Quantum query complexity and distributed computing
Röhrig, H.P.

Citation for published version (APA):
Chapter 3

Property Testing

This chapter is based on joint work with Buhrman, Fortnow, and Newman [33].

3.1 Introduction

Suppose we have a large data set, for example, a large chunk of the worldwide 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 look at the entire data set or even a large portion of it.

To handle these types of problems, Rubinfeld and Sudan [103] and Goldreich, Goldwasser and Ron [65] have developed the notion of property testing. Testable properties come in many varieties including graph properties, e.g., [65, 7, 57, 58, 5, 66], algebraic properties of functions [23, 103, 51], and regular languages [8]. Nice surveys of this area can be found in [102] [56].

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 number of input bits; the set of indices is usually of constant size and chosen probabilistically. Clearly we cannot determine from this small number of bits whether the input sits in some language $L$. However, for many languages we can distinguish the case that the input is in $L$ from the case that the input differs from all inputs in $L$ of the same length by some constant fraction of input bits.

Since there are many examples where quantum computation gives us an advantage over classical computation [22, 109, 108, 69] one may naturally ask whether using quantum computation may lead to better property testers. By using the quantum oracle-query model we can easily extend the definitions of property testing to the quantum setting.
Chapter 3. Property testing

Beals, Buhrman, Cleve, Mosca, and de Wolf [15] have shown that for all total functions we have a polynomial relationship between the number of queries required by a quantum machine and that needed by a deterministic machine. For greater separations one needs to impose a promise on the input. The known examples, such as those due to Simon [109] and Bernstein and Vazirani [22], require considerable structure in the promise. Property testing amounts to the natural promise of either being in the language or far from each 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 every subset of Hadamard codes has a quantum property tester with $O(1)$ queries and that 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, Buhrman, Cleve, Mosca, and de Wolf [15] observed that every $k$-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 [114] 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. Hence, the minimum degree is only a lower bound, which sometimes is not tight.

A priori it is conceivable that 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, every quantum algorithm requires $\Omega(n)$ queries. We then show that a natural explicit property, namely, the range of a $d$-wise independent pseudorandom generator cannot be quantumly tested with less than $(d + 1)/2$ queries for every odd $d \leq n/\log n - 1$.

3.2 Preliminaries

We will use the following formal definition of property testing from Goldreich [64]:

3.2.1. Definition. Let $S$ be a finite set, and $P$ a set of functions mapping $S$ to $\{0, 1\}$. A property tester for $P$ is a probabilistic oracle machine $M$,
3.3. Separating quantum and classical property testing

which given a distance parameter $\varepsilon > 0$ and oracle access to a function $f : S \rightarrow \{0,1\}$, satisfies the following conditions:

1. the tester accepts $f$ if it is in $P$: if $f \in P$ then $\Pr(M^f(\varepsilon) = 1) \geq 2/3$

2. the tester rejects $f$ if it is far from $P$: if $|\{x \in S : f(x) \neq g(x)\}| > \varepsilon \cdot |S|$, for every $g \in P$, then $\Pr(M^f(\varepsilon) = 1) \leq 1/3$.

Here $M^f$ denotes that the machine $M$ is provided with the oracle for $f$.

3.2.2. DEFINITION. The complexity of the tester is the number of oracle queries it makes: A property $P$ has an $(\varepsilon, q)$-tester if there is a tester for $P$ that makes at most $q$ oracle queries for distance parameter $\varepsilon$.

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 $q = q(\varepsilon, n)$ asymptotically for large $n$. We say $L$ is $\varepsilon$-testable with $q(n)$ queries, if for each $n$, $P_n$ has a $(\varepsilon, q(n))$ tester.

To define quantum property testing we simply modify Definition 3.2.1 by allowing $M$ to be a quantum oracle machine.

3.3 Separating Quantum and Classical Property Testing

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

3.3.1. THEOREM. There is a language $L$ that is $\varepsilon$-testable by a quantum test with $O(1/\varepsilon)$ number of queries but for which every probabilistic $1/3$-test requires $\Omega(\log n)$ queries.

