Uselessness for an Oracle Model with Internal Randomness

Aram W. Harrow
Center for Theoretical Physics
Massachusetts Institute of Technology

David J. Rosenbaum*
Department of Computer Science & Engineering
University of Washington

September 24, 2013

Abstract

We consider a generalization of the standard oracle model in which the oracle acts on the target with a permutation selected according to internal random coins. We describe several problems that are impossible to solve classically but can be solved by a quantum algorithm using a single query; we show that such infinity-vs-one separations between classical and quantum query complexities can be constructed from much weaker separations.

We also give conditions to determine when oracle problems—either in the standard model, or in any of the generalizations we consider—cannot be solved with success probability better than random guessing would achieve. In the oracle model with internal randomness where the goal is to gain any nonzero advantage over guessing, we prove (roughly speaking) that k quantum queries are equivalent in power to $2k$ classical queries, thus extending results of Meyer and Pommersheim.

arXiv:1111.1462v2 [quant-ph] 23 Sep 2013

---

*Corresponding author: `djr@cs.washington.edu`

1 Introduction

Oracles are an important conceptual framework for understanding quantum speedups. They may represent subroutines whose code we cannot usefully examine, or an unknown physical system whose properties we would like to estimate. When used by a quantum computer, the most general form of an oracle is a possibly noisy quantum operation that can be applied to an n-qubit input. However, oracles this general have no obvious classical analogue, which makes it difficult to compare the ability of classical and quantum computers to efficiently interrogate oracles. This was the original motivation of the *standard oracle model*, in which f is a function from $[N] = \{1, \ldots, N\}$ to $\{0, 1\}$, and the oracle $\mathcal{O}_f$ acts for a classical computer by mapping x, y to $x, y \oplus f(x)$, and for a quantum computer as a unitary that maps $|x, y\rangle$ to $|x, y \oplus f(x)\rangle$. One way to justify the standard oracle model is that if there is a (not necessarily reversible) classical circuit computing f, then $\mathcal{O}_f$ can be simulated by computing f, XORing the answer onto the target, and uncomputing f.

In this paper, we consider other forms of oracles that are more general than the standard oracle model, but nevertheless permit comparison between classical and quantum query complexities. Meyer and Pommersheim [6] generalized the standard model by letting A be a deterministic classical algorithm that takes the control x of the oracle and computes a value $A(x)$. The oracle then acts by applying a permutation $\pi_{A(x)}$ to the target. We will further generalize the model by replacing A with a randomized classical algorithm. The random coins used by A are internal to the oracle and cannot be accessed externally. We call this concept an *oracle with internal randomness*. Note that even if A takes no input, the oracle can still be interesting since it may apply different permutations depending on its internal coin flips.

Oracles with internal randomness correspond naturally to the situation in which a (quantum or classical) computer seeks to determine properties of a device that acts in a noisy or otherwise non-deterministic manner. One simple example is an oracle that "misfires", i.e. when queried, the oracle does nothing with probability p and responds according to the standard oracle model with probability $1 - p$. This model was considered in [9], which found, somewhat surprisingly, that the square-root advantage of Grover search disappears (i.e. there is an $\Omega(N)$ quantum query lower bound for computing the OR function) for any constant $p > 0$.

The rest of our paper is divided into two parts. First, we explore various examples of oracles with internal randomness that demonstrate the power of the model. We will see that in some cases (e.g. Theorems 1 and 2), this can even result in problems solvable with one quantum query that are completely unsolvable using classical queries.

In the second part, we consider the question of when oracle problems can be solved with any nontrivial advantage; i.e. a probability of success better than could be obtained by simply guessing the answer according to the prior distribution. For an example of when such advantage is *not* possible, consider the parity function on N bits. If these bits are drawn from the uniform distribution, then any classical algorithm making $\leq N - 1$ queries—or any quantum algorithm making $\leq \frac{N}{2} - 1$ queries—will not be able to guess the parity with any nontrivial advantage. In Section 3, we consider the problem of when some number of queries are *useless* for solving an oracle problem. Informally, our main result is roughly that k quantum queries are useless if and only if $2k$ classical queries are useless (this is formalized in Theorem 9). However, a subtlety arises in our theorem when oracles have internal randomness, in that the $2k$ classical queries need to be considered as k pairs, each of which uses a separate sample from the internal randomness of the oracle.

In the unbounded-error query complexity regime, similar results were obtained 15 years ago by Farhi, Goldstone, Gutman and Sipser [4] for the case of the parity function. More recently,

Montanaro, Nishimura and Raymond [8] proved a similar result for any binary function f, using techniques that do not readily generalize to non-binary f. One direction of the special case of our result for deterministic permutative oracles was proved by Meyer and Pommersheim [7]. Our proof is arguably simpler and more operational. We introduce an analogue of gate teleportation [5] for oracles by showing that oracles can be (a) encoded into states analogous to Choi-Jamiołkowski states, and (b) retrieved from those states with an exponentially small, but heralded, success probability (i.e. the procedure outputs a flag that tells us whether it succeeded or failed). We expect that this characterization will be useful for future study of query complexity in the regime where any nonzero advantage is sought.

Finally, our encoding can be used to construct infinity-vs-one separations from *any* separation between classical and quantum uselessness (see Theorems 18 and 19).

2 Examples of infinity-vs-one query-complexity separations

In this section, we discuss problems that can be solved using a single quantum query but cannot be solved classically even with an unlimited number of queries. Such a separation is far stronger even than exponential separations. To achieve such infinity-vs-one separations, it is necessary (but not sufficient) for the oracle to have internal randomness since otherwise one could simulate the quantum algorithm classically with exponential overhead. The key point is that internal randomness effectively causes a different oracle to be used for each query so such a simulation is not possible in this case.

2.1 Distinguishing involutions with no fixed points from cycles

Our first example of an infinity-vs-one separation is given by the problem of distinguishing involutions from cycles. Define $\text{INV} = \left\{ \pi \in S_N : \pi^2 = 1 \text{ and } \pi x \neq x \text{ for all } x \in [N] \right\}$; this is the set of involutions in S_N with no fixed points. Let $\text{CYC} = \{ \pi \in S_N : \pi \text{ is a cycle of length } N \}$. For any nonempty subset S of S_N, define $\mathcal{O}_S$ to be the oracle with a control $x \in [N]$ and a target $y \in [N]$ that acts according to Algorithm 1.

