical queries but only $\mathrm{O}(1)$ quantum queries to test?
- Are there other natural problems that do not have quantum property testers? We conjecture for instance that the language $\{uuvv : u, v \in \Sigma^*\}$ does not have a quantum property tester.
- Beals, et. al. [BBC+98] observed that any k-query quantum algorithm gives rise to a degree-$2k$ polynomial in the input bits, which gives the acceptance probability of the algorithm; thus, a quantum property tester for P gives rise to a polynomial that is on all binary inputs between 0 and 1, that is at least 2/3 on inputs with the property P and at most 1/3 on inputs far from having the property P. Szegedy [Sze99] suggested to algebraically characterize the complexity of classical testing by the minimum degree of such polynomials; as mentioned in the introduction, our results imply that this cannot be the case for classical testers. However, it is an open question whether quantum property testing can be algebraically characterized in this way.

We hope that further research will lead to a greater understanding of what can and cannot be tested with quantum property testers.

Acknowledgments

We thank Ronitt Rubinfeld for discussions and pointers on property testing.

References

[ABI86] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of Algorithms*, 7:567–583, 1986.

[AFKS99] N. Alon, E. Fischer, M. Krivelevich, and M. Szegedy. Efficient testing of large graphs. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, pages 656–666. IEEE, 1999.

[Alo01] N. Alon. Testing subgraphs in large graphs. In *Proceedings of the 42th IEEE Symposium on Foundations of Computer Science*, pages 434–441. IEEE, 2001.

[ANKS99] N. Alon, I. Newman, M. Krivelevich, and M. Szegedy. Regular languages are testable with a constant number of queries. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, pages 645–655, New York, 1999. IEEE.

[BBBV97] C. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.

[BBC+98] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 352–361. IEEE, New York, 1998.

[BH97] G. Brassard and P. Høyer. An exact quantum polynomial-time algorithm for Simon's problem. In *Proceedings of the 5th Israeli Symposium on Theory of Computing and Systems (ISTCS'97)*, pages 12–23, 1997.

[BLR93] M. Blum, M. Luby, and R. Rubinfeld. Self-testing and self-correcting programs, with applications to numerical programs. *Journal of Computer and System Sciences*, 47:549–595, 1993.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.

[EKK+00] F. Ergün, S. Kannan, S. Kumar, R. Rubinfeld, and M. Vishwanathan. Spot-checkers. *Journal of Computer and System Sciences*, 60(3):717–751, 2000.

[Fis01a] E. Fischer. Testing graphs for colorability properties,. In *Proceedings of the 12th ACM-SIAM SODA*, pages 873–882. ACM, 2001.

[Fis01b] E. Fischer. Testing of matrix properties,. In *Proceedings of the* 33^{rd} *ACM STOC*, pages 286–295. ACM, 2001.

[GGR98] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.

[Gol98] O. Goldreich. Combinatorial property testing (a survey), 1998. Manuscript.

[Gro96] L. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on the Theory of Computing*, pages 212–219. ACM, New York, 1996.

[GT01] O. Goldreich and L. Trevisan. Three theorems regarding testing graph properties. In *Proceedings of the 42th IEEE Symposium on Foundations of Computer Science*, pages 460–469. IEEE, Nevada, 2001.

[Høy01] P. Høyer. Fourier sampling. Private communication, 2001.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[Ron00] D. Ron. Property testing. In *Handbook of Randomization*. Kluwer, 2000. To appear.

[RS96] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, April 1996.

[Sho97] P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[Sim97] D. Simon. On the power of quantum computation. *SIAM Journal on Computing*, 26(5):1474–1483, 1997.

[SY96] R. Solovay and A. Yao, 1996. Manuscript.

[Sze99] M. Szegedy. Private communication, 1999.

[Yao77] A. C-C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of 18th IEEE FOCS*, pages 222–227, 1977.

