A Brief Introduction to Quantum Query Complexity

Yassine Hamoudi

Université de Bordeaux, CNRS, LaBRI, France.
ys.hamoudi@gmail.com

Last update: April 11, 2025

Abstract

Quantum query complexity is a fundamental model for analyzing the computational power of quantum algorithms. It has played a key role in characterizing quantum speedups, from early breakthroughs such as Grover's and Simon's algorithms to more recent developments in quantum cryptography and complexity theory. This document provides a structured introduction to quantum query lower bounds, focusing on four major techniques: the hybrid method, the polynomial method, the recording method, and the adversary method. Each method is developed from first principles and illustrated through canonical problems. Additionally, the document discusses how the adversary method can be used to derive upper bounds, highlighting its dual role in quantum query complexity. The goal of this exposition is to offer a self-contained exposition accessible to readers with a basic background in quantum computing, while also serving as an entry point for researchers interested in the study of quantum lower bounds.

1 Introduction

Query complexity (also known as *decision tree complexity*) is the study of algorithms that can access their input solely through an abstract operation, called a *query*. Unlike other models of computation (such as circuit complexity or Turing machines), query complexity has been particularly effective in pinpointing the hardness of various computational problems. Notably, it has played a crucial role in understanding the power and limitations of quantum computing, from the early-days quantum algorithms [DJ92; BV97; Sim97; Gro97] to modern applications in cryptography (e.g., quantum random oracle model). In this respect, it is one of the few computational models in which *exponential* quantum speedups have been rigorously established (e.g., Simon's problem [Sim97], Welded Tree [CCD+03], Forrelation [BS21; SSW23], Yamakawa-Zhandry's problem [YZ24]).

The great success of query complexity lies, in part, in the development of *lower-bound methods* that relate the minimum number of queries required to solve a problem to its combinatorial properties. Together with algorithmic upper bounds, these methods can distinguish between problems of varying complexity – such as constant, logarithmic, cubic root, square root, etc. – thereby establishing an *unconditional* hierarchy of *fine-grained* complexity classes. However, only a handful of such methods extend to the quantum setting. Indeed, many intuitive properties of classical queries do not carry over to *quantum queries*, which can access the input in superposition. A foundational step in understanding the limits of quantum queries was taken in the seminal work of Bennett, Bernstein, Brassard and Vazirani [BBBV97] through the so-called "hybrid method". Since then, additional methods have emerged, revealing deep connections between quantum query complexity and Boolean function analysis, matrix analysis, semidefinite programming, etc.

The goal of this document is to present four of the most prominent and successful methods for establishing quantum query lower bounds, along with some of their basic applications. The

properties sought in lower-bound methods are of various types. Of course, they are expected to reduce the complexity gap with the best-known algorithms, ultimately reaching optimality. However, this can manifest in different ways. Research in query complexity has been driven by challenges such as: problems with *extreme output conditions* (zero error, exponentially small success probability), query models with *physical constraints* (noisy queries, bounded space, short coherence time), query models with *advice* (QMA query complexity, cryptographic settings with auxiliary input), *strong direct product theorems* (complexity of solving multiple instances of the same problem), *composition theorems* (relating the complexity of a problem to the complexities of its components), *lifting theorems* (transferring query complexity lower bounds to communication complexity), complexity of *"inherently" quantum problems* (state conversion), etc. This document primarily focuses on basic applications and does not delve into these advanced aspects, though some are supported by the methods presented in the following sections.

Organization of the document. The study of quantum query complexity requires minimal prior knowledge of quantum computing. It is nevertheless assumed that the reader is familiar with the basics of quantum computing (such as Dirac notation and the circuit model), as found in any standard textbook (e.g., [NC11]). In Section 1.1, we provide a self-contained description of the computational models used in this document. Section 1.2 outlines the main problems that we will use later to illustrate the lower-bound methods.

Each of the four subsequent sections is dedicated to a different lower-bound method: hybrid (Section 2), polynomial (Section 3), recording (Section 4) and adversary (Section 5). The hybrid, recording and adversary methods belong to the same family of techniques, and it is recommended to read them in order.

The final section (Section 6) explores a striking property of the adversary method: its ability to also provide query upper bounds (i.e., quantum algorithms) via the dual of a specific semidefinite program.

Going further. Complementary introductions to quantum query complexity can be found in the survey by Buhrman and de Wolf [BW02], and in the lecture notes of de Wolf [Wol19], Childs [Chi17] and Ben-David [Ben20]. A more advanced treatment of certain lower-bound techniques is provided in the Ph.D. dissertations of Špalek [Špa06], Belovs [Bel14] and Rosmanis [Ros14]. The document will also provide pointers to the scientific literature for readers who wish to explore further.

Acknowledgments. The content of this document was taught at the IAS/PCMI 2023 summer school over the course of five lectures and four problem sessions (nine hours in total). The author wants to thank the organizers of the summer school for the invitation and the great atmosphere throughout the event. The author is also very grateful to Angelos Pelecanos for his work as a teaching assistant.

1.1 How to Model a Quantum Query Algorithm

There are many variants of query complexity, depending on the computational power given to the algorithms, the assumptions made about the input and the conditions required for the output. In this document, we primarily focus on the simple – yet very informative – setup of computing Boolean functions with bounded-error algorithms, as defined next.

(Boolean input alphabet) The input to a query problem is an n-bit string $x = x_1 \dots x_n \in \{0,1\}^n$, where each coordinate x_i is called the *query value* on *query index* i.

(Decision problems) A query problem associates with each input x a unique solution $f(x)$ specified by a Boolean function $f : \{0,1\}^n \to \{0,1\}$.

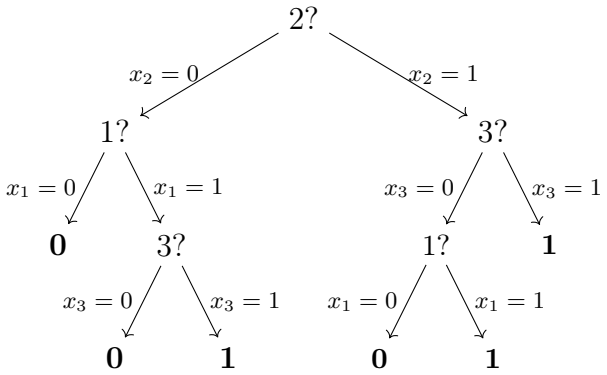

Figure 1: A deterministic algorithm with query complexity 3 that computes the MAJORITY function over three bits (MAJORITY$(x_1x_2x_3) = 1$ when two or three bits are equal to one).

(Worst-case output condition) An algorithm is said to *compute* f if it correctly outputs $f(x)$ with probability at least $2/3$ for all inputs x.

The complexity of an algorithm will be measured by the number of queries made to the input x. For a classical algorithm, a *query* consists of revealing the value of a coordinate x_i for an index i chosen by the algorithm. For a quantum algorithm, the definition of a query is more subtle and will be given later. The query complexity of a function f is the smallest number of queries needed among all algorithms computing f.

We briefly define the two models of classical query complexity – *deterministic* and *randomized* – that are most studied in the literature. A deterministic query algorithm can be conveniently represented as a Boolean decision tree, where each node corresponds to a query and each leaf to an output (see Figure 1) – hence the alternative name of *decision tree complexity*. Since the output is deterministic, the condition for such an algorithm to compute f is to always output $f(x)$. A randomized query algorithm also has the ability to make random choices, which is equivalent to having a distribution over decision trees. In that case, the output condition allows the algorithm to output the wrong value with probability at most $1/3$.

Definition 1 (Deterministic and randomized query complexities)**.** A *deterministic query algorithm* with query complexity T is a rooted, ordered binary tree of depth T, where each internal node is labeled with an index $i \in \{1, \ldots, n\}$, and each leaf is labeled with a Boolean value. The *output* of the algorithm on an input $x \in \{0,1\}^n$ is the value of the leaf obtained by starting at the root and, recursively, moving to the left child when $x_i = 0$ or to the right child when $x_i = 1$, where i is the label of the current node.

A *randomized query algorithm* with query complexity T is a distribution over deterministic query algorithms with query complexity T. The *output* of the algorithm on an input x is the random value obtained by sampling a deterministic query algorithm according to the distribution and evaluating it on x.

The *deterministic (resp. randomized) query complexity* $D(f)$ (resp. $R(f)$) of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is the smallest integer T such that there exists a deterministic (resp. randomized) query algorithm computing f with query complexity T.

It is important to remember that non-query operations are *not counted* toward the complexity of an algorithm. Hence, a function can have low query complexity but require a large effective computation time. However, for most problems of interest, it is observed that the query complexity is a good proxy for their actual difficulty. The query complexity is nevertheless limited to be at most n in the above models, since it always suffices to retrieve the entire input with n queries and apply f on it (this computation can be represented by the perfect binary tree of depth n).

Fact 2. *For all functions $f : \{0,1\}^n \to \{0,1\}$, we have $R(f) \leq D(f) \leq n$.*

The first question to address in defining a model of quantum query complexity is: *how is the algorithm given access to the input?* If we kept the same query operator as before (by letting the algorithm observe a single coordinate at a time), then the query complexity would be *identical* to that in the randomized model. Indeed, the non-query operations could simply be simulated by a classical algorithm (even if they are quantum), since their cost is not part of the query complexity anyway. Hence, the query operator must be "truly quantum" to make the model interesting. This is achieved by the following quantum oracles, which have the ability to query multiple coordinates in superposition.

Definition 3 (Quantum query operators)**.** Consider the Hilbert space of dimension $2n$ spanned by the vectors $|i,b\rangle$ where $i \in \{1,\ldots,n\}$ and $b \in \{0,1\}$. Given an input $x \in \{0,1\}^n$, we define the two following quantum unitary operators $\mathcal{O}_x$ and $\mathcal{O}_x^{\pm}$:

Binary oracle	*Phase oracle*
$\mathcal{O}_x\|i,b\rangle = \|i,b \oplus x_i\rangle$	$\mathcal{O}_x^{\pm}\|i,b\rangle = (-1)^{bx_i}\|i,b\rangle$

where $\oplus$ is the XOR operation. The register holding i is called the *index register*, and the register holding b is the *value register*. We say that an algorithm makes a *quantum query to x* whenever it applies $\mathcal{O}_x$ or $\mathcal{O}_x^{\pm}$.

The first oracle $\mathcal{O}_x$ is the most natural extension of the classical query operator. It writes down the value of the query in an extra register in a reversible manner, and it can act on superpositions: $\mathcal{O}_x\big(\sum_{i,b} \alpha_{i,b}|i,b\rangle\big) = \sum_{i,b} \alpha_{i,b}|i,b \oplus x_i\rangle$ for any amplitudes $\alpha_{i,b}$. The second oracle $\mathcal{O}_x^{\pm}$ encodes the query result into the phase rather than into a quantum register, which is sometimes more convenient for use in applications. This modification makes no change to the query complexity since the two oracles can simulate each other efficiently, by conjugation with a Hadamard gate.

Fact 4 (Equivalence between binary and phase oracles)**.** *We have the following equality,*

$$\mathcal{O}_x^{\pm} = (\mathrm{Id} \otimes H)\mathcal{O}_x(\mathrm{Id} \otimes H)$$

where H is the Hadamard gate. In particular, any quantum algorithm that makes T quantum queries to x using $\mathcal{O}_x$ can be perfectly simulated by an algorithm that makes the same number T of queries but using $\mathcal{O}_x^{\pm}$ (and vice-versa).

We now incorporate the quantum query operator into the definition of quantum algorithms. Since query complexity concerns only the number of query operations, we can drastically simplify the model: all operations performed between two queries (including intermediate measurements – by the deferred measurement principle) can be grouped together into a single unitary.

In general, an algorithm may need some extra workspace beyond the $\lceil \log n \rceil + 1$ qubits used for the query register $|i,b\rangle$. However, the size (or existence) of this extra memory plays no role in the lower-bound methods that will be presented in this document (in fact, it is a major research challenge to find query lower-bound methods that are also sensitive to space constraints). Hence, for the ease of notation, we will only be considering *memoryless* algorithms that need no extra registers beyond $|i,b\rangle$ (an example of such an algorithm is Grover's search).

We summarize the quantum query model in the next definition. The reader can also refer to the picture in Figure 2, which illustrates a quantum query algorithm in the quantum circuit formalism.

Definition 5 (Canonical form of a memoryless quantum query algorithm)**.** A memoryless *quantum query algorithm* with query complexity T is a sequence $U_0,\ldots,U_T$ of unitary operators acting on the Hilbert space spanned by the vectors $|i,b\rangle$ where $i \in \{1,\ldots,n\}$ and $b \in \{0,1\}$.

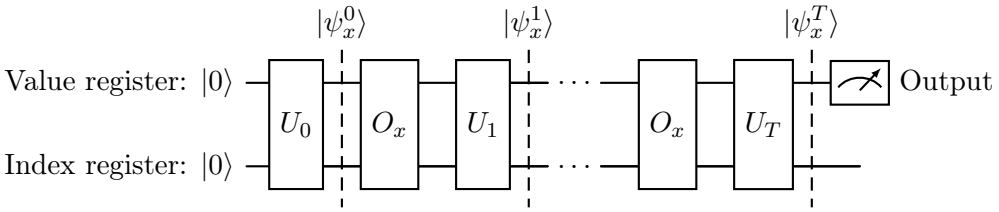

Figure 2: Canonical form of a (memoryless) quantum query algorithm.

The *intermediate state* $|\psi_x^t\rangle$ of the algorithm after $t \in \{0, \ldots, T\}$ queries on the input x is defined as,

$$|\psi_x^t\rangle = U_t \mathcal{O}_x U_{t-1} \mathcal{O}_x \ldots U_1 \mathcal{O}_x U_0 |0, 0\rangle.$$

The *final state* of the algorithm is $|\psi_x^T\rangle$ and its *output* is the random bit obtained by measuring the value register of that state in the standard basis. Hence, the output is 0 with probability $\|(\mathrm{Id} \otimes |0\rangle\langle 0|)|\psi_x^T\rangle\|^2$ and 1 with probability $\|(\mathrm{Id} \otimes |1\rangle\langle 1|)|\psi_x^T\rangle\|^2 = 1 - \|(\mathrm{Id} \otimes |0\rangle\langle 0|)|\psi_x^T\rangle\|^2$. The *success probability* p_{succ} of the algorithm in computing a function f is the smallest probability with which the algorithm outputs the correct value on any input: $p_{\mathrm{succ}} = \min_{x \in \{0,1\}^n} \|(\mathrm{Id} \otimes |f(x)\rangle\langle f(x)|)|\psi_x^T\rangle\|^2$. The algorithm is said to *compute* f if $p_{\mathrm{succ}} \geq 2/3$.

The most general form of query algorithms allows the unitaries U_t's to act on a larger Hilbert space spanned by $|i, b\rangle \otimes |w\rangle$, where $w \in \{1, \ldots, d\}$ represents an extra memory register of arbitrary dimension d. The memoryless assumption amounts to restricting $d = 1$. It is a simple exercise to verify that all the lower bound methods presented in this document extend readily to any value of d.

Finally, the quantum query complexity of a Boolean function is defined as the smallest query complexity among the algorithms that compute that function.

Definition 6 (Quantum query complexity)**.** The *quantum query complexity* $Q(f)$ of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is the smallest integer T such that there exists a quantum algorithm computing f with query complexity T.

It is always possible to simulate a classical query on an index i by querying $\mathcal{O}_x|i, 0\rangle = |i, x_i\rangle$ and measuring the value register. Hence, the quantum query complexity is always less than or equal to the randomized complexity (though the simulation may require considering non-memoryless quantum algorithms to carry it out).

Fact 7. *For all functions $f : \{0,1\}^n \to \{0,1\}$, we have $Q(f) \leq R(f)$.*

We end this section by listing a few basic properties of the operator norm, which are used extensively in the analysis of quantum query lower bounds.

Lemma 8 (Properties of the operator norm)**.** *Let $\|.\|$ denote both the Euclidean vector norm $\||\psi\rangle\| = \sqrt{\sum_x |\psi_x|^2}$, and the induced matrix norm (spectral norm) $\|A\| = \max_{\psi:\||\psi\rangle\|=1} \|A|\psi\rangle\|$ over the same Hilbert space. Then, the following properties hold,*

- *(Triangle inequality)* $\||\psi_1\rangle + |\psi_2\rangle\| \leq \||\psi_1\rangle\| + \||\psi_2\rangle\|$,
- *(Cauchy–Schwarz inequality)* $|\langle\psi_1 \,|\, \psi_2\rangle| \leq \||\psi_1\rangle\| \cdot \||\psi_2\rangle\|$,
- *(Unitary invariance)* $\|U|\psi\rangle\| = \||\psi\rangle\|$,
- *(Submultiplicativity)* $\|A|\psi\rangle\| \leq \|A\| \cdot \||\psi\rangle\|$,

for any vectors $|\psi\rangle, |\psi_1\rangle, |\psi_2\rangle$, unitary U and matrix A.

1.2 Functions of interest

We list a few examples of functions that will serve as applications for the lower-bound methods later in the document. A problem is well suited for query complexity when its difficulty resides in collecting sufficient information about its input (and not, for instance, into some pre- or post-processing operations). Of course, not all problems fall into this category. However, it is often the case that part of their solution involves solving a subproblem encountered in query complexity. A good example is Shor's factoring algorithm, whose core component is a quantum query algorithm for the Period-finding problem. In that case, quantum query complexity may still be helpful in understanding whether a subroutine is optimal or not.

OR and Search. The OR function returns one if and only if the input $x \in \{0,1\}^n$ contains at least one bit equal to one. This is arguably one of the most studied problem in quantum query complexity. The celebrated Grover's algorithm [Gro97] can compute it *quadratically* faster than any classical algorithm. The proof of its optimality will serve as a guiding application throughout this document. An extension of that problem – the Search problem – asks to locate an index i such that $x_i = 1$ when it exists. It turns out to be no more difficult than the decision problem.

Collision. The Collision problem asks to decide if the input contains two coordinates with the same value $x_i = x_j$ (or to locate the indices of such coordinates). In order to make sense of this, the input alphabet must be increased beyond the Boolean domain, which we describe in Section 4. The Collision problem has been instrumental in the development of new quantum lower-bound methods, and it is studied heavily for its applications in cryptography as well.

Parity and Majority. The Parity function returns one if and only if the input $x \in \{0,1\}^n$ has an odd number of bits equal to one. It is an example of a problem for which no significant quantum speedup is possible (we will see that the query complexity decreases by only a factor of two). Another example of such a function is Majority, which returns the Boolean value that appears most frequently in the input.

Connectivity. The Connectivity function views the input $x \in \{0,1\}^{\binom{n}{2}}$ as the adjacency matrix of an undirected graph over n vertices, and it returns one if and only if that graph is connected. We will show how lower-bound methods have also led to the development of new quantum algorithms for this problem.

AND-OR tree. The AND-OR tree function is a composed function acting on $n = m^2$ bits, which applies the OR function to the m consecutive blocks of m input bits, followed by the AND function on the results: $\mathrm{AND}(\mathrm{OR}(x_1, \ldots, x_m), \mathrm{OR}(x_{m+1}, \ldots, x_{2m}), \ldots, \mathrm{OR}(x_{n-m+1}, \ldots, x_n))$. The composition structure of this problem makes it possible to analyze its complexity in a very automated way, by simply multiplying the complexities of the OR and AND functions. This will result from the composition properties of the adversary method detailed in Sections 5 and 6.

