Quantum Property Testing for Bounded-Degree Graphs

Andris Ambainis[1], Andrew M. Childs[2], and Yi-Kai Liu[3]

[1] Faculty of Computing, University of Latvia.
Email: ambainis@lu.lv
[2] Department of Combinatorics & Optimization and
Institute for Quantum Computing, University of Waterloo.
Email: amchilds@uwaterloo.ca
[3] Department of Computer Science, University of California, Berkeley.
Email: yikailiu@eecs.berkeley.edu

Abstract. We study quantum algorithms for testing bipartiteness and expansion of bounded-degree graphs. We give quantum algorithms that solve these problems in time $\tilde{O}(N^{1/3})$, beating the $\Omega(\sqrt{N})$ classical lower bound. For testing expansion, we also prove an $\tilde{\Omega}(N^{1/4})$ quantum query lower bound, thus ruling out the possibility of an exponential quantum speedup. Our quantum algorithms follow from a combination of classical property testing techniques due to Goldreich and Ron, derandomization, and the quantum algorithm for element distinctness. The quantum lower bound is obtained by the polynomial method, using novel algebraic techniques and combinatorial analysis to accommodate the graph structure.

1 Introduction

In *property testing*, one is asked to distinguish between objects that satisfy a property P and objects that are far from satisfying P. The goal is to design algorithms that test properties in sublinear or even constant time, without reading the entire input—a task that is nontrivial even for properties that can be computed in polynomial time. This is motivated by the practical question of how to extract meaningful information from massive data sets that are too large to fit in a single computer's memory and can only be handled in small pieces.

Testing properties of graphs is an interesting special case.[4] Many graph properties, such as connectivity and planarity, can be tested in constant time, independent of the number of vertices N [19, 22]. However, some graph properties are much harder to test. For bounded-degree graphs in the adjacency-list representation, the best classical algorithms for testing *bipartiteness* [20] and *expansion* [21, 16, 25, 29] use $\tilde{O}(\sqrt{N})$ queries.[5] In fact, this is nearly optimal, as there are $\Omega(\sqrt{N})$ query lower bounds for both problems [22]. As a natural extension, we

[4] Here, the graph can be specified by an adjacency matrix (suitable for dense graphs) or by a collection of adjacency lists (for bounded-degree graphs).

[5] We use tilde notation to suppress logarithmic factors.

consider whether these problems can be solved more efficiently using *quantum* queries.

There has been some previous work on quantum property testing. In particular, there are examples of exponential separations between quantum and classical property testing [12], and there are quantum algorithms for testing juntas [9], solvability of black-box groups [24], uniformity and orthogonality of distributions [13, 14], and certain properties related to the Fourier transform [2, 14]. However, aside from concurrent work on testing graph isomorphism [14], we are not aware of previous work on quantum algorithms for testing properties of graphs.[6]

Here, we give quantum algorithms for testing bipartiteness and expansion of bounded-degree graphs in time only $\tilde{O}(N^{1/3})$, beating the $\Omega(\sqrt{N})$ classical lower bounds [22]. Moreover, we prove that any quantum algorithm for testing expansion must use $\tilde{\Omega}(N^{1/4})$ queries, showing that quantum computers cannot achieve a superpolynomial speedup for this problem.

Why might quantum computers offer an advantage for testing bipartiteness and expansion? The classical algorithms for these problems use random walks to explore the graph, so one might hope to do better by using quantum walks, which are a powerful tool for searching graphs [32]. In fact, our algorithms use quantum walks indirectly. The classical algorithm for testing bipartiteness is based on checking whether a pair of short random walks form an odd-length cycle in the graph, thereby certifying non-bipartiteness [20]. The algorithm for testing expansion looks for collisions between the endpoints of short random walks, with a large number of collisions indicating that the walk is not rapidly mixing [21]. In both cases, the property is tested by looking for collisions among a set of $\tilde{O}(\sqrt{N})$ items. By using the quantum walk algorithm for element distinctness [7, 27] to look for these collisions, we can solve the problem using $\tilde{O}(N^{1/3})$ quantum queries. In addition, we show that the above classical algorithms can be derandomized, using $O(\log N)$-wise independent bits. This yields quantum algorithms that run in time $\tilde{O}(N^{1/3})$.

While we have shown a polynomial quantum speedup, one may ask whether an exponential speedup is possible. Quantum computers can give at most a polynomial speedup for total functions [10], but this limitation does not apply to property testing (and indeed, examples of exponential speedup are known [12]). On the other hand, superpolynomial speedup is impossible for symmetric functions [3], even in the case of partial functions such as those arising in property testing. It is an interesting question whether exponential speedups are possible for testing *graph* properties, which may have significantly less symmetry.