A Appendix

Proof of Lemma 4.10. We follow the steps of subroutine Q when passed k linearly independent vectors $z_1, \dots, z_k$ so that all $i_j := \min\{i : z_j[i] = 1\}$ are distinct for $1 \le j \le k$.

$$|0^n\rangle|0\rangle|0^k\rangle \longmapsto \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} |x\rangle|0\rangle|0^k\rangle \longmapsto \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle|0^k\rangle \longmapsto \frac{1}{N} \sum_{x,y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle|f(x)\rangle|0^k\rangle$$

This is the state before the **for** loop is entered. We claim (and proceed to show by induction) that after the Jth execution of the loop body, the state is

$$\frac{\sqrt{2^J}}{N} \sum_{x\in\{0,1\}^n} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\ \forall j\le J}} (-1)^{x\cdot y}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_J\rangle|0^{k-J}\rangle.$$

Executing the body of the loop for $j = J+1$,

$$\frac{\sqrt{2^J}}{N} \sum_{x\in\{0,1\}^n} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\,\forall j\le J}} (-1)^{x\cdot y}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_J\rangle|0\rangle|0^{k-J-1}\rangle \longmapsto$$

$$\frac{\sqrt{2^J}}{N} \sum_{x\in\{0,1\}^n} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\,\forall j\le J}} (-1)^{x\cdot y}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_J\rangle|y[i_{j+1}]\rangle|0^{k-J-1}\rangle =$$

$$\frac{\sqrt{2^J}}{N} \sum_{x\in\{0,1\}^n} \sum_{b\in\{0,1\}} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\,\forall j\le J+1}} (-1)^{x\cdot(y\oplus bz_{J+1})}|y\oplus bz_{J+1}\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_J\rangle|b\rangle|0^{k-J-1}\rangle \longmapsto$$

$$\frac{\sqrt{2^J}}{N} \sum_{x\in\{0,1\}^n} \sum_{b\in\{0,1\}} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\,\forall j\le J+1}} (-1)^{x\cdot(y\oplus bz_{J+1})}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_J\rangle|b\rangle|0^{k-J-1}\rangle =$$

$$\frac{\sqrt{2^{J+1}}}{N} \sum_{x\in\{0,1\}^n} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\,\forall j\le J+1}} (-1)^{x\cdot y}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_J\rangle\frac{1}{\sqrt{2}}\sum_{b\in\{0,1\}}(-1)^{x\cdot(bz_{J+1})}|b\rangle|0^{k-J-1}\rangle \longmapsto$$

$$\frac{\sqrt{2^{J+1}}}{N} \sum_{x\in\{0,1\}^n} \sum_{\substack{y\in\{0,1\}^n \\ y[i_j]=0\,\forall j\le J+1}} (-1)^{x\cdot y}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_{J+1}\rangle|0^{k-J-1}\rangle$$

□

Proof of Lemma 4.13. For $b\in\{0,1\}$ let $D_{b,c} := D_c \cap f^{-1}\{b\} = \{x : f(x) = b \text{ and } x\cdot z_1 = c[1],\ldots,x\cdot z_k = c[k]\}$. Note that the $D_{b,c}$ and D_c also depend on $z_1, \ldots, z_k$ and the $D_{b,c}$ depend on f. Let

$$|\psi_0\rangle := \frac{\sqrt{2^k}}{N}\sum_{x\in\{0,1\}^n}|0\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_k\rangle = \frac{\sqrt{2^k}}{N}\sum_{b\in\{0,1\}}\sum_{c\in\{0,1\}^k}|D_{b,c}||0\rangle|b\rangle|c[1]\rangle\cdots|c[k]\rangle.$$

By Lemma 4.10, at the end of Q, the system is in state

$$|\psi\rangle = |\psi_0\rangle + \frac{\sqrt{2^k}}{N}\sum_{x\in\{0,1\}^n}\sum_{\substack{y\in\{0,1\}^n\setminus\{0\} \\ y[i_j]=0\,\forall j\le k}}(-1)^{x\cdot y}|y\rangle|f(x)\rangle|x\cdot z_1\rangle\cdots|x\cdot z_k\rangle.$$