Total vs. partial functions. The functions studied in this document share a common property: their input x can take *any* value in the Boolean hypercube $\{0,1\}^n$. This necessitates restricting ourselves to problems that can be represented by functions f whose domain is the entire set $\{0,1\}^n$. Such functions are called *total*, in contrast to *partial* functions, which are only defined on a proper subset of $\{0,1\}^n$ to which x is *promised* to belong. The lower-bound methods presented in this document can be adapted to handle partial functions as well. There is, however, a major difference between partial and total functions in terms of the achievable gaps between $Q(f)$, $R(f)$ and $D(f)$. While the gaps can be exponential for partial functions (e.g.,

Deutsch–Jozsa's problem [DJ92], Simon's problem [Sim97], Welded Tree [CCD+03]), they are at most polynomial for total functions. This comes from very generic lower bounds exploiting the absence of structure in the input to total functions. We state these results below for the sake of concreteness. Weaker versions of these statements will be discussed later in the document as well.

Proposition 9 (Relations between query complexities of total functions)**.** *For any total function $f : \{0,1\}^n \to \{0,1\}$, the gaps between its deterministic, randomized and quantum query complexities are at most,*

$$D(f) = O(R(f)^3) \text{ [Nis91]}, \; R(f) = O(Q(f)^4) \text{ [ABK+21]}.$$

Moreover, there exist some functions $f, g, h : \{0,1\}^n \to \{0,1\}$ that achieve the next gaps,

$$D(f) = \widetilde{\Omega}(R(f)^2) \text{ [ABB+17]}, \; D(g) = \widetilde{\Omega}(Q(g)^4) \text{ [ABB+17]}, \; R(h) = \widetilde{\Omega}(Q(h)^3) \text{ [BS21; SSW23]}.$$

These results indicate that, for total functions, the best possible speedup is at most *cubic* in the randomized vs. deterministic setting and at most *quartic* in the quantum vs. deterministic or randomized settings. While Grover's algorithm provides an example of a *quadratic* speedup for the quantum vs. randomized setting, there are recent examples of total functions [BS21; SSW23] exhibiting a *cubic* speedup. It is still an open question to find a *super-cubic* speedup for the quantum vs. randomized setting, or to show that $R(f) = O(Q(f)^3)$.

2 The Hybrid Method

The *hybrid method* is the first lower-bound technique we present in this document. It originates from the work of Bennett, Bernstein, Brassard and Vazirani [BBBV97] on the limits of quantum computers for solving NP problems. This foundational result precluded the existence of a super-quadratic speedup for the OR function. Later, Grover complemented this lower bound with the celebrated quantum search algorithm [Gro97], showing that the optimal speedup for OR is indeed quadratic.

The hybrid method gave rise to a broader family of lower-bound methods known as "adversarial". We will develop two other such methods later in the document (Sections 4 and 5).

2.1 Technique

The hybrid method consists of tracking the distance (or, equivalently, the angle) between the intermediate states of an arbitrary quantum query algorithm on two inputs x and y, that lead to different outcomes, $f(x) \neq f(y)$. Initially, this distance is zero, since the algorithm always starts in the same state regardless of the input. However, for the algorithm to be successful, the distance should progressively increase during the computation, approaching one so that the final states become distinguishable. The challenge lies in upper bounding how far each quantum query can separate the two states. Intuitively, the most important indices to query are those where the inputs differ, i.e., where $x_i \neq y_i$. The hybrid method precisely shows that this increase is governed by the *total amplitude* of such indices in the quantum query.

Theorem 10 (Hybrid method)**.** *Consider a quantum algorithm that computes a function $f : \{0,1\}^n \to \{0,1\}$ using T queries to its input. Let $|\psi_x^t\rangle$ denote the intermediate states of the algorithm after $t \in \{0, \ldots, T\}$ queries on the input $x \in \{0,1\}^n$. Then, for any two inputs $x, y \in \{0,1\}^n$, we have:*

- *(Initial condition)* $\||\psi_x^0\rangle - |\psi_y^0\rangle\| = 0$,

- *(Progress evolution)* $\||\psi_x^{t+1}\rangle - |\psi_y^{t+1}\rangle\| \leq \||\psi_x^t\rangle - |\psi_y^t\rangle\| + 2 \min_{z\in\{x,y\}} \big\|(\sum_{i:x_i\neq y_i} |i\rangle\langle i| \otimes \mathrm{Id})|\psi_z^t\rangle\big\|$,
- *(Final condition) If* $f(x) \neq f(y)$ *then* $\||\psi_x^T\rangle - |\psi_y^T\rangle\| \geq 1/3$.

Proof of the initial and final conditions. The initial condition is immediate, as the states are independent of the inputs: $|\psi_x^0\rangle = |\psi_y^0\rangle = U_0|0,0\rangle$. For the final condition, by applying the Cauchy–Schwarz inequality, we have:

$$\begin{aligned}
\||\psi_x^T\rangle - |\psi_y^T\rangle\|^2 &= 2\big(1 - \mathrm{Re}(\langle\psi_x^T \,|\, \psi_y^T\rangle)\big) \\
&= 2\Big(1 - \sum\nolimits_{b\in\{0,1\}} \mathrm{Re}(\langle\psi_x^T \,|\, (\mathrm{Id}\otimes|b\rangle\langle b|) \,|\, \psi_y^T\rangle)\Big) \\
&\geq 2\Big(1 - \sum\nolimits_{b\in\{0,1\}} \|(\mathrm{Id}\otimes|b\rangle\langle b|)|\psi_x^T\rangle\| \cdot \|(\mathrm{Id}\otimes|b\rangle\langle b|)|\psi_y^T\rangle\|\Big).
\end{aligned}$$

Suppose, without loss of generality, that $f(x) = 0$ and $f(x) = 1$. Then, the correctness of the algorithm implies $\|(\mathrm{Id}\otimes|0\rangle\langle 0|)|\psi_x^T\rangle\|^2 \geq 2/3$ and $\|(\mathrm{Id}\otimes|1\rangle\langle 1|)|\psi_y^T\rangle\|^2 \geq 2/3$. Hence, $\||\psi_x^T\rangle - |\psi_y^T\rangle\|^2 \geq 2(1 - 2\sqrt{2}/3) \geq 1/9$. □

Proof of the progress evolution. The values of x and y are interchangeable in the proof, hence it is sufficient to consider the case where the minimum is achieved by $z = x$ in the progress evolution. By definition, the states of the algorithm at two consecutive time steps satisfy $|\psi_x^{t+1}\rangle = U_{t+1}\mathcal{O}_x|\psi_x^t\rangle$ and $|\psi_y^{t+1}\rangle = U_{t+1}\mathcal{O}_y|\psi_y^t\rangle$. We obtain that,

$$\begin{aligned}
\||\psi_x^{t+1}\rangle - |\psi_y^{t+1}\rangle\| &= \|U_{t+1}(\mathcal{O}_x - \mathcal{O}_y)|\psi_x^t\rangle + U_{t+1}\mathcal{O}_y(|\psi_x^t\rangle - |\psi_y^t\rangle)\| \\
&\leq \|U_{t+1}(\mathcal{O}_x - \mathcal{O}_y)|\psi_x^t\rangle\| + \|U_{t+1}\mathcal{O}_y(|\psi_x^t\rangle - |\psi_y^t\rangle)\| \qquad \text{(triangle inequality)} \\
&= \|(\mathcal{O}_x - \mathcal{O}_y)|\psi_x^t\rangle\| + \||\psi_x^t\rangle - |\psi_y^t\rangle\|. \qquad \text{(unitary invariance of the norm)}
\end{aligned}$$

The oracles $\mathcal{O}_x$ and $\mathcal{O}_y$ coincide when the query register holds an index i such that $x_i = y_i$. Hence, $(\mathcal{O}_x - \mathcal{O}_y)|\psi_x^t\rangle = (\mathcal{O}_x - \mathcal{O}_y)(\sum_{i:x_i\neq y_i}|i\rangle\langle i|\otimes\mathrm{Id})|\psi_x^t\rangle$. Using that $\|\mathcal{O}_x - \mathcal{O}_y\| \leq \|\mathcal{O}_x\| + \|\mathcal{O}_y\| \leq 2$ and the submultiplicativity of the norm, we conclude that $\|(\mathcal{O}_x - \mathcal{O}_y)|\psi_x^t\rangle\| \leq 2\big\|(\sum_{i:x_i\neq y_i}|i\rangle\langle i|\otimes \mathrm{Id})|\psi_x^t\rangle\big\|$. □

The quantity $\big\|(|i\rangle\langle i|\otimes\mathrm{Id})|\psi_x^t\rangle\big\|^2$ is sometimes called the *query weight* of i in the state $|\psi_x^t\rangle$, as it corresponds to the probability of measuring that index in the query register of $|\psi_x^t\rangle$. A direct corollary to the hybrid method states that an algorithm must assign a sufficiently large query weight to indices where $x_i \neq y_i$ in order to successfully distinguish between the two inputs.

Corollary 11 (Query weights lower bound)**.** *Consider a quantum algorithm that computes a function* $f : \{0,1\}^n \to \{0,1\}$ *in* T *queries. Let* $x, y \in \{0,1\}^n$ *be two inputs such that* $f(x) \neq f(y)$*. Then, the query weights must satisfy the inequality* $\sum_{t=0}^{T-1} \sqrt{\sum_{i:x_i\neq y_i} \big\|(|i\rangle\langle i|\otimes\mathrm{Id})|\psi_x^t\rangle\big\|^2} \geq 1/6$.

The quantity $\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i:x_i\neq y_i}\big\|(|i\rangle\langle i|\otimes\mathrm{Id})|\psi_x^t\rangle\big\|^2$ is sometimes more convenient to manipulate, as it has an operational meaning: it represents the probability of finding a witness index i that distinguishes between x and y when the query register is measured at a random step of the algorithm. By combining the above corollary with the Cauchy–Schwarz inequality, this probability must be at least $1/(36T^2)$ for the algorithm to succeed. In comparison, it can be easily shown that for randomized algorithms, this probability is at least $\Omega(1/T)$, thereby enabling stronger lower bounds.

Why is it called "hybrid"? The hybrid method has another formulation that is particularly useful, for instance, in security proofs for cryptographic protocols. It states that, in an arbitrary algorithm, replacing a single unitary step U with another U' induces an error of at most $\|(U-U')|\psi\rangle\|$, where $|\psi\rangle$ is the state of the algorithm immediately before the perturbation occurs. One can recover Corollary 11 by considering a sequence of hybrid algorithms $U_T\mathcal{O}_y U_{T-1}\dots\mathcal{O}_y U_t\mathcal{O}_x\dots\ U_1\mathcal{O}_x U_0$ where the oracle behaves as $\mathcal{O}_x$ until the t-th query, after which it switches to $\mathcal{O}_y$. The error between two consecutive hybrids is at most $\|(\mathcal{O}_x-\mathcal{O}_y)U_t\mathcal{O}_x\dots\ U_1\mathcal{O}_x U_0|0,0\rangle\| = \|(\mathcal{O}_x-\mathcal{O}_y)|\psi_x^t\rangle\| \leq 2\big\|(\sum_{i:x_i\neq y_i}|i\rangle\langle i|\otimes \mathrm{Id})|\psi_x^t\rangle\big\|$. The interested reader can find more details in a survey by Vazirani [Vaz98].

2.2 Applications

We describe two canonical uses of the hybrid method for problems whose output is sensitive to a small perturbations of their input.

Application 1: The OR function. Our first application is to recover the original quantum lower bound $Q(\mathrm{OR}) = \Omega(\sqrt{n})$ for the OR function. In order to use Theorem 10, we must find a way to bound the term $\big\|(\sum_{i:x_i\neq y_i}|i\rangle\langle i|\otimes \mathrm{Id})|\psi_x^t\rangle\big\|$ appearing in the progress evolution. This amounts to finding pairs of inputs (x,y) with $f(x)\neq f(y)$ such that the query weight increases slowly on the bits that differ between the two inputs.

The crucial idea is to focus the analysis on the all-0 input $x^{(0)} = (0,\dots,0)$ and the 1-sparse inputs $y^{(1)} = (1,0,\dots,0), y^{(2)} = (0,1,0,\dots,0),\dots, y^{(n)} = (0,\dots,0,1)$. The merit of this choice is that it simplifies the progress evolution into:

$$\||\psi_{x^{(0)}}^{t+1}\rangle - |\psi_{y^{(i)}}^{t+1}\rangle\| \leq \||\psi_{x^{(0)}}^{t}\rangle - |\psi_{y^{(i)}}^{t}\rangle\| + 2\big\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle\big\|$$

since $x^{(0)}$ and $y^{(i)}$ differ only on the bit at position i. We claim that, on average over i, the query weight $\big\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle\big\|^2$ is $1/n$. Indeed, $\sum_{i=1}^n\big\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle\big\|^2 = 1$ since the states $(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle$ are orthogonal. Hence, it requires $T=\Omega(\sqrt{n})$ queries to progress from $\||\psi_{x^{(0)}}^{0}\rangle - |\psi_{y^{(i)}}^{0}\rangle\| = 0$ (initial condition) to $\||\psi_{x^{(0)}}^{T}\rangle - |\psi_{y^{(i)}}^{T}\rangle\| \geq 1/3$ (final condition). We make this argument more formal below.

Proposition 12 (Hybrid method applied to OR)**.** *The quantum query complexity of the* OR *function is at least* $Q(\mathrm{OR}) \geq \sqrt{n}/6$.

Proof. Consider any quantum algorithm that computes the OR function using some number T of queries. Define the progress measure after $t\in\{0,\dots,T\}$ queries as:

$$\Delta_t = \sum_{i=1}^n \||\psi_{x^{(0)}}^{t}\rangle - |\psi_{y^{(i)}}^{t}\rangle\|$$

where $|\psi_{x^{(0)}}^{t}\rangle$ (resp. $|\psi_{y^{(i)}}^{t}\rangle$) is the state of the algorithm after t queries on input $x^{(0)}$ (resp. $y^{(i)}$). By Theorem 10, the progress must be $\Delta_0 = 0$ initially and at least $\Delta_T \geq n/3$ at the end because $f(x^{(0)})\neq f(y^{(i)})$ for all i. At each query, the progress increases by at most $\Delta_{t+1} \leq \Delta_t + 2\sum_{i=1}^n\big\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle\big\| \leq \Delta_t + 2\sqrt{n}$, since $\sum_{i=1}^n\big\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle\big\| \leq \sqrt{n\sum_{i=1}^n\big\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{x^{(0)}}^{t}\rangle\big\|^2} = \sqrt{n}$ by the Cauchy–Schwarz inequality. Hence, $n/3\leq\Delta_T\leq 2T\sqrt{n}$ and we necessarily have $T\geq\sqrt{n}/6$. □

The hybrid method is optimal in the case of the OR function, as Grover's algorithm [Gro97] provides a matching $O(\sqrt{n})$ upper bound. A careful inspection of that algorithm reveals that its query weights on the inputs $y^{(i)}$ are $\|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi_{y^{(i)}}^{t}\rangle\|^2 \approx t^2/n$. Hence, it falls close to the lower bound stated in Corollary 11 since $\sum_{t=0}^{T-1}\sqrt{t^2/n} = O(1)$ when $T=O(\sqrt{n})$.

Application 2: Block sensitivity. Our second application relates the query complexity and the *block sensitivity* of a Boolean function. The block sensitivity is a combinatorial measure of complexity introduced by Nisan [Nis91] that generalizes the notion of *sensitivity*. Informally, it is the largest number of disjoint bit flips in the input that can change the output of the function.

Definition 13 (Block sensitivity)**.** The *block sensitivity* $\mathrm{bs}(f)$ of a function $f : \{0,1\}^n \to \{0,1\}$ is the largest integer s such that there exist an input $x \in \{0,1\}^n$ and s disjoint subsets $B_1, \ldots, B_s \subseteq \{1, \ldots, n\}$ satisfying $f(x^{B_j}) \neq f(x)$ for all $1 \leq j \leq s$, where $x^{B_j} \in \{0,1\}^n$ is defined by $x_i^{B_j} = 1 - x_i$ when $i \in B_j$ and $x_i^{B_j} = x_i$ otherwise.

One can check that the block sensitivity is maximal (i.e., equal to n) for the OR, AND and Parity functions for instance. This is proved by observing that the output value is flipped on each block $B_j = \{j\}$ when the input is, respectively, $x = 0^n$, $x = 1^n$ and $x \in \{0,1\}^n$.

An interesting example of a function with lower block sensitivity, due to Rubinstein [Rub95], is constructed as follows. Suppose that $n = m^2$ is a square number and m is even. First, define the function $g : \{0,1\}^m \to \{0,1\}$ on m bits such that $g(x) = 1$ when all coordinates of x are zero except at two consecutive indices (i.e., $x_{2j} = x_{2j+1} = 1$ for some j). Next, consider the composed function $f = \mathrm{OR} \bullet g$, which applies g to the m consecutive substrings of size m in $x \in \{0,1\}^n$ and returns the OR of the results, i.e., $f(x) = \mathrm{OR}(g(x_1, \ldots, x_m), \ldots, g(x_{n-m+1}, \ldots, x_n))$. The block sensitivity of f is easily shown to be $\mathrm{bs}(f) = n/2$. The input achieving this optimum is the all-0 string, with the blocks $B_j = \{2j, 2j+1\}$ for all $j \in \{0, \ldots, n/2 - 1\}$. If the blocks were restricted to be of size one (which results in the *sensitivity* of f), then the optimum would only be $\sqrt{n}$.

The block sensitivity was identified by Nisan as characterizing the complexity of computing a function in a certain model of parallel computation. He also described the following connection with classical query complexity.

Proposition 14 (Lemma 4.2 in [Nis91])**.** *The randomized query complexity of any function $f : \{0,1\}^n \to \{0,1\}$ is at least $R(f) \geq \mathrm{bs}(f)/3$.*

Proof. The proof is by contradiction. Let x be an input on which f attains its block sensitivity, and consider $s = \mathrm{bs}(f)$ disjoint blocks $B_1, \ldots, B_s$ with $f(x^{B_j}) \neq f(x)$ for all j. Suppose that the algorithm makes fewer than $s/3$ queries. Then, on input x, there exists a block B_j of indices that is queried with probability less than $1/3$. When the algorithm does not query that block, it must be wrong with probability at least $1/2$ either on input x or x^{B_j} (since it cannot distinguish between the two). Hence, the algorithm fails with probability at least $(1 - 1/3) \cdot 1/2 = 1/3$ on some input. □

The hybrid method also helps us establish a quantum lower bound in terms of the block sensitivity. Another proof of this result will be given later using the polynomial method (Proposition 31).