We use Hadamard codes to provide examples for Theorem 3.3.1:

3.3.2. DEFINITION. 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 $y, i \in F_2^{\log n}$.

Note: the Hadamard mapping $h : \{0,1\}^{\log n} \rightarrow \{0,1\}^n$ is one-to-one. Bernstein and Vazirani [22] 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 3.3.1 follows from the following two lemmas.
3.3.3. **Lemma.** For every $A$, $P_A$ has an $(\epsilon, O(1/\epsilon))$ quantum tester. Furthermore, the test has one-sided error.

3.3.4. **Lemma.** 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 3.3.3 we note that for every $A$, $P_A$ can be tested by a one-sided algorithm with $O(1/\epsilon + \log n)$ queries even nonadaptively; hence, the result of Lemma 3.3.4 is tight. An $\epsilon$-test with $O((\log n)/\epsilon)$ queries follows from Theorem 3.3.5 below. The slightly more efficient test is the following: First we query $x_2^i$, $i = 1, \ldots, \log n$. Note that if $x = h(y)$ then $y_i = x_{2i}$ for $i = 1, \ldots, \log n$. Thus a candidate $y$ for $x = h(y)$ is found. If $y \not\in A$ then $x$ is rejected. Then $k := O(1/\epsilon)$ times the following check is performed: a random index $i \in \{1, \ldots, n\}$ is chosen independently at random and if $x_i \not= y \cdot i$, then $x$ is rejected. Otherwise, $x$ is accepted. Clearly if $x$ is rejected then $x \not\in P_A$.

It is easily verified that if $x$ has Hamming distance more than $\epsilon n$ from every $z$ in $P_A$ then with constant probability $x$ is rejected.

**Proof of Lemma 3.3.3.** $P_A$ can be checked with $O(1/\epsilon)$ 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 $x_i \oplus x_j \not= x_{i \oplus j}$ reject. $k = O(1/\epsilon)$ is sufficient to detect an input $x$ that is $\epsilon n$-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 each $x' \in P_A$, then it is either far from being a Hadamard codeword or it is close to a Hadamard codeword $h(y')$ for a $y' \not\in A$; note that in this case $x$ is far from every $h(y)$, $y \in A$ 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 $y' \not\in A$. We note also that this algorithm has one-sided error. 

**Proof of Lemma 3.3.4.** The lower bound makes use of the Yao principle [118]: let $D$ be an arbitrary probability distribution on positive and negative inputs, i.e., on inputs that either belong to $P_A$ or are $\epsilon n$-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
3.3. Separating quantum and classical property testing

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 algorithm errs on at most 1/8 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 lose much. Let \(T\) be any fixed deterministic decision tree performing at most \(q\) queries in every branch. Then let \(c(T) := \{y | T(h(y)) = \text{accept}\}\) and let \(\mu(T) := |c(T)|/n\), i.e., \(\mu(T)\) is the fraction of inputs that \(T\) accepts. Assume first that \(\mu(T) \leq 1/2\). Since for a random \(y\) we have \(\Pr_y[T(h(y)) = \text{accept}] = \mu(T) \leq 1/2\), it follows by a Chernoff-type bound that \(\Pr_A[|A \cap c(T)| \geq (3/4)|A|] \leq 2^{-n/8}\): However, if \(|A \cap c(T)| < (3/4)|A|\) then \(T\) 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 \(2^{-n/8}\), \(T\) will be correct on at least 7/8 of the inputs. If \(\mu(T) > 1/2\) the same reasoning shows that with probability of at most \(1 - 2^{-n/8}\) it will err on at least a 1/4-fraction of the negative inputs. Hence, in total, for every fixed \(T\), \(\Pr_A[T\text{ is correct on at least }7/8\text{ of the inputs}] \leq 2^{-n/8}\).