Algorithm 1 The oracle for the problem of distinguishing involutions with no fixed points from cycles

---
1: Select $\pi \in S$ uniformly at random
2: Compute $\pi(x)$ where x is the value of the control
3: Add $\pi(x)$ to the target y modulo N

Theorem 1. *Classical algorithms cannot solve the problem of distinguishing cycles from involutions with no fixed points with unbounded error using any number of queries.*

Proof. In this problem, an oracle $\mathcal{O}_S$ is given which is either $\mathcal{O}_{\text{INV}}$ or $\mathcal{O}_{\text{CYC}}$; the problem is to determine which of these is the case. Consider querying the oracle when the control is x. Then $\pi(x)$ is a uniformly random value in $[N] \setminus \{x\}$ for both cases so this problem cannot be solved by a classical algorithm. $\qquad\square$

However, the problem can be solved by a quantum algorithm using a single query to the oracle as shown in Algorithm 2.

Our algorithm will make use of the swap test [2]. Let F denote the swap operator, i.e. $F \ket{\alpha, \beta} = \ket{\beta, \alpha}$. Then the swap test is defined to be the measurement with outcomes $\{\frac{I+F}{2}, \frac{I-F}{2}\}$. The $\frac{I+F}{2}$ outcome is called the "symmetric" outcome and $\frac{I-F}{2}$ the "antisymmetric" outcome. A key property of the swap test is that when applied to pure states $\ket{\alpha} \otimes \ket{\beta}$, it will output "symmetric" with probability $\frac{1+|\langle \alpha | \beta \rangle|^2}{2}$, thus providing an estimate of their overlap.

Algorithm 2 The quantum algorithm for distinguishing involutions with no fixed points from cycles

---

1: Prepare the state $\frac{1}{\sqrt{N}} \sum_{x=1}^{N} \ket{x}$

2: Apply $\mathcal{O}_S$ to obtain the state $\frac{1}{\sqrt{N}} \sum_{x=1}^{N} \ket{x, \pi(x)}$

3: Apply the swap test to $\frac{1}{\sqrt{N}} \sum_{x=1}^{N} \ket{x, \pi(x)}$

4: **if** the swap test outputs "symmetric" **then return** "INV"

5: **else return** "CYC"

6: **end if**

We now show that the above algorithm effectively counts the number of transpositions in an arbitrary permutation which is sufficient to distinguish involutions from cycles.

Theorem 2. *Quantum algorithms can solve the problem of distinguishing cycles from involutions with no fixed points using a single query with one-sided error $1/2$.*

Proof. Consider a general state ρ^{AB} on two identical systems A and B. Then applying the swap test to this system (where the swap exchanges A and B) outputs 0 (symmetric) with probability $\Pr(0) = \frac{1+\operatorname{tr} \rho^{AB} F}{2}$. Applying this formula to the state $\frac{1}{\sqrt{N}} \sum_{x=1}^{N} \ket{x, \pi(x)}$, the probability of observing 0 is

$$\Pr(0) = \frac{1 + (1/N) \sum_{xy} \langle \pi(y)|x \rangle \langle y|\pi(x) \rangle}{2} \tag{1}$$

$$= \frac{1 + (1/N) |\{(x,y) : \pi(x) = y \text{ and } \pi(y) = x\}|}{2} \tag{2}$$

This probability is $1/2$ if $\pi \in$ CYC and is 1 if $\pi \in$ INV. $\qquad \square$

Hence, there is an infinity-vs-one separation in the unbounded-error classical and quantum query complexities for this problem. This analysis can also be applied to obtain an algorithm for estimating the number of transpositions in any permutation.

2.2 An infinity-vs-$\Theta(n)$ separation for a modification of Simon's problem

We now show how to modify Simon's problem [10] to obtain an infinity-vs-$\Theta(n)$ separation between the classical and quantum query complexities. Recall that for Simon's problem, we are given oracle access to a function $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ and $f(x) = f(y)$ if and only if $x = y + a$ for some fixed element $a \in \mathbb{Z}_2^n$ and our task is to determine a. Classically, exponentially many queries are required; however, quantumly at each step we learn a vector that is orthogonal to a so that the expected number of queries required is $\Theta(n)$. The crucial point here is that this algorithm will return a vector orthogonal to a for any f that is constant and distinct on the cosets $\{x, x + a\}$, so if f changes between calls to the oracle and a does not, then the quantum algorithm will not be affected.

3

Our randomized oracle is defined as follows. Fix some unknown $a \in \mathbb{Z}_2^n$. Then construct an oracle $\mathcal{O}_a : |x\rangle |y\rangle \mapsto |x\rangle |y + f(x)\rangle$ where $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ is selected uniformly at random at each call subject to the constraint that $f(x) = f(y)$ if and only if $x = y + a$. The problem is then to determine a.

Classically, this cannot be done since each query to the oracle results in a random number; however, the quantum algorithm still requires only $\Theta(n)$ queries.

2.3   An infinity-vs-one separation for the hidden linear structure problem

Beaudrap, Cleve and Watrous [3] introduced the hidden linear structure problem where we are given a blackbox that performs the mapping $|x\rangle |y\rangle \mapsto |x\rangle |\pi(y + sx)\rangle$ where $\pi \in S_q$ and $s \in GF(q)$ for $q = 2^n$. The problem is to find s. By extending quantum Fourier transforms to $GF(q)$, Beaudrap, Cleve and Watrous [3] show that this problem can be solved exactly using a single quantum query but classical algorithms require $\Omega(\sqrt{q})$ queries to determine s. They are able to achieve such a query complexity separation by using a non-standard (but still deterministic) oracle model. A similar separation ($O(1)$ vs $\Omega(N^{1/4})$) was achieved in the standard oracle model by Aaronson [1].

We propose the following randomized variant of their oracle problem. Fix some (unknown) $s \in GF(q)$. Then define the oracle by $\mathcal{O}_s : |x\rangle |y\rangle \mapsto |x\rangle |\pi(y + sx)\rangle$ where π is selected uniformly at random for each query. The goal is still to determine s. Since the quantum algorithm only uses one query it is unaffected by this change; however, classically the output of the oracle is completely random at each query so we obtain an infinity-vs-one separation.

The three separations shown are examples of a more general phenomenon in which randomness can be used to amplify a modest quantum-vs-classical query separation into an unbounded one. We discuss this further in Section 4.

3   Uselessness for oracles with internal randomness

We now turn to the general problem of when some number of queries are *useless* for solving an oracle problem. Equivalently we can ask when it is possible to answer an oracle problem with any positive advantage over guessing.

To define oracle problems, we use a slightly more compact notation than in previous sections. An oracle π is defined by a collection of permutations $\pi_{x,r} \in S_M$, where $x \in [N]$ is input by the algorithm, and r is the internal randomness which is distributed according to R. Overloading notation, we say that if the oracle is queried k times, then $\boldsymbol{r} = (r_1, \ldots, r_k)$ is distributed according to R^k, which may not necessarily be an i.i.d. distribution.

