THEORY OF COMPUTING, Volume 12 (18), 2016, pp. 1–35
www.theoryofcomputing.org

Upper Bounds on Quantum Query Complexity Inspired by the Elitzur–Vaidman Bomb Tester

Cedric Yen-Yu Lin   Han-Hsuan Lin

Received August 17, 2015; Revised February 29, 2016; Published November 28, 2016

Abstract: Inspired by the Elitzur–Vaidman bomb testing problem (1993), we introduce a new query complexity model, which we call *bomb query complexity*, $B(f)$. We investigate its relationship with the usual quantum query complexity $Q(f)$, and show that $B(f) = \Theta(Q(f)^2)$.

This result gives a new method to derive upper bounds on quantum query complexity: we give a method of finding bomb query algorithms from classical algorithms, which then provide non-constructive upper bounds on $Q(f) = \Theta(\sqrt{B(f)})$. Subsequently, we were able to give explicit quantum algorithms matching our new bounds. We apply this method to the single-source shortest paths problem on unweighted graphs, obtaining an algorithm with $O(n^{1.5})$ quantum query complexity, improving the best known algorithm of $O(n^{1.5}\log n)$ (Dürr et al. 2006, Furrow 2008). Applying this method to the maximum bipartite matching problem gives an algorithm with $O(n^{1.75})$ quantum query complexity, improving the best known (trivial) $O(n^2)$ upper bound.

ACM Classification: F.1.2, F.1.3, F.2.2

AMS Classification: 81P68, 68Q12, 68Q17, 68Q05

Key words and phrases: algorithms, query complexity, quantum algorithms, quantum query complexity, graph algorithms, Elitzur-Vaidman bomb tester, adversary method, maximum bipartite matching

A conference version of this paper appeared in the Proceedings of the 30th Computational Complexity Conference, 2015 [37].

 DOI: 10.4086/toc.2016.v012a018

1 Introduction

Quantum query complexity is an important method of understanding the power of quantum computers. In this model we are given a black box containing a Boolean string $x = x_1 \cdots x_N$, and we would like to calculate some function $f(x)$ with as few quantum accesses to the black box as possible. It is often easier to give bounds on the query complexity than on the time complexity of a problem, and insights from the former often prove useful in understanding the power and limitations of quantum computers. One famous example is Grover's algorithm for unstructured search on a set of size N [25]; by treating the unstructured database as a black box to be queried, it was shown that $\Theta(\sqrt{N})$ queries were required [9], proving that Grover's algorithm's $O(\sqrt{N})$ *time* complexity is optimal.

Several methods have been proposed to bound the quantum query complexity. Upper bounds are almost always proved by finding better query algorithms. Some general methods of constructing quantum algorithms have been proposed, such as quantum walks [4, 50, 38, 31] and learning graphs [7]. For lower bounds, the main methods are the polynomial method [6] and the adversary method [3]. In particular, the general adversary lower bound [30] has been shown to tightly characterize quantum query complexity [47, 48, 36], but calculating such a tight bound seems difficult in general. Nevertheless, the general adversary lower bound is valuable as a theoretical tool, for example in proving composition theorems [48, 36, 33] or showing non-constructive (!) upper bounds [33].

Our work

To improve our understanding of quantum query complexity, we introduce and study an alternative oracle model, which we call the *bomb oracle* (see Section 3 for the precise definition). Our model is inspired by the concept of *interaction-free measurements*, illustrated vividly by the Elitzur-Vaidman bomb testing problem [22], in which a property of a system can be measured without disturbing the system significantly. Like the quantum oracle model, in the bomb oracle model we want to evaluate a function $f(x)$ on a hidden Boolean string $x = x_1 \cdots x_N$ while querying the oracle as few times as possible. In this model, however, the bomb oracle is a controlled quantum oracle with the extra requirement that the algorithm fails if the controlled query returns a 1. This seemingly impossible task can be tackled using the quantum Zeno effect [40], in a fashion similar to the Elitzur-Vaidman bomb tester [35] (Section 2.2).

Our main result (Theorem 4.1) is that the bomb query complexity, $B(f)$, is characterized by the square of the quantum query complexity $Q(f)$:

Theorem 4.1. $B(f) = \Theta(Q(f)^2)$.

We prove the upper bound, $B(f) = O(Q(f)^2)$ (Theorem 4.3), by adapting Kwiat et al.'s solution of the Elitzur-Vaidman bomb testing problem (Section 2.2, [35]) to our model. We prove the lower bound, $B(f) = \Omega(Q(f)^2)$ (Theorem 4.5), by demonstrating that $B(f)$ is lower bounded by the square of the general adversary bound [30], $(\mathrm{Adv}^{\pm}(f))^2$. The aforementioned result that the general adversary bound tightly characterizes the quantum query complexity [47, 48, 36], $Q(f) = \Theta(\mathrm{Adv}^{\pm}(f))$, allows us to draw our conclusion.

The characterization of query complexity given in Theorem 4.1 allows us to give *non-constructive* upper bounds on the quantum query complexity for some problems. For some functions f a bomb query algorithm is easily designed by adapting a classical algorithm. Specifically, we show the following (stated informally).

Theorem 5.4. *Suppose there is a classical algorithm that computes $f(x)$ in T queries, and the algorithm guesses the result of each query (0 or 1), making at most an expected G mistakes for all x. Then we can design a bomb query algorithm that uses $O(TG)$ queries, and hence $B(f) = O(TG)$. By our characterization of Theorem 4.1, $Q(f) = O(\sqrt{TG})$.*

This result inspired us to look for an explicit quantum algorithm that reproduces the query complexity $O(\sqrt{TG})$. We were able to do so.

Theorem 5.5. *Under the assumptions of Theorem 5.4, there is an explicit algorithm (Algorithm 5.9) for f with query complexity $O(\sqrt{TG})$.*

Using Algorithm 5.9, we were able to give an $O(n^{3/2})$ algorithm for the single-source shortest paths (SSSP) problem in an unweighted graph with n vertices, beating the best-known $O(n^{3/2}\log n)$ algorithm [21, 24]. A more striking application is our $O(n^{7/4})$ algorithm for maximum bipartite matching; in this case the best-known upper bound was the trivial $O(n^2)$, although the time complexity of this problem had been studied in [5] (and similar problems in [19]).

Finally, in Section 7 we briefly discuss a related query complexity model, which we call the *projective query complexity* $P(f)$, in which each quantum query to x is immediately followed by a classical measurement of the query result. This model seems interesting to us because its power lies between classical and quantum: we observe that $Q(f) \leq P(f) \leq R(f)$, where $R(f)$ is the classical randomized query complexity, and $P(f) \leq B(f) = \Theta(Q(f)^2)$. We note that Regev and Schiff [46] showed that $P(\mathrm{OR}) = \Theta(N)$.

Past and related work

Mitchison and Jozsa have proposed a different computational model called *counterfactual computation* [41], also based on interaction-free measurement. In counterfactual computation the result of a computation may be learnt without ever running the computer. There has been some discussion on what constitutes counterfactual computation; see for example [29, 42, 28, 51, 27, 49, 52].

There have also been other applications of interaction-free measurement to quantum cryptography. For example, Noh has proposed counterfactual quantum cryptography [45], where a secret key is distributed between parties, even though a particle carrying secret information is not actually transmitted. More recently, Brodutch et al. proposed an adaptive attack [13] on Wiesner's quantum money scheme [53]; this attack is directly based off Kwiat et al.'s solution of the Elitzur-Vaidman bomb testing problem [35].

Our Algorithm 5.9 is very similar to Kothari's algorithm for the oracle identification problem [34], and we refer to his analysis of the query complexity in our paper.

The projective query model we detail in Section 7 was, to our knowledge, first considered by Aaronson in unpublished work in 2002 [1].

Discussion and outlook

Our work raises a number of open questions. The most obvious ones are those pertaining to the application of our recipe for turning classical algorithms into bomb algorithms, Theorem 5.4.

- Can we generalize our method to handle non-Boolean input and output? If so, we might be able to find better upper bounds for the adjacency-list model, or to study graph problems with weighted edges.

- Can our explicit (through Theorem 5.5) algorithm for maximum bipartite matching be made more *time* efficient? The best known quantum algorithm for this task has time complexity $O(n^2 \log n)$ in the adjacency matrix model [5].

- Finally, can we find more upper bounds using Theorem 5.4? For example, could the query complexity of the maximum matching problem on general nonbipartite graphs be improved with Theorem 5.4, by analyzing the classical algorithm of Micali and Vazirani [39]?

Perhaps more fundamental, however, is the possibility that the bomb query complexity model will help us understand the relationship between the classical randomized query complexity, $R(f)$, and the quantum query complexity $Q(f)$. It is known [6] that for all total functions f, $R(f) = O(Q(f)^6)$; the current best known separation is $R(f) = \tilde{\Omega}(Q(f)^{2.5}) = \tilde{\Omega}(B(f)^{1.125})$, for a function f discovered by Ben-David [8, 2] after the initial appearance of our paper. Some more open questions, then, are the following.

- Can we say something about the relationship between $R(f)$ and $B(f)$ for specific classes of functions? For example, is $R(f) = O(B(f)^2)$ for total functions?

- Referring to the notation of Theorem 5.4, we have $B(f) = O(TG)$. Is the quantity G related to other measures used in the study of classical decision-tree complexity, for example the certificate complexity, sensitivity [16], block sensitivity [44], or (exact or approximate) polynomial degree? (For a review, see [14].)

- What about other query complexity models that might help us understand the relationship between $R(f)$ and $Q(f)$? One possibility is the projective query complexity, $P(f)$, considered in Section 7. Regev and Schiff [46] have shown (as a special case of their results) that even with such an oracle, $P(\mathrm{OR}) = \Theta(N)$ queries are needed to evaluate the OR function.

We hope that further study on the relationship between bomb and classical randomized complexity will shed light on the power and limitations of quantum computation.

2 Preliminaries

2.1 Mathematical notation

We will assume that the reader is familiar with basic notations and concepts of quantum computing. See [43] for a reference on quantum computing. We will use the bit-flip gate (the Pauli-X matrix)

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.1}$$

and we define $R(\theta)$ as the rotation matrix $\exp(-i\theta Y)$,

$$R(\theta) \equiv \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{2.2}$$

Note that $\pi/(2\theta)$ applications of $R(\theta)$ rotates $|0\rangle$ to $|1\rangle$.

We now recall a few facts from matrix analysis. For more details, we refer to [12].

Definition 2.1 (matrix norms)**.** We use $\|\cdot\|$ and $\|\cdot\|_F$ to denote the spectral norm and the Frobenius norm, respectively. The spectral norm is defined by

$$\|A\| = \max_{|x|\neq 0} \frac{\|Ax\|}{\|x\|}. \tag{2.3}$$

This is the largest singular value of the matrix. For a hermitian matrix, it is the maximum absolute value of eigenvalues.

The Frobenius norm of a matrix is the square root of the sum of squared absolute values of its entries:

$$\|A\|_F = \sqrt{\sum_i \sum_j |A_{ij}|^2} = \sqrt{\operatorname{tr}(A^\dagger A)}, \tag{2.4}$$

where $\dagger$ indicates conjugate-transpose.

Lemma 2.2. *For any $m,n > 0$ and matrices $X \in \mathbb{C}^{m\times n}$, $Y \in \mathbb{C}^{n\times n}$, $Z \in \mathbb{C}^{n\times m}$, we have*

$$|\operatorname{tr}(XYZ)| \le \|X\|_F \|Y\| \|Z\|_F.$$

This lemma can be proved by using that

$$|\operatorname{tr}(XYZ)| \le \|Y\| \|ZX\|_{\mathrm{tr}} \qquad \text{[12, Exercise IV.2.12], and}$$
$$\|ZX\|_{\mathrm{tr}} \le \|X\|_F \|Z\|_F \qquad \text{[12, Corollary IV.2.6]},$$

where $\|\cdot\|_{\mathrm{tr}}$ denotes the trace norm (sum of singular values). (The trace norm will see no further use in this paper.) A more accessible proof is found online at [15].

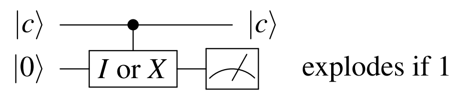
In this paper, log always denotes the logarithm with base 2.

2.2 The Elitzur-Vaidman bomb testing problem

The Elitzur-Vaidman bomb testing problem [22] is a well-known thought experiment to demonstrate how quantum mechanics differs drastically from our classical perceptions. This problem demonstrates dramatically the possibility of *interaction-free measurements*, the possibility of a measurement on a property of a system without disturbing the system.

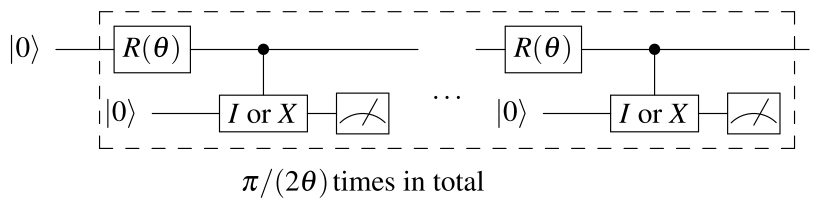
The bomb-testing problem is explained as follows. Assume we have a bomb that is either a dud (defective) or a live bomb. The only way to interact with the bomb is to probe it with a photon. If the bomb is a dud, then the photon passes through unimpeded; if the bomb is live, then the bomb explodes.