Here we prove that testing expansion requires $\tilde{\Omega}(N^{1/4})$ quantum queries, thus ruling out the possibility of an exponential speedup. We use the polynomial method [10]—specifically, a technique of Aaronson based on reduction to a bivariate polynomial [1]. We define a distribution over N-vertex graphs with

[6] Quantum speedups are known for *deciding* certain graph properties, without the promise that the graph either has the property or is far from having it [17, 26, 15]. This turns out to be a fairly different setting, and the results there are not directly comparable to ours.

ℓ connected components (and with another parameter $M \approx N$), such that each component is an expander with high probability. With $\ell = 1$ component, such graphs are almost surely expanders, whereas graphs with $\ell \geq 2$ components are very far from expanders. Our main technical contribution is to show that the acceptance probability of any T-query quantum algorithm, when presented with this distribution, is well-approximated by a bivariate polynomial in M and ℓ of degree $O(T \log T)$. This requires a somewhat involved calculation of a closed-form expression for the acceptance probability as a function of M and ℓ, using algebraic techniques and the combinatorics of partitions. Then it follows by known results on polynomial approximation that $\Omega(N^{1/4}/\log N)$ queries are necessary to test expansion.

This proof may be of independent interest since there are very few techniques available to prove quantum lower bounds for property testing. In particular, the standard quantum adversary method [6] is subject to a "property testing barrier" [23]. Furthermore, graph structure makes it difficult to apply the polynomial method, so our lower bound for testing expansion requires substantial new machinery. These techniques may be applicable to other problems with graph structure. Note also that our approach is an alternative to the classical lower bounds for testing bipartiteness and expansion [22].

We are only aware of a few previous lower bounds for quantum property testing: the result that not all languages can be tested efficiently [12] (which is nonconstructive, using a counting argument), and lower bounds for testing orthogonality and uniformity of distributions [13, 14] and for testing graph isomorphism [14] (which follow by reduction from the collision problem).

Despite this progress, there remain many unanswered questions about quantum testing of graph properties. So far, we have been unable to prove a superconstant lower bound for testing bipartiteness. More generally, is there any graph property testing problem that admits an exponential quantum speedup?

In the remainder of this section, we define the model of quantum property testing. We use the adjacency-list model for graphs with bounded (i.e., constant) degree d. A graph $G = (V, E)$ is represented by a function $f_G \colon V \times \{1, \ldots, d\} \to V \cup \{*\}$, where $f_G(v, i)$ returns the i^{th} neighbor of v in G, or $*$ if v has fewer than i neighbors. A quantum computer is provided with a unitary black box that reversibly computes f_G as $|v, i, z\rangle \mapsto |v, i, z \oplus f_G(v, i)\rangle$. The query complexity of an algorithm is the number of calls it makes to the black box for f_G.

We say that G is *ε-far* from satisfying a property P if one must change at least εnd edges of G in order to satisfy P. We say that an algorithm *ε-tests* P if it accepts graphs that satisfy P with probability at least 2/3, and rejects graphs that are ε-far from satisfying P with probability at least 2/3. (More generally, we may consider algorithms that determine whether a graph satisfies P or is ε-far from satisfying a related property P'.)

We say that a graph G is an *α-expander* if for every $U \subseteq V$ with $|U| \leq |V|/2$, we have $|\partial(U)| \geq \alpha |U|$, where $\partial(U)$ is the set of vertices in $V - U$ adjacent to at least one vertex of U.

2 Quantum Algorithms for Bipartiteness and Expansion

First, recall the classical algorithm for testing bipartiteness [20]. This algorithm performs $T = \Theta(1/\varepsilon)$ repetitions, where during each repetition it chooses a random starting vertex s, then does $K = \sqrt{N}\,\mathrm{poly}(\frac{\log N}{\varepsilon})$ random walks from s, each of length $L = \mathrm{poly}(\frac{\log N}{\varepsilon})$, and looks for "collisions" where two walks from s reach the same vertex v, one after an even number steps, the other after an odd number of steps.

We derandomize each of the T repetitions separately. Each repetition uses $n = O(KL\log d)$ bits of randomness. We claim that it suffices to use k-wise independent random bits for some $k = O(L\log d)$. To see this, consider the analysis given in [20]. Lemma 4.5 of [20] states sufficient conditions for the algorithm to find an odd cycle, and hence reject, with high probability. The proof considers the random variable $X = \sum_{i<j} \eta_{ij}$, where η_{ij} is a Boolean random variable that indicates whether walk i collides with walk j while having different parity. The probability that $X = 0$ is upper bounded using Chebyshev's inequality together with bounds on $\mathrm{E}[X]$ and $\mathrm{Var}[X]$. Note that $\mathrm{E}[X]$ and $\mathrm{Var}[X]$ are linear and quadratic in the η_{ij}, respectively, so they only depend on sets of at most $O(L\log d)$ random bits. Thus they are unchanged by substituting k-wise independent random bits for some $k = O(L\log d)$. This reduces the number of random bits required by the algorithm to $O(k\log n) = O(\mathrm{poly}(\frac{\log N \log d}{\varepsilon}))$.