To describe the problem we want to solve, we follow the notation of Meyer and Pommersheim [6] while adding internal randomness to the oracle. We are promised that our oracle belongs to a set C, which in general may be a strict subset of all functions from $[N] \times \operatorname{supp}(R)$ to S_M. The set C is partitioned into sets $\{C_j\}$, and our goal is to determine which C_j contains π. By an abuse of notation, we say that C is our oracle problem. Queries are made to an oracle $\mathcal{O}_\pi$ which acts by $|x\rangle |y\rangle \mapsto |x\rangle |\pi_{x,r}(y)\rangle$.

The oracle problem C is a worst-case problem for which we demand that algorithms work well for all choices of $\pi \in C$. However, we also consider average-case problems in which π is distributed according to a known distribution μ. The resulting oracle problem is denoted (C, μ).

Before stating our own results, we describe the main result of [7]. In their model there is no internal randomness, so the action of the oracle is simply $\pi_x \in S_M$ for each $x \in [N]$. If $\boldsymbol{x} = (x_1, \ldots, x_k)$ and $\boldsymbol{y} = (y_1, \ldots, y_k)$, then define $\pi_{\boldsymbol{x}}(\boldsymbol{y}) = (\pi_{x_1}(y_1), \ldots, \pi_{x_k}(y_k))$. Their result may then be stated as follows.

Definition 3 (Classical uselessness[7]). *k classical queries are useless for the oracle problem (C, μ) if for all $\boldsymbol{x} \in [N]^k$, $\boldsymbol{y} \in [M]^k$, $\boldsymbol{z} \in [M]^k$ and j, $\Pr(\pi \in C_j \mid \pi_{\boldsymbol{x}}(\boldsymbol{y}) = \boldsymbol{z}) = \Pr(\pi \in C_j)$, where π is distributed according to μ.*

Definition 4 (Quantum uselessness[7]). *k quantum queries are useless for the oracle problem (C, μ) if for any k-query quantum algorithm run on any initial state and any POVM measurement $\{M_s\}$ which is made on the output of the algorithm, $\Pr(\pi \in C_j \mid s) = \Pr(\pi \in C_j)$ for all j and s, where π is distributed according to μ.*

We pause to briefly comment on the connection to unbounded-error query complexity. Unbounded-error query complexity typically refers to binary problems, i.e. when C is partitioned into C_0, C_1 and the goal is to determine which one π belongs to with success probability $> 1/2$. In this case, the statement that k (quantum or classical) queries are useless for (C, μ) (for some μ) is equivalent to the unbounded-error query complexity of C being $> k$. This is stated precisely and proved in Appendix C.

The main result of [7] is the following theorem.

Theorem 5 (Classical uselessness implies quantum uselessness[7]). *For any deterministic oracle problem (C, μ), if $2k$ classical queries are useless then k quantum queries are useless.*

We will give an alternate proof of this theorem, establish a converse, and generalize it to oracles with internal randomness.

3.1 Definitions of Classical Uselessness

In order to characterize uselessness for oracles with internal randomness, we first need to extend the definitions to this case. As above, we define $\pi_{\boldsymbol{x}, \boldsymbol{r}}(\boldsymbol{y}) = (\pi_{x_1, r_1}(y_1), \ldots \pi_{x_k, r_k}(y_k))$. One natural definition of uselessness in this setting is that a classical algorithm ignorant of the oracle's internal randomness should not be able to gain any nontrivial advantage in learning which C_j contains π.

Definition 6 (Weak classical uselessness). *If (C, μ) is an oracle problem, then k classical queries are weakly useless if for all $\boldsymbol{x} \in [N]^k$, $\boldsymbol{y}, \boldsymbol{z} \in [M]^k$ and j, $\Pr(\pi \in C_j \mid \pi_{\boldsymbol{x}, \boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{z}) = \Pr(\pi \in C_j)$, where π and $\boldsymbol{r}$ are distributed according to μ and R^k.*

It is easy to see that if $2k$ classical queries are weakly useless then k quantum queries need not be useless since Algorithm 2 is a counterexample. A much stronger definition of uselessness would be to allow the classical algorithm to see, or equivalently to choose, the internal random bits used by the oracle.

Definition 7 (Strong classical uselessness). *If (C, μ) is an oracle problem, then k classical queries are strongly useless if for all $\boldsymbol{x} \in [N]^k$, $\boldsymbol{y}, \boldsymbol{z} \in [M]^k$ and all possible values $\boldsymbol{r} \in \mathrm{supp}(R^k)$,*

$$\Pr(\pi \in C_j \mid \pi_{\boldsymbol{x}, \boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{z}) = \Pr(\pi \in C_j) \tag{3}$$

for all j, where π is distributed according to μ.

We will see later that strong classical uselessness for $2k$ queries is sufficiently powerful to imply quantum uselessness for k queries. However, it is in fact too strong. A necessary and sufficient condition will come from having each of the k pairs share a seed.

Definition 8 (Pairwise classical uselessness)**.** *If (C, μ) is an oracle problem, then $2k$ classical queries are pairwise useless if for all $\boldsymbol{x}, \boldsymbol{x}' \in [N]^k$, $\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{z}, \boldsymbol{z}' \in [M]^k$ and j, $\Pr(\pi \in C_j \mid \pi_{\boldsymbol{x}, \boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{z}, \pi_{\boldsymbol{x}', \boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{z}') = \Pr(\pi \in C_j)$, where π and $\boldsymbol{r}$ are distributed according to μ and R^k.*

This definition ensures that each pair of query values (x_i, x_i') shares the same random seed r_i. We will see later that this corresponds precisely (in the unbounded error setting) to the power of quantum queries, because the density matrix resulting from a quantum query depends on only one random seed, while the different row and column indices interrogate two different choices of x, y.

It is important to note that weak classical uselessness and pairwise classical uselessness are not comparable: there exist problems that satisfy weak classical uselessness but not pairwise classical uselessness and vice versa. Section 2.1 gives an example where two classical queries are weakly useless but not pairwise useless. For an example of a problem where two classical queries are not weakly useless but are pairwise useless, let C be the set of all balanced binary functions on $\{0, 1\}$ and let f be chosen uniformly at random from C. Consider the task of determining the function implemented by the oracle that acts for the ith query by $|x\rangle \mapsto |x \oplus f(r_i)\rangle$ where r_i is the ith random seed; let r_1 be uniformly distributed in $\{0, 1\}$ and let $r_i = 0$ for $i \geq 2$. Clearly, two classical queries with the random seeds r_1 and r_2 determine f. However, two classical queries that share the random seed r_1 yield no useful information.