We would like to determine whether the bomb is live or not without exploding it. If we pass the photon through a beamsplitter before probing the bomb, we can implement the *controlled probe*, pictured below.



(2.5)

In this figure $c \in \{0,1\}$, and the controlled gate is the identity I if the bomb is a dud, and the bit-flip operation X if it is live. It was shown in [35] how to determine whether a bomb was live with arbitrarily low probability of explosion by making use of the quantum Zeno effect [40]. Specifically, the following circuit determines whether the bomb is live with failure probability $O(\theta)$.



(2.6)

If the bomb is a dud, then the controlled probes do nothing, and repeated application of $R(\theta)$ rotates the control bit from $|0\rangle$ to $|1\rangle$. If the bomb is live, the bomb explodes with $O(\theta^2)$ probability in each application of the probe; in the likely event the bomb does not explode, the control bit is projected back to $|0\rangle$. After $O(1/\theta)$ iterations the control bit stays in $|0\rangle$, with only a $O(\theta)$ probability of explosion. Using $O(1/\theta)$ operations, we can thus tell a dud bomb apart from a live one with only $O(\theta)$ probability of explosion.

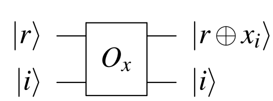
2.3 Quantum query complexity

Throughout this paper, all functions f which we would like to calculate are assumed to have Boolean input, i. e., the domain is $D \subseteq \{0,1\}^N$.

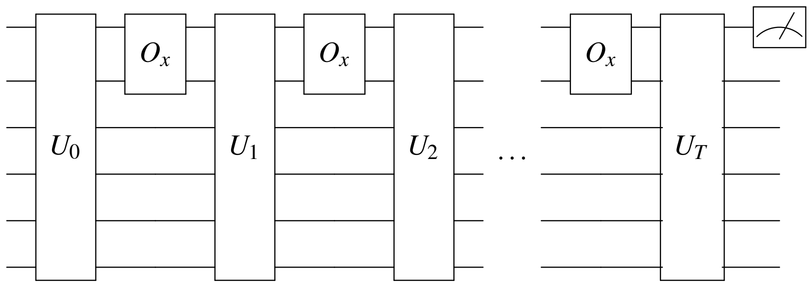
For a Boolean string $x \in \{0,1\}^N$, the quantum oracle O_x is a unitary operator that acts on a one-qubit record register and an N-dimensional index register as follows ($\oplus$ is the XOR function):

$$O_x|r,i\rangle = |r \oplus x_i, i\rangle \tag{2.7}$$

This is represented by the following diagram.



We want to determine the value of a Boolean function $f(x)$ using as few queries to the quantum oracle O_x as possible. Algorithms for f have the general form as the following circuit, where the U_t are unitaries independent of x.



At the end of the circuit the first qubit is measured, and the result is the output of the circuit, which would ideally be $f(x)$ with high probability. The quantum query complexity $Q_\delta(f)$ is the minimum number of applications of O_x in the circuit required to determine $f(x)$ with error not greater than δ for all x. By gap amplification (e. g., by performing the circuit multiple rounds and doing majority voting), it can be shown that the choice of δ only affects the query complexity by a $\log(1/\delta)$ factor. We therefore often set $\delta = 0.01$ and write $Q_{0.01}(f)$ as $Q(f)$.

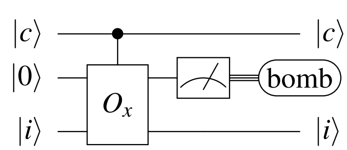
3 Bomb query complexity

In this section we introduce a new query complexity model, which we call the *bomb query complexity*. A circuit in the bomb query model is a restricted quantum query circuit, with the following restrictions on the usage of the quantum oracle.

1. We have an extra control register $|c\rangle$ used to control whether O_x is applied (we call the controlled version CO_x):
$$CO_x|c,0,i\rangle = |c, c \cdot x_i, i\rangle\,,$$
where $\cdot$ indicates Boolean AND.

2. The record register, i. e., the second register in the definition of CO_x above, *must* contain $|0\rangle$ before CO_x is applied.

3. After CO_x is applied, the record register is immediately measured in the computational basis (giving the answer $c \cdot x_i$), and the algorithm *terminates immediately if a 1 is measured* (if $c \cdot x_i = 1$). We refer to this as *the bomb blowing up* or *the bomb exploding*.



explodes if $c \cdot x_i = 1$ (3.1)

We define the *bomb query complexity* $B_{\varepsilon,\delta}(f)$ to be the minimum number of times the above circuit needs to be applied in an algorithm such that the following hold for all input x.

- The algorithm reaches the end without the bomb exploding with probability at least $1 - \varepsilon$. We refer to the probability that the bomb explodes as the *probability of explosion*.

- The total probability that the bomb either explodes or fails to output $f(x)$ correctly is at most δ. Note that this implies $\delta \geq \varepsilon$.

The above implies that the algorithm outputs the correct answer with probability at least $1-\delta$.

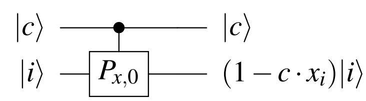
The effect of circuit (3.1) is equivalent to applying the following projector on $|c,i\rangle$.

$$\begin{aligned} M_x = CP_{x,0} &= \sum_{i\in[N]} |0,i\rangle\langle 0,i| + \sum_{i:x_i=0} |1,i\rangle\langle 1,i| \\ &= I - \sum_{i:x_i=1} |1,i\rangle\langle 1,i|\,. \end{aligned}$$

$CP_{x,0}$ (which we will just call M_x in our proofs later on) is the controlled version of $P_{x,0}$, the projector that projects onto the indices i on which $x_i = 0$:

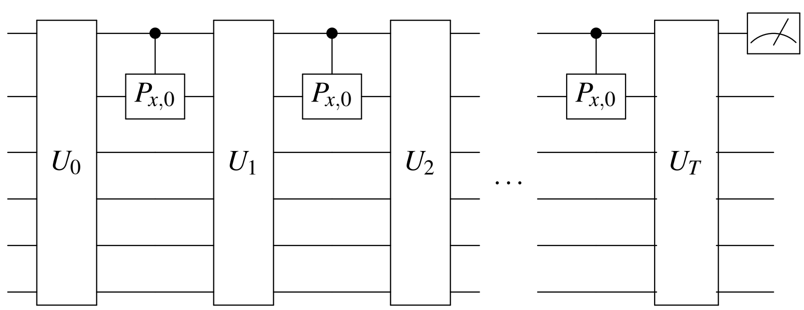
$$P_{x,0} = \sum_{i:x_i=0} |i\rangle\langle i|\,.$$

Thus Circuit 3.1 is equivalent to the following circuit.



In this notation, the square of the norm of a state is the probability that the state has survived to this stage, i. e., the algorithm has not terminated. The norm of $(1-c\cdot x_i)|x_i\rangle$ is 1 if $c\cdot x_i = 0$ (the state survives this stage), and 0 otherwise (the bomb blows up).

A general circuit in this model looks like the following.



(3.2)

As usual, at the end of the circuit the first qubit is measured, and the result is the output of the circuit.

It is not at all clear that gap amplification can be done efficiently in the bomb query model to improve the error δ; repeating the circuit multiple times increases the chance that the bomb blows up. However, it turns out that the complexity $B_{\varepsilon,\delta}(f)$ is closely related to $Q_\delta(f)$, and therefore the choice of δ affects $B_{\varepsilon,\delta}(f)$ by at most a $\log^2(1/\delta)$ factor as long as $\delta \geq \varepsilon$ (see Corollary 4.2). We therefore often omit δ by setting $\delta = 0.01$, and write $B_{\varepsilon,0.01}(f)$ as $B_\varepsilon(f)$. Sometimes we even omit the ε.

Finally, note that the definition of the bomb query complexity $B(f)$ is inherently *asymmetric* with respect to 0 and 1 in the input: querying 1 causes the bomb to blow up, while querying 0 is safe. In Section 5.1, we define a *symmetric* bomb query model and its corresponding query complexity, $\tilde{B}_{\varepsilon,\delta}(f)$. We prove that this generalized symmetric model is asymptotically equivalent to the original asymmetric model, $\tilde{B}_{\varepsilon,\delta}(f) = \Theta(B_{\varepsilon,\delta}(f))$, in Lemma 5.1. This symmetric version of the bomb query complexity will turn out to be useful in designing bomb query algorithms.

4 Main result

Our main result is the following.

Theorem 4.1. *For every function f with Boolean input alphabet, and every ε satisfying* $0 < \varepsilon \leq 0.01$,

$$B_{\varepsilon,0.01}(f) = \Theta\left(\frac{Q_{0.01}(f)^2}{\varepsilon}\right).$$

Here 0.01 can be replaced by any constant not greater than $1/10$.

Proof. The upper bound $B_{\varepsilon,\delta}(f) = O(Q_\delta(f)^2/\varepsilon)$ is proved in Theorem 4.3. The lower bound $B_{\varepsilon,\delta}(f) = \Omega(Q_{0.01}(f)^2/\varepsilon)$ is proved in Theorem 4.5. □

Corollary 4.2. *For all functions f with Boolean input alphabet, and numbers ε, δ satisfying* $0 < \varepsilon \leq \delta \leq 1/10$,

$$B_{\varepsilon,0.1}(f) = O(B_{\varepsilon,\delta}(f)), \qquad B_{\varepsilon,\delta}(f) = O(B_{\varepsilon,0.1}(f)\log^2(1/\delta)).$$

In particular, if δ is constant,

$$B_{\varepsilon,\delta}(f) = \Theta(B_{\varepsilon,0.1}(f)).$$

Proof. Immediate from Theorem 4.3 and $Q_{0.1}(f) = O(Q_\delta(f))$, $Q_\delta(f) = O(Q_{0.1}(f)\log(1/\delta))$. □

Because of this result, we will often omit the 0.01 in $B_{\varepsilon,0.01}$ and write simply B_ε.

4.1 Upper bound

Theorem 4.3. *For all functions f with Boolean input alphabet, and numbers ε, δ satisfying* $0 < \varepsilon \leq \delta \leq 1/10$,

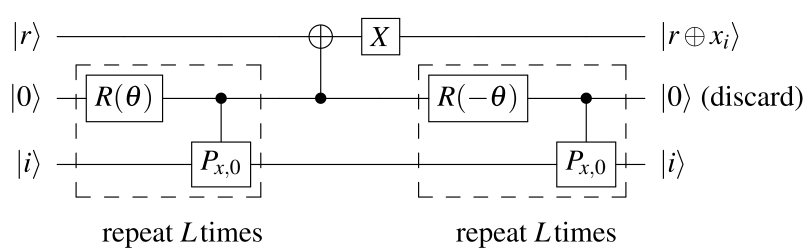
$$B_{\varepsilon,\delta}(f) = O(Q_\delta(f)^2/\varepsilon).$$

The proof follows the solution of Elitzur-Vaidman bomb-testing problem ([35], or Section 2.2). By taking advantage of the Quantum Zeno effect [40], using $O(Q(f)/\varepsilon)$ calls to M_x, we can simulate one call to O_x with probability of explosion $O(\varepsilon/Q(f))$. Replacing all O_x queries in (3.2) with this construction results in a bounded error algorithm with probability of explosion $O((\varepsilon/Q(f))Q(f)) = O(\varepsilon)$.

Proof. Let $\theta = \pi/(2L)$ for some large positive integer L (chosen later), and recall that

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}.$$

We claim that with $2L$ calls to the bomb oracle $M_x = CP_{x,0}$, we can simulate O_x by the following circuit with probability of explosion less than $\pi^2/(2L)$ and error $O(1/L)$.



(4.1)

In words, we simulate O_x acting on $|r,i\rangle$ by the following steps.

1. Append an ancilla qubit $|0\rangle$, changing the state into $|r,0,i\rangle$.
2. Repeat the following L times:
   (a) apply $R(\theta)$ on the second register;
   (b) apply M_x on the third register controlled by the second register.

   At this point, if the bomb hasn't blown up, the second register should contain $1-x_i$.
3. Apply $CNOT$ on the first register controlled by the second register; this copies $1-x_i$ to the first register.
4. Apply a NOT gate to the first register.
5. Repeat the following L times to uncompute the second (ancilla) register:
   (a) apply $R(-\theta)$ on the second register;
   (b) apply M_x on the third register controlled by second register.
6. Discard the second (ancilla) register.

We now calculate explicitly the action of the circuit on an arbitrary state to confirm our claims above. Consider how the circuit acts on the basis state $|r,0,i\rangle$ (the second register being the appended ancilla). We break into cases.

- If $x_i=0$, then $P_{x,0}|i\rangle=|i\rangle$, so the controlled projections do nothing. Thus in Step 2 the rotation $R(\theta)^L=R(\pi/2)$ is applied to the ancilla qubit, rotating it from 0 to 1. After Step 2 then, the state is $|r,1,i\rangle$. Step 3 and 4 together do not change the state, while Step 5 rotates the ancilla back to 0, resulting in the final state $|r,0,i\rangle$.
- If $x_i=1$, then $P_{x,0}|i\rangle=0$, and

  $$M_x|0,i\rangle=|0,i\rangle, \qquad M_x|1,i\rangle=0\,.$$

  Therefore in Step 2 and Step 5, after each rotation $R(\pm\theta)$, the projection $CP_{x,0}$ projects the ancilla back to 0:

  $$M_xR(\theta)|0,i\rangle=M_x(\cos\theta|0\rangle+\sin\theta|1\rangle)|i\rangle=\cos\theta|0,i\rangle\,.$$

  Each application of $M_xR(\theta)$ thus has no change on the state other than to shrink its amplitude by $\cos\theta$. The CNOT in Step 3 has no effect (since the ancilla stays in 0), and Step 4 maps $|r\rangle$ to $|r\oplus 1\rangle$. Since there are $2L$ applications of this shrinkage (in Step 2 and 5), the final state is $\cos^{2L}\theta|r\oplus 1,0,i\rangle$.