Proposition 15 (Hybrid method applied to block sensitivity)**.** *The quantum query complexity of any function $f : \{0,1\}^n \to \{0,1\}$ is at least $Q(f) \geq \sqrt{\mathrm{bs}(f)}/6$.*

Proof. We proceed similarly to the proof of Proposition 12. First, we identify a hard set of inputs: fix $x \in \{0,1\}^n$ and $s = \mathrm{bs}(f)$ disjoint subsets $B_1, \ldots, B_s \subseteq \{1, \ldots, n\}$ such that $f(x^{B_j}) \neq f(x)$. Next, we define the progress measure after t queries as: $\Delta_t = \sum_{j=1}^{s} \| |\psi_x^t\rangle - |\psi_{x^{(B_j)}}^t\rangle \|$. By Theorem 10, the initial and final conditions imply that $\Delta_0 = 0$ and $\Delta_T \geq s/3$. Moreover, the progress increases by at most

$$\Delta_{t+1} \leq \Delta_t + 2 \sum_{j=1}^{s} \Big\| \Big(\sum_{i \in B_j} |i\rangle\langle i| \otimes \mathrm{Id} \Big) |\psi_x^t\rangle \Big\|$$

since x and $x^{(B_j)}$ differ only in the indices in B_j. By the Cauchy–Schwarz inequality, it is at most $\sum_{j=1}^{s} \left\|(\sum_{i\in B_j} |i\rangle\langle i| \otimes \mathrm{Id})|\psi_x^t\rangle\right\| \leq \sqrt{s\sum_{j=1}^{s} \left\|(\sum_{i\in B_j} |i\rangle\langle i| \otimes \mathrm{Id})|\psi_x^t\rangle\right\|^2} \leq \sqrt{s}$. Hence, $s/3 \leq \Delta_T \leq 2T\sqrt{s}$. □

The two above propositions give the best dependence on $\mathrm{bs}(f)$ that one can hope for in general. This is witnessed by the OR function that saturates the bounds: $R(\mathrm{OR}) = \Theta(n) = \Theta(\mathrm{bs}(\mathrm{OR}))$ and $Q(\mathrm{OR}) = \Theta(\sqrt{n}) = \Theta(\sqrt{\mathrm{bs}(\mathrm{OR})})$. In general, however, the block sensitivity may not be equal to the query complexity. There is at most a cubic gap with the deterministic complexity

$$D(f) = O(\mathrm{bs}(f)^3)$$

(see [BBC+01, Lemma 5.3] for an algorithm). Together with Proposition 15, it leads to the polynomial relationship $Q(f) \leq D(f) = O(Q(f)^6)$ between the deterministic and quantum query complexity of any function f. It is a major open problem to determine whether $D(f) = O(\mathrm{bs}(f)^2)$.

3 The Polynomial Method

The *polynomial method* was introduced in a work by Beals, Buhrman, Cleve, Mosca and de Wolf [BBC+01]. It establishes a deep connection between quantum query complexity and the approximation of real functions by low-degree polynomials. One of its great successes was the first optimal lower bound for the Collision problem by Aaronson and Shi [AS04].

This method tends to differ from the other techniques presented in this document, as it does not involve analyzing the increase of a progress measure under each query. Instead, it directly relates the existence of a T-query algorithm computing f to the existence of a $2T$-degree multilinear polynomial approximating f. The primary difficulty lies in lower bounding the degree of such multilinear polynomials, which is purely a matter of Boolean function analysis. In Section 3.2, we will introduce some central tools that can be used to address this challenge.

The theory of polynomial approximation is also instrumental in the design of quantum algorithms, such as in the Quantum Singular Value Transformation framework [GSLW19] or in obtaining a converse to the polynomial method [ABP19].

3.1 Technique

We are interested in polynomials with n variables that are evaluated on Boolean inputs $x_1, \ldots, x_n \in \{0,1\}$. We need only consider multilinear polynomials since $x_i^2 = x_i$ when x_i is Boolean. A multilinear polynomial has the following form.

Definition 16. A real, *multilinear polynomial* over the variables $x_1, \ldots, x_n$ is a polynomial of the form

$$p(x_1, \ldots, x_n) = \sum_{S\subseteq\{1,\ldots,n\}} a_S \prod_{i\in S} x_i$$

with real coefficients $a_S \in \mathbb{R}$. The *degree* of p is the size of the largest subset $S \subseteq \{1, \ldots, n\}$ with a non-zero coefficient: $\deg(p) = \max_{a_S\neq 0}|S|$.

A fundamental result in Boolean function analysis is that any function defined on the Boolean hypercube can be represented as a multilinear polynomial. Moreover, this representation is unique.

Lemma 17. *For any function $f : \{0,1\}^n \to \mathbb{R}$, there exists a unique multilinear polynomial p_f over the variables $x_1, \ldots, x_n$ such that $f(x) = p_f(x)$ for all $x \in \{0,1\}^n$.*

Proof. We construct a multilinear polynomial p_f that coincides with f. The proof of uniqueness is left to the reader. We start with indicator functions. For any $y \in \{0,1\}^n$, let $1_y : \{0,1\}^n \to \{0,1\}$ denote the indicator function that evaluates to 1 if and only if the input is $x = y$. The multilinear polynomial $p_y(x) = \prod_{i:y_i=1} x_i \prod_{i:y_i=0}(1-x_i)$ coincides with 1_y on all Boolean inputs. If f is an arbitrary function, it suffices to take the linear combination $p_f(x) = \sum_{y\in\{0,1\}^n} f(y)p_y(x)$. □

There are several measures of complexity associated with the polynomial representation of a Boolean function. It is often relevant to also consider the polynomials that are close to f in some sense. Here, we will focus on the exact degree $\deg(f) = \deg(p_f)$, and the approximate degree $\widetilde{\deg}(f)$, which is obtained by minimizing over all polynomials that *pointwise approximate* f.

Definition 18 (Exact and approximate degrees)**.** Consider a Boolean function $f : \{0,1\}^n \to \{0,1\}$. The *(exact) degree* $\deg(f)$ of f is the degree of its multilinear polynomial representation: $\deg(f) = \deg(p_f)$. The *approximate degree* $\widetilde{\deg}(f)$ of f is the smallest integer d such that there exists a polynomial p of degree d that approximates f in the sense

$$|p(x) - f(x)| \leq 1/3$$

for all $x \in \{0,1\}^n$.

Notice that the exact degree is always at most n, and the approximate degree is less than or equal to it. On the other hand, it was recently established [ABK+21, Theorems 1 and 18] that the gap between the two quantities is at most quadratic,

$$\widetilde{\deg}(f) \leq \deg(f) \leq \min\{n, 9\widetilde{\deg}(f)^2\}. \tag{1}$$

The OR function provides an example that saturates this bound (a direct proof of the lower bound $\widetilde{\deg}(\mathrm{OR}) = \Omega(\sqrt{n})$ will be given in Proposition 29).

Proposition 19 (Example 3.11 in [NS94])**.** *The exact and approximate degrees of the* OR *function satisfy, respectively,* $\deg(\mathrm{OR}) = n$ *and* $\widetilde{\deg}(\mathrm{OR}) = O(\sqrt{n})$.

Proof. The exact degree of OR is easily obtained from the fact that its polynomial representation is $\mathrm{OR}(x_1,\ldots,x_n) = 1 - (1-x_1)(1-x_2)\ldots(1-x_n)$.

For the approximate degree, we observe that it suffices to construct a *univariate* polynomial $q(z)$ of degree $O(\sqrt{n})$ such that $q(0) \in [0,1/3]$ and $q(z) \in [2/3,1]$ for all $z \in \{1,\ldots,n\}$. Indeed, the multivariate polynomial $p(x) = q(x_1 + \cdots + x_n)$ is an approximation of the OR function of degree $\deg(p) \leq \deg(q)$. The desired polynomial q can be obtained out of the k'th Chebyshev polynomial $T_k(z)$ of degree $k = O(\sqrt{n})$, by choosing $q(z) = aT_k(bz) + c$ for some constants $a, b, c \in \mathbb{R}$. The construction is detailed in [NS94] or [BT22, Lemma 7]. □

One can expect that the functions with high degrees are harder to compute. A result in that direction is easy to establish for the deterministic and randomized query complexities.

Proposition 20 (Classical polynomial method)**.** *The deterministic and randomized query complexities of any function* $f : \{0,1\}^n \to \{0,1\}$ *satisfy, respectively,* $D(f) \geq \deg(f)$ *and* $R(f) \geq \widetilde{\deg}(f)$.

Proof. We first consider the case of deterministic query algorithms. By Definition 1, there exists a decision tree of depth $T = D(f)$ computing f. We construct a polynomial of degree at most T in the variables $x_1,\ldots,x_n$ that coincides with the output of the tree. The proof proceeds by induction on the depth. If the tree consists of a single leaf ($T = 0$), the algorithm makes no query and the corresponding polynomial is simply the constant output associated with the leaf. For depth $T \geq 1$, let i be the index queried at the root of the tree, and let p_L (resp. p_R) be the polynomial of degree at most $T-1$ constructed recursively for the left (resp. right) subtree of

the root. Then, the polynomial $p = (1 - x_i)p_L + x_i p_R$ is of degree at most T and $p(x) = f(x)$ for all $x \in \{0,1\}^n$. We conclude that $\deg(f) \leq \deg(p) \leq T = D(f)$ (the equality $\deg(f) = \deg(p)$ holds if p is also multilinear, which happens, for instance, when the algorithm never queries the same index twice).

Now, consider a randomized query algorithm with query complexity $T = R(f)$. Such an algorithm corresponds to a probabilistic distribution over decision trees D of depth at most T. Using the same construction as above, we associate with each tree D a polynomial p_D of degree at most T that coincides with the output of that tree. We then define the polynomial p as the linear combination of the polynomials $\alpha_D p_D$, where $\alpha_D \in [0,1]$ is the probability of selecting D in the distribution. The crucial observation is that $p(x)$ corresponds to the probability that the randomized algorithm outputs 1 on input $x \in \{0,1\}^n$. By assumption, this probability is at least $p(x) \geq 2/3$ when $f(x) = 1$, and at most $p(x) \leq 1/3$ when $f(x) = 0$. Thus, p is an approximating polynomial for f. Since the approximate degree is the smallest degree of an approximating polynomial for f, we conclude that $\widetilde{\deg}(f) \leq \deg(p) \leq T = R(f)$. □

The example of the OR function shows that the lower bound on $R(f)$ in terms of the approximate degree is not necessarily tight, since $\widetilde{\deg}(\mathrm{OR}) = O(\sqrt{n})$ but $R(\mathrm{OR}) = \Omega(n)$. This comes as no surprise since the crux of the polynomial method will be to show that $\widetilde{\deg}(f)/2$ is in fact a lower bound on the quantum query complexity. The proof exploits the following fundamental property, stating that the amplitudes of a quantum state after one query are univariate polynomials in the input bits $x_1, \ldots, x_n$.

Lemma 21. *For any input $x \in \{0,1\}^n$, the effect of the quantum oracle operator on a basis state is*

$$\mathcal{O}_x|i,b\rangle = (1 - x_i)|i,b\rangle + x_i|i,b \oplus 1\rangle$$

for all $i \in \{1,\ldots,n\}$ and $b \in \{0,1\}$.

Proof. This follows directly from the definition: $\mathcal{O}_x|i,b\rangle = |i,b \oplus x_i\rangle$. □

By applying the lemma repeatedly, one obtains that the amplitudes after T queries are multilinear polynomials of degree T. Since the probability of outputting 1 is the *square* of an amplitude, it follows that any algorithm can be transformed into an approximating polynomial for f whose degree is twice the number of quantum queries.

Theorem 22 (Polynomial method)**.** *Consider a quantum algorithm that computes a function $f : \{0,1\}^n \to \{0,1\}$ using T queries to its input. Let $p(x) \in [0,1]$ denote the probability that the algorithm outputs 1 on the input $x \in \{0,1\}^n$. Then,*

- *(Approximation) $|p(x) - f(x)| \leq 1/3$ for all $x \in \{0,1\}^n$,*
- *(Polynomial degree) $\deg(p) \leq 2T$.*

In particular, the approximate degree of f is at most $\widetilde{\deg}(f) \leq 2T$.

Proof. The first point is immediate, as the probability of outputting 1 must be $p(x) \geq 2/3$ when $f(x) = 1$ and $p(x) \leq 1/3$ when $f(x) = 0$.

For the second point, let $|\psi_x^t\rangle$ denote the intermediate states of the algorithm after $t \in \{0,\ldots,T\}$ queries on the input $x \in \{0,1\}^n$. By definition, the probability of outputting 1 is

$$p(x) = \|(\mathrm{Id} \otimes |1\rangle\langle 1|)|\psi_x^T\rangle\|^2 = \sum_{1 \leq i \leq n} |\langle i,1 \,|\, \psi_x^T\rangle|^2.$$

It is sufficient to show that, for all $i \in \{1,\ldots,n\}$ and $b \in \{0,1\}$, the complex-valued function $x \mapsto \langle i,b \,|\, \psi_x^t\rangle$ is a polynomial in x of degree at most t. We prove it by induction on t. The

base case is trivial since $|\psi_x^0\rangle = U_0|0,0\rangle$ and $\langle i,b\,|\,\psi_x^0\rangle$ is independent of x. Suppose that the statement holds for t. Then, for $t+1$, the inner product is $\langle i,b\,|\,\psi_x^{t+1}\rangle = \langle i,b\,|\,U_{t+1}\mathcal{O}_x\,|\,\psi_x^t\rangle$. Let $U_{t+1}^\dagger|i,b\rangle = \sum_{1\le j\le n, c\in\{0,1\}} \alpha_{j,c}^\dagger|j,c\rangle$ denote the decomposition of the state $U_{t+1}^\dagger|i,b\rangle$ in the standard basis, where the complex numbers $\alpha_{j,c}$ are *independent* of x. By Lemma 21, the inner product is

$$\langle i,b\,|\,\psi_x^{t+1}\rangle = \sum_{1\le j\le n, c\in\{0,1\}} \alpha_{j,c}\big((1-x_j)\langle j,c\,|\,\psi_x^t\rangle + x_j\langle j,c\oplus 1\,|\,\psi_x^t\rangle\big).$$

By the induction hypothesis, each term $\langle j,c\,|\,\psi_x^t\rangle$ and $\langle j,c\oplus 1\,|\,\psi_x^t\rangle$ are multivariate polynomials in x of degree at most t. Hence, $\langle i,b\,|\,\psi_x^{t+1}\rangle$ is of degree at most $t+1$. □

This result has several interesting consequences in approximation theory and quantum query lower bounds. For instance, the existence of Grover's algorithm provides an alternative proof of Proposition 19 on the approximate degree of the OR function. These types of "algorithmically-inspired" polynomials are discussed in more detail in [BT22, Section 4]. Another crucial implication in query complexity is that no algorithm can compute a function f using fewer than $\widetilde{\deg}(f)/2$ queries, as this would result in an approximating polynomial of degree less than $\widetilde{\deg}(f)$.

Corollary 23. *The quantum query complexity of any function $f : \{0,1\}^n \to \{0,1\}$ is at least $Q(f) \ge \widetilde{\deg}(f)/2$.*

Together with Equation (1), it also implies that $Q(f) \ge \sqrt{\deg(f)}/6$. The latter inequality yields, for instance, that $Q(\mathrm{OR}), Q(\textsc{Parity}) \ge \sqrt{n}/6$ (which is not optimal in case of the PARITY function). We will develop more direct methods for analyzing the approximate degree in the next section.

3.2 Applications

We present three applications of the polynomial method, each exploiting the multilinear polynomial constructed in Theorem 22 in different ways.

Application 1: Distinguishing distributions. We begin with a simple application of the polynomial method, although in a problem setting that slightly differs from what we have considered so far. Instead of computing a Boolean function f, we consider the problem of distinguishing between two probability distributions μ_0 and μ_1.

Definition 24 (Distinguishing problem)**.** Let μ_0 and μ_1 be two distributions over the set $\{0,1\}^n$. In the (μ_0,μ_1)-*distinguishing problem*, the algorithm is given oracle access to an input $x \in \{0,1\}^n$ sampled either from μ_0 or μ_1, and it must output a bit $b \in \{0,1\}$ that maximizes the *distinguishing advantage*:

$$\big|\Pr[b=1 \mid x\sim\mu_0] - \Pr[b=1 \mid x\sim\mu_1]\big|.$$

We say that two distributions are *indistinguishable* by a class of algorithms if no algorithm in that class can make the advantage nonzero. Distributions that are indistinguishable from the uniform distribution μ_{unif} are called *pseudorandom*.

The construction of pseudorandom distributions is a topic of central importance in cryptography and in complexity theory. We show that the following *k-wise independence* condition is sufficient for a distribution to be pseudorandom against quantum algorithms that make few queries.

Definition 25 (k-wise independence)**.** We say that a distribution μ over $\{0,1\}^n$ is *k-wise independent* if for all subset $S \subseteq \{1,\dots,n\}$ of size at most k, its marginal distribution on the coordinates indexed by S is uniform: $\Pr_{x\sim\mu}[x_i = a_i, \forall i \in S] = 2^{-|S|}$ for all choices of $a_i \in \{0,1\}$.

It is easy to see that k-wise independent distributions are pseudorandom for classical algorithms that make at most k queries, since the knowledge of any k input bits provides no information on whether the distribution is uniform or not. Using the polynomial method, we demonstrate a similar result for quantum algorithms, although with a slightly stronger assumption on the number of queries.

Proposition 26 (Polynomial method applied to the distinguishing problem)**.** *Let μ be a $2k$-wise independent distribution over $\{0,1\}^n$. Then, μ is pseudorandom for the class of quantum quantum algorithms that make at most k queries.*

Proof. Fix any quantum algorithm that outputs a bit b after making k queries to its input x. By Theorem 22, the probability $p(x)$ that it outputs $b = 1$ on input x is given by a function $p : \{0,1\}^n \to [0,1]$ of degree at most $\deg(p) \leq 2k$. Let $\{a_S\}_{S\subseteq\{1,\ldots,n\},|S|\leq 2k}$ denote the coefficients of the multilinear polynomial that coincides with it, $p(x) = \sum_{S\subseteq\{1,\ldots,n\},|S|\leq 2k} a_S \prod_{i\in S} x_i$. The distinguishing advantage of the algorithm is,

$$\begin{aligned}\left|\Pr[b=1 \mid x\sim\mu] - \Pr[b=1 \mid x\sim\mu_{\text{unif}}]\right| &= \left|\mathbb{E}_{x\sim\mu}[p(x)] - \mathbb{E}_{x\sim\mu_{\text{unif}}}[p(x)]\right| \\ &= \left|\sum_{S:|S|\leq 2k} a_S \left(\mathbb{E}_{x\sim\mu}\left[\prod_{i\in S} x_i\right] - \mathbb{E}_{x\sim\mu_{\text{unif}}}\left[\prod_{i\in S} x_i\right]\right)\right|.\end{aligned}$$

Since μ is $2k$-wise independent, its marginal distribution over any set S of at most $2k$ indices is uniform. It implies that the product of any $2k$ coordinates has the same expected value under the distributions μ and μ_{unif}. Hence, $\mathbb{E}_{x\sim\mu}[\prod_{i\in S} x_i] = \mathbb{E}_{x\sim\mu_{\text{unif}}}[\prod_{i\in S} x_i]$ and the distinguishing advantage of the algorithm is zero. □

This type of application of the polynomial method can be generalized to other problems relevant in cryptography, such as polynomial interpolation [KK11].

Application 2: Symmetrization. The polynomial method leads to the study of approximating polynomials whose number n of variables grows with the input size of the problem. Symmetrization refers to a family of techniques that exploit the symmetries of the problem to construct another approximating polynomial with far fewer variables – ideally one or two – without increasing the degree. The resulting polynomial typically inherits certain fluctuation properties from the original polynomial (such as the number of roots, large derivative, etc.), which can be exploited to lower bound its degree.