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 for all queries on all branches and a Boolean function \(f : \{0,1\}^q \rightarrow \{\text{accept, reject}\}\) 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 \(T \cdot 2^{-n/8} = o(1)\), which shows that for most \(A\) as above, every \(\varepsilon\)-test that queries at most \((\log n)/2\) many queries has error probability of at least 1/8. Standard amplification techniques then imply that for some constant \(c\) every algorithm that performs \(c \log n\) many queries has error at least 1/3. \(\square\)

3.3.5. Theorem. Let \(P \subseteq \{0,1\}^n\) be a property with \(|P| = s > 0\). For any \(\varepsilon > 0\), \(P\) can be \(\varepsilon\)-tested by a one-sided classical algorithm using \(O((\log s)/\varepsilon)\) many queries.

Proof. Denote the input by \(y \in \{0,1\}^n\). Consider the following algorithm: query the input \(y\) in \(k := \ln(3s^2)/\varepsilon\) random places; accept if there is at least one \(x \in P\) consistent with the bits from the input and reject otherwise. Clearly, if \(y \in P\), this algorithm works correctly.

If \(y\) is \(\varepsilon\)-far from each \(x \in P\), then for every specific \(x \in P\), \(\Pr[x_i = y_i] \leq 1 - \varepsilon\) when choosing an \(i \in [n]\) uniformly at random. With \(k\) indices chosen
independently and uniformly at random, the probability for no disagreement with \( z \) becomes \((1 - \varepsilon)^k \leq 1/(3s^2)\). Therefore, the probability that there is no disagreement for at least one of the \( s \) members of \( P \) is at most \( 1/(3s) \), so with probability \( 2/3 \) for a \( y \) that is far from \( P \), we will rule out every \( x \in P \) as being consistent with \( y \).

\[ \square \]

### 3.4 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.

**3.4.1. Theorem.** There exists a language \( L \) that for every \( \varepsilon = \Omega(1) \) is \((\varepsilon, \log n \log \log n)\) quantumly testable but every probabilistic \( 1/8 \)-test for \( L \) requires \( n^{\Omega(1)} \) queries.

The language that we provide is inspired by Simon's problem [109] and our quantum testing algorithm makes use of Brassard and Høyer's algorithm for Simon's problem [26]. Simon's problem is to find \( s \in \{0,1\}^n \setminus \{0^n\} \) from a 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 \( z \)s. Essentially, our quantum tester uses this subroutine to try to extract information about \( s \) until it fails repeatedly. Høyer [74] and also Friedl et al. [61] analyzed this approach in group-theoretic terms, obtaining an alternative proof to Theorem 3.4.3.

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 \( j \)th bit of \( x \), i.e., \( x = x[1] \ldots x[n] \). 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 3.4.1 follows from the following two theorems.

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

**3.4.3. Theorem.** 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 3.4.2. We again apply the Yao principle [118] as in the proof of Lemma 3.3.4: we construct two distributions, $P$ and $U$, on positive and at least $N/8$-far negative inputs, respectively, such that every deterministic adaptive decision tree $T$ with few queries 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 $T$ be any deterministic decision tree. Let $v$ be a vertex of $T$. Let $\Pr_P(v)$ and $\Pr_U(v)$ be the probability that an input chosen according to $P$ and $U$, respectively, is consistent with $v$. We will show that for every vertex $v$ of $T$ we have $|\Pr_P(v) - \Pr_U(v)| = o(1)$; hence, $T$ has error $1/2 - o(1)$ if with probability $1/2$ we choose $v$ according to $P$ and with probability $1/2$ from $U$.

The distribution $P$ is defined as follows: We first choose $s \in \{0,1\}^n$ 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, these functions 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 nonzero 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:

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

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

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

In particular, a direct consequence of Lemma 3.4.5 is that with probability $1 - e^{-\Omega(N)}$ 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:
Chapter 3. Property testing

3.4.6. **Lemma.** Let \( T \) be any fixed deterministic decision tree and let \( v \) be a vertex of depth \( d \) in \( T \). 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 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 \( s \):

3.4.7. **Definition.** Let \( T \) be a deterministic decision tree and \( u \) a vertex in \( T \) at depth \( d \). We denote the path from the root of \( T \) to \( u \) by \( \text{path}(u) \). Every vertex \( v \) in \( T \) 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 \( B_u := \{(s,f_s) : s \neq x_v \oplus x_w \text{ for all } v,w \in \text{path}(u)} \).