We can now combine the two cases: by linearity, the application of the circuit on a general state $\sum_{r,i} a_{r,i}|r,i\rangle$ (removing the ancilla) is

$$\begin{aligned}\sum_{r,i} a_{r,i}|r,i\rangle &\to \sum_{r\in\{0,1\},x_i=0} a_{r,i}|r,i\rangle + \sum_{r\in\{0,1\},x_i=1} a_{r,i}\cos^{2L}(\theta)|r\oplus 1,i\rangle \\ &= \sum_{r,i} a_{r,i}\cos^{2Lx_i}\left(\frac{\pi}{2L}\right)|r\oplus x_i,i\rangle \equiv \left|\psi'\right\rangle.\end{aligned}$$

Thus the effect of this construction simulates the usual quantum oracle $|r,i\rangle \to |r\oplus x_i,i\rangle$ with probability of explosion not greater than

$$1-\cos^{4L}\left(\frac{\pi}{2L}\right) \le 1-\left(1-\frac{\pi^2}{4L^2}\right)^{2L} \le \frac{\pi^2}{2L}.$$

Moreover, the difference between the output of our circuit, $|\psi'\rangle$, and the output on the quantum oracle, $|\psi\rangle = \sum_{r,i} a_{r,i}|r\oplus x_i,i\rangle$, is

$$\begin{aligned}\left\|\left|\psi'\right\rangle - |\psi\rangle\right\| &= \left\|\sum_{r\in\{0,1\},x_i=1} a_{r,i}(1-\cos^{2L}(\theta))|r\oplus 1,i\rangle\right\| \\ &\le 1-\cos^{2L}\frac{\pi}{2L} \le \frac{\pi^2}{4L}.\end{aligned}$$

Given this construction, we can now prove our theorem. Suppose we are given a quantum algorithm that finds $f(x)$ with $Q_{\delta'}(f)$ queries, making at most $\delta' = \delta - \varepsilon$ error. We construct an algorithm using bomb oracles instead by replacing each of the applications of the quantum oracle O_x by our circuit construction (4.1), where we choose

$$L = \left\lceil \frac{\pi^2}{2\varepsilon} Q_{\delta'}(f) \right\rceil.$$

Then the probability of explosion is not greater than

$$\frac{\pi^2}{2L} Q_{\delta'}(f) \le \varepsilon$$

and the difference between the final states, $\left|\psi_f\right\rangle$ and $\left|\psi'_f\right\rangle$, is at most

$$\left\|\left|\psi'_f\right\rangle - \left|\psi_f\right\rangle\right\| \le \frac{\pi^2}{4L} Q_{\delta'}(f) \le \frac{\varepsilon}{2}. \tag{4.2}$$

Therefore for any projector P (in particular, the projector that projects onto the classical answer at the end of the algorithm),

$$\begin{aligned}\left|\left\langle\psi'_f\middle|P\middle|\psi'_f\right\rangle - \left\langle\psi_f\middle|P\middle|\psi_f\right\rangle\right| &\le \left|\left\langle\psi'_f\middle|P\middle|\psi'_f\right\rangle - \left\langle\psi_f\middle|P\middle|\psi'_f\right\rangle\right| + \left|\left\langle\psi'_f\middle|P\middle|\psi_f\right\rangle - \left\langle\psi_f\middle|P\middle|\psi_f\right\rangle\right| \\ &\le \left\|\left|\psi'_f\right\rangle\right\| \left\|P\left(\left|\psi'_f\right\rangle - \left|\psi_f\right\rangle\right)\right\| + \left\|P\left(\left|\psi'_f\right\rangle - \left|\psi_f\right\rangle\right)\right\| \left\|\left|\psi_f\right\rangle\right\| \\ &\le \varepsilon/2 + \varepsilon/2 = \varepsilon\end{aligned}$$

where the first inequality follows by the triangle inequality, the second by Cauchy-Schwarz, and the last by (4.2). The algorithm accumulates at most ε extra error at the end, giving a total error not greater than $\delta' + \varepsilon = \delta$. This algorithm makes

$$2LQ_{\delta'}(f) < \frac{\pi^2}{\varepsilon} Q_{\delta'}^2(f) + 2Q_{\delta'}(f)$$

queries to the bomb oracle, and therefore

$$B_{\varepsilon,\delta}(f) < \frac{\pi^2}{\varepsilon} Q_{\delta-\varepsilon}(f)^2 + 2Q_{\delta-\varepsilon}(f) = O\left(\frac{Q_{\delta-\varepsilon}(f)^2}{\varepsilon}\right). \tag{4.3}$$

From this we can derive that $B_{\varepsilon,\delta}(f) = O(Q_\delta(f)^2/\varepsilon)$:

$$\begin{aligned} B_{\varepsilon,\delta}(f) &< B_{\varepsilon/2,\delta}(f) \\ &= O\left(\frac{Q_{\delta-\varepsilon/2}(f)^2}{\varepsilon}\right), && \text{by (4.3)}, \\ &= O\left(\frac{Q_{\delta}(f)^2}{\varepsilon}\right), && \text{since } \frac{\delta}{2} \le \delta - \frac{\varepsilon}{2}. \end{aligned}$$

□

4.2 Lower bound

Theorem 4.5. *For all functions f with Boolean input alphabet, and numbers ε, δ satisfying $0 < \varepsilon \le \delta \le 1/10$,*

$$B_{\varepsilon,\delta}(f) = \Omega(Q_{0.01}(f)^2/\varepsilon).$$

Before we give the proof of the general result that $B(f) = \Omega(Q(f)^2)$ (Theorem 4.5), we will illustrate the proof by means of an example, the special case where f is the AND function.

Theorem 4.4. *For $\delta < 1/10$, $B_{\varepsilon,\delta}(\mathrm{AND}) = \Omega(N/\varepsilon)$.*

Proof. Let $|\psi_t^0\rangle$ be the unnormalized state of the algorithm with $x = 1^n$, and $|\psi_t^k\rangle$ be the unnormalized state with $x = 1\cdots 101 \cdots 1$, $x_k = 0$, right before the $(t+1)$-th call to M_x. Then

$$|\psi_{t+1}^k\rangle = U_{t+1} M_x |\psi_t^k\rangle$$

for some unitary U_{t+1}. For ease of notation, we'll write $M_0 \equiv M_{1^n}$ and $M_k = M_{1\cdots 101\cdots 1}$, where the k-th bit is 0 in the latter case. When acting on the control and index bits,

$$\begin{aligned} M_0 &= \sum_{i=1}^N |0,i\rangle\langle 0,i|, \\ M_k &= \sum_{i=1}^N |0,i\rangle\langle 0,i| + |1,k\rangle\langle 1,k|, \qquad k = 1,\cdots,N. \end{aligned}$$

Since the operators M_i are projectors, $M_i^2 = M_i$. Define

$$\varepsilon_t^i = \left\langle \psi_t^i \middle| (I - M_i) \middle| \psi_t^i \right\rangle , \qquad i = 0, 1, \cdots, N . \tag{4.4}$$

Note that $\langle \psi_{t+1}^i | \psi_{t+1}^i \rangle = \left\langle \psi_t^i \middle| M_i^2 \middle| \psi_t^i \right\rangle = \left\langle \psi_t^i \middle| M_i \middle| \psi_t^i \right\rangle = \langle \psi_t^i | \psi_t^i \rangle - \varepsilon_t^i$, for all $i = 0, \cdots, N$ (including 0!), and hence

$$\sum_{t=0}^{T-1} \varepsilon_t^i = \langle \psi_0^i | \psi_0^i \rangle - \langle \psi_T^i | \psi_T^i \rangle \leq \varepsilon .$$

We now define the progress function. Let

$$W_t^k = \langle \psi_t^0 | \psi_t^k \rangle$$

and let the progress function be a sum over the W^k:

$$W_t = \sum_{k=1}^{N} W_t^k = \sum_{k=1}^{N} \langle \psi_t^0 | \psi_t^k \rangle .$$

We can lower bound the total change in the progress function by

$$W_0 - W_T \geq (1 - 2\sqrt{\delta(1-\delta)})N . \tag{4.5}$$

(See Ambainis [3] for a proof; his proof equally applies to unnormalized states.)

We now proceed to upper bound $W_0 - W_T$. Note that

$$\begin{aligned} W_t^k - W_{t+1}^k &= \langle \psi_t^0 | \psi_t^k \rangle - \left\langle \psi_t^0 \middle| M_0 M_k \middle| \psi_t^k \right\rangle \\ &= \left\langle \psi_t^0 \middle| (I - M_0) M_k \middle| \psi_t^k \right\rangle + \left\langle \psi_t^0 \middle| M_0 (I - M_k) \middle| \psi_t^k \right\rangle + \left\langle \psi_t^0 \middle| (I - M_0)(I - M_k) \middle| \psi_t^k \right\rangle \end{aligned}$$

and since $M_0(I - M_k) = 0$, $(I - M_0)M_k = |1,k\rangle\langle 1,k|$, we have

$$\begin{aligned} W_t^k - W_{t+1}^k &\leq \langle \psi_t^0 | 1,k \rangle \langle 1,k | \psi_t^k \rangle + \left\| (I - M_0) \middle| \psi_t^0 \right\rangle \left\| \, \right\| (I - M_k) \left| \psi_t^k \right\rangle \right\| \\ &\leq \left| \langle 1,k | \psi_t^0 \rangle \right| + \sqrt{\varepsilon_t^0 \varepsilon_t^k} , \end{aligned}$$

where we used (4.4) in the second inequality. Summing over k and t, we obtain

$$\begin{aligned} W_0 - W_T &\leq \sum_{t=0}^{T-1} \sum_{k=1}^{N} \left[\left\| \langle 1,k | \psi_t^0 \rangle \right\| + \sqrt{\varepsilon_t^0 \varepsilon_t^k} \right] \\ &\leq \sqrt{TN} \sqrt{\sum_{t=0}^{T-1} \sum_{k=1}^{N} \langle \psi_t^0 | 1,k \rangle \langle 1,k | \psi_t^0 \rangle} + \sum_{k=1}^{N} \sqrt{\sum_{t=0}^{T-1} \varepsilon_t^0 \sum_{t'=0}^{T-1} \varepsilon_{t'}^k} \\ &\leq \sqrt{TN} \sqrt{\sum_{t=0}^{T-1} \left\langle \psi_t^0 \middle| (I - M_0) \middle| \psi_t^0 \right\rangle} + N\varepsilon \\ &\leq \sqrt{TN \sum_{t=0}^{T-1} \varepsilon_t^0} + N\varepsilon \\ &\leq \sqrt{\varepsilon TN} + N\varepsilon \end{aligned}$$

where in the second line we used Cauchy-Schwarz twice, and in the third line we recalled that

$$\sum_{t=0}^{T-1} \varepsilon_t^i \leq \varepsilon$$

for all i. Combined with (4.5), this gives

$$T \geq \frac{(1-2\sqrt{\delta(1-\delta)}-\varepsilon)^2 N}{\varepsilon}. \qquad \square$$

We now proceed to prove the general result. This proof follows the presentation given in A. Childs's online lecture notes [15], which we found quite illuminating.

Theorem 4.5. *For all functions f with Boolean input alphabet, and numbers ε, δ satisfying $0 < \varepsilon \leq \delta \leq 1/10$,*

$$B_{\varepsilon,\delta}(f) = \Omega(Q_{0.01}(f)^2/\varepsilon).$$

Proof. We prove the lower bound on $B_{\varepsilon,\delta}$ by showing that it is lower bounded by $\Omega(\mathrm{Adv}^{\pm}(f)^2/\varepsilon)$, where $\mathrm{Adv}^{\pm}(f)$ is the generalized (i. e., allowing negative weights) adversary bound [30] for f. We can then derive our theorem from the result [36] that $Q(f) = O(\mathrm{Adv}^{\pm}(f))$.

We generalize the bound on the $f = \mathrm{AND}$ case to an adversary bound for $B_{\varepsilon,\delta}$ on arbitrary f. Define the projectors

$$\Pi_0 = \sum_{i=1}^{N} |0,i\rangle\langle 0,i|,$$
$$\Pi_i = |1,i\rangle\langle 1,i|, \qquad i = 1,\cdots,N.$$

It is clear that

$$\Pi_0 + \sum_{i=1}^{N} \Pi_i = I.$$

Note that $M_x = CP_{x,0}$ is

$$M_x = \Pi_0 + \sum_{i:x_i=0} \Pi_i.$$

Define $|\psi_t^x\rangle$ as the state of the algorithm right before the $(t+1)$-th query with input x; then

$$\left|\psi_{t+1}^x\right\rangle = U_{t+1} M_x |\psi_t^x\rangle$$

for some unitary U_{t+1}. Now if we let

$$\varepsilon_t^x = \langle\psi_t^x|(I-M_x)|\psi_t^x\rangle$$

then it follows that $\langle\psi_t^x|\psi_t^x\rangle - \langle\psi_{t+1}^x|\psi_{t+1}^x\rangle = \varepsilon_t^x$, and thus

$$\sum_{t=0}^{T-1} \varepsilon_t^x = \langle\psi_0^x|\psi_0^x\rangle - \langle\psi_T^x|\psi_T^x\rangle \leq \varepsilon.$$

We proceed to define the progress function. Let D be the set of allowable input strings x. Let Γ be an *adversary matrix*, i. e., an $|D| \times |D|$ matrix such that

1. $\Gamma_{xy} = \Gamma_{yx} \quad \forall x, y \in D$; and
2. $\Gamma_{xy} = 0 \quad$ if $f(x) = f(y)$.