We describe the most standard symmetrization technique, due to Minsky and Papert [MP69], which consists of averaging the polynomial over all inputs with the same Hamming weight. This approach works well for problems that are invariant under permutations of the input bits.

Proposition 27 (Minsky-Papert symmetrization)**.** *Let $p(x_1,\ldots,x_n)$ be a multilinear polynomial over the variables $x_1,\ldots,x_n$. Then, there exists a univariate polynomial $p_{\text{sym}}(k)$ of degree at most $\deg(p_{\text{sym}}) \leq \deg(p)$ such that, for all integers $k \in \{0,\ldots,n\}$,*

$$p_{\text{sym}}(k) = \mathbb{E}_{x\in\{0,1\}^n:|x|=k}[p(x)].$$

Proof. It is sufficient to prove the existence of p_{sym} when p consists of a single monomial $p(x) = \prod_{i\in S} x_i$ (the general case follows by linearity of expectation). Let $d = |S|$ and consider the univariate polynomial $p_{\text{sym}}(k) = \frac{k(k-1)\cdots(k-d+1)}{n(n-1)\cdots(n-d+1)}$ of degree d. It is a simple calculation to check that,

$$p_{\text{sym}}(k) = \begin{cases} 0 & = \mathbb{E}_{x\in\{0,1\}^n:|x|=k}[\prod_{i\in S} x_i] \quad \text{when } k \in \{0,\ldots,d-1\}, \\ \frac{\binom{n-d}{k-d}}{\binom{n}{k}} & = \mathbb{E}_{x\in\{0,1\}^n:|x|=k}[\prod_{i\in S} x_i] \quad \text{when } k \in \{d,\ldots,n\}. \end{cases}$$

□

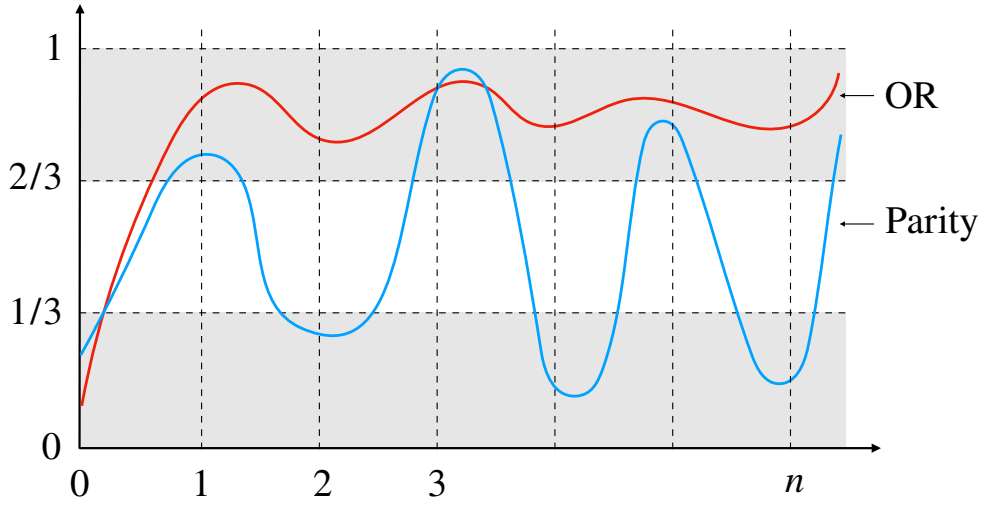

Figure 3: Possible representations of symmetrized polynomials p_{sym} obeying the constraints for the OR and PARITY functions.

Notice that the dependence on n has shifted from the number of variables in p to the size of the domain used to characterize p_{sym}. This often simplifies the degree analysis. We describe three applications of the Minsky-Papert symmetrization.

The first application is to show that the approximate degree of the PARITY function is n, which is the largest possible value and must be equal to the exact degree (indeed, $\mathrm{PARITY}(x_1,\ldots,x_n) = (1-(1-2x_1)\cdots(1-2x_n))/2$). By Corollary 23, this implies that the quantum query complexity is at least $Q(\mathrm{PARITY}) \geq n/2$. This is quadratically better than the lower bound $Q(\mathrm{PARITY}) \geq \sqrt{n}/6$ obtained with the hybrid method (Proposition 15). It is also optimal since Deutsch's algorithm [Deu85] can compute the parity of two bits in one quantum query (the algorithm makes no error, and hence the parity of n bits can be obtained by repeating the algorithm on $n/2$ different pairs of bits when n is even).

Proposition 28 (Symmetrization method applied to PARITY)**.** *The approximate degree of the* PARITY *function is equal to* $\widetilde{\deg}(\mathrm{PARITY}) = n$*. Hence, the quantum query complexity is at least* $Q(\mathrm{PARITY}) \geq n/2$*.*

Proof. Consider any multilinear polynomial $p(x_1,\ldots,x_n)$ that approximates the PARITY function. The proof consists of showing that, necessarily, $\deg(p) \geq \deg(p_{\mathrm{sym}}) \geq n$. The first inequality is immediate by Proposition 27. The second inequality uses the property that,

$$|p_{\mathrm{sym}}(k)| \leq 1/3 \quad \text{for } k \in \{0,2,4,\ldots\} \quad \text{and} \quad |p_{\mathrm{sym}}(k)-1| \leq 1/3 \quad \text{for } k \in \{1,3,5,\ldots\}$$

since PARITY evaluates to 0 on inputs with an even Hamming weight, and to 1 on inputs with an odd Hamming weight. A possible representation of the polynomial p_{sym} on the interval $[0,n]$ is shown in Figure 3. The above property implies that the polynomial $1-2p_{\mathrm{sym}}$ changes sign at least n times over the interval $[0,n]$. Hence, its number of roots must be at least n and $\deg(p_{\mathrm{sym}}) \geq n$. □

The next application provides another proof that the quantum query complexity of OR is $\Omega(\sqrt{n})$. The symmetrized polynomial p_{sym} is slightly harder to analyse here, as it requires using an inequality from approximation theory.

Proposition 29 (Symmetrization method applied to OR)**.** *The approximate degree of the* OR *function is at least* $\widetilde{\deg}(\mathrm{OR}) \geq \sqrt{n/6}$*. Hence, the quantum query complexity is at least* $Q(\mathrm{OR}) \geq \sqrt{n/24}$*.*

Proof. Consider any multilinear polynomial p that approximates the OR function. The polynomial $p_{\mathrm{sym}}(k)$ derived from Proposition 27 has the property that,

$$|p_{\mathrm{sym}}(0)| \leq 1/3 \quad \text{and} \quad |p_{\mathrm{sym}}(k) - 1| \leq 1/3 \ \text{ for } k \in \{1, \ldots, n\}$$

since OR evaluates to 0 on inputs of Hamming weight $|x| = 0$, and to 1 on inputs of Hamming weight $|x| \in \{1, \ldots, n\}$. A possible representation of the polynomial p_{sym} on the interval $[0, n]$ is shown in Figure 3. By the Mean Value Theorem, there must exist a real $x \in [0, 1]$ such that the derivative at x of the function p_{sym} is at least $p'_{\mathrm{sym}}(x) \geq 1/3$. We evoke the following lower bound on the degree of polynomials with such large derivatives.

Lemma 30 (Ehlich-Zeller and Rivlin-Cheney Theorem)**.** *Let $a, b, c \in \mathbb{R}$ (with $a < b$ and $c > 0$), $n \in \mathbb{N}$ and $p : \mathbb{R} \to \mathbb{R}$ be a polynomial such that $p(k) \in [a, b]$ for all integers $k \in \{0, 1, \ldots, n\}$ and $|p'(x)| \geq c$ for some real $x \in [0, n]$. Then, $\deg(p) \geq \sqrt{cn/(c + b - a)}$.*

A direct application of this lemma to the polynomial p_{sym} with $a = -1/3$, $b = 4/3$ and $c = 1/3$ leads to the conclusion that $\deg(p_{\mathrm{sym}}) \geq \sqrt{n/6}$. □

The next application extends the previous result to a general lower bound on the approximate degree in terms of the block sensitivity (Definition 13).

Proposition 31 (Symmetrization method applied to block sensitivity)**.** *The approximate degree of any function $f : \{0, 1\}^n \to \{0, 1\}$ is at least $\widetilde{\deg}(f) \geq \sqrt{\mathrm{bs}(f)/6}$. Hence, the quantum query complexity is at least $Q(f) \geq \sqrt{\mathrm{bs}(f)/24}$.*

Proof. Let $x \in \{0, 1\}^n$ be an input on which f attains its block sensitivity. Suppose, without loss of generality, that $f(x) = 0$ and fix $s = \mathrm{bs}(f)$ disjoint blocks $B_1, \ldots, B_s$ with $f(x^{B_j}) = 1$ for all j. We define the function $\pi : \{0, 1\}^s \to \{0, 1\}^n$ that associates with each $y \in \{0, 1\}^s$ the bitstring obtained by flipping all blocks in x indexed by y, i.e., $\pi(y)_i = 1 - x_i$ if $i \in B_j$ and $y_j = 1$, and $\pi(y)_i = x_i$ if $i \in B_j$ and $y_j = 0$.

Given any polynomial p that approximates f, we can construct another multilinear polynomial $q(y_1, \ldots, y_s)$, over s variables, such that

$$q(y_1, \ldots, y_s) = p(\pi(y))$$

for all $y \in \{0, 1\}^s$ and $\deg(q) \leq \deg(p)$ (it suffices to replace the i-th variable in p with $x_i(1 - y_j) + (1 - x_i)y_j$ if $i \in B_j$). In particular, $q(y)$ evaluates to $p(x)$ on the all-0 string, and to $p(x^{B_j})$ on the string y with a single 1 at position j. Hence, its symmetrized polynomial must satisfy,

$$|q_{\mathrm{sym}}(0)| \leq 1/3, \quad |q_{\mathrm{sym}}(1) - 1| \leq 1/3, \quad q_{\mathrm{sym}}(k) \in [-1/3, 4/3] \ \text{ for } k \in \{2, \ldots, s\}.$$

The Ehlich-Zeller and Rivlin-Cheney Theorem yields that $\deg(q_{\mathrm{sym}}) \geq \sqrt{s/6} = \sqrt{\mathrm{bs}(f)/6}$. □

The reader interested in problems that are more challenging to symmetrize can continue her reading with the lower bounds for the Collision [AS04], and AND-OR tree [Kre21] problems.

Application 3: Dual polynomials. We conclude with a more recent and powerful technique for lower bounding the approximate degree. For convenience, in this section, we express the domain and range of Boolean functions as,

$$f : \{-1, 1\}^n \to \{-1, 1\}.$$

This amounts to replacing the bit $b \in \{0, 1\}$ with $1 - 2b$, which is an operation that preserves the (approximate) degree.

The central idea of the dual polynomial method is to view the approximate degree as being given by the following pair of primal-dual linear programs.

Primal linear program

$$\begin{array}{lll} \min_{\epsilon,p} & \epsilon & \\ \text{s.t.} & |p(x) - f(x)| \leq \epsilon & \forall x \in \{-1,1\}^n \\ & \deg(p) < d & \end{array}$$

Dual linear program

$$\begin{array}{lll} \max_{\phi} & \sum_{x\in\{-1,1\}^n} \phi(x) \cdot f(x) & \\ \text{s.t.} & \sum_x |\phi(x)| = 1 & \\ & \sum_x \phi(x) \cdot p(x) = 0 & \forall p, \deg(p) < d \end{array}$$

The variables of the primal program are the approximation parameter ϵ and the $\binom{n}{<d}$ coefficients needed to represent a polynomial $p : \{-1,1\}^n \to \{-1,1\}$ of degree less than d. The variables of the dual program are the 2^n values $\phi(x)$ needed to specify a function $\phi : \{-1,1\}^n \to \mathbb{R}$.

It is straightforward to relate the primal program with the approximate degree of f: for a fixed value of d, the approximate degree is *at most* $\widetilde{\deg}(f) < d$ if and only if the optimal value is at most $\epsilon \leq 1/3$. The dual program is more interesting to interpret: by weak duality, the approximate degree is *at least* $\widetilde{\deg}(f) \geq d$ if one can identify a so-called *dual polynomial* $\phi : \{-1,1\}^n \to \mathbb{R}$ satisfying the next conditions.

Proposition 32 (Method of Dual Polynomials)**.** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a Boolean function and $d \in \{0,\ldots,n\}$ be an integer. Suppose that there exists a real-valued function $\phi : \{-1,1\}^n \to \mathbb{R}$ such that,*

- *(Correlation)* $\sum_{x\in\{-1,1\}^n} \phi(x) \cdot f(x) > 1/3$,
- *(Normalization)* $\sum_{x\in\{-1,1\}^n} |\phi(x)| = 1$,
- *(Pure high degree)* $\sum_{x\in\{-1,1\}^n} \phi(x) \cdot p(x) = 0$ *for all polynomials* $p : \{-1,1\}^n \to \{-1,1\}$ *of degree at most* $\deg(p) < d$.

Then the approximate degree of f must be at least $\widetilde{\deg}(f) \geq d$.

In words, $2^n\phi$ must not deviate significantly from f (correlation and normalization), while having no monomial of degree less than d (pure high degree). It suffices to exhibit one such function ϕ to certify that the approximate degree of f is at least d. A simple case of application is for Parity.

Proposition 33 (Dual polynomial method applied to Parity)**.** *The approximate degree of the* Parity *function is* $\widetilde{\deg}(\textsc{Parity}) = n$.

Proof. The polynomial representation of the Parity function over $\{-1,1\}$ is given by the degree-n monomial $\textsc{Parity}(x_1,\ldots,x_n) = x_1 \cdots x_n$. It is easy to check that $\phi(x) = \frac{1}{2^n} x_1 \cdots x_n$ is a valid dual polynomial: $\sum_x \phi(x) \cdot f(x) = 1$, $\sum_x |\phi(x)| = 1$ and $\deg(\phi) = n$. □

Unfortunately, we will not provide additional examples of dual polynomials, as they quickly become complicated to construct (even for the OR function). The reader is invited to consult the survey by Bun and Thaler [BT22] for more details and intuition on this method.

4 The Recording Method

The *recording method* (also called the *compressed oracle* technique) was introduced in a recent work by Zhandry [Zha19]. It was originally intended for security proofs in the quantum random oracle model (QROM), where the input $x \in \{0,\ldots,m-1\}^n$ is often non-Boolean ($m > 2$) and is meant to represent a random hash functions $H(i) = x_i$. Since then, it has also proven useful in query lower bounds, although there are some limitations to the type of problems it applies to.

The recording method is best suited for problems that are hard *on average* when the input is drawn from some sufficiently *unstructured* distribution. It also requires the problem to be based on a *local property*, in the sense that the output is determined by a small subset of input coordinates satisfying a certain predicate. Some of these limitations have been partially lifted since then, but we will not touch upon these improvements here. Our presentation follows the approach in [HM21].

A typical case of application, which will be developed in the applications section, is the problem of finding a coordinate equals to $x_i = 1$ in a random input x (with alphabet size $m \approx n$). This is a variant of the OR problem called (preimage) SEARCH, also solved with $O(\sqrt{n})$ queries by Grover's algorithm. Here, the input is drawn from the uniform distribution, so each coordinate is independently a solution with probability $1/m$.

4.1 Technique

The recording method starts by placing a certain hard distribution μ on the input x. The most well-understood case is when the coordinates x_i are *independent and identically distributed* under μ. To handle some problems of interest, it is best to extend the input alphabet beyond the Boolean domain. Hence, we will detail the method when the input is drawn from the distribution,

$$\mu_{\text{unif}} : \text{uniform distribution on } \{0, \dots, m-1\}^n = \Sigma^n$$

for some integer $m \geq 2$. The quantum query framework stated in Section 1.1 must first be extended in three directions:

(Large input alphabet) The algorithm is given access to an input $x \in \{0, \dots, m-1\}^n$ through the oracle $\mathcal{O}_x|i, b\rangle = |i, b + x_i \bmod m\rangle$, where $i \in \{1, \dots, n\}$ and $b, x_i \in \{0, \dots, m-1\}$. Equivalently, it is given access to the phase oracle $\mathcal{O}_x^{\pm}|i, b\rangle = \omega^{bx_i}|i, b\rangle$, where $\omega = e^{2\mathbf{i}\pi/m}$ is the m-th root of unity.

(Relational problems) The problem can have multiple valid solutions: each input x is associated with a set $\mathcal{F}_x \subseteq \{1, \dots, n\}$ of solutions, and the algorithm succeeds if it outputs any element from that set.

(Average-case output condition) The algorithm must output a valid solution with probability at least $2/3$, where the randomness is both over the actions of the algorithm (e.g., the outcome of a measurement) *and* the distribution μ of the input.

We make a few comments on the above definitions. First, it will be more convenient to use the phase oracle $\mathcal{O}_x^{\pm}|i, b\rangle = \omega^{bx_i}|i, b\rangle$ in the recording method. Its equivalence with the oracle $\mathcal{O}_x$ follows by the same arguments as in Fact 4. Next, the second item offers more flexibility in the type of problems that can be considered, as the solution need not be Boolean, nor unique. The set of valid solutions is now represented by a *relation* $\mathcal{F} = \{(x, i) : x \in \{0, \dots, m-1\}^n, i \in \mathcal{F}_x\}$, as opposed to a function $f(x)$ (which would restrict the relation to singletons $\mathcal{F}_x = \{f(x)\}$). For simplifying the model, we assume that the solutions can be encoded as numbers from the set $\{1, \dots, n\}$, but other sets can work as well (the COLLISION problem, defined in the applications section, would require encoding the solutions over $\{1, \dots, n\} \times \{1, \dots, n\}$). Finally, the average-case condition is also a relaxation of the output condition stated in Section 1.1. Here, the algorithm must perform well for a *large fraction* of inputs (under the given distribution), instead of being successful on all of them. The complexity of the optimal algorithm under a given distribution is called the average-case quantum query complexity.

Definition 34 (Average-case query complexity)**.** The *average-case quantum query complexity* $Q_\mu(\mathcal{F})$ of a relation $\mathcal{F}$ under a distribution μ is the smallest integer T such that there exists

a quantum algorithm with query complexity T that, given an input x sampled according to μ, outputs $y \in \mathcal{F}_x$ with probability at least $2/3$. Similarly, we define $R_\mu(\mathcal{F})$ as the average-case randomized query complexity.

Query record. The core idea of the recording method is to construct a quantum object that keeps track of what queries an algorithm has made. If the queries were classical, it would suffice to take the record $R_t = ((i_1, x_{i_1}), (i_2, x_{i_2}), \ldots, (i_t, x_{i_t}))$ of all the query-answer pairs seen by the algorithm after t queries (this is a random variable, which depends on the random input x and the random actions of the algorithm). This object is very helpful in quantifying the knowledge gained by an algorithm toward solving a problem. However, it becomes more difficult to define a similar record for quantum queries. For instance, a single quantum query can learn information about the whole input by querying all coordinates in superposition. The solution will be to construct a record that is itself a quantum state – entangled with the state of the algorithm. The construction proceeds in three steps:

1. *Input purification:* the input distribution is encoded into a bipartite quantum state, one system being the state of the algorithm and the other being a purification register.

2. *Record state:* the purification register is mapped to a different basis in which it can be interpreted as a quantum record.

3. *Recording oracle:* the query operator is modified to operate directly on the record state with some desirable properties.