3.4.8. **Lemma.** Let \( v \) be a vertex of depth \( d \) in a decision tree \( T \). Then \( \Pr_P[B_v] \geq 1 - (d-1)/N \)

**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 \((d-1)\) pairs. Each of these pairs excludes exactly one \( s \) and there are \( N \) possible \( s \)'s.

3.4.9. **Lemma.** Let \( v \) be a vertex of depth \( d \) in a decision tree \( T \) 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 \( T \) is a deterministic decision tree of depth \( d = o(\sqrt{N}) \) and let \( v \) be any leaf of \( T \). Then by Lemmas 3.4.8 and 3.4.9, we get that \( \Pr_p(f \text{ is consistent with } v) = (1 - o(1))2^{-d} \). 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 3.4.5 and 3.4.6 we get that \( \Pr_{U'}[f \text{ is consistent with } v] = (1 - o(1))2^{-d} \) and hence \( T \) has only \( o(1) \) bias of being right on every leaf. This implies that its error probability is \( 1/2 - o(1) \).

**Proof of Theorem 3.4.3.** We give a quantum algorithm making \( O(\log N \log \log N) \) 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 \( \varepsilon N \). Pseudo code for our algorithm is given on page 65; it consists of a classical main program SimonTester and a quantum subroutine SimonSampler adapted from Brassard and Høyer's algorithm for Simon's problem [26, Section 4]. The
3.4. An exponential separation

Procedure SimonTester
1: for \( k = 0 \) to \( n - 1 \) do
2: \( l \leftarrow 0 \)
3: repeat
4: \( z \leftarrow \text{SimonSampler}(z_1, \ldots, z_k) \)
5: \( l \leftarrow l + 1 \)
6: until \( z \neq 0 \) or \( l > 2(\log n)/e^2 \)
7: if \( z = 0 \) then
8: accept
9: else
10: \( z_{k+1} \leftarrow z \)
11: reject

Procedure SimonSampler(\( z_1, \ldots, z_k \))
1: input: \( z_1, \ldots, z_k \in \{0, 1\}^n \)
2: output: \( z \in \{0, 1\}^n \)
3: quantum workspace: \( \mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z} \) where
4: \( \mathcal{X} \) is \( n \) qubits \( \mathcal{X} = \mathcal{X}_1 \otimes \cdots \otimes \mathcal{X}_n, \mathcal{X}_i = \mathbb{C}^2, \)
5: \( \mathcal{Y} = \mathbb{C}^2 \) is one qubit, and
6: \( \mathcal{Z} \) is \( k \) qubits \( \mathcal{Z} = \mathcal{Z}_1 \otimes \cdots \otimes \mathcal{Z}_k, \mathcal{Z}_j = \mathbb{C}^2 \)
7: initialize the workspace to \( |0^n\rangle|0\rangle|0^k\rangle \)
8: apply \( H_{2^n} \) to \( \mathcal{X} \)
9: apply \( U_f \) to \( \mathcal{X} \otimes \mathcal{Y} \)
10: apply \( H_{2^n} \) to \( \mathcal{X} \)
11: for \( j = 1 \) to \( k \) do
12: \( i \leftarrow \min\{i : z_j[i] = 1\} \)
13: apply CNOT with control \( \mathcal{X}_i \) and target \( \mathcal{Z}_j \)
14: apply \( |x\rangle \leftrightarrow |x \oplus z_j\rangle \) to \( \mathcal{X} \) conditional on \( \mathcal{Z}_j \)
15: apply \( H_2 \) to \( \mathcal{Z}_j \)
16: return measurement of \( \mathcal{X} \)
quantum gates used are the $2^{n}$-dimensional Hadamard transform $H_{2^n}$, which applies
\[
\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}
\]
individually to each of $n$ qubits, the quantum oracle query $U_f$, and classical reversible operations run in quantum superposition.