We then combine this derandomized classical algorithm with Ambainis' quantum algorithm for element distinctness [7, 27, 35]. (For details, see the full version of this paper [8].) This shows

Theorem 1. *There is a quantum algorithm that always returns "true" when G is bipartite, returns "false" with constant probability when G is ε-far from bipartite, and runs in time $O(N^{1/3}\,\mathrm{poly}(\frac{\log N}{\varepsilon}))$.*

Using similar ideas, we can also give an $\tilde{O}(N^{1/3})$-time quantum algorithm for testing expansion. We start with the classical algorithm of [21], derandomize it using k-wise independent random variables, and apply the quantum algorithm for element distinctness. There is a slight complication, because we need to count collisions, not just detect them. However, the number of collisions is small—roughly $O(N^{2\mu})$ where μ is chosen to be a small constant—so we can count the collisions using brute force. See [8] for details.

3 Quantum Lower Bound for Testing Expansion

3.1 Overview

We now turn to lower bounds for testing expansion. Specifically, we prove

Theorem 2. *Any quantum algorithm for testing expansion of bounded-degree graphs must use $\Omega(N^{1/4}/\log N)$ queries.*

Proof. We consider random graphs G on N vertices, sampled from the following distribution $P_{M,l}$ (where $M \geq N$ and l divides M):

1. We start by constructing a random graph G' on M vertices, as follows: First, we partition the vertices into l sets $V_1, \ldots, V_l$, with each set V_i containing M/l vertices. Then, on each set V_i, we create a random subgraph by randomly choosing c perfect matchings on V_i and taking their union. (Here c is some sufficiently large constant.)
2. We then construct G as follows: First, we pick a subset of vertices $v_1, \ldots, v_N$ from G'. To pick v_1, we choose one of the sets $V_1, \ldots, V_l$ uniformly at random, call it V_j, and we let v_1 be a random vertex from V_j. For each subsequent vertex v_i, we again select a set V_j uniformly at random, and choose v_i uniformly at random among those vertices of V_j that were not chosen in the previous steps. Then we let G be the induced subgraph of G' on $v_1, \ldots, v_N$.

The process above fails if we try to choose more than M/l vertices from the same V_j. However, the probability of that happening is small—on average, N/l vertices are chosen in each V_j. We choose $M = (1 + \Theta(N^{-0.1}))N$. Then a straightforward application of Chernoff bounds implies that the process fails with probability at most $e^{-\Omega(N^{0.55})}$. For more detail, see Section C.1 in [8].

Note that the resulting graph G has degree at most c. The reason for choosing G as a subgraph of G' (rather than constructing G directly) is that this leads to simpler formulas for the probabilities of certain events, e.g., the probability that vertices v_1, v_2 and v_3 all belong to the same component of G is $1/l^2$. This seems essential for our use of the polynomial method.

If $l = 1$, then this process generates an expander with high probability. It is well known [31, 28] that the graph on M vertices generated by taking c perfect matchings is an expander with high probability. In Section C.2 in [8], we show that the subgraph that we choose is also an expander. (Informally, the main reason is that only a $\Theta(N^{-1/4})$ fraction of the vertices of G' are not included in G. This allows us to carry out the proof of [31, 28] without substantial changes.)

If $l = 2$, then this process generates a disconnected graph with two connected components, each of size roughly $N/2$. Such a graph is very far from any expander graph—specifically, for any α', it is at least about $(\alpha'/2d)$-far from an α'-expander of maximum degree d.

Therefore, if a quantum algorithm tests expansion, it must accept a random graph generated according to $P_{M,1}$ with probability at least $2/3$, and a random graph generated according to $P_{M,2}$ with probability at most $1/3$. (Graphs drawn from $P_{M,l}$ with $l > 2$ must also be accepted with probability at most $1/3$, although this fact is not used in the analysis.)

The strategy of the proof is as follows. We show that for any quantum algorithm run on a random graph from the distribution $P_{M,l}$, the acceptance probability of the algorithm can be approximated by a bivariate polynomial in M and l, where the number of queries used by the algorithm corresponds to the degree of this polynomial. (This is our main technical contribution.) We then lower bound the degree of this polynomial.

In more detail, we will prove the following lemma (see Section 3.2):

Lemma 1. *Let A be a quantum algorithm using T queries. The acceptance probability of A (for the probability distribution $P_{M,l}$) is approximated (up to an additive error of $e^{-\Omega(N^{0.55})}$) by a fraction $\frac{f(M,l)}{g(M,l)}$, where $f(M,l)$ and $g(M,l)$ are polynomials of degree $O(T \log T)$ and $g(M,l)$ is a product of factors $(M-(2k-1)l)$ for $k \in \{1, \dots, T\}$, with $(M-(2k-1)l)$ occurring at most $2T/k$ times.*

Now choose $a = 1 + \Theta(N^{-0.1})$ such that aN is even. We say that a pair (M,l) is δ-*good* if $M \in [aN - \delta^{3/2}, aN + \delta^{3/2}]$, $l \leq \delta$, and l divides M.