We detail each step of the construction below. The reader can refer in parallel to the pictures in Figures 4 and 5, which summarize the main ideas of the process.

1. Input purification. We first address the question of representing the state of an algorithm whose input is random. This will also be useful in Section 5 when describing the adversary method. Recall that we denote the state of an algorithm after t queries on a fixed input x as $|\psi_x^t\rangle$. There are two equivalent ways of representing the average state of the algorithm when x is chosen randomly with some probability $\mu(x)$,

$$\rho^t = \sum_x \mu(x) |\psi_x^t\rangle\langle\psi_x^t| \quad \longleftrightarrow \quad |\psi^t\rangle = \sum_x \sqrt{\mu(x)} |\psi_x^t\rangle \otimes |x\rangle.$$

The first representation uses the density matrix formalism. The system is described by the pure-state ensemble $\{\mu(x), |\psi_x^t\rangle\}_x$ corresponding to the density operator ρ^t. The second representation is a purification of ρ^t, where the purification register holds a copy of the input x. In this section, we will work with the latter representation, as it retains the *joint state* of the algorithm and input distribution (whereas ρ^t encodes only the marginal state on the algorithm registers).

We extend the formalism given in Definition 5 to define a query algorithm using the joint state $|\psi^t\rangle$. The purification register acts as a control state for the joint oracle $\mathcal{O}^\pm$ defined as: $\mathcal{O}^\pm(|i, b\rangle \otimes |x\rangle) = (\mathcal{O}_x^\pm |i, b\rangle) \otimes |x\rangle = \omega^{bx_i} |i, b\rangle \otimes |x\rangle$. The unitaries U_t applied by the algorithm do not depend on x, hence they are extended to act as the identity on the purification register. The new formalism is given in the next definition and displayed in Figure 4.

Definition 35 (Joint state of a quantum query algorithm with input distribution)**.** A (memoryless) *quantum query algorithm* with query complexity T and input alphabet $\Sigma = \{0, \ldots, m-1\}$ is a sequence $U_0, \ldots, U_T$ of unitary operators acting on the Hilbert space spanned by the vectors $|i, b\rangle$ where $i \in \{1, \ldots, n\}$ and $b \in \Sigma$.

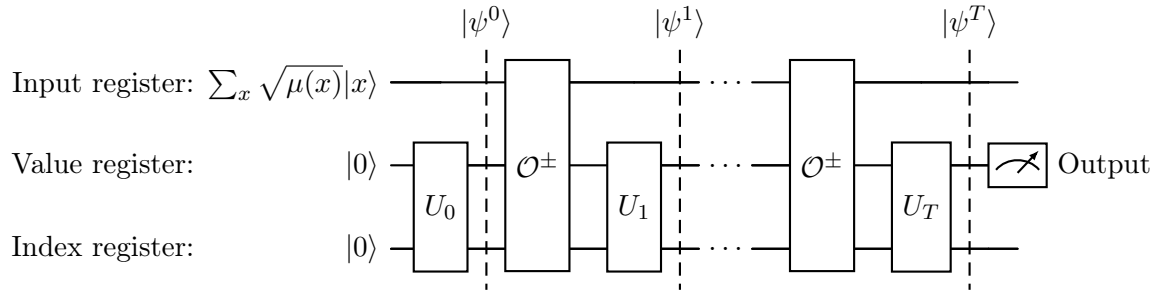

Figure 4: Canonical form of a (memoryless) quantum query algorithm with input distribution.

The *joint phase oracle* is $\mathcal{O}^{\pm} = \sum_{x\in\Sigma^n} \mathcal{O}_x^{\pm} \otimes |x\rangle\langle x|$, and the *joint state* $|\psi^t\rangle$ *of the algorithm and input* after $t \in \{0,\dots,T\}$ queries on an input distribution $(\mu(x))_{x\in\Sigma^n}$ is defined as,

$$|\psi^t\rangle = (U_t \otimes \mathrm{Id})\mathcal{O}^{\pm} \dots (U_1 \otimes \mathrm{Id})\mathcal{O}^{\pm}(U_0 \otimes \mathrm{Id})\Big(|0,0\rangle \otimes \sum_{x\in\Sigma^n} \sqrt{\mu(x)}|x\rangle\Big).$$

Equivalently, $|\psi^t\rangle = \sum_{x\in\Sigma^n} \sqrt{\mu(x)}|\psi_x^t\rangle \otimes |x\rangle$ where $|\psi_x^t\rangle = U_t\mathcal{O}_x^{\pm} \dots U_1\mathcal{O}_x^{\pm}U_0|0,0\rangle$ is the state of the algorithm on the input x. The reduced density matrix representing the state of the algorithm after t queries is denoted ρ^t and is equal to $\sum_{x\in\Sigma^n} \mu(x)|\psi_x^t\rangle\langle\psi_x^t|$.

The *output* of the algorithm is the random value i obtained by measuring the index register of $|\psi^T\rangle$ in the standard basis. The (average) *success probability* p_{succ}^{μ} of the algorithm in computing a relation $\mathcal{F}$ on the input distribution μ is the probability of measuring a correct output for a random input: $p_{\mathrm{succ}}^{\mu} = \sum_{x\in\Sigma^n, i\in\mathcal{F}_x} \big\|(|i\rangle\langle i| \otimes \mathrm{Id} \otimes |x\rangle\langle x|)|\psi^T\rangle\big\|^2$.

2. Record state. We explain how to construct the quantum record by embedding the input register of $|\psi^t\rangle$ into a Hilbert space of higher dimension and applying a certain unitary on it. The input register is originally supported over the m^n basis states $|x\rangle = |x_1,\dots,x_n\rangle$ where $x_i \in \{0,\dots,m-1\}$. The latter alphabet is augmented with a new "empty" symbol $\varnothing$, which will be used to represent the absence of knowledge from the algorithm about a coordinate of the input. The *record space* is the Hilbert space of dimension $(m+1)^n$ spanned by the vectors,

$$|x_1,\dots,x_n\rangle = |x_1\rangle \otimes \dots \otimes |x_n\rangle \quad \text{where} \quad x_i \in \{0,\dots,m-1\} \cup \{\varnothing\}.$$

Equivalently, it is the n-fold tensor product of the Hilbert space of dimension $m+1$ spanned by the vectors $|0\rangle,\dots,|m-1\rangle,|\varnothing\rangle$. The input register of $|\psi^t\rangle$ is renamed the *record register* when it is interpreted as living in the record space. We aim to define a *recording unitary* $\mathcal{R}$ – operating on the record space – that prepares a state $|\psi_{\mathrm{rec}}^t\rangle = (\mathrm{Id} \otimes \mathcal{R})|\psi^t\rangle$ whose record register contains the *record state* of the algorithm. This unitary will depend solely on the input distribution μ_{unif} and inherit its product structure. We can consider two situations that guide the choice of this operator:

- *(Initial state)* The initial state of the algorithm conveys no information on the input, so the record should start in the all-empty state $|\varnothing\rangle^{\otimes n}$. This corresponds to the initial joint state $|\psi^0\rangle$ being mapped to,

$$|\psi^0\rangle = (U_0|0,0\rangle) \otimes \left(\frac{1}{m^{n/2}} \sum_{x\in\Sigma^n} |x\rangle\right) \xmapsto{\mathrm{Id}\otimes\mathcal{R}} |\psi_{\mathrm{rec}}^0\rangle = (U_0|0,0\rangle) \otimes |\varnothing\rangle^{\otimes n}.$$

  Hence, the unitary $\mathcal{R}$ shall map the uniform superposition to the state $|\varnothing\rangle^{\otimes n}$.

- *(Phase kickback)* If the algorithm queries the state $|i,b\rangle \otimes \left(\frac{1}{m^{n/2}} \sum_{x\in\Sigma^n} |x\rangle\right)$ with $b \neq 0$ then the post-query state is (up to the normalization factor $1/m^{n/2}$):

$$\begin{aligned} |i,b\rangle \otimes \textstyle\sum_{x\in\Sigma^n} |x\rangle \quad &\stackrel{\mathcal{O}^{\pm}}{\longmapsto} \quad \textstyle\sum_{x\in\Sigma^n} \omega^{bx_i} |i,b\rangle \otimes |x\rangle \\ &= \quad |i,b\rangle \otimes \left(\textstyle\sum_{x_1\ldots x_{i-1}\in\Sigma^{i-1}} |x_1 \ldots x_{i-1}\rangle\right) \otimes \left(\textstyle\sum_{x_i\in\Sigma} \omega^{bx_i} |x_i\rangle\right) \\ &\qquad\qquad \otimes \left(\textstyle\sum_{x_{i+1}\ldots x_n\in\Sigma^{n-i}} |x_{i+1}\ldots x_n\rangle\right). \end{aligned}$$

  Hence, a query can equivalently be seen as a modification of the record state, rather than of the state of the algorithm. Furthermore, when querying the index i, only the i-th subsystem of the record register is modified and becomes *orthogonal* to the initial uniform superposition. This provides a natural criterion for including an input coordinate in the record state: if the state of its register is orthogonal to the uniform superposition, then the unitary $\mathcal{R}$ should keep it intact into the record; otherwise, it should replace it with an empty record,

$$\sum_{x\in\Sigma^n} \omega^{bx_i} |i,b\rangle \otimes |x\rangle \stackrel{\mathrm{Id}\otimes\mathcal{R}}{\longmapsto} |i,b\rangle \otimes |\varnothing\rangle^{\otimes i-1} \otimes \Big(\sum_{x_i\in\Sigma} \omega^{bx_i} |x_i\rangle\Big) \otimes |\varnothing\rangle^{\otimes(n-i)}.$$

It is easy to find a unitary $\mathcal{R}$ that satisfies all of the above conditions. Due to the symmetries of the input distribution μ_{unif}, it suffices to take the tensor product $\mathcal{R} = \mathcal{R}_1 \otimes \cdots \otimes \mathcal{R}_n$ of n identical operators $\mathcal{R}_i$ acting on each subsystem of the record register as follows.

Definition 36 (Recording operator)**.** Let $\mathcal{R}_i$ denote the unitary and Hermitian operator acting on the i-th subsystem of the record space as follows:

$$\mathcal{R}_i : \begin{cases} \frac{1}{\sqrt{m}} \sum_{x_i\in\Sigma} |x_i\rangle & \mapsto \quad |\varnothing\rangle, \\ \frac{1}{\sqrt{m}} \sum_{x_i\in\Sigma} \omega^{bx_i} |x_i\rangle & \mapsto \quad \frac{1}{\sqrt{m}} \sum_{x_i\in\Sigma} \omega^{bx_i} |x_i\rangle \quad \text{if } b \in \{1,\ldots,m-1\}, \\ |\varnothing\rangle & \mapsto \quad \frac{1}{\sqrt{m}} \sum_{x_i\in\Sigma} |x_i\rangle. \end{cases}$$

Given the joint state $|\psi^t\rangle$ of an algorithm and input distribution μ_{unif} (Definition 35), we define the *joint state* $|\psi^t_{\mathrm{rec}}\rangle$ *of the algorithm and record* as,

$$|\psi^t_{\mathrm{rec}}\rangle = (\mathrm{Id} \otimes (\mathcal{R}_1 \otimes \cdots \otimes \mathcal{R}_n))|\psi^t\rangle = (\mathrm{Id} \otimes \mathcal{R})|\psi^t\rangle$$

where $\mathcal{R} = \mathcal{R}_1 \otimes \cdots \otimes \mathcal{R}_n$ is the *recording operator* acting on the record space.

As a first simple observation, the size of the quantum record (i.e., the maximum number of non-$\varnothing$ entries in the basis states over which it is supported) cannot be larger than the number of quantum queries made so far.

Fact 37 (Record size)**.** *The state* $|\psi^t_{\mathrm{rec}}\rangle = (\mathrm{Id} \otimes \mathcal{R})|\psi^t\rangle$ *after* t *queries is supported only over basis states* $|i,b\rangle \otimes |x_1,\ldots,x_n\rangle$ *whose record size is at most* $|\{j : x_j \neq \varnothing\}| \leq t$*.*

Proof. By tracking the action of the phase oracle $\mathcal{O}^{\pm}$ during t queries, one can see that the joint state of the algorithm and input admits a decomposition of the form,

$$|\psi^t\rangle = \sum_{i\in\{1,\ldots,n\}, b\in\Sigma, c\in\Sigma^n} \alpha_{i,b,c} |i,b\rangle \otimes \sum_{x\in\Sigma^n} \omega^{c_1x_1+\cdots+c_nx_n} |x\rangle$$

for some complex coefficients $\alpha_{i,b,c}$ that can be non-zero only when $|\{j : c_j \neq 0\}| \leq t$. For a given c, the state $\sum_{x\in\Sigma^n} \omega^{c_1x_1+\cdots+c_nx_n}|x\rangle = \left(\sum_{x_1\in\Sigma} \omega^{c_1x_1}|x_1\rangle\right) \otimes \cdots \otimes \left(\sum_{x_n\in\Sigma} \omega^{c_nx_n}|x_n\rangle\right)$ is mapped by the recording operator $\mathcal{R}$ to the product state $|R_1\rangle \otimes \cdots \otimes |R_n\rangle$ where $|R_j\rangle = \sqrt{m}|\varnothing\rangle$ if $c_j = 0$, and $|R_j\rangle = \sum_{x_j\in\Sigma} \omega^{c_jx_j}|x_j\rangle$ otherwise. Hence, $|\psi^t_{\mathrm{rec}}\rangle = (\mathrm{Id} \otimes \mathcal{R})|\psi^t\rangle$ is supported over basis states that have at most t coordinates different from $\varnothing$. □

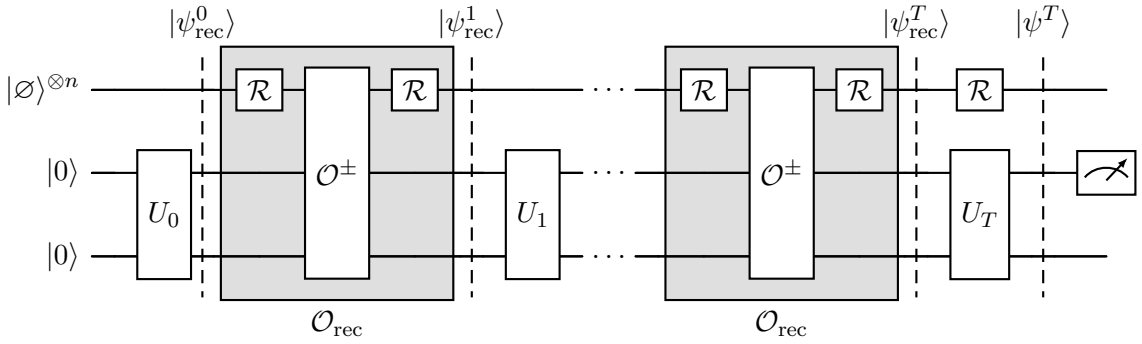

Figure 5: Canonical form of a (memoryless) quantum query algorithm with recording oracle.

The quantum record can behave very differently from its classical counterpart. For instance, its size can decrease over time if the algorithm uses query parameters (i, b) that make the i-th subsystem return to the uniform superposition. This phenomenon is unavoidable to preserve the reversibility of the computation. The purpose of the next section is to better understand how the record evolves after each query.

3. Recording oracle. It is not clear yet that the quantum record has any meaningful properties beyond that of Fact 37. We provide a different viewpoint on the construction of the quantum record, which better explains its evolution during the computation. To that end, we define a third query operator – the *recording oracle* $\mathcal{O}_{\text{rec}}$ – that conjugates the phase oracle $\mathcal{O}^{\pm}$ with the recording operator $\text{Id} \otimes \mathcal{R}$. Unlike the binary and phase oracles, the recording oracle depends on the chosen input distribution encoded into $\mathcal{R}$.

Definition 38 (Recording oracle)**.** The *recording oracle* for the input distribution μ_{unif} is the unitary operator acting on the joint state of the algorithm and record as follows,

$$\mathcal{O}_{\text{rec}} = (\text{Id} \otimes \mathcal{R})\mathcal{O}^{\pm}(\text{Id} \otimes \mathcal{R}),$$

where the phase oracle is extended to the record space as $\mathcal{O}^{\pm} = \sum_{x \in (\Sigma \cup \{\varnothing\})^n} \mathcal{O}^{\pm}_x \otimes |x\rangle\langle x|$ with $\mathcal{O}^{\pm}_x |i, b\rangle = \omega^{bx_i} |i, b\rangle$ when $x_i \in \Sigma$, and $\mathcal{O}^{\pm}_x |i, b\rangle = |i, b\rangle$ when $x_i = \varnothing$.

The need to specify the behavior of the phase oracle on $x_i = \varnothing$ is an artifact of the construction. This case will never occur in practice, but it makes the analysis simpler.

The recording oracle $\mathcal{O}_{\text{rec}}$ has two central properties that are at the core of the recording method. First, the joint state $|\psi^t_{\text{rec}}\rangle$ can equally be viewed as the state obtained by replacing the phase oracle with the recording oracle in the algorithm. This is represented in Figure 5. Second, the evolution of the record state upon querying $\mathcal{O}_{\text{rec}}$ follows *almost* the same rules as a classical record: if a coordinate is not in the record prior to the query then a uniform superposition is recorded; and if it is already in the record, the state remains (almost) unchanged.

Theorem 39 (Recording method)**.** *Consider a quantum algorithm that accesses a random input drawn from the uniform distribution μ_{unif}. Let $|\psi^t\rangle$ denote the joint state of the algorithm and input after t queries (Definition 35). Then,*

- *(Indistinguishability) The joint state $|\psi^t_{\text{rec}}\rangle = (\text{Id} \otimes \mathcal{R})|\psi^t\rangle$ of the algorithm and record is also equal to $|\psi^t_{\text{rec}}\rangle = (U_t \otimes \text{Id})\mathcal{O}_{\text{rec}}(U_{t-1} \otimes \text{Id})\mathcal{O}_{\text{rec}}(U_0 \otimes \text{Id})(|0, 0\rangle \otimes |\varnothing, \ldots, \varnothing\rangle)$.*
- *(Recording action) The action of the recording oracle $\mathcal{O}_{\text{rec}}$ on a basis state $|i, b\rangle \otimes |x_1, \ldots, x_n\rangle$ where $i \in \{1, \ldots, n\}$, $b \in \Sigma \setminus \{0\}$ and $x \in (\Sigma \cup \{\varnothing\})^m$ is given by the equations,*

$$(x_i = \varnothing) \quad \mathcal{O}_{\mathrm{rec}}|i,b\rangle \otimes |\dots x_{i-1}, \varnothing, x_{i+1}\dots\rangle = |i,b\rangle \otimes |\dots x_{i-1}\rangle \Big(\frac{1}{\sqrt{m}} \sum_{x'_i \in \Sigma} \omega^{bx'_i} |x'_i\rangle\Big) |x_{i+1}\dots\rangle,$$

$$(x_i \in \Sigma) \quad \mathcal{O}_{\mathrm{rec}}|i,b\rangle \otimes |\dots x_{i-1}, x_i, x_{i+1}\dots\rangle = |i,b\rangle \otimes |\dots x_{i-1}\rangle \big(\omega^{bx_i}|x_i\rangle + |\mathrm{error}_{x_i}\rangle\big) |x_{i+1}\dots\rangle,$$

$$\textit{where } |\mathrm{error}_{x_i}\rangle = \tfrac{\omega^{bx_i}}{\sqrt{m}}|\varnothing\rangle + \sum_{x'_i \in \Sigma} \tfrac{1-\omega^{bx_i}-\omega^{bx'_i}}{m} |x'_i\rangle.$$

If $b = 0$ then $\mathcal{O}_{\mathrm{rec}}$ makes no change to the state $|i,b\rangle \otimes |x_1, \dots, x_n\rangle$.