The following technical lemma captures the operation of the quantum subroutine SimonSampler. For $i_1, \ldots, i_J$ fixed, let $Y_j := \{ y \in \{0,1\}^n : \forall j \leq J \, y[i_j] = 0 \}$ denote the length-$n$ binary strings that are 0 at positions $i_1, \ldots, i_J$.

3.4.10. Lemma. When SimonSampler is passed $k$ vectors $z_1, \ldots, z_k$ so that all $i_j := \min\{i : z_j[i] = 1\}$ are distinct for $1 \leq j \leq k$, then the state $|\psi\rangle$ before the measurement is
\[
\frac{\sqrt{2^k}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_k} (-1)^{x \cdot y} |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_k\rangle.
\]

Proof. We follow the steps of subroutine SimonSampler.
\[
|0^n\rangle|0\rangle|0^k\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |x\rangle|0\rangle|0^k\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |x\rangle|f(x)\rangle|0^k\rangle \mapsto \frac{1}{N} \sum_{x,y \in \{0,1\}^n} (-1)^{x \cdot y} |f(x)\rangle|0^k\rangle
\]
This is the state before the for loop is entered. We claim and proceed to show by induction that after the $J$th execution of the loop body, the state is
\[
\frac{\sqrt{2^J}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_j} (-1)^{x \cdot y} |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_J\rangle|0^{k-J}\rangle.
\]
Executing the body of the loop for $j = J + 1$,
\[
\frac{\sqrt{2^J}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_J} (-1)^{x \cdot y} |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_J\rangle|0^{k-J-1}\rangle
\]
\[
\mapsto \frac{\sqrt{2^J}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_J} (-1)^{x \cdot y} |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_J\rangle |y[i_{j+1}]\rangle|0^{k-J-1}\rangle
\]
\[
= \frac{\sqrt{2^J}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_{J+1}} \sum_{b \in \{0,1\}} (-1)^{x \cdot (y \oplus bz_{J+1})} |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_J\rangle |b\rangle|0^{k-J-1}\rangle
\]
(Here, we used the fact that \( Y_J = Y_{J+1} \cup (z_{J+1} + Y_{J+1}) \).)

\[
\frac{\sqrt{2^J}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_{J+1}} (-1)^{x \cdot (y \oplus b_{J+1})} |y\rangle |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_j\rangle |b\rangle |0^{k-J-1}\rangle
\]

\[
= \frac{\sqrt{2^{J+1}}}{N} \sum_{x \in \{0,1\}^n} \sum_{y \in Y_{J+1}} (-1)^{x \cdot y} |y\rangle |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_J\rangle
\]

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

This establishes the following invariants for SimonTester:

3.4.11. **Lemma.** If measuring the first register, \( X \), yields a nonzero value \( z \), then

1. \( \{z_1, \ldots, z_k, z\} \) 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 every \( s \neq 0 \) such that \( f(x) = f(x \oplus s) \) for all \( x \).

**Proof.** If we measure the state from Lemma 3.4.10, then for the value \( z \) of the first register holds \( z \in Y_k \). This implies 2, from which follows 1. For 3: as in Simon's original algorithm, if there is a \( s \neq 0 \) so that for all \( x \), \( f(x) = f(x \oplus s) \), then we can rewrite the state from Lemma 3.4.10 as

\[
\frac{\sqrt{2^k}}{N} \sum_{x : x < x \oplus s} y \left( (-1)^x \cdot y |f(x)\rangle + (-1)^{(x \oplus s) \cdot y} |f(x \oplus s)\rangle \right) |x \cdot z_1\rangle \cdots |x \cdot z_k\rangle
\]

\[
= \frac{\sqrt{2^k}}{N} \sum_{x : x < x \oplus s} \sum_{y \in Y_k} |y\rangle (-1)^x \cdot y (1 + (-1)^x \cdot y) |f(x)\rangle |x \cdot z_1\rangle \cdots |x \cdot z_k\rangle .
\]

Hence, only \( y \) with \( s \cdot y = 0 \) will have nonzero amplitude. \( \square \)
Next, we want to assess the probability of obtaining \( z = 0 \) in SimonTester Line 4. 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, \( \|P_0|\psi\|_2^2 \) 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:

### 3.4.12. Definition

For \( c \in \{0,1\}^k \) and \( z_1, \ldots, z_k \in \{0,1\}^n \) we define \( D_c := \{ x \in \{0,1\}^n : x \cdot z_1 = c[1], \ldots, x \cdot z_k = c[k] \} \).