Let a be the normalized eigenvector of Γ with eigenvalue $\pm\|\Gamma\|$, where $\pm\|\Gamma\|$ is the largest (by absolute value) eigenvalue of Γ. Define the progress function

$$W_t = \sum_{x,y \in D} \Gamma_{xy} a_x^* a_y \langle \psi_t^x | \psi_t^y \rangle \,.$$

For $\varepsilon \leq \delta < 1/10$ we have that[1] (see [30] for a proof; their proof applies equally well to unnormalized states)

$$|W_0 - W_T| \geq (1 - 2\sqrt{\delta(1-\delta)} - 2\delta)\|\Gamma\| \,.$$

We now proceed to upper bound $|W_0 - W_T| \leq \sum_t |W_t - W_{t-1}|$. Note that

$$\begin{aligned} W_t - W_{t+1} &= \sum_{x,y \in D} \Gamma_{xy} a_x^* a_y \left(\langle \psi_t^x | \psi_t^y \rangle - \langle \psi_{t+1}^x | \psi_{t+1}^y \rangle \right) \\ &= \sum_{x,y \in D} \Gamma_{xy} a_x^* a_y \left(\langle \psi_t^x | \psi_t^y \rangle - \langle \psi_t^x | M_x M_y \middle| \psi_t^y \rangle \right) \\ &= \sum_{x,y \in D} \Gamma_{xy} a_x^* a_y \left(\langle \psi_t^x | (I - M_x) M_y \middle| \psi_t^y \rangle + \langle \psi_t^x | M_x (I - M_y) \middle| \psi_t^y \rangle + \langle \psi_t^x | (I - M_x)(I - M_y) \middle| \psi_t^y \rangle \right) \end{aligned} \tag{4.6}$$

We bound the three terms separately. For the first two terms, use

$$(I - M_x) M_y = \sum_{i: x_i = 1, y_i = 0} \Pi_i = (I - M_x) \sum_{i: x_i \neq y_i} \Pi_i \,.$$

Define the $|D| \times |D|$ matrix Γ_i as

$$\Gamma_i = \begin{cases} \Gamma_{xy} & \text{if } x_i \neq y_i, \\ 0 & \text{if } x_i = y_i. \end{cases}$$

The first term of (4.6) is

$$\begin{aligned} \sum_{x,y \in D} \sum_{i: x_i \neq y_i} \Gamma_{xy} a_x^* a_y \langle \psi_t^x | (I - M_x) \Pi_i \middle| \psi_t^y \rangle &= \sum_{x,y \in D} \sum_{i=1}^N (\Gamma_i)_{xy} a_x^* a_y \langle \psi_t^x | (I - M_x) \Pi_i \middle| \psi_t^y \rangle \\ &= \sum_{i=1}^N \operatorname{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \end{aligned}$$

where

$$\begin{aligned} Q_i &= \sum_{x \in D} a_x \Pi_i | \psi_t^x \rangle \langle x | \,, \\ \tilde{Q}_i &= \sum_{x \in D} a_x \Pi_i (I - M_x) | \psi_t^x \rangle \langle x | \,. \end{aligned}$$

[1] As described in [30], the 2δ term can be removed if the output is Boolean (0 or 1).

Although both Q_i and $\tilde{Q}_i$ depend on t, we suppress the t dependence in the notation. Similarly, the second term of (4.6) is equal to

$$\sum_{i=1}^{N} \operatorname{tr}\left(\tilde{Q}_i \Gamma_i Q_i^\dagger\right).$$

We can also rewrite the third term of (4.6) as

$$\sum_{x,y\in D} \Gamma_{xy} a_x^* a_y \langle \psi_t^x | (I - M_x)(I - M_y) \left| \psi_t^y \right\rangle = \operatorname{tr}(Q' \Gamma Q'^\dagger)$$

where

$$Q' = \sum_{x\in D} a_x (I - M_x) |\psi_t^x\rangle\langle x|.$$

Therefore, adding absolute values,

$$|W_t - W_{t+1}| \le \sum_{i=1}^{N} \left[\left| \operatorname{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \right| + \left| \operatorname{tr}(\tilde{Q}_i \Gamma_i Q_i^\dagger)) \right| \right] + \left| \operatorname{tr}(Q' \Gamma Q'^\dagger) \right|. \tag{4.7}$$

Then by Lemma 2.2,

$$\sum_{i=1}^{N} \left| \operatorname{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger) \right| \le \sum_{i=1}^{N} \|\Gamma_i\| \|Q_i\|_F \|\tilde{Q}_i\|_F. \tag{4.8}$$

We can calculate

$$\begin{aligned}
\sum_{i=1}^{N} \|Q_i\|_F^2 &= \sum_{i=1}^{N} \sum_{x\in D} |a_x|^2 \, \|\Pi_i |\psi_t^x\rangle\|^2 \\
&= \sum_{x\in D} |a_x|^2 \langle \psi_t^x | \sum_{i=1}^{N} \Pi_i |\psi_t^x\rangle \\
&\le \sum_{x\in D} |a_x|^2 \\
&= 1
\end{aligned}$$

and

$$\begin{aligned}
\sum_{i=1}^{N} \|\tilde{Q}_i\|_F^2 &= \sum_{i=1}^{N} \sum_{x\in D} |a_x|^2 \, \|\Pi_i (I - M_x) |\psi_t^x\rangle\|^2 \\
&= \sum_{x\in D} |a_x|^2 \langle \psi_t^x | (I - M_x) \left(\sum_{i=1}^{N} \Pi_i \right) (I - M_x) |\psi_t^x\rangle \\
&\le \sum_{x\in D} |a_x|^2 \langle \psi_t^x | (I - M_x) |\psi_t^x\rangle \\
&= \sum_{x\in D} |a_x|^2 \varepsilon_t^x.
\end{aligned}$$

Then by Cauchy-Schwarz,

$$\sum_{i=1}^{N} \|Q_i\|_F \|\tilde{Q}_i\|_F \leq \sqrt{\sum_{x\in D} |a_x|^2 \varepsilon_t^x}\,. \tag{4.9}$$

Therefore by (4.8) and (4.9),

$$\sum_{i=1}^{N} \left|\mathrm{tr}(Q_i \Gamma_i \tilde{Q}_i^\dagger)\right| \leq \sqrt{\sum_{x\in D} |a_x|^2 \varepsilon_t^x} \max_{i\in[N]} \|\Gamma_i\|\,.$$

Similarly for $\mathrm{tr}(Q'\Gamma Q'^\dagger)$, we have

$$\begin{aligned}
\|Q'\|_F^2 &= \sum_{x\in D} |a_x|^2 \left\|(I-M_x)|\psi_t^x\rangle\right\|^2 \\
&= \sum_{x\in D} |a_x|^2 \langle\psi_t^x|(I-M_x)|\psi_t^x\rangle \\
&= \sum_{x\in D} |a_x|^2 \varepsilon_t^x
\end{aligned}$$

and using Lemma 2.2,

$$\mathrm{tr}(Q'\Gamma Q'^\dagger) \leq \|Q'\|_F^2 \|\Gamma\| = \sum_{x\in D} |a_x|^2 \varepsilon_t^x \|\Gamma\|\,.$$

Thus continuing from (4.7) we have that

$$|W_t - W_{t+1}| \leq 2\sqrt{\sum_{x\in D} |a_x|^2 \varepsilon_t^x} \max_{i\in[N]} \|\Gamma_i\| + \sum_{x\in D} |a_x|^2 \varepsilon_t^x \|\Gamma\|\,.$$

Finally, if we sum the above over t we obtain

$$|W_0 - W_T| \leq 2\max_{i\in[N]} \|\Gamma_i\| \sum_{t=0}^{T-1} \sqrt{\sum_{x\in D} |a_x|^2 \varepsilon_t^x} + \sum_{t=0}^{T-1} \sum_{x\in D} |a_x|^2 \varepsilon_t^x \|\Gamma\|\,.$$

The first term can be bounded using Cauchy-Schwarz:

$$\sum_{t=0}^{T-1} \sqrt{\sum_{x\in D} |a_x|^2 \varepsilon_t^x} \leq \sqrt{T \sum_{t=0}^{T-1} \sum_{x\in D} |a_x|^2 \varepsilon_t^x} \leq \sqrt{\varepsilon T}$$

where we used $\sum_t \varepsilon_t^x \leq \varepsilon$ and $\sum_{x\in D} |a_x|^2 = 1$. The second term can be summed easily:

$$\sum_{t=0}^{T-1} \sum_{x\in D} |a_x|^2 \varepsilon_t^x \|\Gamma\| \leq \sum_{x\in D} |a_x|^2 \varepsilon \|\Gamma\| = \varepsilon \|\Gamma\|\,.$$

Therefore

$$|W_0 - W_T| \leq 2\sqrt{\varepsilon T} \max_{i\in[N]} \|\Gamma_i\| + \varepsilon \|\Gamma\|\,.$$

Combined with our lower bound $|W_0 - W_T| \geq (1 - 2\sqrt{\delta(1-\delta)} - 2\delta)\|\Gamma\|$, this immediately gives

$$T \geq \frac{(1 - 2\sqrt{\delta(1-\delta)} - 2\delta - \varepsilon)^2}{4\varepsilon} \frac{\|\Gamma\|^2}{\max_{i \in [N]} \|\Gamma_i\|^2}\,.$$

Recalling from [30] that

$$\mathrm{Adv}^{\pm}(f) = \max_{\Gamma} \frac{\|\Gamma\|}{\max_{i \in [N]} \|\Gamma_i\|}\,,$$

we obtain[2]

$$T \geq \frac{(1 - 2\sqrt{\delta(1-\delta)} - 2\delta - \varepsilon)^2}{4\varepsilon} \mathrm{Adv}^{\pm}(f)^2\,.$$

We now use the tight characterization of the quantum query complexity by the general weight adversary method, given in [36, Theorem 1.1].

Theorem 4.6 (Lee, Mittal, Reichardt, Špalek, Szegedy)**.** *Let $f : D \to E$, $D \subseteq \{0,1\}^N$. Then $Q_{0.01}(f) = O(\mathrm{Adv}^{\pm}(f))$.*

Combined with our result above, we obtain

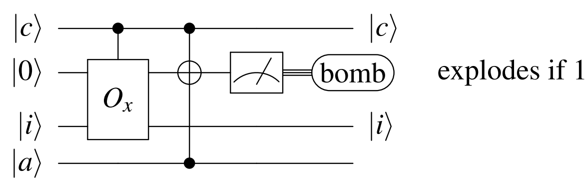
$$B_{\varepsilon,\delta}(f) = \Omega\left(\frac{Q_{0.01}(f)^2}{\varepsilon}\right)\,. \qquad \square$$

5 Generalizations and applications

We now discuss applications of the result $B_\varepsilon(f) = \Theta(Q(f)^2/\varepsilon)$ that could be useful.

5.1 Generalizing the bomb query model

We consider modifying the bomb query model as follows. We require that the input string x can only be accessed by the following circuit.



Compare with Circuit 3.1; the difference is that there is now an extra register $|a\rangle$, and the bomb explodes only if both $x_i \neq a$ and the control bit is 1. In other words, the bomb explodes if $c \cdot (x_i \oplus a) = 1$. The three registers c, i, and a are allowed to be entangled, however. If we discard the second register afterwards, the effect of this circuit, written as a projector, is

$$\widetilde{M}_x = \sum_{i \in [N], a \in \{0,1\}} |0,i,a\rangle\langle 0,i,a| + \sum_{i,a: x_i = a} |1,i,a\rangle\langle 1,i,a|\,. \tag{5.1}$$

[2]For Boolean output (0 or 1) the 2δ term can be dropped, as we previously noted (Footnote 1).

Let $\tilde{B}_{\varepsilon,\delta}(f)$ be the required number of queries to this modified bomb oracle $\widetilde{M}_x$ to calculate $f(x)$ with error not greater than δ, with a probability of explosion not greater than ε. Using Theorem 4.1, we show that $\tilde{B}$ and B are equivalent up to a constant.

Lemma 5.1. *If $f : D \to E$, where $D \subseteq \{0,1\}^N$, $\delta \le 1/10$ is a constant, and $\varepsilon \le \delta$ is arbitrary, then $B_{\varepsilon,\delta}(f) = \Theta(\tilde{B}_{\varepsilon,\delta}(f))$.*

Proof. It should be immediately obvious that $B_{\varepsilon,\delta}(f) \ge \tilde{B}_{\varepsilon,\delta}(f)$, since a query in the asymmetric model can easily be simulated by a query in the symmetric model by simply setting $a = 0$. In the following we show that $B_{\varepsilon,\delta}(f) = O(\tilde{B}_{\varepsilon,\delta}(f))$.