Proof of the indistinguishability. The proof is by induction on t. The base case ($t = 0$) uses that U_0 and $\mathcal{R}$ are acting on different registers, hence their order can be inverted: $|\psi^0_{\mathrm{rec}}\rangle = (\mathrm{Id} \otimes \mathcal{R})|\psi^0\rangle = (\mathrm{Id} \otimes \mathcal{R})(U_0 \otimes \mathrm{Id})\big(|0,0\rangle \otimes \frac{1}{m^{n/2}} \sum_{x \in \Sigma^n} |x\rangle\big) = (U_0 \otimes \mathrm{Id})(\mathrm{Id} \otimes \mathcal{R})\big(|0,0\rangle \otimes \frac{1}{m^{n/2}} \sum_{x \in \Sigma^n} |x\rangle\big) = (U_0 \otimes \mathrm{Id})(|0,0\rangle \otimes |\varnothing, \dots, \varnothing\rangle)$. The induction step applies the same argument to U_{t+1} and uses the fact that $\mathcal{R}$ squares to the identity: $|\psi^{t+1}_{\mathrm{rec}}\rangle = (\mathrm{Id} \otimes \mathcal{R})(U_{t+1} \otimes \mathrm{Id})\mathcal{O}^{\pm}|\psi^t\rangle = (U_{t+1} \otimes \mathrm{Id})(\mathrm{Id} \otimes \mathcal{R})\mathcal{O}^{\pm}|\psi^t\rangle = (U_{t+1} \otimes \mathrm{Id})(\mathrm{Id} \otimes \mathcal{R})\mathcal{O}^{\pm}(\mathrm{Id} \otimes \mathcal{R})(\mathrm{Id} \otimes \mathcal{R})|\psi^t\rangle = (U_{t+1} \otimes \mathrm{Id})\mathcal{O}_{\mathrm{rec}}|\psi^t_{\mathrm{rec}}\rangle$. □

Proof of the recording action. We assume $b \neq 0$, as it is easy to see that $\mathcal{O}_{\mathrm{rec}}$ acts as the identity otherwise. We decompose the action of $\mathcal{O}_{\mathrm{rec}} = (\mathrm{Id} \otimes \mathcal{R})\mathcal{O}^{\pm}(\mathrm{Id} \otimes \mathcal{R})$ when $x_i = \varnothing$,

$$\begin{aligned}
|i,b\rangle \otimes |\dots x_{i-1}, \varnothing, x_{i+1}\dots\rangle &\xmapsto{\mathrm{Id}\otimes\mathcal{R}} |i,b\rangle \otimes (\cdots \otimes \mathcal{R}_{i-1}|x_{i-1}\rangle)\Big(\frac{1}{\sqrt{m}} \sum_{x'_i \in \Sigma} |x'_i\rangle\Big)(\mathcal{R}_{i+1}|x_{i+1}\rangle \otimes \dots) \\
&\xmapsto{\mathcal{O}^{\pm}} |i,b\rangle \otimes (\cdots \otimes \mathcal{R}_{i-1}|x_{i-1}\rangle)\Big(\frac{1}{\sqrt{m}} \sum_{x'_i \in \Sigma} \omega^{bx'_i}|x'_i\rangle\Big)(\mathcal{R}_{i+1}|x_{i+1}\rangle \otimes \dots) \\
&\xmapsto{\mathrm{Id}\otimes\mathcal{R}} |i,b\rangle \otimes |\dots x_{i-1}\rangle\Big(\frac{1}{\sqrt{m}} \sum_{x'_i \in \Sigma} \omega^{bx'_i}|x'_i\rangle\Big)|x_{i+1}\dots\rangle.
\end{aligned}$$

The last step uses that $\mathcal{R}_j^2 = \mathrm{Id}$ for all j, hence all registers are restored to their initial basis state, except the i-th record register.

We now consider the case of $x_i \in \Sigma$. For the ease of notation, we only track the evolution of the i-th record register (the other registers do not change, for the same reasons as above),

$$\begin{aligned}
|x_i\rangle &\xmapsto{\mathcal{R}_i} |x_i\rangle + \frac{1}{\sqrt{m}}|\varnothing\rangle - \frac{1}{n} \sum_{x'_i \in \Sigma} |x'_i\rangle \\
&\xmapsto{\mathcal{O}^{\pm}} \omega^{bx_i}|x_i\rangle + \frac{1}{\sqrt{m}}|\varnothing\rangle - \frac{1}{n} \sum_{x'_i \in \Sigma} \omega^{bx'_i}|x'_i\rangle \\
&\xmapsto{\mathcal{R}_i} \omega^{bx_i}|x_i\rangle + \frac{\omega^{bx_i}}{\sqrt{m}}|\varnothing\rangle - \frac{1}{m} \sum_{x'_i \in \Sigma} \omega^{bx_i}|x'_i\rangle + \frac{1}{m} \sum_{x'_i \in \Sigma} |x'_i\rangle - \frac{1}{m} \sum_{x'_i \in \Sigma} \omega^{bx'_i}|x'_i\rangle.
\end{aligned}$$

The first step has been obtained by rewriting the action of the recording operator (Definition 36) in the standard basis. □

4.2 Applications

We illustrate the recording method on two problems that are ubiquitous in cryptography: finding a preimage or a collision in a random (hash) function. Beyond establishing the query complexity of these problems, the recording method will give very tight upper bounds on the best average success probability that can be achieved with a given number of quantum queries. This refinement is useful, for instance, when choosing the security parameters (e.g., key length) of cryptographic schemes based on hash functions.

Application 1: The Search problem. This problem is a variant of OR, where instead of deciding if the input contains a particular value (e.g., the bit 1), the task is to find such a value (when it exists). We will be studying the hardness of this problem under the uniform input distribution.

Definition 40 (Search)**.** The Search problem is to find an index $i \in \{1, \dots, n\}$ such that $x_i = 1$ in an input $x \in \{0, \dots, m-1\}^n$.

The lower bound will be given as a function of the alphabet size m. The input size n appears indirectly in the result, as it constrains the values of m for which the problem is non-trivial under the uniform distribution (for instance, if $m \gg n$ then a random input has no solution at all with high probability). If it helps the reader, one can fix $m = n/2$ since it guarantees the existence of two solutions in expectation, and the probability of having no solution is small $(1 - 2/n)^n \leq e^{-2} \leq 1/7$ (this event can be ignored as long as its probability is below the allowed failure probability, e.g. 1/3).

It is straightforward to argue that classical algorithms need $\Omega(m)$ queries to solve the Search problem. We provide a detailed proof that will help understanding the quantum case afterward.

Proposition 41 (Classical recording method applied to Search)**.** *The average-case randomized complexity of the* Search *problem under the uniform distribution is at least* $R_{\mu_{\text{unif}}}(\text{Search}) \geq 2m/3 - 1$.

Proof. We measure the progress of an algorithm by the probability of the event E_t: "the algorithm queries at least one coordinate equals to $x_i = 1$ during its first t queries". The probability that the $t+1$-th query triggers the event is at most $\Pr[E_{t+1} \,|\, \overline{E_t}] \leq 1/m$, irrespective of the behavior of the algorithm (since the x_i's are independent, identically distributed random variables). Hence, the progress behaves as follows,

- (Initial condition) $\Pr[E_0] = 0$,
- (Progress evolution) $\Pr[E_{t+1}] = \Pr[E_t] + \Pr[E_{t+1} \mid \overline{E_t}] \cdot \Pr[\overline{E_t}] \leq \Pr[E_t] + 1/m$.

It remains to relate the average success probability $p_{\text{succ}}^{\mu_{\text{unif}}}$ of an algorithm making T queries to its final progress $\Pr[E_T]$. If the algorithm never queried a coordinate equal to 1 (i.e., the event E_T does not happen), then it is left to guess randomly where such a coordinate can be. This is somewhat the same as the event E_{T+1} given $\overline{E_T}$, except that the algorithm cannot see the result of the last query.

- (Final condition) $p_{\text{succ}}^{\mu_{\text{unif}}} \leq \Pr[E_T] + 1/m$.

By combining the three bullet points together, we obtain that the best possible success probability after T queries must be at most $p_{\text{succ}}^{\mu_{\text{unif}}} \leq (T+1)/m$. In particular, succeeding with average probability at least $p_{\text{succ}}^{\mu_{\text{unif}}} \geq 2/3$ requires making at least $T \geq 2m/3 - 1$ queries. □

We are now going to mimic the above proof in the quantum query model, using the recording method formalism. The main challenge is to adapt the progress measure when the query record (hence, the event E_t) is no longer properly defined as a random variable. The solution is to measure the progress as the *total amplitude* (norm) of the part of the quantum record that contains a solution. The quadratic decrease in the lower bound can be traced back to manipulating amplitudes in the proof, rather than probabilities (squared norm).

Proposition 42 (Quantum recording method applied to Search)**.** *The average-case quantum complexity of the* Search *problem under the uniform distribution is at least* $Q_{\mu_{\text{unif}}}(\text{Search}) \geq \sqrt{m/15} - \sqrt{1/5}$.

Proof. Consider any quantum algorithm that solves the SEARCH problem using some number T of queries. Define the progress measure Δ_t after $t \in \{0, \dots, T\}$ queries as the norm of the state obtained by projecting $|\psi^t_{\mathrm{rec}}\rangle$ (Definition 36) onto the records containing at least one coordinate equals to $x_i = 1$,

$$\Delta_t = \|\Pi|\psi^t_{\mathrm{rec}}\rangle\| \quad \text{where} \quad \Pi = \mathrm{Id} \otimes \sum_{\substack{x \in (\Sigma \cup \{\varnothing\})^n, \\ \exists i, x_i = 1}} |x\rangle\langle x|.$$

We show that the progress Δ_t and success probability $p^{\mu_{\mathrm{unif}}}_{\mathrm{succ}}$ of the algorithm obey the following inequalities,

- (Initial condition) $\Delta_0 = 0$,
- (Progress evolution) $\Delta_{t+1} \leq \Delta_t + \sqrt{10/m}$,
- (Final condition) $p^{\mu_{\mathrm{unif}}}_{\mathrm{succ}} \leq (\Delta_T + \sqrt{2/m})^2$.

It follows immediately that $p^{\mu_{\mathrm{unif}}}_{\mathrm{succ}} \leq (\sqrt{10}T + \sqrt{2})^2/m$. Hence, succeeding with probability at least $p^{\mu_{\mathrm{unif}}}_{\mathrm{succ}} \geq 2/3$ requires making at least $T \geq \sqrt{m/15} - \sqrt{1/5}$ quantum queries.

Proof of the initial condition. The record is initially empty, $|\psi^0_{\mathrm{rec}}\rangle = (U_0|0,0\rangle) \otimes |\varnothing\rangle^{\otimes n}$, hence the progress starts at $\Delta_0 = \|\Pi|\psi^0_{\mathrm{rec}}\rangle\| = 0$.

Proof of the progress evolution. We first prove an analogous statement to the equality $\Pr[E_{t+1}] = \Pr[E_t] + \Pr[E_{t+1}, \overline{E_t}]$ used in the classical setting. We substitute the use of the law of total probability with the triangle inequality. The progress increases at most by the norm of the part of the state recording a value $x_i = 1$ for the first time.

$$\begin{aligned}
\Delta_{t+1} &= \|\Pi(U_{t+1} \otimes \mathrm{Id})\mathcal{O}_{\mathrm{rec}}|\psi^t_{\mathrm{rec}}\rangle\| && \text{(definition of } |\psi^{t+1}_{\mathrm{rec}}\rangle\text{)} \\
&= \|(U_{t+1} \otimes \mathrm{Id})\Pi\mathcal{O}_{\mathrm{rec}}|\psi^t_{\mathrm{rec}}\rangle\| && (\Pi \text{ and } U_{t+1} \otimes \mathrm{Id} \text{ commute}) \\
&= \|\Pi\mathcal{O}_{\mathrm{rec}}|\psi^t_{\mathrm{rec}}\rangle\| && \text{(unitary invariance of the norm)} \\
&\leq \|\Pi\mathcal{O}_{\mathrm{rec}}\Pi|\psi^t_{\mathrm{rec}}\rangle\| + \|\Pi\mathcal{O}_{\mathrm{rec}}(\mathrm{Id} - \Pi)|\psi^t_{\mathrm{rec}}\rangle\| && \text{(triangle inequality)} \\
&\leq \|\Pi|\psi^t_{\mathrm{rec}}\rangle\| + \|\Pi\mathcal{O}_{\mathrm{rec}}(\mathrm{Id} - \Pi)|\psi^t_{\mathrm{rec}}\rangle\| && \text{(submultiplicativity of the norm)} \\
&= \Delta_t + \|\Pi\mathcal{O}_{\mathrm{rec}}(\mathrm{Id} - \Pi)|\psi^t_{\mathrm{rec}}\rangle\|.
\end{aligned}$$

Next, we show that $\|\Pi\mathcal{O}_{\mathrm{rec}}(\mathrm{Id}-\Pi)|\psi^t_{\mathrm{rec}}\rangle\| \leq \sqrt{10/m}\|(\mathrm{Id}-\Pi)|\psi^t_{\mathrm{rec}}\rangle\|$, in analogy to the statement $\Pr[E_{t+1}, \overline{E_t}] = \Pr[E_{t+1} \mid \overline{E_t}] \cdot \Pr[\overline{E_t}] \leq 1/m \Pr[\overline{E_t}]$. The proof is carried out in greater generality, replacing $(\mathrm{Id} - \Pi)|\psi^t_{\mathrm{rec}}\rangle$ with any state $|\psi\rangle = \sum_{i,b,x} \alpha_{i,b,x}|i,b\rangle \otimes |x\rangle$ in the support of $\mathrm{Id} - \Pi$ (i.e., no record in the support of $|\psi\rangle$ shall contain the value 1). We decompose such a state into $n+2$ mutually orthogonal components $|\psi\rangle = |\psi_{\mathrm{id}}\rangle + |\psi_{\varnothing}\rangle + \sum_{y \in \Sigma}|\psi_y\rangle$ defined as follows:

- $|\psi_{\mathrm{id}}\rangle = \sum_{i,b,x:b=0} \alpha_{i,b,x}|i,b\rangle \otimes |x\rangle$ (null query),
- $|\psi_{\varnothing}\rangle = \sum_{i,b,x:x_i=\varnothing, b\neq 0} \alpha_{i,b,x}|i,b\rangle \otimes |x\rangle$ (non-null query, empty record),
- $|\psi_y\rangle = \sum_{i,b,x:x_i=y, b\neq 0} \alpha_{i,b,x}|i,b\rangle \otimes |x\rangle$ (non-null query, nonempty record).

The component $|\psi_1\rangle$ is zero by definition of $\mathrm{Id} - \Pi$. We analyse how much the norms of the other components decrease after applying $\Pi\mathcal{O}_{\mathrm{rec}}$. The action of the oracle $\mathcal{O}_{\mathrm{rec}}$ on a basis state is dictated by Theorem 39. Notice that the only way for these states to be in the support of Π after applying $\mathcal{O}_{\mathrm{rec}}$ is to record $x_i = 1$ at the position i indicated by the index register, since the rest of the record stays unchanged. The norms are:

- $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_{\mathrm{id}}\rangle\| = 0$: The state becomes zero since the record does not change when the value register holds a zero.

– $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle\| = \frac{1}{\sqrt{m}}\||\psi_\varnothing\rangle\|$: The state is equal to $\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle = \sum_{x_i=\varnothing,b\neq 0} \alpha_{i,b,x}\frac{\omega^b}{\sqrt{m}}|i,b\rangle \otimes |x^{\{i\}}\rangle$ with $x_i^{\{i\}} = 1$ and $x_j^{\{i\}} = x_j$ if $j \neq i$. Thus, $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle\|^2 = \sum_{x_i=\varnothing,b\neq 0}\frac{|\alpha_{i,b,x}|^2}{m} = \frac{1}{m}\||\psi_\varnothing\rangle\|^2$.

– $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_y\rangle\| \leq \frac{3}{m}\||\psi_y\rangle\|$: The state is equal to $\Pi\mathcal{O}_{\mathrm{rec}}|\psi_y\rangle = \sum_{x_i=y,b\neq 0}\alpha_{i,b,x}\frac{1-\omega^b-\omega^{by}}{m}|i,b\rangle\otimes |x^{\{i\}}\rangle$. Thus, $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_y\rangle\|^2 = \sum_{x_i=y,b\neq 0}\frac{|(1-\omega^b-\omega^{by})\alpha_{i,b,x}|^2}{m^2} \leq \frac{9}{m^2}\||\psi_y\rangle\|^2$.

Finally, using the triangle and Cauchy–Schwarz inequalities, we conclude that the norm of the overall state is $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi\rangle\| \leq \|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle\| + \sum_{y\in\Sigma\setminus\{1\}}\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_y\rangle\| \leq \frac{1}{\sqrt{m}}\||\psi_\varnothing\rangle\| + \frac{3}{m}\sum_{y\in\Sigma\setminus\{1\}}\||\psi_y\rangle\| \leq \sqrt{\frac{10}{m}}\||\psi\rangle\|$.

Proof of the final condition. The average success probability is defined as $p_{\mathrm{succ}}^{\mu_{\mathrm{unif}}} = \|\Pi_{\mathrm{succ}}|\psi^T\rangle\|^2$ where Π_{succ} is the projector onto the states $|i,b\rangle\otimes|x\rangle$ with $x_i = 1$. Using the relationship between the joint states $|\psi^T\rangle$ and $|\psi_{\mathrm{rec}}^T\rangle$ (Definition 36), we have

$$\begin{aligned}
p_{\mathrm{succ}}^{\mu_{\mathrm{unif}}} &= \|\Pi_{\mathrm{succ}}(\mathrm{Id}\otimes\mathcal{R})|\psi_{\mathrm{rec}}^T\rangle\|^2 && \text{(since } |\psi^T\rangle = (\mathrm{Id}\otimes\mathcal{R})|\psi_{\mathrm{rec}}^T\rangle\text{)}\\
&\leq \left(\|\Pi_{\mathrm{succ}}(\mathrm{Id}\otimes\mathcal{R})\Pi|\psi_{\mathrm{rec}}^T\rangle\| + \|\Pi_{\mathrm{succ}}(\mathrm{Id}\otimes\mathcal{R})(\mathrm{Id}-\Pi)|\psi_{\mathrm{rec}}^T\rangle\|\right)^2 && \text{(triangle inequality)}\\
&\leq \left(\|\Pi|\psi_{\mathrm{rec}}^T\rangle\| + \|\Pi_{\mathrm{succ}}(\mathrm{Id}\otimes\mathcal{R})(\mathrm{Id}-\Pi)|\psi_{\mathrm{rec}}^T\rangle\|\right)^2 && \text{(submultiplicativity of the norm)}\\
&= \left(\Delta_T + \|\Pi(\mathrm{Id}\otimes\mathcal{R})(\mathrm{Id}-\Pi)|\psi_{\mathrm{rec}}^T\rangle\|\right)^2.
\end{aligned}$$

The analysis of $\|\Pi(\mathrm{Id}\otimes\mathcal{R})(\mathrm{Id}-\Pi)|\psi_{\mathrm{rec}}^T\rangle\|$ follows that of $\|\Pi\mathcal{O}_{\mathrm{rec}}(\mathrm{Id}-\Pi)|\psi_{\mathrm{rec}}^T\rangle\|$ done previously. Using the same notation as before $|\psi\rangle = |\psi_{\mathrm{id}}\rangle + |\psi_\varnothing\rangle + \sum_{y\in\Sigma\setminus\{1\}}|\psi_y\rangle$ for a state in the support of $\mathrm{Id}-\Pi$, we have $\|\Pi(\mathrm{Id}\otimes\mathcal{R})|\psi_{\mathrm{id}}\rangle\| = 0$, $\|\Pi(\mathrm{Id}\otimes\mathcal{R})|\psi_\varnothing\rangle\| = \frac{1}{\sqrt{m}}\||\psi_\varnothing\rangle\|$ and $\|\Pi(\mathrm{Id}\otimes\mathcal{R})|\psi_y\rangle\| = \frac{1}{m}\||\psi_y\rangle\|$ (the action of $\mathcal{R}$ in the standard basis is given, for instance, in the proof of Theorem 39). We conclude that $\|\Pi(\mathrm{Id}\otimes\mathcal{R})|\psi\rangle\| \leq \sqrt{\frac{2}{m}}\||\psi\rangle\|$. □

Application 2: The Collision problem. We sketch a second application of the recording method for the Collision problem, defined as follows.