### 3.4.13. Lemma

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

1. \( \|P_0|\psi\|_2^2 = 1 \) if and only if for every \( c \in \{0,1\}^k \), \( f \) is constant when restricted to \( D_c \).

2. If \( \|P_0|\psi\|_2^2 \geq 1 - \varepsilon^2/2 \), then \( f \) differs in at most \( \varepsilon N \) points from some function \( g \) that is constant when restricted to \( D_c \) for every \( c \in \{0,1\}^k \).

#### Proof

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

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

\[
= \frac{\sqrt{2^k}}{N} \sum_{b \in \{0,1\}} \sum_{c \in \{0,1\}^k} |D_{b,c}| |0\rangle|b\rangle|c[1]\rangle \cdots |c[k]\rangle.
\]

By Lemma 3.4.10, at the end of SimonSampler the system is in state \(|\psi\rangle = |\psi_0\rangle + |\psi_0^\perp\rangle\) for some \(|\psi_0^\perp\rangle\) orthogonal to \(|\psi_0\rangle\). We consider the case \( \|P_0|\psi\|_2^2 = 1 \). 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

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

(3.1)

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,

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

(3.2)
3.4. An exponential separation

\[ |D_{0,c}|^2 + |D_{1,c}|^2 \leq N^2/2^{2k}, \]
but in order for equation (3.1) 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 (3.1) holds, therefore \(||\psi_0|| = 1\) and \(|\psi| = |\psi_0|\). This concludes the proof of case 1 of the lemma.

If \(||P_0|\psi||^2 = ||\psi_0||^2 \geq 1 - \delta\), then

\[
\sum_{b \in \{0,1\}} \sum_{c \in \{0,1\}^k} |D_{b,c}|^2 \geq (1 - \delta) \frac{N^2}{2^k}. \tag{3.3}
\]

Still, the constraints (3.2) hold; let \(r 2^k\) be the number of \(c \in \{0,1\}^k\) so that \(\min\{|D_{0,c}|, |D_{1,c}|\} \geq \gamma N/2^k\). Then

\[
\sum_{b \in \{0,1\}} \sum_{c \in \{0,1\}^k} |D_{b,c}|^2 \leq r 2^k (\gamma^2 + (1 - \gamma)^2) \frac{N^2}{2^{2k}} + (1 - r) 2^k \frac{N^2}{2^k},
\]

and using (3.3), we obtain \(r \leq \delta/(1 - \gamma^2 - (1 - \gamma)^2)\). With \(\delta = \varepsilon^2/2\) and \(\gamma = \varepsilon/2\), this implies \(r \leq \varepsilon\). But then

\[
\sum_{c \in \{0,1\}^k} \min\{|D_{0,c}|, |D_{1,c}|\} \leq r 2^k \frac{N}{2^{k+1}} + (1 - r) 2^k \gamma \frac{N}{2^k} \leq \varepsilon N .
\]

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.

3.4.14. LEMMA. 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 \(k < n\).

Proof. If \(k < n\), then there exists an \(s \neq 0\) with \(s \cdot z_1 = 0, \ldots, s \cdot z_k = 0\). For each such \(s\) and all \(x\), 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 nontrivial subspace of \(\{0,1\}^n\), therefore \(S^1 = \{z : z \cdot s = 0 \forall s \in S\}\) is a proper subspace of \(\{0,1\}^n\).

Let \(z_1, \ldots, z_k\) be an arbitrary basis of \(S^1\). \(\Box\)

3.4.15. LEMMA. Let \(0 < q < 1\), and \(|\varphi_1\), \ldots, \(|\varphi_m|\) be quantum states satisfying \(||P_0|\varphi_j||^2 < 1 - \delta\) 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 \(X\) register of \(|\varphi_1\), \ldots, \(|\varphi_m|\) will yield \(m\) times outcome 0.
Chapter 3. Property testing

Proof.  