We then approximate the fraction $\frac{f(M,l)}{g(M,l)}$ (from Lemma 1) by $\frac{f(M,l)}{(aN)^{\deg g(M,l)}}$. For each term $M-(2k-1)l$, we first replace it by M and then by aN. The first step introduces multiplicative error of $1 - \frac{(2k-1)l}{M} \geq 1 - \frac{2kl}{N} \approx e^{-2kl/N}$. For all terms together, the error introduced in this step is at most $\prod_{k=1}^{T} (e^{-2kl/N})^{2T/k} = e^{-4T^2 l/N}$. If $T = O(N^{1/4}/\log N)$ and $l = O(N^{1/2})$, the multiplicative error is $1 - o(1)$.

The second approximation step introduces multiplicative error of

$$(\frac{M}{aN})^{O(T\log T)} \approx (e^{(M-aN)/aN})^{O(T\log T)} \leq (e^{\delta^{3/2}/aN})^{O(T\log T)}.$$

If $\delta = O(N^{1/2})$ and $T = O(N^{1/4}/\log N)$, this can be upper bounded by $1+\epsilon$ for arbitrarily small $\epsilon > 0$, by appropriately choosing the big-O constant in $T = O(N^{1/4}/\log N)$.

Next, we prove a second lemma, which lower bounds the degree of a bivariate polynomial:

Lemma 2. *Let $f(M,l)$ be a polynomial such that $|f(aN,1) - f(aN,2)| \geq \epsilon$ for some fixed $\epsilon > 0$ and, for any δ-good (M,l), $|f(M,l)| \leq 1$. Then the degree of $f(M,l)$ is $\Omega(\sqrt{\delta})$.*

The proof of this lemma follows the collision lower bounds of Aaronson and Shi [1, 33] and is included in Section C.3 in [8] for completeness.

We now set $\delta = \Theta(N^{1/2})$ and apply Lemma 2 to $\frac{f(M,l)}{2(aN)^{\deg g(M,l)}}$. This is a polynomial in M and ℓ, because the denominator is a constant. With $M = aN$, its values at $l = 1$ and $l = 2$ are bounded away from each other by at least $1/3$ since the algorithm works. Its values at δ-good pairs (M,l) have magnitude at most 1 because the acceptance probability of the algorithm is in $[0,1]$, so $|\frac{f(M,l)}{2(aN)^{\deg g(M,l)}}| \leq \frac{1}{2} + o(1)$. Thus we find that the degree of $f(M,l)$ must be $\Omega(N^{1/4})$. It follows that $T = \Omega(N^{1/4}/\log N)$ queries are necessary.

3.2 Proof of Lemma 1

Here we assume that the process generating a graph G from the probability distribution $P_{M,l}$ does not fail. (The effect of this process possibly failing is considered in Section C.1 in [8].) The acceptance probability of A is a polynomial P_A of degree at most $2T$ in Boolean variables $x_{u,v,j}$, where $x_{u,v,j} = 1$ iff (u,v) is an edge in the j^{th} matching.

P_A is a weighted sum of monomials. It suffices to show that the expectation of every such monomial has the rational form described in Lemma 1. If this is shown, then $\mathrm{E}[P_A]$ is a sum of such fractions: $\mathrm{E}[P_A] = \frac{f_1(M,l)}{g_1(M,l)} + \frac{f_2(M,l)}{g_2(M,l)} + \cdots$. We put these fractions over a common denominator, obtaining $\mathrm{E}[P_A] = \frac{f(M,l)}{g(M,l)}$ where $g(M,l) = \mathrm{lcm}(g_1(M,l), g_2(M,l), \ldots)$. In this common denominator, $(M-(2k-1)l)$ occurs at most $2T/k$ times. Therefore, the degree of $g(M,l)$ is at most $2T\sum_{k=1}^{2T}\frac{1}{k} = O(T\log T)$. Similarly, the degree of $f(M,l)$ is at most $O(T\log T) + \deg g(M,l) = O(T\log T)$.

Now consider a particular monomial $P = x_{u_1,v_1,j_1}x_{u_2,v_2,j_2}\cdots x_{u_d,v_d,j_d}$, where $d = \deg P$. Let G_P be the graph with edges $(u_1,v_1),\ldots,(u_d,v_d)$ (i.e., with the edges relevant to P) where the edge (u_a,v_a) comes from the j_a^{th} matching. Let $C_1,\ldots,C_k$ be the connected components of G_P. For each component C_i, let X_i be the event that every edge (u_a,v_a) in C_i (viewed as a subgraph of G_P) is present in the random graph G as part of the j_a^{th} matching. We have to find an expression for the expectation

$$\mathrm{E}[P] = \Pr[X_1 \cap X_2 \cap \ldots \cap X_k].$$

We first consider $\Pr[X_i]$. Let v_i be the number of vertices in C_i, and for each matching j, let $d_{i,j}$ be the number of variables $x_{u,v,j}$ in P that have $u,v\in C_i$ and label j. Note that