For each string $x \in \{0,1\}^N$, define the string $\tilde{x} \in \{0,1\}^{2N}$ by concatenating two copies of x and flipping every bit of the second copy. In other words,

$$\tilde{x}_i = \begin{cases} x_i & \text{if } i \le N, \\ 1 - x_{i-N} & \text{if } i > N. \end{cases} \tag{5.2}$$

Let $\tilde{D} = \{\tilde{x} : x \in D\}$. Given a function $f : D \to \{0,1\}$, define $\tilde{f} : \tilde{D} \to \{0,1\}$ by $\tilde{f}(\tilde{x}) = f(x)$.

We claim that a query in the symmetric model to x can be simulated by a query in the original asymmetric model to $\tilde{x}$. This can be seen by comparing $\widetilde{M}_x$ (Equation (5.1)) and the projector $M_{\tilde{x}}$, defined as follows.

$$M_{\tilde{x}} = \sum_{\tilde{i} \in [2N]} |0,\tilde{i}\rangle\langle 0,\tilde{i}| + \sum_{\tilde{i} \in [2N] : \tilde{x}_{\tilde{i}} = 0} |1,\tilde{i}\rangle\langle 1,\tilde{i}| \,.$$

Recalling the definition of $\tilde{x}$ in (5.2), we see that these two projectors are exactly equal if we encode $\tilde{i}$ as (i,a), where $i \equiv \tilde{i} \bmod N$ and $a = \lfloor i/N \rfloor$.

Since $\tilde{f}(\tilde{x}) = f(x)$, we thus have $\tilde{B}_{\varepsilon,\delta}(f) = B_{\varepsilon,\delta}(\tilde{f})$. Our result then readily follows; it can easily be checked that $Q(f) = Q(\tilde{f})$, and therefore by Theorem 4.1,

$$\tilde{B}_{\varepsilon,\delta}(f) = B_{\varepsilon,\delta}(\tilde{f}) = \Theta\left(\frac{Q(\tilde{f})^2}{\varepsilon}\right) = \Theta\left(\frac{Q(f)^2}{\varepsilon}\right). \qquad \square$$

There are some advantages to allowing the projector $\widetilde{M}_x$ instead of M_x. First of all, the inputs 0 and 1 in x are finally manifestly symmetric, unlike that in M_x (the bomb originally blew up if $x_i = 1$, but not if $x_i = 0$). Moreover, we now allow the algorithm to *guess* an answer a to the query x_i (this answer may be entangled with the index register i), and the bomb blows up only in the case $c \cdot (a \oplus i) = 1$: the guess is wrong, and the control bit c is 1. This flexibility may allow more leeway in designing algorithms for the bomb query model, as we soon utilize.

5.2 Using classical algorithms to design bomb query algorithms

We now demonstrate the possibility that we can prove *non-constructive* upper bounds on $Q(f)$ for some functions f, by creating bomb query algorithms and using that $Q(f) = \Theta(\sqrt{\varepsilon B_\varepsilon(f)})$. Consider for example the following classical algorithm for the OR function.

Algorithm 5.2 (Classical algorithm for OR)**.** *Pick some arbitrary ordering of the N bits, and query them one by one, terminating as soon as a 1 is seen. Return 1 if a 1 was seen; otherwise return 0.*

We can convert this immediately to a bomb query algorithm for OR, by using the construction in the proof of Theorem 4.3. That construction allows us to implement the operation O_x in $O(\varepsilon^{-1})$ queries, with $O(\varepsilon)$ error and probability of explosion if $x_i = 1$, but *no* error if $x_i = 0$. Thus we have the following algorithm.

Algorithm 5.3 (Bomb algorithm for OR)**.** *Query the N bits one-by-one, and apply the construction of Theorem 4.3 one bit at a time, using $O(1/\varepsilon)$ operations each time. Terminate as soon as a 1 is seen, and return 1; otherwise return 0 if all bits are 0.*

Since the algorithm ends as soon as a 1 is found, the algorithm only accumulates ε error in total. Thus this shows $B_\varepsilon(\mathrm{OR}) = O(N/\varepsilon)$.

Note, however, that we have already shown that $Q(f) = \Theta(\sqrt{\varepsilon B_\varepsilon(f)})$ for Boolean f. An $O(N/\varepsilon)$ bomb query algorithm for OR therefore implies that $Q(\mathrm{OR}) = O(\sqrt{N})$. We have shown the existence of an $O(\sqrt{N})$ quantum algorithm for the OR function, without actually constructing one!

We formalize the intuition in the above argument by the following theorem.

Theorem 5.4. *Let $f : D \to E$, where $D \subseteq \{0,1\}^N$. Suppose there is a classical randomized query algorithm $\mathcal{A}$, that makes at most T queries, and evaluates f with bounded error. Let the query results of $\mathcal{A}$ on random seed $s_{\mathcal{A}}$ be $x_{p_1}, x_{p_2}, \ldots, x_{p_{\tilde{T}(x)}}$, $\tilde{T}(x) \leq T$, where x is the hidden query string.*

Suppose there is another (not necessarily time-efficient) randomized algorithm $\mathcal{G}$, with random seed $s_{\mathcal{G}}$, which takes as input $x_{p_1}, \ldots, x_{p_{t-1}}$ and $s_{\mathcal{A}}$, and outputs a guess for the next query result x_{p_t} of $\mathcal{A}$. Note that $\mathcal{G}$ is given the random seed $s_{\mathcal{A}}$ of $\mathcal{A}$ as well as all past query results, so it can predict the next query index of $\mathcal{A}$. Assume that $\mathcal{G}$ does not make more than an expected total of G mistakes (for all inputs x), i. e.,

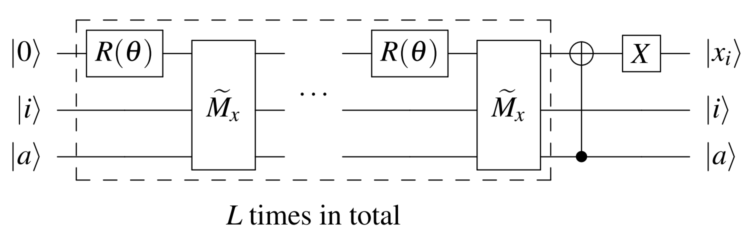
$$\mathrm{E}_{s_{\mathcal{A}}, s_{\mathcal{G}}} \left\{ \sum_{t=1}^{\tilde{T}(x)} \left| \mathcal{G}(x_{p_1}, \ldots, x_{p_{t-1}}, s_{\mathcal{A}}, s_{\mathcal{G}}) - x_{p_t} \right| \right\} \leq G \quad \forall x \,. \tag{5.3}$$

Then $B_\varepsilon(f) = O(TG/\varepsilon)$, and thus (by Theorem 4.1) $Q(f) = O(\sqrt{TG})$.

As an example, in our simple classical example for OR we have $T = N$ (the algorithm takes at most N steps) and $G = 1$ (the guessing algorithm always guesses the next query to be 0; since the algorithm terminates on a 1, it makes at most one mistake).

Proof. For the purposes of this proof, we use the characterization of B by the modified bomb construction given in Section 5.1. This proof is substantially similar to that of Theorem 4.3.

The following circuit finds x_i with zero probability of explosion if $x_i = a$, and with an $O(1/L)$ probability of explosion if $x_i \neq a$ (in both cases the value of x_i found by the circuit is always correct).



(5.4)

where $\theta = \pi/(2L)$ for some large number L to be picked later, and

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}.$$

The boxed part of the circuit is then simply $[\widetilde{M}_x(R(\theta)\otimes I\otimes I)]^L$, applied to the state $|0,i,a\rangle$. We can analyze this circuit by breaking into cases.

- If $x_i = a$, then $\widetilde{M}_x|\psi\rangle|i,a\rangle = |\psi\rangle|i,a\rangle$ for any state $|\psi\rangle$ in the control register. Thus the operators $\widetilde{M}_x$ act as identities, and the circuit simply applies the rotation $R(\theta)^L = R(\pi/2)$ to the control register, rotating it from 0 to 1. We thus obtain the state $|1,i,a\rangle$; the final CNOT and X gates add $a\oplus 1 = x_i \oplus 1$ to the first register, giving $|x_i,i,a\rangle$.

- If $x_i \neq a$, then
$$\widetilde{M}_x|0,i,a\rangle = |0,i,a\rangle\,, \qquad \widetilde{M}_x|1,i,a\rangle = 0\,.$$
Therefore after each rotation $R(\theta)$, the projection $\widetilde{M}_x$ projects the control qubit back to 0:
$$\widetilde{M}_x(R(\theta)\otimes I\otimes I)|0,i,a\rangle = \widetilde{M}_x(\cos\theta|0\rangle + \sin\theta|1\rangle)|i,a\rangle = \cos\theta|0,i,a\rangle \quad (\text{for } x_i \neq a)\,.$$
In this case the effect of $\widetilde{M}_x(R(\theta)\otimes I\otimes I)$ is to shrink the amplitude by $\cos(\theta)$; L applications results in the state $\cos^L(\theta)|0,i,a\rangle$. The final CNOT and X gates add $a\oplus 1 = x_i$ to the first register, giving $|x_i,i,a\rangle$.

The probability of explosion is 0 if $x_i = a$. If $x_i \neq a$, the probability of explosion is

$$1 - \cos^{2L}\left(\frac{\pi}{2L}\right) \leq \frac{\pi^2}{4L}\,.$$

Pick

$$L = \left\lceil \frac{\pi^2 G}{4\varepsilon} \right\rceil\,.$$

Then the probability of explosion is 0 if $x_i = a$, and not greater than ε/G if $x_i \neq a$. If the bomb does not explode, then the circuit *always* finds the correct value of x_i.

We now construct the bomb query algorithm based on $\mathcal{A}$ and $\mathcal{G}$. The bomb query algorithm follows $\mathcal{A}$, with each classical query replaced by the above construction. There are at most $TL \approx \pi^2 TG/(4\varepsilon)$ bomb queries. At each classical query, we pick the guess a to be the guess provided by $\mathcal{G}$. The bomb only has a chance of exploding if the guess is incorrect; hence for all x, the total probability of explosion is not greater than

$$\frac{\varepsilon}{G}\,\mathrm{E}_{s_{\mathcal{A}},s_{\mathcal{G}}}\left\{\sum_{t=1}^{\tilde{T}(x)} \left|\mathcal{G}(x_{p_1},\cdots,x_{p_{t-1}},s_{\mathcal{A}},s_{\mathcal{G}}) - x_{p_t}\right|\right\} \leq \varepsilon\,.$$

Thus replacing the classical queries of $\mathcal{A}$ with our construction gives a bomb query algorithm with probability of explosion not greater than ε; aside from the probability of explosion, this bomb algorithm makes no extra error over the classical algorithm $\mathcal{A}$. The number of queries this algorithm uses is

$$\tilde{B}_{\varepsilon,\delta+\varepsilon}(f) \leq \left\lceil \frac{\pi^2 G}{4\varepsilon} \right\rceil T\,,$$

where δ is the error rate of the classical algorithm. Therefore by Lemma 5.1 and Lemma 4.2,

$$B_\varepsilon(f) = O(B_{\varepsilon,\delta+\varepsilon}(f)) = O(\tilde{B}_{\varepsilon,\delta+\varepsilon}(f)) = O(TG/\varepsilon)\ . \qquad \square$$

5.3 Explicit quantum algorithm for Theorem 5.4

In this section we give an explicit quantum algorithm, in the setting of Theorem 5.4, that reproduces the given query complexity. This algorithm is very similar to the one given by Robin Kothari for the oracle identification problem [34].

Theorem 5.5. *Under the assumptions of Theorem 5.4, there is an explicit quantum algorithm for f with query complexity $O(\sqrt{TG})$.*

Proof. We will construct this algorithm (Algorithm 5.9) shortly. We need the following quantum search algorithm as a subroutine.

Theorem 5.6 (Finding the first marked element in a list)**.** *Suppose there is an ordered list of N elements, and each element is either marked or unmarked. Then there is a bounded-error quantum algorithm that finds the **first** marked element in the list (or determines that no marked elements exist), such that the following hold.*

- *If the first marked element is the d-th element of the list, then the algorithm uses an expected $O(\sqrt{d})$ queries.*
- *If there are no marked elements, then the algorithm uses $O(\sqrt{N})$ queries, but always determines correctly that no marked elements exist.*

This algorithm has been used by Kothari in [34].

Proof. We give an algorithm that has the stated properties. We first recall a quantum algorithm for finding the minimum in a list of items.

Theorem 5.7 (Dürr–Høyer [21])**.** *Given a function $g : [N] \to \mathbb{R}$, there is a quantum algorithm that finds the minimum of g with expected $O(\sqrt{N})$ evaluations of g, making $\delta < 1/10$ error.*

We now give our algorithm for finding the first marked element in a list. For simplicity, assume that N is a power of 2 (i. e., $\log_2 N$ is an integer).

Algorithm 5.8.

1. *For $\ell = 2^0, 2^1, 2^2, \cdots, 2^{\log_2 N} = N$:*
   - *Find the first marked element within the first ℓ elements, or determine no marked element exists. This can be done by defining*

$$g(i) = \begin{cases} \infty & \text{if } i \text{ is unmarked,} \\ i & \text{if } i \text{ is marked,} \end{cases}$$

and using Theorem 5.7 to find the minimum of g. This takes $O(\sqrt{\ell}) = O(\sqrt{d})$ queries and makes $\delta < 1/10$ error for each ℓ. If a marked element i^* is found, the algorithm outputs i^* and stops.

2. *If no marked element was found in Step 1, the algorithm decides that no marked element exists.*

We now claim that Algorithm 5.8 has the desired properties. Let us break into cases.