Pr [m times 0 | \forall j : \| P_0 | \varphi_j \| ^2 < 1 - \delta ] < (1 - \delta )^m = (1 - \delta )^{\log q / \log (1 - \delta )} = q . \qed

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 3.4.11 the nonzero \( z \) computed by the algorithm lie in \( S^\perp \) and are linearly independent, therefore after \( \dim S^\perp \) rounds of for loop in SimonTester, we measure \( z = 0 \) with certainty. Since \( f \in L \), \( \dim S > 0 \) and thus \( \dim S^\perp < n \).

If \( f \) is \( \varepsilon n \)-far from being in \( L \), then by Lemma 3.4.14 \( f \) is \( \varepsilon n \)-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 3.4.13 case 2, for all \( k < n \), \( \| P_0 | \psi \| ^2 < 1 - \varepsilon^2 / 2 \). Thus, Lemma 3.4.15 guarantees that we accept with probability at most \( 1 / 3 \) if we let \( q = 1 / (3n) \) and thus \( m = O((\log n) / \varepsilon^2) \).

This concludes the proof of Theorem 3.4.3. \qed

3.5 Quantum Lower Bounds

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

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

Proof. Fix \( n \), a small \( \varepsilon \), 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

\[ P_\varepsilon := \{ y : d( x, y ) < \varepsilon n \ \text{for some} \ x \in P \} ; \]

using \( \sum_{k=0}^{n} \binom{n}{k} \leq 2^{H(\varepsilon)n} \), where

\[ H(\varepsilon) := -\varepsilon \log \varepsilon - (1 - \varepsilon) \log (1 - \varepsilon) , \]

we obtain \( |P_\varepsilon| \leq 2^{(1/20 + H(\varepsilon))n} \). In order for \( A \) to test properties of size \( 2^{n/20} \), it needs to reject with high probability on at least \( 2^n - 2^{(1/20 + H(\varepsilon))n} \) inputs; but then, the probability that \( A \) accepts with high probability on a random \( x \in \{0, 1\}^n \) is bounded by \( 2^{(1/20 + H(\varepsilon))n} / 2^n \) and therefore the probability that \( A \) accepts with high probability on \( |P| \) random inputs is bounded by

\[ 2^{-(1/20 - H(\varepsilon))n |P|} = 2^{-2^{n/20 + \Theta(\log n)}} . \]
3.5. Quantum lower bounds

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 [15], 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 [20] and Solovay and Yao [110], every 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 \( 2^{O(n)} \) over some fixed denominator. There are less than \( 2^{nH(2q/n)} \) degree-\( q \) monomials in \( n \) variables, thus we can limit ourselves to \( 2^{nO(1)2^{nH(2q/n)}} \leq 2^{(n/20)\cdot(91/100)+\Theta(\log n)} \) 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. For \( m \ll n \), a pseudorandom number generator is a function \( f : \{0,1\}^m \to \{0,1\}^n \) that maps a small seed \( s \in \{0,1\}^m \) to a large binary string \( f(s) \in \{0,1\}^n \); if \( s \) is chosen uniformly at random, the distribution \( f(s) \) of \( n \)-bit strings should have certain properties of the uniform distribution over \( n \)-bit strings. One such property is independence: if \( x \in \{0,1\}^n \) is chosen uniformly at random, the values of its bits are independent, i.e., \( x[i] \) and \( x[j] \) are independent random variables for \( i \neq j \). Accordingly, random \( s, f(s)[i] \) and \( f(s)[j] \) should be independent, i.e., for fixed seed \( s \) and index \( i \) and each index \( j \neq i \), the sets of seeds

\[
S_{s,i,j,0} := \{ s' : f(s')[j] = 0 \text{ and } f(s')[i] = f(s)[i] \}
\]
\[
S_{s,i,j,1} := \{ s' : f(s')[j] = 1 \text{ and } f(s')[i] = f(s)[i] \}
\]

should have the same size. This independence requirement readily extends to fixing up to \( d \) bit positions and requiring that for each of the remaining bit positions \( j \), there are as many strings in the image \( f(\{0,1\}^m) \) with the \( j \)th bit 0 as there are with the \( j \)th bit 1. This corresponds to the \((d+1)\)-wise independence of the pseudorandom values \( f(\{0,1\}^m) \). Of course, choosing \( x \in \{0,1\}^n \) uniformly at random gives \( n \)-wise independence, but for many applications \( d \)-wise independence with \( d < n \) is sufficient and permits small seed sizes \( m \).