Definition 43 (Collision)**.** The Collision problem is to find two distinct indices $i \neq j \in \{1,\ldots,n\}$ such that $x_i = x_j$ in an input $x \in \{0,\ldots,m-1\}^n$.

Again, there is a range of parameters n, m for which the problem is relevant under the uniform distribution. For instance, if $m = n$ then there are $\frac{1}{m}\binom{n}{2} = (n-1)/2$ collision pairs in expectation, one of which can be found using $O(\sqrt{m})$ classical queries (birthday attack) or $O(m^{1/3})$ quantum queries (BHT [BHT98] or Ambainis [Amb07] algorithms).

The classical complexity of the Collision problem can be understood using the event E_t: "the algorithm queries two coordinates with equal values $x_i = x_j$ during its first t queries". The progress evolves as $\Pr[E_{t+1}] = \Pr[E_t] + t/m$ since the probability that the $t+1$-th query returns one of the t values observed before (i.e., produces a collision) is at most t/m. This yields that the progress after T queries is $\Delta_T = 1/m + 2/m + \cdots + (T-1)/m = O(T^2/m)$, hence requiring $T = \Omega(\sqrt{m})$ queries to make it sufficiently large. We can transpose this proof to the quantum setting using the following progress measure.

Proposition 44 (Quantum recording progress for Collision)**.** *Consider any quantum algorithm with access to a random input drawn from the uniform distribution μ_{unif}. Let $|\psi_{\mathrm{rec}}^t\rangle$ denote the joint state of the algorithm and record after t queries to the input. Define the progress measure Δ_t as the norm of the state obtained by projecting $|\psi_{\mathrm{rec}}^t\rangle$ onto the records containing at least two equal coordinates $x_i = x_j$,*

$$\Delta_t = \|\Pi|\psi_{\mathrm{rec}}^t\rangle\| \quad \textit{where} \quad \Pi = \mathrm{Id}\otimes \sum_{\substack{x\in(\Sigma\cup\{\varnothing\})^n,\\ \exists i\neq j, x_i=x_j\neq\varnothing}} |x\rangle\langle x|.$$

Then the progress Δ_t obeys the inequality $\Delta_{t+1} \leq \Delta_t + \sqrt{\frac{10t}{m}}$.

Proof. Using the same argument as in the proof of Proposition 42, the progress increases after each query by at most $\Delta_{t+1} \leq \Delta_t + \|\Pi\mathcal{O}_{\mathrm{rec}}(\mathrm{Id} - \Pi)|\psi^t_{\mathrm{rec}}\rangle\|$. Let $|\psi\rangle = \sum_{i,b,x} \alpha_{i,b,x}|i,b\rangle \otimes |x\rangle$ denote any state in the support of $\mathrm{Id} - \Pi$ *and* with records of size at most $|\{j : x_j \neq \varnothing\}| \leq t$. Notice that the latter condition is satisfied by the state $(\mathrm{Id} - \Pi)|\psi^t_{\mathrm{rec}}\rangle$ according to Fact 37. Hence, it suffices to show that $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi\rangle\| \leq \sqrt{\frac{10t}{m}}\||\psi\rangle\|$.

Let $|\psi\rangle = |\psi_{\mathrm{id}}\rangle + |\psi_\varnothing\rangle + \sum_{y\in\Sigma}|\psi_y\rangle$ be the decomposition of $|\psi\rangle$ as defined in the proof of Proposition 42. We claim that $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_{\mathrm{id}}\rangle\| = 0$, $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle\| \leq \sqrt{\frac{t}{m}}\||\psi_\varnothing\rangle\|$ and $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_y\rangle\| \leq \frac{3t}{m}\||\psi_y\rangle\|$. The first statement is immediate since $\mathcal{O}_{\mathrm{rec}}|\psi_{\mathrm{id}}\rangle = |\psi_{\mathrm{id}}\rangle$. Let us detail the second statement (the last one is similar). By Theorem 39, the state evolves into

$$\begin{aligned}\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle &= \Pi \sum_{i,b,x:x_i=\varnothing,b\neq 0} \alpha_{i,b,x}|i,b\rangle \otimes |\dots x_{i-1}\rangle\Big(\frac{1}{\sqrt{m}}\sum_{x'_i\in\Sigma}\omega^{bx'_i}|x'_i\rangle\Big)|x_{i+1}\dots\rangle \\ &= \sum_{i,b,x:x_i=\varnothing,b\neq 0} \alpha_{i,b,x}|i,b\rangle \otimes |\dots x_{i-1}\rangle\Big(\frac{1}{\sqrt{m}}\sum_{\substack{x'_i\in\Sigma,\\ \exists j\neq i, x_j=x'_i}}\omega^{bx'_i}|x'_i\rangle\Big)|x_{i+1}\dots\rangle.\end{aligned}$$

The norm is at most $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi_\varnothing\rangle\|^2 = \sum_{i,b,x:x_i=\varnothing,b\neq 0}|\alpha_{i,b,x}|^2\frac{|\{x'_i\in\Sigma:\exists j,x_j=x'_i\}|}{m} \leq \frac{t}{m}\||\psi_\varnothing\rangle\|^2$ since the records with non-zero amplitudes $\alpha_{i,b,x} \neq 0$ are of size at most t by assumption. Finally, using the triangle and Cauchy–Schwarz inequalities, we have that $\|\Pi\mathcal{O}_{\mathrm{rec}}|\psi\rangle\| \leq \sqrt{\frac{10t}{m}}\||\psi\rangle\|$. □

We leave it to the reader to prove that any algorithm solving Collision using T queries must satisfy the final condition $p^{\mu_{\mathrm{unif}}}_{\mathrm{succ}} = (\Delta_T + O(\sqrt{T/m}))^2 = O(T^3/m)$ (our definition of the computational model should be slightly adapted to allow for the output of two indices). This entails that the average-case quantum complexity must be at least $Q_{\mu_{\mathrm{unif}}}(\textsc{Collision}) = \Omega(m^{1/3})$, which is optimal since it matches the complexity of the existing quantum algorithms [BHT98; Amb07].

5 The Adversary Method

The *adversary method* is arguably the most popular and versatile technique for proving quantum query lower bounds. It comes in many flavors, the simplest of which is the hybrid method presented in Section 2. The most evolved versions have virtually no limits, as they can always provide the optimal complexity (the catch being the difficulty in applying such methods to concrete problems). In this section, we will present the modern formulation of the adversary method, based on the spectral properties of an adversially chosen matrix with real-weight entries [HLŠ07].

5.1 Technique

Our presentation of the adversary method extends the approaches introduced in the hybrid and recording methods. Unlike in the previous section, we revert to the model with a Boolean input alphabet $x \in \{0,1\}^n$, Boolean decision problems $f : \{0,1\}^n \to \{0,1\}$, and a joint binary oracle $\mathcal{O}(|i,b\rangle \otimes |x\rangle) = |i, b \oplus x_i\rangle \otimes |x\rangle$.

First, the adversary method introduces the possibility of assigning weights to the pairs of inputs considered in the hybrid method, as follows.

Weighted Gram matrix. Recall that the inner product $\langle \psi_x^T \,|\, \psi_y^T \rangle$ between two final states with $f(x) \neq f(y)$ relates to the probability with which the algorithm can be correct (final condition in Theorem 10). The adversary method exploits the entire Gram matrix $G_t = (\langle \psi_x^t \,|\, \psi_y^t \rangle)_{x,y \in \{0,1\}^n}$ and assigns weights $\Gamma_{x,y}$ to its entries (a somewhat unintuitive aspect of the method is that it allows for negative weights as well). The inner products $\langle \psi_x^t \,|\, \psi_y^t \rangle$ evolve as queries are made to indices i with $x_i \neq y_i$ (progress evolution in Theorem 10). The adversary method quantifies this evolution using the "punctured" weights $(\Gamma_i)_{x,y}$ obtained by zeroing out all entries of Γ with $x_i = y_i$. The constraints on the resulting matrices are summarized in the following definition.

Definition 45 (Adversary matrix)**.** Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function over n variables. We say that $\Gamma \in \mathbb{R}^{2^n \times 2^n}$ is an *adversary matrix* for f if it satisfies the two conditions,

- (Symmetric) $\Gamma_{x,y} = \Gamma_{y,x}$ for all $x, y \in \{0,1\}^n$,
- (f-sparse) $\Gamma_{x,y} = 0$ when $f(x) = f(y)$.

For all $i \in \{1, \ldots, n\}$, we define the *i-th punctured adversary matrix* $\Gamma_i \in \mathbb{R}^{2^n \times 2^n}$ of Γ as the matrix satisfying also the condition,

- (Punctured) $(\Gamma_i)_{x,y} = 0$ when $x_i = y_i$, and $(\Gamma_i)_{x,y} = \Gamma_{x,y}$ otherwise.

While the adversary matrix provides a way to emphasize pairs of inputs that are difficult to distinguish, another source of hardness can be introduced by placing a distribution on the input, as was done in the recording method. This is achieved below by extending the purification technique introduced in the last section.

Generalized input purification. We described in Definition 35 how the state of an algorithm operating under an input distribution μ can be represented as the bipartite state $|\psi^t\rangle = \sum_{x \in \{0,1\}^n} \sqrt{\mu(x)} |\psi_x^t\rangle \otimes |x\rangle$. The amplitude $\sqrt{\mu(x)}$ can, in fact, be replaced with any complex number a_x satisfying $|a_x|^2 = \mu(x)$, since the reduced density matrix of the algorithm remains equal to $\rho^t = \sum_{x \in \Sigma^n} \mu(x) |\psi_x^t\rangle\langle\psi_x^t|$. In the adversary method, the choice of these amplitudes determines the initial value Δ_0 of the progress measure, which we aim to minimize in order to allow for a longer progress evolution. The progress is quantified by the expression

$$-|\langle \psi^t \,|\, (\mathrm{Id} \otimes \Gamma) \,|\, \psi^t \rangle| = -\Big| \sum_{x,y} \Gamma_{x,y} a_x a_y^* \langle \psi_x^t \,|\, \psi_y^t \rangle \Big|,$$

which is minimized at $t = 0$ when $(a_x)_x$ is chosen as the principal eigenvector of the adversary matrix Γ. With this choice, and by applying an offset of $\|\Gamma\|$ to initialize the progress at 0, the adversary method is formulated as follows.

Theorem 46 (Adversary method)**.** *Consider a quantum algorithm that accesses a random input $x \in \{0,1\}^n$ drawn from a distribution μ. Let $a \in \mathbb{C}^{2^n}$ be a unit vector such that $\mu(x) = |a_x|^2$ for all x. Define the joint state of the algorithm and input after t queries as,*

$$|\psi^t\rangle = \sum_{x \in \{0,1\}^n} a_x |\psi_x^t\rangle \otimes |x\rangle.$$

Given a function $f : \{0,1\}^n \to \{0,1\}$ and a non-zero adversary matrix $\Gamma \in \mathbb{R}^{2^n \times 2^n}$ for f with principal eigenvector a, define the following progress measure,

$$\Delta_t = \|\Gamma\| - |\langle \psi^t \,|\, (\mathrm{Id} \otimes \Gamma) \,|\, \psi^t \rangle|$$

Then the progress obeys the following inequalities,

- *(Initial condition)* $\Delta_0 = 0$,

- *(Progress evolution)* $\Delta_{t+1} \le \Delta_t + 2\max_{i\in\{1,\dots,n\}}\|\Gamma_i\|$.

Furthermore, the average success probability p^μ_{succ} *of the algorithm in computing* f *after* T *queries is at most,*

- *(Final condition)* $p^\mu_{\mathrm{succ}} \le \frac{1}{2} + \sqrt{\Delta_T/(2\|\Gamma\|)}$.

Consequently, the average-case quantum query complexity of f *under the distribution* μ *is at least* $Q_\mu(f) \ge \frac{\|\Gamma\|}{36\max_{i\in\{1,\dots,n\}}\|\Gamma_i\|}$.

Proof of the initial condition. The initial state is $|\psi^0\rangle = |0,0\rangle \otimes |a\rangle$ where $|a\rangle = \sum_{x\in\{0,1\}^n} a_x|x\rangle$ is a principal unit eigenvector of Γ. Hence, $\Delta_0 = \|\Gamma\| - |\langle a \,|\, \Gamma \,|\, a\rangle| = 0$. □

Proof of the progress evolution. The progress increase is bounded by $\Delta_{t+1} - \Delta_t \le |\langle \psi^{t+1} \,|\, (\mathrm{Id} \otimes \Gamma) \,|\, \psi^{t+1}\rangle - \langle \psi^t \,|\, (\mathrm{Id}\otimes\Gamma) \,|\, \psi^t\rangle|$. By definition, $|\psi^{t+1}\rangle = (U_{t+1}\otimes \mathrm{Id})\mathcal{O}|\psi^t\rangle$ where $\mathcal{O}(|i,b\rangle\otimes|x\rangle) = (\mathcal{O}_x|i,b\rangle)\otimes|x\rangle = |i, b\oplus x_i\rangle \otimes |x\rangle$ is the joint binary oracle. Since $(U_{t+1}\otimes\mathrm{Id})$ and $(\mathrm{Id}\otimes\Gamma)$ commute, we obtain that

$$\Delta_{t+1} - \Delta_t \le |\langle \psi^t \,|\, \mathcal{O}(\mathrm{Id}\otimes\Gamma)\mathcal{O} \,|\, \psi^t\rangle - \langle \psi^t \,|\, (\mathrm{Id}\otimes\Gamma) \,|\, \psi^t\rangle|.$$

The operator $\mathcal{O}(\mathrm{Id}\otimes\Gamma)\mathcal{O} - \mathrm{Id}\otimes\Gamma$ can be expressed in the standard basis as $\sum_{i,b,x,y}(|i,b\oplus x_i\rangle\langle i, b\oplus y_i| - |i,b\rangle\langle i,b|)\otimes \Gamma_{x,y}|x\rangle\langle y|$. Its action on the value register is represented by the two-by-two matrix $\sum_{b\in\{0,1\}}|b\oplus x_i\rangle\langle b\oplus y_i| - |b\rangle\langle b|$, which equals 0 when $x_i = y_i$, and $X - \mathrm{Id}$ when $x_i \neq y_i$ (where X is the Pauli-X matrix). The condition on (x_i, y_i) can be absorbed into the matrix Γ_i, since $(\Gamma_i)_{xy} = 0$ when $x_i = y_i$ by definition (punctured property). Hence, we can continue the above inequality as follows,

$$\begin{aligned}
\Delta_{t+1} - \Delta_t &\le \Big| \sum_{i\in\{1,\dots,n\}} \langle \psi^t \,|\, (|i\rangle\langle i| \otimes (X - \mathrm{Id}) \otimes \Gamma_i) \,|\, \psi^t\rangle \Big| \\
&\le \sum_{i\in\{1,\dots,n\}} \|\mathrm{Id}\otimes(X-\mathrm{Id})\otimes\Gamma_i\| \cdot \|(|i\rangle\langle i|\otimes \mathrm{Id})|\psi^t\rangle\|^2 \\
&\qquad\qquad \text{(Cauchy-Schwarz inequality)} \\
&\le \max_{i\in\{1,\dots,n\}} \|\mathrm{Id}\otimes(X-\mathrm{Id})\otimes\Gamma_i\| \cdot \sum_{i\in\{1,\dots,n\}} \|(|i\rangle\langle i|\otimes\mathrm{Id})|\psi^t\rangle\|^2 \\
&= 2\max_{i\in\{1,\dots,n\}}\|\Gamma_i\|.
\end{aligned}$$

□

Proof of the final condition. The average success probability of the algorithm is defined as $p^\mu_{\mathrm{succ}} = \|\Pi_{\mathrm{succ}}|\psi^T\rangle\|^2$, where Π_{succ} is the projector onto the states $|i,b\rangle\otimes|x\rangle$ with $b = f(x)$. The progress after T queries satisfies the equality:

$$\begin{aligned}
\|\Gamma\| - \Delta_T &= |\langle \psi^T \,|\, (\mathrm{Id}\otimes\Gamma) \,|\, \psi^T\rangle| \\
&= \big|\langle \psi^T \,|\, \Pi_{\mathrm{succ}}(\Gamma\otimes\mathrm{Id})\Pi_{\mathrm{succ}} \,|\, \psi^T\rangle + \langle \psi^T \,|\, (\mathrm{Id}-\Pi_{\mathrm{succ}})(\Gamma\otimes\mathrm{Id})(\mathrm{Id}-\Pi_{\mathrm{succ}}) \,|\, \psi^T\rangle \\
&\quad + \langle \psi^T \,|\, \Pi_{\mathrm{succ}}(\Gamma\otimes\mathrm{Id})(\mathrm{Id}-\Pi_{\mathrm{succ}}) \,|\, \psi^T\rangle + \langle \psi^T \,|\, (\mathrm{Id}-\Pi_{\mathrm{succ}})(\Gamma\otimes\mathrm{Id})\Pi_{\mathrm{succ}} \,|\, \psi^T\rangle\big| \\
&= |\langle \psi^T \,|\, \Pi_{\mathrm{succ}}(\Gamma\otimes\mathrm{Id})(\mathrm{Id}-\Pi_{\mathrm{succ}}) \,|\, \psi^T\rangle + \langle \psi^T \,|\, (\mathrm{Id}-\Pi_{\mathrm{succ}})(\Gamma\otimes\mathrm{Id})\Pi_{\mathrm{succ}} \,|\, \psi^T\rangle|
\end{aligned}$$

where the last equality uses that $\Gamma_{xy} = 0$ when $f(x) = f(y)$ (f-sparse property). By the Cauchy-Schwarz inequality, we obtain

$$\begin{aligned}
\|\Gamma\| - \Delta_T &\le 2\|\Gamma\otimes\mathrm{Id}\| \cdot \|\Pi_{\mathrm{succ}}|\psi^T\rangle\| \cdot \|(\mathrm{Id}-\Pi_{\mathrm{succ}})|\psi^T\rangle\| \\
&= 2\|\Gamma\| \cdot \sqrt{p^\mu_{\mathrm{succ}}(1-p^\mu_{\mathrm{succ}})} \\
&\le 2\|\Gamma\| \cdot (1/4 + p^\mu_{\mathrm{succ}}(1-p^\mu_{\mathrm{succ}})) = 2\|\Gamma\| \cdot \big(1/2 - (p^\mu_{\mathrm{succ}} - 1/2)^2\big).
\end{aligned}$$

Reordering the terms, we obtain that $p^{\mu}_{\text{succ}} \leq 1/2 + \sqrt{\Delta_T/(2\|\Gamma\|)}$. Using the bound on the progress evolution, the success probability after T queries must be at most $p^{\mu}_{\text{succ}} \leq 1/2 + \sqrt{T \cdot \max_{i\in\{1,\ldots,n\}}\|\Gamma_i\|/\|\Gamma\|}$, hence the algorithm requires at least $T \geq \frac{\|\Gamma\|}{36\max_{i\in\{1,\ldots,n\}}\|\Gamma_i\|}$ queries to succeed with probability $p^{\mu}_{\text{succ}} \geq 2/3$. □

The adversary method is often stated under the worst-case output condition, where the algorithm must be correct on *any* input with probability at least 2/3 (as was defined in Section 1 and Definition 6). In this case, the lower bound depends on the *adversary value* $\text{Adv}(f)$, which is obtained by maximizing the above argument over the entire set of adversary matrices (or, equivalently, over all input distributions).