$$d_{i,1} + d_{i,2} + \cdots + d_{i,c} \geq v_i - 1 \tag{1}$$

because a connected graph with v_i vertices must have at least $v_i - 1$ edges. We have

$$\Pr[X_i] = \frac{1}{l^{v_i-1}}\prod_{j=1}^{c}\prod_{j'=1}^{d_{i,j}}\frac{1}{M/l-(2j'-1)} = \frac{1}{l^{v_i-1}}\prod_{j=1}^{c}\prod_{j'=1}^{d_{i,j}}\frac{l}{M-(2j'-1)l}. \tag{2}$$

Here $l^{-(v_i-1)}$ is the probability that all v_i vertices are put into the same set V_j (for some $1\leq j\leq l$) (which is a necessary condition for having edges among them), and $\prod_{j'=1}^{d_{i,j}}\frac{1}{M/l-(2j'-1)}$ is the probability that $d_{i,j}$ particular edges from the j^{th} matching are present. (For the first edge (u,v) in the j^{th} matching, the probability that it is present is $\frac{1}{M/l-1}$, since u is equally likely to be matched with any of M/l vertices in V_j except for u itself; for the second edge (u',v') in the j^{th} matching, the probability that it is present is $\frac{1}{M/l-3}$, since u' can be matched with any of M/l vertices except u,v,u'; and so on. Note that without loss of generality, we can assume that the edges in P from the j^{th} matching are distinct. If P contains the same edge twice from the same matching, then we can remove one of the duplicates without changing the value of P.)

We can rewrite (2) as $\Pr[X_i] = \frac{1}{l^{v_i-1}}\prod_{j=1}^{c}R_{d_{i,j}}$, where we define

$$R_d = \prod_{j'=1}^{d}\frac{l}{M-(2j'-1)l}. \tag{3}$$

We now extend this to deal with multiple components C_i at once, i.e., we want to evaluate $\Pr[\bigcap_{i \in S} X_i]$, where $S \subseteq \{1, \ldots, k\}$. Let E_S be the event that the vertices in $\bigcup_{i \in S} C_i$ (i.e., in any of the components indicated by S) are all put into one set V_j. Then $\Pr[\bigcap_{i \in S} X_i | E_S] = \prod_{j=1}^{c} R_{\sum_{i \in S} d_{i,j}}$. The event E_S happens with probability $l^{-(\sum_{i \in S} v_i)+1}$, since the total number of vertices in $\bigcup_{i \in S} C_i$ is $\sum_{i \in S} v_i$.

Let $L = (S_1, \ldots, S_t)$ be a partition of $\{1, 2, \ldots, k\}$. We call $S_1, \ldots, S_t$ *classes* of the partition L. We say that $S \in L$ if S is one of $S_1, \ldots, S_t$. Let $|L| = t$. We say that L is a refinement of L' (denoted $L < L'$) if L can be obtained from L' by splitting some of the classes of L' into two or more parts. We write $L \leq L'$ if $L < L'$ or $L = L'$. When $L < L'$, let $c_{L,L'}$ be the number of sequences $L = L_0 < L_1 < \cdots < L_j = L'$, with sequences of even length j counting as $+1$ and sequences of odd length j counting as -1. We define $c_{L,L'} = 1$ when $L = L'$. We have the following partition identity, which will be useful later; the proof is given in Section C.4 in [8].

Proposition 1 *Suppose $L'' < L$. Then $\sum_{L' :\, L'' \leq L' \leq L} c_{L',L} = 0$.*

We define the expressions

$$f_L(M, l) = \prod_{S \in L} \prod_{j=1}^{c} R_{\sum_{i \in S} d_{i,j}} \tag{4}$$

$$f'_L(M, l) = \sum_{L' :\, L' \leq L} c_{L',L} f_{L'}(M, l). \tag{5}$$

We can now evaluate $\Pr[X_1 \cap X_2 \cap \ldots \cap X_k]$ as follows. For any partition L of $\{1, 2, \ldots, k\}$, let E_L be the event $\bigcap_{S \in L} E_S$. Let E'_L be the event that E_L happens but no $E_{L'}$ with $L < L'$ happens (i.e., L is the least refined partition that describes the event). Then

$$\Pr[X_1 \cap X_2 \cap \ldots \cap X_k] = \sum_{L} \Pr[E'_L] f_L(M, l).$$

By inclusion-exclusion, $\Pr[E'_L] = \sum_{L' :\, L \leq L'} c_{L,L'} \Pr[E_{L'}]$. Now substitute into the previous equation, reorder the sums, and use the definition of $f'_L(M, l)$:

$$\Pr[X_1 \cap X_2 \cap \ldots \cap X_k] = \sum_{L'} \Pr[E_{L'}] \sum_{L :\, L \leq L'} c_{L,L'} f_L(M, l) = \sum_{L} \Pr[E_L] f'_L(M, l).$$