- If no marked elements exist, then no marked element can possibly be found in Step 1, so the algorithm correctly determines that no marked elements exist in Step 2. The number of queries used is

$$O\left(\sum_{i=0}^{\log_2 N} \sqrt{2^i}\right) = O(\sqrt{N})$$

as desired.

- Suppose the first marked element is the d-th item in the list. Then in Step 1(a), if $\ell \geq d$, there is at least a $1-\delta$ probability that the algorithm will detect that a marked element exists in the first ℓ elements and stop the loop. Letting $\alpha = \lceil \log_2 d \rceil$, the total expected number of queries is thus

$$\begin{aligned} O\left(\sum_{i=0}^{\alpha-1} \sqrt{2^i} + \sqrt{d} + \sum_{i=\alpha}^{\log_2 N} \delta^{i-\alpha}\sqrt{2^i}\right) &\leq O\left(\frac{2^{\alpha/2}-1}{\sqrt{2}-1} + \sqrt{2^\alpha}\frac{1}{1-\sqrt{2}\delta} + \sqrt{d}\right) \\ &= O(\sqrt{2^\alpha}) + O(\sqrt{d}) \\ &= O(\sqrt{d})\,. \end{aligned}$$

The probability of not finding the marked element at the first $\ell \geq d$ is at most δ , and thus the total error of the algorithm is bounded by δ. □

We now give our explicit quantum algorithm.

Algorithm 5.9 (Simulating a classical query algorithm by a quantum one)**.**

Input. *Classical randomized algorithm $\mathcal{A}$ that computes f with bounded error. Classical randomized algorithm $\mathcal{G}$ that guesses queries of $\mathcal{A}$. Oracle O_x for the hidden string x.*

Output. *$f(x)$ with bounded error.*

The quantum algorithm proceeds by attempting to produce the list of queries and results that $\mathcal{A}$ would have made. More precisely, given a random seed $s_{\mathcal{A}}$, the quantum algorithm outputs (with constant error) a list of pairs $(p_1(x), x_{p_1(x)}), \ldots, (p_{\widetilde{T}(x)}(x), x_{p_{\widetilde{T}(x)}(x)})$. This list is such that on random seed $s_{\mathcal{A}}$, the i-th query of algorithm $\mathcal{A}$ is made at the position $p_i(x)$, and the query result is $x_{p_i(x)}$. The quantum algorithm then determines the output of $\mathcal{A}$ using this list.

The main idea for the algorithm is this. We first assume that the guesses made by $\mathcal{G}$ are correct. By repeatedly feeding the output of $\mathcal{G}$ back into $\mathcal{A}$ and $\mathcal{G}$, we can obtain a list of query values for $\mathcal{A}$ without any queries to the actual black box. We then search for the first deviation of the string x from the *predictions of $\mathcal{G}$; assuming the first deviation is the d_1-th query, by Theorem 5.6 the search takes $O(\sqrt{d_1})$ queries (ignoring error for now). We then know that all the guesses made by $\mathcal{G}$ are correct up to the (d_1-1)-th query, and incorrect for the d_1-th query.*

With the corrected result of the first d_1 queries, we now continue by assuming again the guesses made by $\mathcal{G}$ are correct starting from the (d_1+1)-th query, and search for the location of the next deviation, d_2. This takes $O(\sqrt{d_2-d_1})$ queries; we then know that all the guesses made by $\mathcal{G}$ are correct from the (d_1+1)-th to (d_2-1)-th query, and incorrect for the d_2-th one. Continuing in this manner, we eventually determine all query results of $\mathcal{A}$ after an expected G iterations.

We proceed to spell out our algorithm. For the time being, we assume that algorithm for Theorem 5.6 has no error and thus requires no error reduction.

1. *Initialize random seeds $s_{\mathcal{A}}$ and $s_{\mathcal{G}}$ for $\mathcal{A}$ and $\mathcal{G}$. We will simulate the behavior of $\mathcal{A}$ and $\mathcal{G}$ on these seeds. Initialize $d=0$. d is such that we have determined the values of all query results of $\mathcal{A}$ up to the d-th query. Initialize an empty list $\mathcal{L}$ of $(\text{query position}, \text{query result})$ pairs.*

2. *Repeat until either all query results of $\mathcal{A}$ are determined, or $100G$ iterations of this loop have been executed.*

   (a) *Assuming that $\mathcal{G}$ always guesses correctly starting from the $(d+1)$-th query, compute from $\mathcal{A}$ and $\mathcal{G}$ a list of at most $T-d$ query positions $p_{d+1}, p_{d+2}, \ldots$ and results $\tilde{a}_{d+1}, \tilde{a}_{d+2}, \ldots$. This requires no queries to the black box.*

   (b) *Using our algorithm for finding the first marked element (Theorem 5.6, Algorithm 5.8), find the first index $d^* > d$ such that the actual query result of $\mathcal{A}$ differs from the guess by $\mathcal{G}$, i. e., $x_{p_d} \neq \tilde{a}_d$; or return that no such d^* exists. This takes $O(\sqrt{d^*-d})$ queries in the former case, and $O(\sqrt{T-d})$ queries in the latter.*

   (c) *We break into cases.*

      i. *If an index d^* was found in Step 2b, then the algorithm decides the next mistake made by $\mathcal{G}$ is at position d^*. It thus adds the pairs $(p_{d+1}, \tilde{a}_{d+1}), \ldots, (p_{d^*-1}, \tilde{a}_{d^*-1})$, and the pair $(p_{d^*}, 1-\tilde{a}_{d^*})$, to the list $\mathcal{L}$. Also set $d = d^*$.*

      ii. *If no index d^* was found in Step 2b, the algorithm decides that all remaining guesses by $\mathcal{G}$ are correct. Thus the query pairs $(p_{d+1}, \tilde{a}_{d+1}), \ldots, (p_{\tilde{T}(x)}, \tilde{a}_{\tilde{T}(x)})$ are added to $\mathcal{L}$, where $\tilde{T}(x) \leq T$ is the number of queries made by $\mathcal{A}$.*

3. *If the algorithm found all query results of $\mathcal{A}$ in $100G$ iterations of Step 2, use $\mathcal{L}$ to calculate the output of $\mathcal{A}$; otherwise the algorithm fails.*

We now count the total number of queries. Suppose $g \leq 100G$ is the number of iterations of Step 2; if all query results have been determined, g is the number of wrong guesses by $\mathcal{G}$. Say the list of indices d^* found is $d_0 = 0, d_1, \ldots, d_g$. Let $d_{g+1} = T$. Step 2 is executed for $g+1$ times, and the total number of queries is

$$O\left(\sum_{i=1}^{g+1} \sqrt{d_i - d_{i-1}}\right) = O\left(\sqrt{Tg}\right) = O\left(\sqrt{TG}\right)$$

by the Cauchy-Schwarz inequality.

We now analyze the error in our algorithm. The first source of error is cutting off the loop in Step 2: by Markov's inequality, for at least 99% of random seeds $s_{\mathcal{A}}, s_{\mathcal{G}}$, $\mathcal{G}$ does not make more than $100G$ wrong guesses. For these random seeds all query results of $\mathcal{A}$ are determined. Cutting off the loop thus gives at most 0.01 error.

The other source of error is the error of Algorithm 5.8 used in Step 2b: we had assumed that it could be treated as zero-error, but we now remove this assumption. Assuming each iteration gives error δ', the total error accrued could be up to $O(g\delta')$. It seems as if we would need to set $\delta' = O(1/G)$ for the total error to be constant, and thus gain an extra logarithmic factor in the query complexity.

However, in his paper for oracle identification [34], Kothari showed that multiple calls to Algorithm 5.8 can be composed to obtain a bounded-error algorithm based on span programs without an extra logarithmic factor in the query complexity; we refer to [34, Section 3] for the details. Therefore we can replace the iterations of Step 2 with Kothari's span program construction and get a bounded error algorithm with complexity $O(\sqrt{TG})$. □

Note that while Algorithm 5.9 has query complexity $O(\sqrt{TG})$, the time complexity may be much higher. After all, Algorithm 5.9 proceeds by simulating $\mathcal{A}$ query-by-query, although the number of actual queries to the oracle is smaller. Whether or not we can get an algorithm faster than $\mathcal{A}$ using this approach may depend on the problem at hand.

6 Improved upper bounds on quantum query complexity

We now use Theorem 5.5 to improve the quantum query complexity of certain graph problems.

6.1 Single source shortest paths for unweighted graphs

Problem 6.1 (Single source shortest paths (SSSP) for unweighted graphs)**.** The adjacency matrix of a directed n-vertex graph G is provided as a black box; a query on the pair (v,w) returns 1 if there is an edge from v to w, and 0 otherwise. We are given a fixed vertex v_{start}. Call the length of a shortest path from v_{start} to another vertex w the *distance* d_w of w from v_{start}; if no path exists, define $d_w = \infty$. Our task is to find d_w for all vertices w in G.

In this section we shall show the following result.

Theorem 6.2. *The quantum query complexity of single-source shortest paths in an unweighted graph is $\Theta(n^{3/2})$ in the adjacency matrix model.*

Proof. The lower bound of $\Omega(n^{3/2})$ is known [20]. We show the upper bound by applying Theorem 5.5 to a classical algorithm. The following well-known classical algorithm (commonly known as *breadth first search*, BFS) solves this problem.

Algorithm 6.3 (Classical algorithm for unweighted SSSP)**.**

1. *Initialize $d_w := \infty$ for all vertices $w \neq v_{\text{start}}$, $d_{v_{\text{start}}} := 0$, and $\mathcal{L} := (v_{\text{start}})$. $\mathcal{L}$ is the ordered list of vertices for which we have determined the distances, but whose outgoing edges we have not queried.*

2. *Repeat until $\mathcal{L}$ is empty:*

   - *Let v be the first (in order of time added to $\mathcal{L}$) vertex in $\mathcal{L}$. For all vertices w such that $d_w = \infty$:*
     - *Query (v,w).*
     - *If (v,w) is an edge, set $d_w := d_v + 1$ and add w to the end of $\mathcal{L}$.*
   - *Remove v from $\mathcal{L}$.*

We omit the proof of correctness of this algorithm (see for example [17]). This algorithm uses up to $T = O(n^2)$ queries. If the guessing algorithm always guesses that (v,w) is not an edge, then it makes at most $G = n-1$ mistakes; hence $Q(f) = O(\sqrt{TG}) = O(n^{3/2})$.[3] □

The previous best known quantum algorithm for unweighted SSSP, to our best knowledge, was given by Furrow [24]; that algorithm has query complexity $O(n^{3/2}\log n)$.

We now consider the quantum query complexity of unweighted k-source shortest paths (finding k shortest-path trees rooted from k beginning vertices). If we apply Algorithm 6.3 on k different starting vertices, then the expected number of wrong guesses is not greater than $G = k(n-1)$; however, the total number of edges we query need not exceed $T = O(n^2)$, since an edge never needs to be queried more than once. Therefore

Corollary 6.4. *The quantum query complexity of unweighted k-source shortest paths in the adjacency matrix model is $O(k^{1/2}n^{3/2})$, where n is the number of vertices.*

We use this idea—that T need not exceed $O(n^2)$ when dealing with graph problems – again in the following section.

6.2 Maximum bipartite matching

Problem 6.5 (Maximum bipartite matching)**.** We are given as black box the adjacency matrix of an n-vertex bipartite graph $G = (V = X \cup Y, E)$, where the undirected set of edges E only run between the bipartite components X and Y. A *matching* of G is a list of edges of G that do not share vertices. Our task is to find a maximum matching of G, i. e., a matching that contains the largest possible number of edges.

In this section we show that

Theorem 6.6. *The quantum query complexity of maximum bipartite matching is $O(n^{7/4})$ in the adjacency matrix model, where n is the number of vertices.*

Proof. Once again we apply Theorem 5.5 to a classical algorithm. Classically, this problem is solved in $O(n^{5/2})$ time by the Hopcroft-Karp [26] algorithm (here $n = |V|$). We summarize the algorithm as follows. (This summary roughly follows that of [5].)

Algorithm 6.7 (Hopcroft-Karp algorithm for maximum bipartite matching [26])**.**

1. *Initialize an empty matching $\mathcal{M}$. $\mathcal{M}$ is a matching that will be updated until it is maximum.*

[3] It seems difficult to use our method to give a corresponding result for the adjacency list model; the result of a query is much harder to guess when the input alphabet is non-Boolean.

2. *Repeat the following steps until $\mathcal{M}$ is a maximum matching:*

   (a) *Define the* directed *graph $H = (V', E')$ as follows:*

$$\begin{aligned}
V' &= X \cup Y \cup \{s,t\}\,,\\
E' &= \{(s,x) \mid x \in X, (x,y) \notin \mathcal{M} \textit{ for all } y \in Y\}\\
&\quad \cup \{(x,y) \mid x \in X, y \in Y, (x,y) \in E, (x,y) \notin \mathcal{M}\}\\
&\quad \cup \{(y,x) \mid x \in X, y \in Y, (x,y) \in E, (x,y) \in \mathcal{M}\}\\
&\quad \cup \{(y,t) \mid y \in Y, (x,y) \notin \mathcal{M} \textit{ for all } x \in X\}
\end{aligned}$$

   where s and t are two extra auxiliary vertices. Note that if $(s, x_1, y_1, x_2, y_2, \ldots, x_\ell, y_\ell, t)$ is a path in H from s to t, then $x_i \in X$ and $y_i \in Y$ for all i. Additionally, the edges (aside from the first and last) alternate from being in $\mathcal{M}$ and not being in $\mathcal{M}$: $(x_i, y_i) \notin \mathcal{M}$, $(y_i, x_{i+1}) \in \mathcal{M}$. Such a path is called an augmenting path *in the literature.*

   We note that a query to the adjacency matrix of E' can be simulated by a query to the adjacency matrix of E.