It is easy to show that uselessness does not depend on the distribution $\mu(\pi \in C_j)$ over the classes provided the probability of each class is positive. However, it does depend on the conditional distribution of the oracle within each class. Consider the problem of determining the parity of a binary function $f : [N] \to \{0, 1\}$; by tweaking the conditional distribution of f for each parity, we can cause $f(1)$ to be equal to the parity of f with high probability so a single query to f wouldn't be useless. On the other hand, if the conditional distribution for f were uniform, $N - 1$ classical queries would be useless.

3.2 Uselessness results

Our main result in this section is the following equivalence:

Theorem 9. *For any oracle problem (C, μ), k quantum queries are useless if and only if $2k$ classical queries are pairwise useless.*

For deterministic oracles, weak, pairwise and strong classical uselessness are all the same. In this case, Theorem 9 can be simplified to the following strengthening of Theorem 5.

Corollary 10. *For any deterministic oracle problem (C, μ), k quantum queries are useless if and only if $2k$ classical queries are useless.*

3.3 Encoding oracles in states

In this section we will prove Theorem 9. Our strategy will be to show that in the unbounded-error setting, the optimal algorithms make a series of fixed queries and then measure the resulting states. The key ingredient is to show that oracles can be encoded in states in a way that is perfectly

efficient in terms of queries (i.e. one oracle call creates one state, and one state simulates one oracle call), albeit at a cost of producing the output "I don't know" most of the time. We define these encodings first for deterministic oracles.

Definition 11. *Let $\mathcal{O}_\pi$ be a deterministic permutation oracle that maps $|x,y\rangle \in \mathbb{C}^N \otimes \mathbb{C}^M$ to $|x, \pi_x(y)\rangle$. Then define the* encoding *of π to be $|\psi_\pi\rangle = \frac{1}{\sqrt{NM}} \sum_{x\in[N],y\in[M]} |x\rangle^X |y\rangle^Y |\pi_x(y)\rangle^Z$. Here X, Y, Z label different registers for notational convenience.*

Clearly one use of $\mathcal{O}_\pi$ allows the creation of one copy of $|\psi_\pi\rangle$; simply prepare the state $\frac{1}{\sqrt{NM}} \sum_{x,y} |x\rangle^X |y\rangle^Y |y\rangle^Z$ and apply $\mathcal{O}_\pi$ to registers XZ. We will see shortly that one copy of $|\psi_\pi\rangle$ can in turn simulate one use of $\mathcal{O}_\pi$, albeit with a very high, but heralded, failure probability. Before proving this result, we show how Definition 11 generalizes to oracles with internal randomness.

Definition 12. *Let $\mathcal{O}_\pi$ be an oracle whose action is defined by $\mathcal{O}_\pi(|x\rangle\langle x'|\otimes|y\rangle\langle y'| = \mathbb{E}_{r\sim R} |x\rangle\langle x'|\otimes |\pi_{x,r}(y)\rangle\langle\pi_{x',r}(y')|$. For each r, define the deterministic oracle $\mathcal{O}_{\pi,r}$ by $\mathcal{O}_{\pi,r} |x,y\rangle = |x, \pi_{x,r}(y)\rangle$ and define the encoding for fixed r to be $|\psi_{\pi,r}\rangle = \frac{1}{\sqrt{NM}} \sum_{x\in[N],y\in[M]} |x\rangle^X |y\rangle^Y |\pi_{x,r}(y)\rangle^Z$.*

Now we define encodings of oracles with randomness.

Definition 13. *If $\mathcal{O}_\pi$ is an oracle with internal randomness, then define the* encoding *of $\mathcal{O}_\pi$ to be $\rho_\pi = \mathbb{E}_r \, \psi_{\pi,r}$.*

In this last definition, we use the convention that $\psi = |\psi\rangle\langle\psi|$. The utility of considering encodings comes from the following operational equivalence.

Theorem 14.

1. *One use of $\mathcal{O}_\pi$ can create one copy of ρ_π.*

2. *It is possible to consume one copy of ρ_π and simulate $\mathcal{O}_\pi$ with success probability $1/NM^2$. The simulation outputs a classical flag indicating success or failure.*

In both cases, the run time required is linear in the number of qubits, i.e. $O(\log NM)$.

We point out that in the simulation, failure destroys not only the encoding, but also the state input to the oracle. Nevertheless, this simulation is enough to distinguish the case when k queries are useless from the case when they are not.

Additionally, Theorem 14 is stated implicitly in terms of a distribution R. In the case of k queries correlated according to R^k, we have the following variant:

Theorem 15.

1. *k uses of $\mathcal{O}_\pi$ can create $\rho_\pi^k = \mathbb{E}_{\boldsymbol{r}\sim R^k}[\psi_{\pi,r_1} \otimes \cdots \otimes \psi_{\pi,r_k}]$.*

2. *It is possible to consume ρ_π^k and simulate k uses of $\mathcal{O}_\pi$ with success probability $1/N^k M^{2k}$, again with a flag indicating success or failure.*

As a corollary, for correlated internal randomness in the unbounded-error scenario, we can permit algorithms to make the k oracle calls in any order. We will prove Theorem 15 only, since it subsumes Theorem 14.

Proof. To create ρ_π^k, we simply apply $\mathcal{O}_\pi$ k times to $\left(\frac{1}{\sqrt{NM}} \sum_{x,y} |x\rangle^X |y\rangle^Y |y\rangle^Z\right)^{\otimes k}$.

For the second reduction, suppose we are given a copy of ρ_π^k and would like to apply $\mathcal{O}_\pi$ to simulate the ith query of some algorithm. If we condition on $\boldsymbol{r}$, then ρ_π^k becomes the state $\psi_{\pi,r_1} \otimes \cdots \otimes \psi_{\pi,r_k}$. We will use the ith component of this state to simulate our query.

Suppose we want to simulate the action of $\mathcal{O}_{\pi,r_i}$ on the state $|x'\rangle^{X'} |y'\rangle^{Y'}$. Define

$$A = \sum_{x\in[N]} |x\rangle^X \langle x,x|^{XX'} \otimes \frac{1}{\sqrt{M}} \sum_{y\in[M]} \langle y,y|^{YY'}$$

Since $AA^\dagger = \sum_x |x\rangle\langle x| = I_N$, it follows that $A^\dagger A \le I_{N^2 M^2}$ and $\{A, \sqrt{I - A^\dagger A}\}$ comprise valid Kraus operators for a quantum operation. Our simulation will apply this operation, with outcome A labeled success, and $\sqrt{I - A^\dagger A}$ labeled failure.