Note that $\Pr[E_L] = \prod_{S \in L} \Pr[E_S] = \prod_{S \in L} l^{-(\sum_{i \in S} v_i)+1} = l^{-(\sum_{i=1}^{k} v_i)+|L|}$. Thus we have

$$\Pr[X_1 \cap X_2 \cap \ldots \cap X_k] = \sum_{L} l^{-(\sum_{i=1}^{k} v_i)+|L|} f'_L(M, l). \tag{6}$$

We have now written $\Pr[X_1 \cap X_2 \cap \ldots \cap X_k]$ as a sum of rational functions of M and l. We can combine these into a single fraction $\frac{f(M,l)}{g(M,l)}$. It remains to show that this fraction has the properties claimed in Lemma 1.

First, we claim that the denominator $g(M,l)$ contains at most $2T/k$ factors of $M-(2k-1)l$. Observe that each $f_L(M,l)$ is a fraction whose denominator consists of factors $M-(2k-1)l$. The number of factors in the denominator is equal to the number of variables in the monomial P, which is at most $2T$. By the form of (3), for each $M-(2k-1)l$ in the denominator, we also have $M-l$, $M-3l$, …, $M-(2k-3)l$ in the denominator. Therefore, if we have t factors of $M-(2k-1)l$ in the denominator, then the total degree of the denominator is at least tk. Since $tk \leq 2T$, we have $t \leq 2T/k$. This statement holds for $f_L(M,l)$ for every L. Thus, when we sum the $f_L(M,l)$ to obtain first $f'_L(M,l)$ and then $\Pr[X_1 \cap X_2 \cap \ldots \cap X_k]$, and put all the terms over a common denominator $g(M,l)$, this statement also holds for $g(M,l)$.

In $\Pr[X_1 \cap X_2 \cap \ldots \cap X_k]$, when we sum the $f'_L(M,l)$ in (6), we also have factors of $l^{(\sum_{i=1}^k v_i)-|L|}$ in the denominator. Proposition 2 shows that these factors are cancelled out by corresponding factors in the numerator.

Proposition 2 *$f'_L(M,l)$ is equal to a fraction whose denominator is a product of factors $(M-(2k-1)l)$ and whose numerator is divisible by $l^{(\sum_{i=1}^k v_i)-|L|}$.*

When we combine the different $f'_L(M,l)$ in (6) into a single fraction $\frac{f(M,l)}{g(M,l)}$, we see that f and g have the desired form. Also note that f and g have degree $O(T \log T)$, by repeating the same argument used earlier to combine the different monomials P. This completes the proof of Lemma 1; it remains to show Proposition 2.

Proof (of Proposition 2). Note that R_d contains an obvious factor of l^d. We define

$$R'_d = \frac{R_d}{l^d} = \prod_{j'=1}^{d} \frac{1}{M-(2j'-1)l}$$

and we redefine $f_L(M,l)$ and $f'_L(M,l)$ (equations (4) and (5)) using R'_d instead of R_d. This removes a factor of l^d from the numerator of R_d and a factor of $l^{\sum_{i,j} d_{i,j}}$ from the numerator of $f_L(M,l)$. By equation (1), this factor is at least $l^{(\sum_i v_i)-k}$. Therefore, it remains to show that the numerator of the redefined $f'_L(M,l)$ is divisible by $l^{k-|L|}$.

Recall that $f'_L(M,l)$ is a sum of terms $f_{L'}(M,l)$ for all $L' \leq L$. Let us write each term as $f_{L'}(M,l) = 1/\prod_{k \in K(L')}(M-kl)$, where $K(L')$ is a multiset. We put these terms over a common denominator $\beta_L(M,l) = \prod_{k \in B(L)}(M-kl)$, where $B(L) \supseteq K(L')$ for all $L' \leq L$. Then we have

$$f_{L'}(M,l) = \frac{\alpha_{L'}(M,l)}{\beta_L(M,l)}, \qquad \alpha_{L'}(M,l) = \prod_{k \in B(L)-K(L')} (M-kl),$$

$$f'_L(M,l) = \frac{\alpha'_L(M,l)}{\beta_L(M,l)}, \qquad \alpha'_L(M,l) = \sum_{L' :\, L' \leq L} c_{L',L}\,\alpha_{L'}(M,l).$$

Let $m = |B(L)|$. Also, let $\tilde{m} = |K(L')| = \sum_{S \in L'} \sum_{j=1}^{c} \sum_{i \in S} d_{i,j} = \sum_{i=1}^{k} \sum_{j=1}^{c} d_{i,j}$, which is independent of L'. Let $m' = |B(L) - K(L')| = m - \tilde{m}$, which depends on L but not on L'.