   (b) *Using the breadth-first search algorithm (Algorithm 6.3), in the graph H, find the length of the shortest path, or distance, of all vertices from s. Let the distance from s to t be $2\ell + 1$.*

   (c) *Find a maximal set S of vertex-disjoint shortest paths from s to t in the graph H. In other words, S should be a list of paths from s to t such that each path has length $2\ell + 1$, and no pair of paths share vertices except for s and t. Moreover, all other shortest paths from s to t share at least one vertex (except for s and t) with a path in S. We describe how to find such a maximal set in Algorithm 6.8.*

   (d) *If S is empty, i. e., there are no paths from s to t in the graph H, then it follows that $\mathcal{M}$ is a maximum matching, and we terminate. Otherwise continue to Step 2(e):*

   (e) *Let $(s, x_1, y_1, x_2, y_2, \ldots, x_\ell, y_\ell, t)$ be a path in S. Remove the $\ell - 1$ edges (x_{i+1}, y_i) from $\mathcal{M}$, and insert the ℓ edges (x_i, y_i) into $\mathcal{M}$. This increases $|\mathcal{M}|$ by 1. Repeat for all paths in S; there are no conflicts since the paths in S are vertex-disjoint.*

Once again, we omit the proof of correctness of this algorithm; the correctness is guaranteed by Berge's Lemma [10], which states that a matching is maximum if there are no more augmenting paths for the matching. Moreover, $O(\sqrt{n})$ iterations of Step 2 suffice [26].

We now describe how to find a maximal set of shortest-length augmenting paths in Step 2(c). This algorithm is essentially a modified version of depth-first search.

Algorithm 6.8 (Finding a maximal set of vertex-disjoint shortest-length augmenting paths)**.**

Input. The directed graph H defined in Algorithm 6.7, as well as the distances d_v of all vertices v from s (calculated in Step 2(b) of Algorithm 6.7).

1. *Initialize a set of paths $S := \emptyset$, set of vertices $R := \{s\}$, and a stack*[4] *of vertices $\mathcal{L} := (s)$. $\mathcal{L}$ contains the ordered list of vertices that we have begun, but not yet finished, processing. R is the set of*

[4] A stack is a data structure such that elements that are first inserted into the stack are removed last. We say that the last vertex added to (and still in) the stack is at the *top* of the stack; removing the top element is *popping* from the stack.

vertices that we have processed. S is the set of vertex-disjoint shortest-length augmenting paths that we have found.

2. *Repeat until $\mathcal{L}$ is empty:*

   (a) *If the top vertex of $\mathcal{L}$ (i. e., the* last *vertex added to, and still in, $\mathcal{L}$) is t, we have found a new vertex-disjoint path from s to t:*

   - *Trace the path from t back to s by removing elements from the front of $\mathcal{L}$ until s is at the front. Add the corresponding path to S.*
   - *Start again from the beginning of Step 2.*

   (b) *Let v be the top vertex of $\mathcal{L}$ (i. e., the vertex* last *added to, and still in, $\mathcal{L}$). Recall the distance from s to v is d_v.*

   (c) *Find w such that $w \notin R$, $d_w = d_v + 1$, and (v,w) (as an edge in H) has not been queried in this algorithm. If no such vertex w exists, pop (remove) v from $\mathcal{L}$ and start from the beginning of Step 2.*

   (d) *Query (v,w) on the graph H.*

   (e) *If (v,w) is an edge, push w into $\mathcal{L}$. If $w \neq t$, add w to R.*

3. *Output S, the maximal set of vertex-disjoint shortest-length augmenting paths.*

We now return to Algorithm 6.7 and count T and G. There is obviously no need to query the same edge more than once, so $T = O(n^2)$. If the algorithm always guesses, on a query (v,w), that there is no edge between (v,w), then it makes at most $G = O(n^{3/2})$ mistakes: in Step 2(b) there are at most $O(n)$ mistakes (see Algorithm 6.3), while in Step 2(c)/Algorithm 6.8 for each vertex $v \neq t$ there is at most one queried edge pointing to v, and edges pointing to t can be computed without queries to the adjacency matrix of H. Since Step 2 is executed $O(\sqrt{n})$ times, our counting follows.

Thus there is a quantum query algorithm with complexity $Q = O(\sqrt{TG}) = O(n^{7/4})$. □

To our knowledge, this is the first known nontrivial upper bound on the query complexity of maximum bipartite matching.[5] The time complexity of this problem was studied by Ambainis and Špalek in [5]; they gave an upper bound of $O(n^2 \log n)$ time in the adjacency matrix model. A lower bound of $\Omega(n^{3/2})$ for the query complexity of this problem was given in [11, 54].

For readers familiar with network flow, the arguments in this section also apply to Dinic's algorithm for maximum flow [18] on graphs with unit capacity, i. e., where the capacity of each edge is 0 or 1. On graphs with unit capacity, Dinic's algorithm is essentially the same as Hopcroft-Karp's, except that augmenting paths are over a general, nonbipartite flow network. (The set S in Step 2(c) of Algorithm 6.7 is generally referred to as a *blocking flow* in this context.) It can be shown that only $O(\min\{m^{1/2}, n^{2/3}\})$ iterations of Step 2 are required [32, 23], where m is the number of edges of the graph. Thus $T = O(n^2)$, $G = O(\min\{m^{1/2}, n^{2/3}\}n)$, and therefore

Theorem 6.9. *The quantum query complexity of the maximum flow problem in graphs with unit capacity is $O(\min\{n^{3/2}m^{1/4}, n^{11/6}\})$, where n and m are the number of vertices and edges in the graph, respectively.*

[5] The trivial upper bound is $O(n^2)$, where all pairs of vertices are queried.

It is an open question whether a similar result for maximum matching in a general nonbipartite graph can be proven, perhaps by applying Theorem 5.5 to the classical algorithm of Micali and Vazirani [39].

7 Projective query complexity

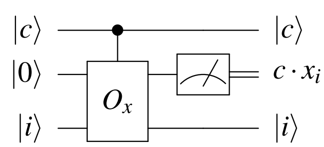
We end this paper with a brief discussion on another query complexity model, which we will call the *projective query complexity*. This model is similar to the bomb query model in that the only way of accessing x_i is through a classical measurement; however, in the projective query model the algorithm does not terminate if a 1 is measured. Our motivation for considering the projective query model is that its power is intermediate between the classical and quantum query models. To the best of our knowledge, this model was first considered in 2002 in unpublished results by Scott Aaronson [1].

A circuit in the projective query complexity model is a restricted quantum query circuit, with the following restrictions on the use of the quantum oracle.

1. We have an extra control register $|c\rangle$ used to control whether O_x is applied (we call the controlled version CO_x):
$$CO_x|c,r,i\rangle = |c, r \oplus (c \cdot x_i), i\rangle\,.$$
where $\cdot$ indicates Boolean AND.

2. The record register, $|r\rangle$ in the definition of CO_x above, *must* contain $|0\rangle$ before CO_x is applied.

3. After CO_x is applied, the record register is immediately measured in the computational basis, giving the answer $c \cdot x_i$. The result, a classical bit, can then be used to control further quantum unitaries (although only controlling the next unitary is enough, since the classical bit can be stored).



(7.1)

We wish to evaluate a function $f(x)$ with as few calls to this *projective oracle* as possible. Let the number of oracle calls required to evaluate $f(x)$, with at most δ error, be $P_\delta(f)$. By gap amplification, the choice of δ affects $P_\delta(f)$ by at most a factor of $\log(1/\delta)$, and thus we will often omit δ.

We can compare the definition in this section with the definition of the bomb query complexity in Section 3; the only difference is that if $c \cdot x_i = 1$, the algorithm terminates in the bomb model, while the algorithm can continue in the projective model. Therefore the following is evident.

Observation 7.1. *$P_\delta(f) \le B_{\varepsilon,\delta}(f)$, and therefore $P(f) = O(Q(f)^2)$.*

Moreover, it is clear that the projective query model has power intermediate between classical and quantum (a controlled query in the usual quantum query model can be simulated by appending a 0 to the input string), and therefore letting $R_\delta(f)$ be the classical randomized query complexity,

Observation 7.2. *$Q_\delta(f) \le P_\delta(f) \le R_\delta(f)$.*

For explicit bounds on P, Regev and Schiff [46] have shown that for computing the OR function, the projective query complexity loses the Grover speedup.[6]

Theorem 7.3 (Regev–Schiff)**.** $P(\mathrm{OR}) = \Omega(N)$.

Note that this result says nothing about $P(\mathrm{AND})$, since the definition of $P(f)$ is asymmetric with respect to 0 and 1 in the input—in Circuit 7.1 the bit $c \cdot x_i$ is measured, and this is asymmetric with respect to x_i being 0 and 1.[7]

We observe that there could be a separation in both parts of the inequality $Q \le P \le B$.

$$Q(\mathrm{OR}) = \Theta(\sqrt{N}), \quad P(\mathrm{OR}) = \Theta(N), \quad B(\mathrm{OR}) = \Theta(N)$$
$$Q(\mathrm{PARITY}) = \Theta(N), \quad P(\mathrm{PARITY}) = \Theta(N), \quad B(\mathrm{PARITY}) = \Theta(N^2)\,.$$

In the latter equation we used the fact that $Q(\mathrm{PARITY}) = \Theta(N)$ [6]. It therefore seems difficult to adapt our lower bound method in Section 4.2 to $P(f)$.

It is also worth noting that Ben-David recently discovered a total function f such that $R(f) = \tilde{\Omega}(Q(f)^{2.5})$ [8, 2] , and thus $R(f) = \tilde{\Omega}(P(f)^{1.125})$. A separation between R and P is therefore possible even for total functions—this fact is perhaps surprising considering the separation between $P(\mathrm{OR})$ and $Q(\mathrm{OR})$, i. e., that Grover's algorithm breaks down in the projective query model.

It would be interesting to find a general lower bound for $P(f)$, or to establish more clearly the relationship between $Q(f)$, $P(f)$, and $R(f)$.

Acknowledgements

We are grateful to Scott Aaronson and Aram Harrow for many useful discussions, and Scott Aaronson and Shelby Kimmel for valuable suggestions on a preliminary draft. We also thank Andrew Childs for giving us permission to make use of his online proof of the general adversary lower bound in [15]. Special thanks to Robin Kothari for pointing us to his paper [34], and in particular his analysis showing that logarithmic factors can be removed from the query complexity of Algorithm 5.9. We also thank the anonymous referees from QIP, CCC, and ToC for their helpful comments. This work is supported by the ARO grant Contract Number W911NF-12-0486. CYL gratefully acknowledges support from the Natural Sciences and Engineering Research Council of Canada.

References

[1] SCOTT AARONSON: Personal communication, 2014. 3, 29

[6] More precisely, Regev and Schiff [46] studied the following oracle model for the Grover problem: for each oracle application with probability $(1-p)$ the oracle is applied correctly, and with probability p the identity is applied instead. Claim 2 in their paper shows that for $p = 1/2$, their model is equivalent to the projective query model.

[7] We could have defined a symmetric version of P, say $\tilde{P}$, by allowing an extra guess on the measurement result, similar to our construction of $\tilde{B}$ in Section 5.1. Unfortunately, Regev and Schiff's result, Theorem 7.3, does not apply to this case, and we see no obvious equivalence between P and $\tilde{P}$.