Upon outcome $\sqrt{I - A^\dagger A}$, the algorithm declares failure. If this occurs at any step of a multi-query algorithm, then the algorithm should guess j according to the *a priori* distribution μ. Thus, for the purposes of determining whether the algorithm outperforms the best guessing strategy, it then suffices to consider only the cases when outcome A occurs.

Upon outcome A, $|\psi_{\pi,r_i}\rangle^{XYZ} |x'\rangle^{X'} |y'\rangle^{Y'}$ is mapped to the (unnormalized) state $\frac{1}{\sqrt{NM^2}} |x'\rangle^X |\pi_{x,r_i}(y')\rangle^Z$. Since the normalization is independent of the input, this means that A occurs with probability $1/NM^2$ regardless of the input state. Conditioned on this outcome, the resulting map is precisely the action of $\mathcal{O}_{\pi,r_i}$.

The overall algorithm succeeds when each of the k queries succeeds. Since each query succeeds with probability $1/NM^2$, the overall algorithm succeeds with probability $1/N^k M^{2k}$. $\qquad\square$

Armed with our notion of encoding, it is straightforward to characterize quantum uselessness.

Corollary 16. *Define $\sigma_j = \mathbb{E}_{\pi\in C_j} \rho_\pi^k$. Then k quantum queries are useless if and only if all the σ_j are the same.*

Proof. By Theorem 15, any k-query algorithm can WLOG create ρ_π^k, resulting in the state σ_j if π is drawn randomly from C_j. The algorithm then proceeds to determine which σ_j it holds, using no further oracle queries. If all the σ_j are equal, then it can learn nothing about j. Conversely, if some σ_j is different from the others, then there is a measurement that will be able to guess j with positive advantage. $\qquad\square$

To conclude the proof of Theorem 9, observe that the quantity on the LHS of Definition 8 is precisely $\text{tr}\left(|\boldsymbol{x}\rangle\langle\boldsymbol{x'}| \otimes |\boldsymbol{y}\rangle\langle\boldsymbol{y'}| \otimes |\boldsymbol{z}\rangle\langle\boldsymbol{z'}|\right) \mathbb{E}_{\pi\in C_j} \mu(\pi)\rho_\pi^k = \text{tr}\left(|\boldsymbol{x}\rangle\langle\boldsymbol{x'}| \otimes |\boldsymbol{y}\rangle\langle\boldsymbol{y'}| \otimes |\boldsymbol{z}\rangle\langle\boldsymbol{z'}|\right)\sigma_j$ which will be independent of j for all $\boldsymbol{x}, \boldsymbol{x'}, \boldsymbol{y}, \boldsymbol{y'}, \boldsymbol{z}, \boldsymbol{z'}$ if and only if all of the σ_j are identical. Combined with Corollary 16, this completes the proof of Theorem 9.

Theorem 17. *Suppose that for some oracle problem k classical queries are weakly useless but k quantum queries are not useless. Then there exists an oracle problem in which this separation holds where the oracle acts by bitwise XOR.*

Proof. Consider an oracle problem (C, μ) for which k classical queries are weakly useless but k quantum queries are not useless. The oracle acts by $\mathcal{O} : |x\rangle |y\rangle \mapsto |x\rangle |\pi_{x,r_i}(y)\rangle$ on the ith call. We can define a new oracle $\mathcal{O}' : |x\rangle |y\rangle |z\rangle \mapsto |x\rangle |y\rangle |z \oplus \pi_{x,r_i}(y)\rangle$. Our new oracle $\mathcal{O}'$ can be used to prepare the encoding for $\mathcal{O}$ so k queries to $\mathcal{O}'$ can simulate any quantum algorithm that uses k queries to $\mathcal{O}$. Classically, $\mathcal{O}'$ can be simulated using $\mathcal{O}$ so we conclude that k classical queries to $\mathcal{O}'$ are weakly useless. $\qquad\square$

4 Amplifying separations

We now leverage our results to obtain a general method of amplifying any separation between classical and quantum uselessness. Let (C, μ) be an oracle problem where C is partitioned into $\{C_i\}$ and r_j is the jth random seed. For each $\pi \in C$, we have an oracle $\mathcal{O}_\pi$. Suppose that k classical queries are weakly useless but k quantum queries are not useless. Let us define the oracle $\mathcal{O}_i : |x_1\rangle \cdots |x_k\rangle |y_1\rangle \cdots |y_k\rangle \mapsto |x_1\rangle \cdots |x_k\rangle |\pi_{x_1, r_1}(y_1)\rangle \cdots |\pi_{x_k, r_k}(y_k)\rangle$ where π is selected from C_i according to μ (this is done independently for each query), $\boldsymbol{r}$ is distributed according to R^k and a fresh random seed $\boldsymbol{r}$ is used for every query to $\mathcal{O}_i$. Consider the problem of determining i where the oracle $\mathcal{O}_i$ is given with probability $\mu(\pi \in C_i)$.

Theorem 18. *Any number of classical queries to the oracle $\mathcal{O}_i$ is weakly useless for determining i.*

Proof. Clearly, a single query to $\mathcal{O}_i$ is equivalent to k queries to the original oracle which are weakly useless by assumption. We conclude that a single classical query to the new oracle is weakly useless. We now show that ℓ classical queries are weakly useless for any $\ell \geq 1$. Let $\boldsymbol{x}_j \in [N]^k$, $\boldsymbol{y}_j, \boldsymbol{z}_j \in [M]^k$ and let each $\boldsymbol{r}_j$ be sampled independently from R^k where $1 \leq j \leq \ell$. We must prove that

$$\Pr(i \mid \pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j, j = 1, \ldots, \ell) = \Pr(i) \tag{4}$$

where each π^j is sampled independently from C_i according to μ. This condition is equivalent to

$$\Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots, \ell \mid i) = \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j, j = 1, \ldots, \ell) \tag{5}$$

Note that by construction, $\Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots, \ell \mid i) = \prod_j \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j \mid i)$. By our assumption that k classical queries to the original oracle are weakly useless, we have that $\Pr(i \mid \pi_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j) = \Pr(i)$ or equivalently $\Pr(\pi_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j \mid i) = \Pr(\pi_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j)$. Therefore,

$$\Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = \boldsymbol{z}_j, j = 1, \ldots, \ell) = \sum_i \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots, \ell \mid i) \Pr(i) \tag{6}$$

$$= \sum_i \left(\prod_j \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j \mid i) \right) \Pr(i) \tag{7}$$

$$= \sum_i \left(\prod_j \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j) \right) \Pr(i) \tag{8}$$

$$= \prod_j \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j) \tag{9}$$

$$= \prod_j \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j \mid i) \tag{10}$$

$$= \Pr(\pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots, \ell \mid i) \tag{11}$$

which is the desired result. $\qquad \square$

We conclude that no matter how many classical queries are made to $\mathcal{O}_i$, no information is obtained about i. On the other hand, we have the following result:

Theorem 19. *A single quantum query to $\mathcal{O}_i$ is* not *useless for determining i.*

Proof. One can use a single quantum query to $\mathcal{O}_i$ to construct the state ρ_π^k as described in Theorem 15. Applying Theorem 15, this state may be used to guess i with higher probability than random guessing since k quantum queries are not useless. $\square$

Thus, we have constructed an infinity-vs-one separation in unbounded-error classical and quantum query complexities from an arbitrary initial separation. One can also construct an infinity-vs-one separation in the bounded-error setting from an arbitrary separation in the unbounded setting; the construction is straightforward and we defer the details to Appendix B.