We consider the case $\|P_0|\psi\rangle\|^2 = 1$. Then the register $\mathcal{X}$ must be in state $|0\rangle$ and thus $|\psi\rangle = |\psi_0\rangle$. Since the state has norm 1, we know that

$$\sum_{b\in\{0,1\}}\sum_{c\in\{0,1\}^k}|D_{b,c}|^2 = \frac{N^2}{2^k}. \tag{1}$$

The $D_{b,c}$ partition $\{0,1\}^n$ and the $D_c = D_{0,c}\cup D_{1,c}$ have the same size for all $c\in\{0,1\}^k$ (because they are cosets of D_0.) Therefore,

$$\sum_{b\in\{0,1\}}\sum_{c\in\{0,1\}^k}|D_{b,c}| = N \qquad \text{and} \qquad |D_{0,c}|+|D_{1,c}| = \frac{N}{2^k} \text{ for all } c\in\{0,1\}^k. \tag{2}$$

$|D_{0,c}|^2 + |D_{1,c}|^2 \le N^2/2^{2k}$, but in order for equation (1) to hold, $|D_{0,c}|^2 + |D_{1,c}|^2$ must be exactly $N^2/2^{2k}$. This can only be achieved if either $D_{0,c}$ or $D_{1,c}$ is empty. So f must be constant when restricted to D_c for any $c \in \{0,1\}^k$. Conversely, if f is constant when restricted to D_c for any $c \in \{0,1\}^k$, then equation (1) holds, therefore $\||\psi_0\rangle\| = 1$ and $|\psi\rangle = |\psi_0\rangle$. This concludes the proof of case 1 of the lemma.

If $\|P_0|\psi\rangle\|^2 = \||\psi_0\rangle\|^2 \ge 1 - \delta$, then

$$\sum_{b\in\{0,1\}} \sum_{c\in\{0,1\}^k} |D_{b,c}|^2 \ge (1-\delta)\,\frac{N^2}{2^k} \tag{3}$$

Still, the constraints (2) hold; let $r2^k$ be the number of $c \in \{0,1\}^k$ so that $\min\{|D_{0,c}|, |D_{1,c}|\} \ge \gamma N/2^k$. Then

$$\sum_{b\in\{0,1\}} \sum_{c\in\{0,1\}^k} |D_{b,c}|^2 \le r2^k(\gamma^2 + (1-\gamma)^2)\frac{N^2}{2^{2k}} + (1-r)2^k\frac{N^2}{2^{2k}},$$

and using (3), we obtain $r \le \delta/(1-\gamma^2-(1-\gamma)^2)$. With $\delta = \epsilon^2/2$ and $\gamma = \epsilon/2$, this implies $r \le \epsilon$. But then

$$\sum_{c\in\{0,1\}^k} \min\{|D_{0,c}|, |D_{1,c}|\} \le r2^k\frac{N}{2^{k+1}} + (1-r)2^k\gamma\frac{N}{2^k} \le \epsilon N$$

$\square$

Proof of Lemma 4.14. If $k < n$, then there exists an s with $s \cdot z_1 = 0, \ldots, s \cdot z_k = 0$. For any such s and any x, we have $x \cdot z_1 = (x \oplus s)\cdot z_1, \ldots, x \cdot z_k = (x \oplus s) \cdot z_k$ and $x \in D_{f(x), x\cdot z_1, \ldots, x \cdot z_k}$ and $x \oplus s \in D_{f(x\oplus s), x\cdot z_1, \ldots, x\cdot z_k}$, therefore $f(x) = f(x \oplus s)$. Conversely, for $f \in L$, $S := \{s : \forall x f(x) = f(x \oplus s)\}$ is a non-trivial subspace of $\{0,1\}^n$, therefore $S^\perp = \{z : z \cdot s = 0 \forall s \in S\}$ is a proper subspace of $\{0,1\}^n$. Let $z_1, \ldots, z_k$ be any basis of $S^\perp$. $\square$