[2] SCOTT AARONSON, SHALEV BEN-DAVID, AND ROBIN KOTHARI: Separations in query complexity using cheat sheets. In *Proc. 48th STOC*, pp. 863–876. ACM Press, 2016. [doi:10.1145/2897518.2897644, arXiv:1511.01937] 4, 30

[3] ANDRIS AMBAINIS: Quantum lower bounds by quantum arguments. *J. Comput. System Sci.*, 64(4):750–767, 2002. Preliminary version in STOC'00. [doi:10.1006/jcss.2002.1826, arXiv:quant-ph/0002066] 2, 13

[4] ANDRIS AMBAINIS: Quantum walk algorithm for element distinctness. *SIAM J. Comput.*, 37(1):210–239, 2007. Preliminary version in FOCS'04. [doi:10.1137/S0097539705447311, arXiv:quant-ph/0311001] 2

[5] ANDRIS AMBAINIS AND ROBERT ŠPALEK: Quantum algorithms for matching and network flows. In *Proc. 23rd Symp. Theoretical Aspects of Comp. Sci. (STACS'06)*, volume 3884 of *LNCS*, pp. 172–183. Springer, 2006. [doi:10.1007/11672142_13, arXiv:quant-ph/0508205] 3, 4, 26, 28

[6] ROBERT BEALS, HARRY BUHRMAN, RICHARD CLEVE, MICHELE MOSCA, AND RONALD DE WOLF: Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. Preliminary version in FOCS'98. [doi:10.1145/502090.502097, arXiv:quant-ph/9802049] 2, 4, 30

[7] ALEKSANDRS BELOVS: Span programs for functions with constant-sized 1-certificates: extended abstract. In *Proc. 44th STOC*, pp. 77–84. ACM Press, 2012. [doi:10.1145/2213977.2213985, arXiv:1105.4024] 2

[8] SHALEV BEN-DAVID: A super-Grover separation between randomized and quantum query complexities. *Electron. Colloq. on Comput. Complex. (ECCC)*, 2015. ECCC. [arXiv:1506.08106] 4, 30

[9] CHARLES H. BENNETT, ETHAN BERNSTEIN, GILLES BRASSARD, AND UMESH V. VAZIRANI: Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997. [doi:10.1137/S0097539796300933, arXiv:quant-ph/9701001] 2

[10] CLAUDE BERGE: Two theorems in graph theory. *Proc. Nat. Acad. Sci. U.S.A.*, 43(9):842–844, 1957. [doi:10.1073/pnas.43.9.842] 27

[11] AIJA BERZINA, ANDREJ DUBROVSKY, RUSINS FREIVALDS, LELDE LACE, AND OKSANA SCEGULNAJA: Quantum query complexity for some graph problems. In *Proc. 30th Conf. Current Trends Theory and Pract. Comput. Sci. (SOFSEM'04)*, volume 2932 of *LNCS*, pp. 140–150. Springer, 2004. [doi:10.1007/978-3-540-24618-3_11] 28

[12] RAJENDRA BHATIA: *Matrix Analysis*. Springer, 1997. [doi:10.1007/978-1-4612-0653-8] 5

[13] AHARON BRODUTCH, DANIEL NAGAJ, OR SATTATH, AND DOMINIQUE UNRUH: An adaptive attack on Wiesner's quantum money. *Quantum Inf. Comput.*, 16(11&12):1048–1070, 2016. [arXiv:1404.1507] 3

[14] HARRY BUHRMAN AND RONALD DE WOLF: Complexity measures and decision tree complexity: a survey. *Theoret. Comput. Sci.*, 288(1):21–43, 2002. [doi:10.1016/S0304-3975(01)00144-X] 4

[15] ANDREW CHILDS: The adversary method, 2013. Available at author's website. 5, 14, 30

[16] STEPHEN A. COOK, CYNTHIA DWORK, AND RÜDIGER REISCHUK: Upper and lower time bounds for parallel random access machines without simultaneous writes. *SIAM J. Comput.*, 15(1):87–97, 1986. [doi:10.1137/0215006] 4

[17] THOMAS H. CORMEN, CHARLES E. LEISERSON, RONALD L. RIVEST, AND CLIFFORD STEIN: *Introduction to Algorithms (3. ed.)*. MIT Press, 2009. Available at publisher's website. 26

[18] YEFIM A. DINITZ: Algorithm for solution of a problem of maximum flow in a network with power estimation. *Soviet Math. Doklady*, 11(5):1277–1280, 1970. Available at author's website. 28

[19] SEBASTIAN DÖRN: Quantum algorithms for matching problems. *Theory Comput. Syst.*, 45(3):613–628, 2009. [doi:10.1007/s00224-008-9118-x] 3

[20] CHRISTOPH DÜRR, MARK HEILIGMAN, PETER HØYER, AND MEHDI MHALLA: Quantum query complexity of some graph problems. *SIAM J. Comput.*, 35(6):1310–1328, 2006. Preliminary version in ICALP'04. [doi:10.1137/050644719, arXiv:quant-ph/0401091] 25

[21] CHRISTOPH DÜRR AND PETER HØYER: A quantum algorithm for finding the minimum, 1996. [arXiv:quant-ph/9607014] 3, 22

[22] AVSHALOM C. ELITZUR AND LEV VAIDMAN: Quantum mechanical interaction-free measurements. *Found. Phys.*, 23(7):987–997, 1993. [doi:10.1007/BF00736012, arXiv:hep-th/9305002] 2, 5

[23] SHIMON EVEN AND ROBERT ENDRE TARJAN: Network flow and testing graph connectivity. *SIAM J. Comput.*, 4(4):507–518, 1975. Preliminary version in STOC'74. [doi:10.1137/0204043] 28

[24] BARTHOLOMEW FURROW: A panoply of quantum algorithms. *Quantum Inf. Comput.*, 8(8):834–859, 2008. ACM DL. [arXiv:quant-ph/0606127] 3, 26

[25] LOV K. GROVER: A fast quantum mechanical algorithm for database search. In *Proc. 28th STOC*, pp. 212–219. ACM Press, 1996. [doi:10.1145/237814.237866, arXiv:quant-ph/9605043] 2

[26] JOHN E. HOPCROFT AND RICHARD M. KARP: An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2(4):225–231, 1973. Preliminary version in SWAT'71. [doi:10.1137/0202019] 26, 27

[27] ONUR HOSTEN AND PAUL G. KWIAT: Weak measurements and counterfactual computation, 2006. [arXiv:quant-ph/0612159] 3

[28] ONUR HOSTEN, MATTHEW T. RAKHER, JULIO T. BARREIRO, NICHOLAS A. PETERS, AND PAUL G. KWIAT: Counterfactual computation revisited, 2006. [arXiv:quant-ph/0607101] 3

[29] ONUR HOSTEN, MATTHEW T. RAKHER, JULIO T. BARREIRO, NICHOLAS A. PETERS, AND PAUL G. KWIAT: Counterfactual quantum computation through quantum interrogation. *Nature*, 439:949–952, 2006. [doi:10.1038/nature04523] 3

[30] PETER HØYER, TROY LEE, AND ROBERT ŠPALEK: Negative weights make adversaries stronger. In *Proc. 39th STOC*, pp. 526–535. ACM Press, 2007. [doi:10.1145/1250790.1250867, arXiv:quant-ph/0611054] 2, 14, 15, 18

[31] STACEY JEFFERY, ROBIN KOTHARI, AND FRÉDÉRIC MAGNIEZ: Nested quantum walks with quantum data structures. In *Proc. 24th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'13)*, pp. 1474–1485. ACM Press, 2013. [doi:10.1137/1.9781611973105.106, arXiv:1210.1199] 2

[32] ALEXANDER V. KARZANOV: O nakhozhdenii maksimal'nogo potoka v setyakh spetsial'nogo vida i nekotorykh prilozheniyakh ("On finding maximum flow in networks of a special kind and in certain applications," in Russian). In *Matematicheskie Voprosy Upravleniya Proizvodstvom (Mathematical Questions of Management)*, volume 5, pp. 81–94. Moscow State Univ. Press, 1973. 28

[33] SHELBY KIMMEL: Quantum adversary (upper) bound. *Chicago J. Theor. Comput. Sci.*, 2013(4), 2013. Preliminary version in ICALP'12. [doi:10.4086/cjtcs.2013.004, arXiv:1101.0797] 2

[34] ROBIN KOTHARI: An optimal quantum algorithm for the oracle identification problem. In *Proc. 31st Symp. Theoretical Aspects of Comp. Sci. (STACS'14)*, volume 25 of *LIPIcs*, pp. 482–493. Schloss Dagstuhl, 2014. [doi:10.4230/LIPIcs.STACS.2014.482, arXiv:1311.7685] 3, 22, 25, 30

[35] PAUL KWIAT, HARALD WEINFURTER, THOMAS HERZOG, ANTON ZEILINGER, AND MARK A. KASEVICH: Interaction-free measurement. *Phys. Rev. Lett.*, 74(24):4763–4766, 1995. [doi:10.1103/PhysRevLett.74.4763] 2, 3, 6, 9

[36] TROY LEE, RAJAT MITTAL, BEN W. REICHARDT, ROBERT ŠPALEK, AND MARIO SZEGEDY: Quantum query complexity of state conversion. In *Proc. 52nd FOCS*, pp. 344–353. IEEE Comp. Soc. Press, 2011. [doi:10.1109/FOCS.2011.75, arXiv:1011.3020] 2, 14, 18

[37] CEDRIC YEN-YU LIN AND HAN-HSUAN LIN: Upper bounds on quantum query complexity inspired by the Elitzur-Vaidman bomb tester. In *Proc. 30th IEEE Conf. on Computational Complexity (CCC'15)*, pp. 537–566. IEEE Comp. Soc. Press, 2015. [doi:10.4230/LIPIcs.CCC.2015.537, arXiv:1410.0932] 1

[38] FRÉDÉRIC MAGNIEZ, ASHWIN NAYAK, JÉRÉMIE ROLAND, AND MIKLOS SANTHA: Search via quantum walk. *SIAM J. Comput.*, 40(1):142–164, 2011. Preliminary version in STOC'07. [doi:10.1137/090745854, arXiv:quant-ph/0608026] 2

[39] SILVIO MICALI AND VIJAY V. VAZIRANI: An $O(\sqrt{|V|}\cdot|E|)$ algorithm for finding maximum matching in general graphs. In *Proc. 21st FOCS*, pp. 17–27. IEEE Comp. Soc. Press, 1980. [doi:10.1109/SFCS.1980.12] 4, 29

[40] BAIDYANATH MISRA AND E. C. GEORGE SUDARSHAN: The Zeno's paradox in quantum theory. *J. Math. Phys.*, 18(4):756–763, 1977. [doi:10.1063/1.523304] 2, 6, 9

[41] GRAEME MITCHISON AND RICHARD JOZSA: Counterfactual computation. *Proc. Royal Soc. A*, 457(2009):1175–1194, 2001. [doi:10.1098/rspa.2000.0714, arXiv:quant-ph/9907007] 3

[42] GRAEME MITCHISON AND RICHARD JOZSA: The limits of counterfactual computation, 2006. [arXiv:quant-ph/0606092] 3

[43] MICHAEL A. NIELSEN AND ISAAC L. CHUANG: *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000. 4

[44] NOAM NISAN: CREW PRAMs and decision trees. *SIAM J. Comput.*, 20(6):999–1007, 1991. Preliminary version in STOC'89. [doi:10.1137/0220062] 4

[45] TAE-GON NOH: Counterfactual quantum cryptography. *Phys. Rev. Lett.*, 103(23):230501, 2009. [doi:10.1103/PhysRevLett.103.230501, arXiv:0809.3979] 3

[46] ODED REGEV AND LIRON SCHIFF: Impossibility of a quantum speed-up with a faulty oracle. In *Proc. 35th Internat. Colloq. on Automata, Languages and Programming (ICALP'08)*, volume 5125 of *LNCS*, pp. 773–781. Springer, 2008. [doi:10.1007/978-3-540-70575-8_63, arXiv:1202.1027] 3, 4, 30

[47] BEN W. REICHARDT: Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function. In *Proc. 50th FOCS*, pp. 544–551. IEEE Comp. Soc. Press, 2009. [doi:10.1109/FOCS.2009.55, arXiv:0904.2759] 2

[48] BEN W. REICHARDT: Reflections for quantum query algorithms. In *Proc. 22nd Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'11)*, pp. 560–569. ACM Press, 2011. [doi:10.1137/1.9781611973082.44, arXiv:1005.1601] 2

[49] HATIM SALIH, ZHENG-HONG LI, M. AL-AMRI, AND M. SUHAIL ZUBAIRY: Protocol for direct counterfactual quantum communication. *Phys. Rev. Lett.*, 110(17):170502, 2013. [doi:10.1103/PhysRevLett.110.170502, arXiv:1206.2042] 3

[50] MARIO SZEGEDY: Quantum speed-up of Markov chain based algorithms. In *Proc. 45th FOCS*, pp. 32–41. IEEE Comp. Soc. Press, 2004. [doi:10.1109/FOCS.2004.53] 2

[51] LEV VAIDMAN: The impossibility of the counterfactual computation for all possible outcomes. *Phys. Rev. Lett.*, 98(16):160403, 2007. [doi:10.1103/PhysRevLett.98.160403, arXiv:quant-ph/0610174] 3

[52] LEV VAIDMAN: Comment on "Protocol for direct counterfactual quantum communication". *Phys. Rev. Lett.*, 112(20):208901, 2014. [doi:10.1103/PhysRevLett.112.208901, arXiv:1304.6689] 3

[53] STEPHEN WIESNER: Conjugate coding. *ACM SIGACT News*, 15(1):78–88, 1983. [doi:10.1145/1008908.1008920] 3

[54] SHENGYU ZHANG: On the power of Ambainis lower bounds. *Theoret. Comput. Sci.*, 339(2-3):241–256, 2005. Preliminary version in ICALP'04. [doi:10.1016/j.tcs.2005.01.019, arXiv:quant-ph/0311060] 28

AUTHORS

Cedric Yen-Yu Lin
Joint Center for Quantum Information and Computer Science
University of Maryland, College Park, MD
cedricl@umiacs.umd.edu

Han-Hsuan Lin
Centre for Quantum Technologies
National University of Singapore, Singapore
han.manatsummit@gmail.com

ABOUT THE AUTHORS

CEDRIC YEN-YU LIN received his Ph. D. in Physics from MIT in 2015, under the supervision of Edward Farhi. His research interests are quantum algorithms and complexity, including query complexity, quantum algorithms for algebraic problems, Hamiltonian complexity, and quantum structural complexity. He is currently a postdoctoral fellow at the Joint Center for Quantum Information and Computer Science at the University of Maryland.

HAN-HSUAN LIN received his Ph. D. from MIT in 2015 under the supervision of Edward Farhi. His research interests include quantum algorithms and quantum query complexity. He currently holds a short-term postdoctoral position at the Institute of Information Science, Academia Sinica in Taiwan. He will join the Centre for Quantum Technologies in Singapore on December 2, 2016 as a research fellow.

This work was completed while the authors were at the Center for Theoretical Physics, MIT.