What we show is that for an arbitrary fixed \( f : \{0,1\}^m \to \{0,1\}^n \) that is a \( d \)-wise independent pseudorandom number generator, testing whether some \( x \in \{0,1\}^n \) is close to satisfying \( x \in f(\{0,1\}^m) \) requires many queries on a quantum computer. Intuitively, this means that such pseudorandom numbers look in a certain way random even to a quantum computer.
3.5.2. **Theorem.** The range of a \(d\)-wise independent pseudorandom number 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:

3.5.3. **Lemma** (See [6]). Suppose \(n = 2^k - 1\) and \(d = 2t+1 \leq n\). Then there exists a uniform probability space \(\Omega\) of size \(2(n+1)^t\) and \(d\)-wise independent random variables \(\xi_1, \ldots, \xi_n\) over \(\Omega\), each of which takes the values 0 and 1 with probability 1/2.

The proof of Lemma 3.5.3 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 \(z \in \Omega = \{1, \ldots, 2(n+1)^t\}\). As a warmup, observe that 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 a \(z\) such that \(\xi_{i_1}(z) \ldots \xi_{i_d}(z) = v_1 \ldots v_d\). On the other hand, \(|\log |\Omega|\| + 1 = O(d\log n)\) queries are always sufficient.

**Proof of Theorem 3.5.2.** We first consider the decision problem and then extend the lower bound to testing. 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 \(2T\) multilinear \(n\)-variable approximating polynomial \(p(x) = p(x_1, \ldots, x_n)\) [15]. 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 \(E[p(x)|x \in P] \geq 2/3\) and \(E[p(x)|x \not\in P] \leq 1/3\). Since \(|P| = o(2^n)\), \(E[p(x)] \leq 1/3 + o(1)\) and thus \(\Delta := E[p(x)|x \in P] - E[p(x)] \geq 1/3 - o(1)\). 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 \(E[p(x_1, \ldots, x_n)] = \sum_i \alpha_i E[m_i(x_1, \ldots, x_n)]\). Because of the \(d\)-wise independence of the bits of each \(x \in P\), for every \(m_i\) of degree at most \(d\) holds \(E[m_i(x)] = E[m_i(x)|x \in P]\). 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 \(P_\varepsilon := \{y : d(x, y) < \varepsilon n\text{ for some } x \in P\}\). Then

\[
|P_\varepsilon| \leq 2^{H(\varepsilon)n}|P| = O(2^{H(\varepsilon)n + d\log n})
\]

so

\[
E[p(x)] = \frac{|P_\varepsilon|}{2^n} E[p(x)|x \in P_\varepsilon] + \left(1 - \frac{|P_\varepsilon|}{2^n}\right) E[p(x)|x \not\in P_\varepsilon] \leq \frac{1}{3} + o(1)
\]
for every $d = n/\log n - \omega(1/\log n)$ and every $\varepsilon$ with $H(\varepsilon) = 1 - \omega(1/n)$. Again, we have $\Delta > 1/3 - o(1)$ and we need monomials of degree greater than $d$. 

3.6 Further Research

The research presented in this chapter initiated the study 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 $O(1)$ quantum queries to test?

- Are there other natural problems that do not have quantum property testers? The language $\{uvuv : u, v \in \Sigma^*\}$ appears to be a good candidate for not having a quantum property tester.

- Beals, Buhrman, Cleve, Mosca, and de Wolf [15] observed that every $k$-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 [114] 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.

- Høyer [74] and Friedl et al. [61] put quantum property testing into a group theoretic context. Is a characterization of quantum property testing possible in group-theoretic terms?