We want to show that $\alpha'_L(M,l)$ is divisible by $l^{k-|L|}$. First, we multiply out each term $\alpha_{L'}(M,l)$ to get $\alpha_{L'}(M,l) = \sum_{i=0}^{m'} e_i(B(L) - K(L'))M^{m'-i}(-l)^i$, where e_i is the i^{th} elementary symmetric polynomial (i.e., $e_i(B(L) - K(L'))$ is the sum of all products of i variables chosen without replacement from the multiset $B(L) - K(L')$). We can then write $\alpha'_L(M,l)$ as

$$\alpha'_L(M,l) = \sum_{i=0}^{m'} \theta_{L,i} M^{m'-i}(-l)^i, \qquad \theta_{L,i} = \sum_{L' \colon L' \leq L} c_{L',L} e_i(B(L) - K(L')).$$

It suffices to show that, for all $0 \leq i \leq k - |L| - 1$, the coefficient $\theta_{L,i}$ is 0. Note that if L is the finest possible partition L_*, then $|L| = k$ and the above claim is vacuous, so we can assume that $L_* < L$. Also note that $\theta_{L,0} = 0$ by Proposition 1 with $L'' = L_*$, so it suffices to consider $i > 0$.

For any set of variables E and any $a \geq 0$, define the power-sum polynomial $T_a(E) = \sum_{k \in E} k^a$. We can write $e_i(B(L) - K(L'))$ in terms of power sums:

$$e_i(B(L) - K(L')) = \Lambda_{i,L}[T_a(B(L) - K(L')) \colon a = 0, 1, 2, \ldots, i],$$

where $\Lambda_{i,L}$ is a polynomial function of the power sums $T_a(B(L) - K(L'))$ of total degree i in the variables $k \in B(L) - K(L')$. Note that the polynomial $\Lambda_{i,L}$ only depends on the size of the set $B(L) - K(L')$, hence it only depends on L, and not on L'. To simplify things, we can write $T_a(B(L) - K(L')) = T_a(B(L)) - T_a(K(L'))$ and absorb the $T_a(B(L))$ term into the polynomial $\Lambda_{i,L}$ to get a new polynomial $\tilde{\Lambda}_{i,L}$. Then we have $e_i(B(L) - K(L')) = \tilde{\Lambda}_{i,L}[T_a(K(L')) \colon a = 0, 1, 2, \ldots, i]$, and

$$\theta_{L,i} = \sum_{L' \colon L' \leq L} c_{L',L} \tilde{\Lambda}_{i,L}[T_a(K(L')) \colon a = 0, 1, 2, \ldots, i].$$

It suffices to show that, for all $0 \leq i \leq k - |L| - 1$, the above sum vanishes term-by-term, i.e., for all sequences $\{a_j\}$ such that $a_j \geq 0$ and $\sum_j a_j \leq i$, we have

$$\sum_{L' \colon L' \leq L} c_{L',L} \prod_j T_{a_j}(K(L')) = 0. \tag{7}$$

We have $T_a(K(L')) = \sum_{S \in L'} \sum_{j=1}^{c} T_a(\{1, 3, 5, \ldots, 2(\sum_{i \in S} d_{i,j}) - 1\})$, by the definition of $K(L')$. Note that, for any integer s, $T_a(\{1, 3, 5, \ldots, 2s - 1\}) = T_a(\{1, 2, 3, \ldots, 2s\}) - 2^a T_a(\{1, 2, 3, \ldots, s\})$, and by Faulhaber's formula, this equals a polynomial $Q_a(s)$ of degree $a + 1$, with rational coefficients and no constant term. We have $T_a(K(L')) = \sum_{S \in L'} \sum_{j=1}^{c} Q_a(\sum_{i \in S} d_{i,j})$. Let $q_{a,\alpha}$ ($\alpha = 1, \ldots, a + 1$) be the coefficients of Q_a. Then we can rewrite this as

$$T_a(K(L')) = \sum_{\alpha=1}^{a+1} q_{a,\alpha} S_\alpha(L'), \text{ where } S_\alpha(L') = \sum_{S \in L'} \sum_{j=1}^{c} \Big(\sum_{i \in S} d_{i,j}\Big)^\alpha.$$

It suffices to show that the sum in equation (7) vanishes term-by-term, i.e., for all $0 \leq i \leq k - |L| - 1$ and for all sequences $\{\alpha_j\}$ such that $\alpha_j \geq 1$ and $\sum_j (\alpha_j - 1) \leq i$, we have

$$\sum_{L' :\, L' \leq L} c_{L',L} \prod_j S_{\alpha_j}(L') = 0.$$

This final claim is shown in Section C.5 in [8]. This completes the proof of Proposition 2.

Acknowledgments. We thank the anonymous referees for several helpful comments. AA was supported by ESF project 1DP/1.1.1.2.0/09/APIA/VIAA/044, FP7 Marie Curie Grant PIRG02-GA-2007-224886 and FP7 FET-Open project QCS. AMC and YKL acknowledge the hospitality of the Kavli Institute for Theoretical Physics, where this research was supported in part by the National Science Foundation under Grant No. PHY05-51164. AMC received support from MITACS, NSERC, QuantumWorks, and the US ARO/DTO. YKL received support from an NSF postdoctoral fellowship and ARO/NSA. This work was done in part while YKL was at the Institute for Quantum Information at Caltech.