Acknowledgments

A   Alternate proofs of uselessness

In this appendix, we present alternate proofs of various uselessness theorems. These proofs do not rely on the idea of encoding oracles into states, but instead give direct arguments, so they are more self-contained, although also longer. First we prove that pairwise classical uselessness implies quantum uselessness.

Proof. The proof is an extension of the technique used by Meyer and Pommersheim. Suppose that $2k$ classical queries are pairwise useless. Consider an oracle π that acts by $\mathcal{O}_\pi^i : |x, y, z\rangle \mapsto |x, \pi_{x,r_i}, z\rangle$ for the ith query. Note that, as before, the r_i variables may obey an arbitrary joint distribution so different queries are not necessarily independent. Consider an arbitrary k-query quantum algorithm with initial state ρ_0 and POVM $\{M_s\}$. For the ith query, the algorithm queries the oracle and then applies an arbitrary unitary transformation U_i. This yields the final state

$$\rho_\pi = U_k \mathcal{O}_\pi^k \ldots U_1 \mathcal{O}_\pi^1 \rho_0 \mathcal{O}_\pi^{1\dagger} U_1^\dagger \ldots \mathcal{O}_\pi^{k\dagger} U_k^\dagger \tag{12}$$

Let us fix the random seed used for the ith query as r_i. The final state is then

$$\rho_{\pi,\boldsymbol{r}} = U_k P_{r_k} \ldots U_1 P_{r_1} \rho_0 P_{r_1}^\dagger U_1^\dagger \ldots P_{r_k}^\dagger U_k^\dagger \tag{13}$$

where P_{r_i} denotes the permutative action $|x, y, z\rangle \mapsto |x, \pi_{x,r_i}(y), z\rangle$ of the oracle when the random seed is fixed to r_i. Let A be a matrix, $L = (x, y, z)$ and $L' = (x', y', z')$. Then

$$\left(P_{r_i} A P_{r_i}^\dagger \right)_{L,L'} = \langle x, \pi_{x,r_i}^{-1}(y), z | A | x', \pi_{x',r_i}^{-1}(y'), z' \rangle \tag{14}$$

$$= A_{\pi_{\cdot,r_i}(L), \pi_{\cdot,r_i}(L')} \tag{15}$$

where $\pi_{\cdot,r_i}(L) = (x, \pi_{x,r_i}^{-1}(y), z)$. Then the state after the $i + 1^{\text{th}}$ query (for the fixed values $\boldsymbol{r}$ of the seeds) is

$$\rho_{i+1,\boldsymbol{r}} = U_{i+1} P_{r_{i+1}} \rho_{i,\boldsymbol{r}} P_{r_{i+1}}^\dagger U_{i+1}^\dagger \tag{16}$$

so that the matrix elements are

$$(\rho_{i+1,\boldsymbol{r}})_{L,L'} = \sum_{K,K'} (U_{i+1})_{L,K} (\rho_{i,\boldsymbol{r}})_{\pi_{\cdot,r_{i+1}}(K), \pi_{\cdot,r_{i+1}}(K')} (U_{i+1}^\dagger)_{K',L'} \tag{17}$$

This value is a function of L, L', $\pi_{\cdot,r_{i+1}}(K)$ and $\pi_{\cdot,r_{i+1}}(K')$. Therefore, the final state $\rho_{\pi,\boldsymbol{r}} = \rho_{k,\boldsymbol{r}}$ may be written as

$$\rho_{\pi,\boldsymbol{r}} = \sum_I Q_I(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}), \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}')) \tag{18}$$

where $I = (L_1, \ldots, L_k, L'_1 \ldots, L'_k)$. Let $\mathbb{E}_{\pi|\pi\in C_j}$ denote the expectation over π according to the distribution $\Pr(\pi \mid \pi \in C_j)$. Then for any j,

$$\mathbb{E}_{\pi|\pi\in C_j} \rho_{\pi,\boldsymbol{r}} = \sum_I \mathbb{E}_{\pi|\pi\in C_j} Q_I(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}), \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}')) \tag{19}$$

$$= \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} \mathbb{E}_{\pi|\pi\in C_j} Q_I(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}), \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}'))[\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}']) \tag{20}$$

where $\boldsymbol{w} = (w_1, \ldots, w_k)$ and $\boldsymbol{w}' = (w'_1, \ldots, w'_k)$

$$= \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}') \mathbb{E}_{\pi|\pi\in C_j} [\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}']) \tag{21}$$

$$= \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}') \Pr(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}' \mid \pi \in C_j) \tag{22}$$

$$\tag{23}$$

Taking the expectation over the random seeds $\boldsymbol{r}$,

$$\mathbb{E}_{\pi|\pi\in C_j} \mathbb{E}_{\boldsymbol{r}} \rho_{\pi,\boldsymbol{r}} = \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}') \mathbb{E}_{\boldsymbol{r}} \Pr(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}' \mid \pi \in C_j) \tag{24}$$

$$= \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}') \Pr(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}' \mid \pi \in C_j) \tag{25}$$

$$= \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}') \Pr(\pi \in C_j \mid \pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}') \tag{26}$$

$$\cdot \frac{\Pr(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}')}{\Pr(\pi \in C_j)} \tag{27}$$

$$= \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}') \Pr(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}') \tag{28}$$

by pairwise classical uselessness

$$= \mathbb{E}_\pi \mathbb{E}_{\boldsymbol{r}} \sum_I \sum_{\boldsymbol{w},\boldsymbol{w}'} Q_I(\boldsymbol{w}, \boldsymbol{w}')[\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}) = \boldsymbol{w}, \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}') = \boldsymbol{w}'] \tag{29}$$

$$= \mathbb{E}_\pi \mathbb{E}_{\boldsymbol{r}} \sum_I Q_I(\pi_{\boldsymbol{x},\boldsymbol{r}}(\boldsymbol{y}), \pi_{\boldsymbol{x}',\boldsymbol{r}}(\boldsymbol{y}')) \tag{30}$$

$$= \mathbb{E}_\pi \mathbb{E}_{\boldsymbol{r}} \rho_{\pi,\boldsymbol{r}} \tag{31}$$

$$\tag{32}$$

Defining $\rho_\pi = \mathbb{E}_{\boldsymbol{r}} \rho_{\pi,\boldsymbol{r}}$, this may be written as

$$\mathbb{E}_{\pi|\pi\in C_j} \rho_\pi = \mathbb{E}_\pi \rho_\pi \tag{33}$$

Note that for a random $\pi \in C$, the state after running the algorithm is $\mathbb{E}_\pi \rho_\pi$ and for a random $\pi \in C_j$ the state is $\mathbb{E}_{\pi|\pi\in C_j} \rho_\pi$. Now, consider the probability that $\pi \in C_j$ given the measurement outcome s. We have