References

[1] S. Aaronson. Quantum lower bound for the collision problem. In STOC, pages 635-642. 2002.

[2] S. Aaronson. BQP and the polynomial hierarchy. In STOC, pages 141-150. 2010.

[3] S. Aaronson and A. Ambainis. The need for structure in quantum speedups. In Innovations in Computer Science, pages 338-352. 2011.

[4] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. J. of Algorithms 7 (4):567-583, 1986.

[5] N. Alon, O. Goldreich, J. Hastad, and R. Peralta. Simple constructions of almost k-wise independent random variables. Random Structures and Algorithms 3 (3):289-304, 1992.

[6] A. Ambainis. Quantum lower bounds by quantum arguments. J. of Computer and System Sciences 64 (4):750-767, 2002.

[7] A. Ambainis. Quantum walk algorithm for element distinctness. SIAM J. on Computing 37 (1):210-239, 2007.

[8] A. Ambainis, A.M. Childs and Y.-K. Liu, Quantum property testing for bounded-degree graphs, arXiv:1012.3174, 2010.

[9] A. Atici and R. Servedio. Quantum algorithms for learning and testing juntas. Quantum Information Processing 6 (5):323-348, 2007.

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

[11] H. Buhrman, C. Durr, M. Heiligman, P. Hoyer, F. Magniez, M. Santha, and R. de Wolf. Quantum algorithms for element distinctness. SIAM J. on Computing 34 (6):1324-1330, 2005.

[12] H. Buhrman, L. Fortnow, I. Newman, and H. Rohrig. Quantum property testing. SIAM J. on Computing 37 (5):1387-1400, 2008.

[13] S. Bravyi, A. W. Harrow, and A. Hassidim. Quantum algorithms for testing properties of distributions. In STACS, pages 131-142, 2010.

[14] S. Chakraborty, E. Fischer, A. Matsliah, and R. de Wolf. New results on quantum property testing. In FSTTCS, pages 145-156. 2010.

[15] A. M. Childs and R. Kothari. Quantum query complexity of minor-closed graph properties. To appear in STACS. 2011.

[16] A. Czumaj and C. Sohler. Testing expansion in bounded-degree graphs. In FOCS, pages 570-578. 2007.

[17] C. Durr, M. Heiligman, P. Hoyer, and M. Mhalla. Quantum query complexity of some graph problems. SIAM J. on Computing 35 (6):1310-1328, 2006.

[18] O. Goldreich. Randomized Methods in Computation, 2001. Lecture notes available at http://www.wisdom.weizmann.ac.il/∼oded/rnd.html, Lecture 2.

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

[20] O. Goldreich and D. Ron. A sublinear bipartiteness tester for bounded degree graphs. Combinatorica 19 (3):335-373, 1999.

[21] O. Goldreich and D. Ron. On testing expansion in bounded-degree graphs, 2000. ECCC report TR00-020.

[22] O. Goldreich and D. Ron. Property testing in bounded degree graphs. Algorithmica 32 (2):302-343, 2002.

[23] P. Hoyer, T. Lee, and R. Spalek. Negative weights make adversaries stronger. In STOC, pages 526-535. 2007.

[24] Y. Inui and F. L. Gall. Quantum property testing of group solvability. In LATIN, pages 772-783. 2008.

[25] S. Kale and C. Seshadhri. Testing expansion in bounded-degree graphs, 2007. ECCC report TR07-076.

[26] F. Magniez, M. Santha, and M. Szegedy. Quantum algorithms for the triangle problem. SIAM J. on Computing 37 (2):413-424, 2007.

[27] F. Magniez, A. Nayak, J. Roland, and M. Santha. Search via quantum walk. In STOC, pages 575-584. 2007.

[28] R. Motwani and P. Raghavan. Randomized Algorithms, 1995. Cambridge University Press.

[29] A. Nachmias and A. Shapira. Testing the expansion of a graph. Information and Computation 208:309-314, 2010.

[30] R. Paturi. On the degree of polynomials that approximate symmetric Boolean functions (preliminary version). In STOC, pages 468-474. 1992.

[31] M. Pinsker. On the complexity of a concentrator. In Proceedings of the 7th International Teletraffic Conference, pages 318/1-318/4. 1973.

[32] M. Santha. Quantum walk based search algorithms. In Theory and Applications of Models of Computation, pages 31-46. 2008.

[33] Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. In FOCS, pages 513-519. 2002.

[34] D. R. Simon. On the power of quantum computation. SIAM J. on Computing 26 (5):1474-1483, 1997.

[35] M. Szegedy. Quantum speed-up of Markov chain based algorithms. In FOCS, pages 32-41. 2004.