12

$$\Pr(\pi \in C_j \mid s) = \frac{\Pr(s \mid \pi \in C_j) \Pr(\pi \in C_j)}{\Pr(s)} \tag{34}$$

$$= \frac{\operatorname{tr} M_s \, \mathbb{E}_{\pi \mid \pi \in C_j} \, \rho_f}{\operatorname{tr} M_s \, \mathbb{E}_{\pi} \, \rho_\pi} \Pr(\pi \in C_j) \tag{35}$$

$$= \Pr(\pi \in C_j) \tag{36}$$

as claimed. $\qquad\square$

Next, we prove that quantum uselessness implies classical uselessness, but in the special case of standard oracles that act via XOR but with internal randomness. Specifically, consider an oracle that acts by $\mathcal{O}_f^i : |x, y, z\rangle \mapsto |x, y \oplus f(x, r_i), z\rangle$ for the ith query. As before, we allow the r_i variables to be drawn from an arbitrary joint distribution.

Proof. Suppose that k quantum queries are useless. This means that for any POVM $\{M_s\}$ and quantum algorithm run on any initial state ρ_0, $\Pr(f \in C_j \mid s) = \Pr(f \in C_j)$ for all j. Since $\Pr(f \in C_j \mid s) = \frac{\Pr(s \mid f \in C_j) \Pr(f \in C_j)}{\Pr(s)}$, this implies that

$$\Pr(s \mid f \in C_j) = \Pr(s) \tag{37}$$

for all j. Let us choose the initial state

$$\rho_0 = \left(\frac{1}{N} \sum_{x, x'} |x\rangle \langle x'| \otimes |0\rangle \langle 0| \right)^{\otimes k} \tag{38}$$

and the algorithm defined by the unitary operator $\bigotimes_{i=1}^k \mathcal{O}_f^i$. The result of running the algorithm assuming a particular function f and fixed seeds $\boldsymbol{r}$ is then

$$\rho_{f,\boldsymbol{r}} = \left(\bigotimes_{i=1}^k \mathcal{O}_f^i \right) \rho_0 \left(\bigotimes_{i=1}^k \mathcal{O}_f^i \right)^\dagger \tag{39}$$

$$= \frac{1}{N^k} \bigotimes_{i=1}^k \sum_{x, x'} |x, f(x, r_i)\rangle \langle x', f(x', r_i)| \tag{40}$$

For a particular function f, the state after running the algorithm is

$$\rho_f = \mathbb{E}_{\boldsymbol{r}} \, \rho_{f,\boldsymbol{r}} \tag{41}$$

$$= \frac{1}{N^k} \mathbb{E}_{\boldsymbol{r}} \bigotimes_{i=1}^k \sum_{x, x'} |x, f(x, r_i)\rangle \langle x', f(x', r_i)| \tag{42}$$

Now

$$\Pr(s) = \mathop{\mathbb{E}}_{f} \Pr(s \mid f) \tag{43}$$

$$= \operatorname{tr} M_s \mathop{\mathbb{E}}_{f} \rho_f \tag{44}$$

$$= \operatorname{tr} M_s \rho_C \tag{45}$$

Similarly,

$$\Pr(s \mid f \in C_j) = \mathop{\mathbb{E}}_{f \mid f \in C_j} \Pr(s \mid f) \tag{46}$$

$$= \operatorname{tr} M_s \mathop{\mathbb{E}}_{f \mid f \in C_j} \rho_f \tag{47}$$

$$= \operatorname{tr} M_s \rho_{C_j} \tag{48}$$

Since $\Pr(s \mid f \in C_j) = \Pr(f \in C_j)$, this implies that

$$\operatorname{tr} M_s (\rho_{C_j} - \rho_C) = 0 \tag{49}$$

for *all* POVMs $\{M_s\}$ which means that

$$\rho_{C_j} = \rho_C \tag{50}$$

$$\frac{1}{N^k} \mathop{\mathbb{E}}_{\boldsymbol{r}} \mathop{\mathbb{E}}_{f \mid f \in C_j} \bigotimes_{i=1}^{k} \sum_{x,x'} |x, f(x, r_i)\rangle \langle x', f(x', r_i)| = \frac{1}{N^k} \mathop{\mathbb{E}}_{\boldsymbol{r}} \mathop{\mathbb{E}}_{f} \bigotimes_{i=1}^{k} \sum_{x,x'} |x, f(x, r_i)\rangle \langle x', f(x', r_i)| \tag{51}$$

Equating the $((x_1, y_1, \ldots, x_k, y_k), (x'_1, y'_1, \ldots, x'_k, y'_k))$ elements of these matrices, we have that

$$\mathop{\mathbb{E}}_{\boldsymbol{r}} \mathop{\mathbb{E}}_{f \mid f \in C_j} [f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}'] = \mathop{\mathbb{E}}_{\boldsymbol{r}} \mathop{\mathbb{E}}_{f} [f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}'] \tag{52}$$

$$\mathop{\mathbb{E}}_{\boldsymbol{r}} \Pr(f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}' \mid f \in C_j) = \mathop{\mathbb{E}}_{\boldsymbol{r}} \Pr(f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}') \tag{53}$$

$$\Pr(f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}' \mid f \in C_j) = \Pr(f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}') \tag{54}$$

Applying Bayes' rule, we have

$$\Pr(f \in C_j \mid f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}') = \frac{\Pr(f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}' \mid f \in C_j) \Pr(f \in C_j)}{\Pr(f(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{y}, f(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{y}')} \tag{55}$$

$$= \Pr(f \in C_j) \tag{56}$$

which is precisely the definition of pairwise classical uselessness in the case of oracles that act by XOR. $\qquad\square$

Combining this with Theorem 9, we have the following result

Corollary 20. *For any oracle problem* (C, μ) *in the standard model with internal randomness,* k *quantum queries are useless if and only if* $2k$ *classical queries are pairwise useless*

Since pairwise classical uselessness is equivalent to classical uselessness when f is deterministic, we have the following corollary.

Corollary 21. *If* k *quantum queries are useless for an oracle problem* (C, μ) *in the standard model, then* $2k$ *classical queries are useless.*

B   Bounded-error infinity-vs-one separations

We now show how to obtain an infinity-vs-one separation in the bounded-error regime from an arbitrary separation between the classical and quantum uselessness. Consider the oracle $\mathcal{O}_i$ as defined above. By Theorem 19, there exists a single-query quantum algorithm A, a POVM $\{M_s\}$ and an i' such that for some s, $\Pr(i = i' \mid s) > \Pr(i = i')$. Equivalently,

$$\Pr(s \mid i = i') > \Pr(s) \tag{57}$$

$$\Pr(s \mid i = i')(1 - \Pr(i = i')) > \Pr(s \mid i \neq i')\Pr(i \neq i') \tag{58}$$

$$\Pr(s \mid i = i') > \Pr(s \mid i \neq i') \tag{59}$$

$$\Pr(s \mid i = i') = \Pr(s \mid i \neq i') + \epsilon \tag{60}$$

for some $\epsilon > 0$. Consider the problem of deciding if $i = i'$ by querying $\mathcal{O}_i$. By running A some large number of times T and using majority voting and Chernoff bounds, we may decide if $i = i'$ with bounded error. Although T may be quite large, the gap is large since it is a separation between an infinite number of classical queries and a finite number of classical queries.

Corollary 22. *The bounded-error quantum query complexity of deciding if $i = i'$ using $\mathcal{O}_i$ is finite.*

By Theorem 18, $\Pr(i \mid \pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots) = \Pr(i)$ for all $\ell \geq 1$ and $\boldsymbol{x}_j \in [N]^k$, $\boldsymbol{y}_j, \boldsymbol{z}_j \in [M]^k$. Thus, $\Pr(i = i' \mid \pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots) = \Pr(i = i')$ and $\Pr(i \neq i' \mid \pi^j_{\boldsymbol{x}_j, \boldsymbol{r}_j}(\boldsymbol{y}_j) = z_j, j = 1, \ldots) = \Pr(i \neq i')$ so ℓ queries are weakly useless for deciding if $i = i'$.

Corollary 23. *Any number of classical queries to the oracle $\mathcal{O}_i$ is weakly useless for deciding if $i = i'$; thus no classical algorithm can decide if $i = i'$ with unbounded error no matter how many queries are made.*

We can construct a new oracle $\mathcal{O}'_i$ that simulates T queries to $\mathcal{O}_i$ using an independent random seed for each query. From this we obtain the following.

Corollary 24. *The bounded-error quantum query complexity of deciding if $i = i'$ using $\mathcal{O}'_i$ is 1.*

Corollary 25. *Any number of classical queries to the oracle $\mathcal{O}'_i$ is weakly useless for deciding if $i = i'$; thus no classical algorithm can decide if $i = i'$ with unbounded error no matter how many queries are made.*

Thus, we have constructed an infinity-vs-one separation between the bounded-error quantum query complexity and the unbounded-error classical query complexity from an arbitrary initial separation. This comes at the price of large inputs for the constructed oracle.

C   Relation between uselessness and unbounded query complexity

In this section, we define *binary* oracle problems to be those where our goal is to output a single bit, or equivalently, where C is partitioned into only two sets C_0, C_1, and our goal is to determine whether $\pi \in C_0$ or $\pi \in C_1$.

Proposition 26. *Let C be a binary oracle problem. Then the unbounded quantum (resp. classical) query complexity of C is $> k$ if and only if there exists a distribution μ with $\mu(C_0) = \mu(C_1) = 1/2$ such that k quantum (resp. classical) queries are useless for (C, μ).*

Equivalently we could demand that $0 < \mu(C_0) < 1$ because reweighting 0-inputs and 1-inputs does not affect the uselessness properties of a distribution. (The same does *not* hold for changing the probabilities within the class of 0-inputs or 1-inputs.) However, we need to avoid the trivial case in which a distribution is useless because the answer is already known perfectly from the prior distribution μ.

Proof. The "if" direction is easy. If such a μ exists, then by the definition of uselessness, no algorithm can achieve success probability $> 1/2$ with $\leq k$ queries.

For the converse, we use Yao's minimax principle, which states that there exists a distribution μ for which no k-query algorithm can achieve success probability $> 1/2$. Since it is always possible to achieve success probaiblity $\max(\mu(C^{-1}(0)), \mu(C^{-1}(1)))$ by guessing, we must also have $\mu(C^{-1}(0)) = \mu(C^{-1}(1)) = 1/2$. $\qquad\square$

A natural generalization of unbounded-error query complexity to non-binary problems would be to define success as guessing the right answer with probability $> \max_j \mu(C_j)$. In this case, uselessness is now a strictly stronger statement whenever μ is such that $\mu(C_j)$ is not the same for each j. To see this, let ν be the distribution over π obtained after making some number of queries. Uselessness states that $\mu(C_j) = \nu(C_j)$ for each j, whereas unbounded-error query complexity depends only on whether $\max_j \mu(C_j) = \max_j \nu(C_j)$.

References

[1] S. Aaronson. BQP and the polynomial hierarchy. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 141–150, 2010, 0910.4698.

[2] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87:167902, 2001.

[3] J. N. de Beaudrap, R. Cleve, and J. Watrous. Sharp quantum vs. classical query separations. *ArXiv Quantum Physics e-prints*, 2001, quant-ph/0011065.

[4] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. Limit on the speed of quantum computation in determining parity. *Phys. Rev. Lett.*, 81(24):5442–5444, 1998, quant-ph/9802045.

[5] D. Gottesman and I. Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402:390–393, 1999, quant-ph/9908010.

[6] D. A. Meyer and J. Pommersheim. Single query learning from abelian and non-abelian Hamming distance oracles. *ArXiv Quantum Physics e-prints*, 2009, 0912.0583.

[7] D. A. Meyer and J. Pommersheim. On the uselessness of quantum queries. *Theoretical Computer Science*, 412(51):7068–7074, 2011, 1004.1434.

[8] A. Montanaro, H. Nishimura, and R. Raymond. Unbounded-error quantum query complexity. In *Algorithms and Computation*, volume 5369 of *Lecture Notes in Computer Science*, pages 919–930. 2008.

[9] O. Regev and L. Schiff. Impossibility of a quantum speed-up with a faulty oracle. In *Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part I*, pages 773–781, 2008, 1202.1027.

[10] D. R. Simon. On the power of quantum computation. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 116–123, 1994.