Definition 47 (Adversary value)**.** The *adversary value* of a function $f : \{0,1\}^n \to \{0,1\}$ is the non-negative real number defined as,

$$\text{Adv}(f) = \max_{\Gamma} \frac{\|\Gamma\|}{\max_{i\in\{1,\ldots,n\}}\|\Gamma_i\|}$$

where the maximum is taken over all non-zero adversary matrices $\Gamma \in \mathbb{R}^{2^n\times 2^n}$ for f.

Corollary 48. *The quantum query complexity of any function $f : \{0,1\}^n \to \{0,1\}$ is at least $Q(f) \geq \text{Adv}(f)/36$.*

Unlike other combinatorial measures of complexity – such as the block sensitivity – the adversary value is always equal to the optimal query complexity, up to a constant factor. The proof of the upper bound $Q(f) = O(\text{Adv}(f))$ will be the focus of Section 6. The crucial insight is that $\text{Adv}(f)$ can be expressed as the optimum of a semidefinite program, whose dual can be turned into a quantum algorithm.

Another interesting aspect of the adversary method is that it naturally scales with function composition, in the sense that applying it to a composed function $f \bullet g$ only requires understanding the method separately for f and g.

Proposition 49 (Function composition [HLŠ07; Rei09])**.** *Let $f : \{0,1\}^n \to \{0,1\}$ and $g : \{0,1\}^m \to \{0,1\}$ be two Boolean functions. Define their composition $f \bullet g : \{0,1\}^{nm} \to \{0,1\}$ as $f \bullet g(x) = f(g(x_1,\ldots,x_m),\ldots,g(x_{(n-1)m+1},\ldots,x_{nm}))$. Then, $\text{Adv}(f \bullet g) = \text{Adv}(f)\text{Adv}(g)$.*

The reader interested in the proof of this result may attempt to demonstrate the case $\text{Adv}(\text{OR} \bullet g) \geq \sqrt{n} \cdot \text{Adv}(g)$ by combining an optimal adversary matrix for g with the adversary matrix for the OR function described in the next section.

5.2 Applications

One of the most challenging aspects of the adversary method is identifying which adversary matrices maximize the ratio $\|\Gamma\|/(\max_i\|\Gamma_i\|)$. In this section, we develop some applications where Γ belongs to $\{0,1\}^{2^n\times 2^n}$ (sometimes referred to as the "basic adversary method"). This restriction on the entries of Γ can severely limit the adversary method in general, but it makes it somewhat more intuitive. The merits of introducing more general classes of matrices were identified later (see [ŠS06] for instance), with the most general case – allowing negative entries – discovered by Høyer, Lee and Špalek [HLŠ07].

Application 1: The OR function. We present a simple adversary matrix for the OR function, demonstrating that its query complexity is at least $\Omega(\sqrt{n})$. Note that the constant factor in this lower bound is slightly smaller than those established in Propositions 12 and 29.

Proposition 50 (Adversary method applied to OR)**.** *The quantum query complexity of the* OR *function is at least $Q(\text{OR}) \geq \sqrt{n}/36$.*

$$\Gamma = \begin{array}{c} \begin{array}{cccc} x^{(0)} & y^{(1)} & \dots & y^{(n)} \end{array} \\ \begin{pmatrix} 0 & 1 & \dots & 1 \\ 1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \dots & 0 \end{pmatrix} \end{array} \begin{array}{c} x^{(0)} \\ y^{(1)} \\ \vdots \\ y^{(n)} \end{array}$$

$$\Gamma_i = \begin{array}{c} \begin{array}{ccccccc} x^{(0)} & y^{(1)} & \dots & y^{(i)} & \dots & \dots & y^{(n)} \end{array} \\ \begin{pmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ \vdots & 0 & \dots & \dots & \dots & \dots & 0 \\ 0 & \vdots & \ddots & & & & \vdots \\ 1 & \vdots & & \ddots & & & \vdots \\ 0 & \vdots & & & \ddots & & \vdots \\ \vdots & \vdots & & & & \ddots & \vdots \\ 0 & 0 & \dots & \dots & \dots & \dots & 0 \end{pmatrix} \end{array} \begin{array}{c} x^{(0)} \\ y^{(1)} \\ \vdots \\ y^{(i)} \\ \vdots \\ \vdots \\ y^{(n)} \end{array}$$

Figure 6: Adversary matrix for the OR function.

Proof. We consider the $(0,1)$-adversary matrix Γ with nonzero weights placed on the same hard-to-distinguish pairs of inputs $(x^{(0)}, y^{(i)})$ as defined in the application of the hybrid method (Proposition 12). The non-zero parts of the adversary and punctured adversary matrices are represented in Figure 6.

It is a simple calculation to check that $\|\Gamma\| = \sqrt{n}$ and $\|\Gamma_i\| = 1$, hence $Q(\mathrm{OR}) \geq \|\Gamma\|/(36\|\Gamma_i\|) = \sqrt{n}/36$. The fact that $\left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2n}}, \dots, \frac{1}{\sqrt{2n}}\right)$ is a principal eigenvector of Γ informs us that the lower bound holds also for the average-case complexity $Q_\mu(\mathrm{OR}) \geq \sqrt{n}/36$ under the input distribution $\mu(x^{(0)}) = 1/2$, $\mu(y^{(1)}) = \cdots = \mu(y^{(n)}) = 1/(2n)$. □

Application 2: Combinatorial formulation of the basic adversary method. The adversary method was originally introduced by Ambainis [Amb02] for adversary matrices with $(0,1)$ entries. Here, we reproduce the combinatorial formulation of this approach and develop its application to the Connectivity problem in the next section. The basic adversary method represents hard-to-distinguish pairs of inputs as a bipartite graph and relates the adversary value (and, therefore, the query complexity) to the degree expansion properties of that graph.

Proposition 51 (Basic adversary method [Amb02])**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. Choose two sets of inputs $V_0 \subseteq \{x : f(x) = 0\}$, $V_1 \subseteq \{x : f(x) = 1\}$ and a relation $E \subseteq V_0 \times V_1$. Let G be the bipartite graph with vertex set $V = V_0 \cup V_1$ and edge set E. For each index $i \in \{1, \dots, n\}$, define the subgraph G_i of G obtained by removing all edges (x, y) for which $x_i = y_i$.*

For each $x \in \{0,1\}^n$, let $d(x)$ be the degree of the vertex x in the graph G and $d(x,i)$ be its degree in the graph G_i. Then, the adversary value of f is at least

$$\mathrm{Adv}(f) \geq \sqrt{\frac{\min_{(x,y)\in V_0\times V_1} d(x)d(y)}{\max_{(x,y)\in E, i\in\{1,\dots,n\}} d(x,i)d(y,i)}}.$$

Proof. We define Γ as the adjacency matrix of the graph G (i.e., $\Gamma_{x,y} = 1$ if and only if $(x,y) \in E$). One can check that Γ is indeed an adversary matrix. Let $m_0 = \min_{x\in V_0} d(x)$ and $m_1 = \min_{y\in V_1} d(y)$ be the minimal left and right degrees in the bipartite graph G. We can assume without loss of generality that $m_0, m_1 \geq 1$ (otherwise the lower bound is vacuous).

We show first that $\|\Gamma\| \geq \sqrt{m_0 m_1}$. Let $a \in \mathbb{R}^n$ be the vector defined as $a_x = \sqrt{m_0/(2m_1|E|)}$ when $x \in V_0$ and $a_y = \sqrt{m_1/(2m_0|E|)}$ when $y \in V_1$. Then, $\|a\|^2 = m_0|V_0|/(2m_1|E|) + m_1|V_1|/(2m_0|E|) \leq 1$ and $\|\Gamma a\|^2 = \sum_{x\in V_0} d(x)^2 m_1/(2m_0|E|) + \sum_{y\in V_1} d(y)^2 m_0/(2m_1|E|) \geq m_0 m_1 |V_0|/(2|E|) + m_0 m_1 |V_1|/(2|E|) = m_0 m_1$. Hence, $\|\Gamma\| \geq \|\Gamma a\|/\|a\| \geq \sqrt{m_0 m_1}$.

We claim next that $\|\Gamma_i\| \leq \max_{(x,y)\in E} \sqrt{d(x,i)d(y,i)}$ for all i. This follows immediately from the next inequality on the spectral norm of symmetric $(0,1)$-matrices.

Lemma 52 (Appendix A in [ŠS06]). *The spectral norm of a symmetric $(0,1)$-matrix A is at most $\|A\| \leq \max_{x,y:A_{x,y}=1} \sqrt{r_x(A)c_y(A)}$, where $r_x(A)$ (resp. $c_y(A)$) is the sum of the elements in the x-th row (resp. y-th column) of A.*

The proposition follows by definition of $\mathrm{Adv}(f) \geq \sqrt{\|\Gamma\| / \max_{i\in\{1,\ldots,n\}} \|\Gamma_i\|}$. □

Application 3: The Connectivity function. We study the complexity of determining whether an undirected n-vertex graph is connected or not. The input is encoded over $\binom{n}{2}$ bits $x \in \{0,1\}^{\binom{n}{2}}$ representing the adjacency matrix of the graph (where $x_{\{i,j\}} = 1$ if and only if there is an edge between vertices i and j). This input is *not* meant to represent the graphs G introduced in the basic adversary method (Proposition 51), which are distinct objects used to analyze the adversary value.

Definition 53 (Connectivity)**.** The Connectivity problem is to output 1 if the graph represented by the input $x \in \{0,1\}^{\binom{n}{2}}$ is connected, and 0 otherwise.

The query complexity of this problem was first established by Dürr, Heiligman, Høyer and Mhalla [DHHM06]. The classical query complexity is easily shown to be maximal through a reduction from the OR problem.

Proposition 54. *The randomized query complexity of the* Connectivity *function is at least* $R(\text{Connectivity}) = \Omega(n^2)$.

Proof. Assume that n is even. We describe a reduction from the OR problem over $n^2/4$ bits to the Connectivity problem over $\binom{n}{2}$ bits. Fix P_1 and P_2 to be two path graphs of lengths $n/2$ over the vertices $\{1,\ldots,n/2\}$ and $\{n/2+1,\ldots,n\}$, respectively. Given an input $y \in \{0,1\}^{n/2\times n/2}$ to the OR problem (indexed as a square matrix for convenience), define the graph obtained by the union of P_1, P_2 and all the edges $\{i, j+n/2\}$ for which $y_{i,j} = 1$. Then, the resulting graph is connected if and only if $\mathrm{OR}(y) = 1$. Since computing $\mathrm{OR}(y)$ requires $\Omega(n^2)$ queries, the reduction implies that the same lower bound applies to the Connectivity problem as well. □

The same reduction yields an $\Omega(\sqrt{n^2/4}) = \Omega(n)$ quantum lower bound for Connectivity. This is, however, not optimal, as shown in the next proposition based on the basic adversary method (we will describe a matching upper bound in Proposition 68, using the dual to the adversary method).

Proposition 55 (Adversary method applied to Connectivity)**.** *The quantum query complexity of the* Connectivity *function is at least* $Q(\text{Connectivity}) = \Omega(n^{3/2})$.

Proof. We first construct the two sets of inputs V_0 and V_1 needed to apply Proposition 51. The set V_0 consists of the graphs $x \in \{0,1\}^{\binom{n}{2}}$ made of two disjoint cycles, each of length at least $n/3$. The set V_1 consists of the graphs $x \in \{0,1\}^{\binom{n}{2}}$ made of a single cycle of length n. We define the relation $E \subseteq V_0 \times V_1$ as all pairs of graphs $(x,y) \in V_0 \times V_1$ that are related by the following process: y can be obtained by disconnecting one edge from each cycle in x and gluing the two resulting paths together into a cycle of length n.

Using the notations of Proposition 51, each input $x \in V_0$ belongs to at least $d(x) \geq (n/3)^2$ pairs in E, since there are at least $n/3$ choices to disconnect each cycle in x. Conversely, each input $y \in V_1$ belongs to at least $d(y) \geq n^2/6$ pairs since there are at least $n^2/6$ choices to disconnect two edges in y at distance at least $n/3$ from each other. We leave it to the reader to verify that the number of pairs remaining, when an edge $\{i,j\}$ must be in one graph but not in the other, satisfies the relation $d(x,\{i,j\})d(y,\{i,j\}) = O(n)$. By Proposition 51 and Corollary 48, the quantum query complexity is at least $Q(\text{Connectivity}) = \Omega(\mathrm{Adv}(\text{Connectivity})) = \Omega(\sqrt{n^2 \cdot n^2/n}) = \Omega(n^{3/2})$. □

CONNECTIVITY is part of a larger family of graph problems – the non-trivial monotone graph properties – whose complexities attract a lot of attention. The interested reader can refer to the Aanderaa-Karp-Rosenberg conjectures (e.g., [ABK+21, Section 5]).

6 Algorithmic Dual to the Adversary Method

The study of lower-bound methods is often not entirely separate from that of upper bounds, i.e., algorithm design. This principle is particularly well illustrated by the adversary method. Indeed, any solution to its dual, defined through semidefinite optimization, can be adapted into a surprisingly efficient converse algorithm. This result, first established by Reichardt [Rei11], provides a tight characterization of quantum query complexity in terms of the adversary value, up to constant factors: $Q(f) = \Theta(\mathrm{Adv}(f))$. This stands in contrast to the dual of the polynomial method, as described in Proposition 32, which does not have an algorithmic interpretation in general (although this can be addressed by a recent extension of the polynomial method [ABP19]).

This section presents the dual of the adversary method and explains how its solutions can be converted into quantum algorithms. We follow the approach of [LMR+11; Wol19]. In the application section, we derive optimal algorithms for the OR, AND-OR TREE and CONNECTIVITY functions.

6.1 Technique

The adversary value $\mathrm{Adv}(f)$ was defined in the previous section as the supremum of the ratio $\|\Gamma\| / \max_{i\in\{1,\dots,n\}} \|\Gamma_i\|$, evaluated over the set of adversary matrices Γ (Definition 47). It is not complicated to see that this optimization problem can be phrased as a semidefinite program (SDP), i.e., a generalization of a linear program where, in addition to linear constraints, some variables X (structured as square matrices) must satisfy the positive semidefinite constraint (PSD) $X \succeq 0$.

The dual of an SDP often provides valuable insights into its optimal solutions. Below, we state the primal-dual formulation of the adversary value. The dual program associates a PSD matrix $V^{(i)}$ with each query index i, and it minimizes the largest diagonal entry of $\sum_i V^{(i)}$ under the constraint that the off-diagonal terms of the partial sum $\sum_{i:x_i\neq y_i} V^{(i)}_{x,y}$ are equal to 1 whenever $f(x) \neq f(y)$.

Proposition 56 (Dual program for $\mathrm{Adv}(f)$, proven in [Rei09] or Theorem 3.29 in [Bel14])**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. Define the two following semidefinite programs.*

Primal semidefinite program

$$\begin{array}{lll} \max\limits_{\Gamma} & \dfrac{\|\Gamma\|}{\max_{i\in\{1,\dots,n\}}\|\Gamma_i\|} & \\ s.t. & \Gamma_{x,y} = \Gamma_{y,x} & \forall x,y \in \{0,1\}^n \\ & \Gamma_{x,y} = 0 & \forall x,y : f(x) = f(y) \\ & \Gamma \in \mathbb{R}^{2^n\times 2^n} & \end{array}$$

Dual semidefinite program

$$\begin{array}{lll} \min\limits_{V^{(1)},\dots,V^{(n)}} & \max\limits_{x\in\{0,1\}^n} \sum\limits_{i\in\{1,\dots,n\}} V^{(i)}_{x,x} & \\ s.t. & V^{(i)} \succeq 0 & \forall i \in \{1,\dots,n\} \\ & \sum_{i:x_i\neq y_i} V^{(i)}_{x,y} = 1 & \forall x,y : f(x) \neq f(y) \\ & V^{(1)},\dots,V^{(n)} \in \mathbb{C}^{2^n\times 2^n} & \end{array}$$

Then, the two programs are dual to each other, and they satisfy the strong duality property. In particular, their optimum are both equal to $\mathrm{Adv}(f)$.

By (weak) duality, one can certify that the adversary value is at most $\mathrm{Adv}(f) \leq T$ by exhibiting a feasible solution to the dual program with value T. The striking property, which we establish next, is the ability to also derive a quantum algorithm for computing f with complexity $O(T)$. The rest of this section is dedicated to the description and analysis of this algorithm. We start by rewriting the solutions of the dual program in terms of the vector realizations of the PSD matrices.

Lemma 57 (Vector realization of a dual solution)**.** *Let $V^{(1)}, \ldots, V^{(n)} \in \mathbb{C}^{2^n \times 2^n}$ be a feasible solution to the dual program from Proposition 56 with value $T = \max_{x\in\{0,1\}^n} \sum_{i\in\{1,\ldots,n\}} V^{(i)}_{x,x}$. Then, there exist an integer d and a set of complex vectors $\{|w^{(x,i)}\rangle\}_{x\in\{0,1\}^n, i\in\{1,\ldots,n\}} \subset \mathbb{C}^d$ such that $\sum_{i:x_i\neq y_i} \langle w^{(x,i)} \,|\, w^{(y,i)}\rangle = 1$ when $f(x) \neq f(y)$, and $T = \max_{x\in\{0,1\}^n} \sum_{i\in\{1,\ldots,n\}} \|w^{(x,i)}\|^2$.*

Proof. A necessary (and sufficient) condition for a matrix $V \in \mathbb{C}^{2^n\times 2^n}$ to be PSD is to be equal to a Gram matrix $V_{x,y} = \langle w^{(x)} \,|\, w^{(y)}\rangle$ for some vectors $|w^{(0)}\rangle, \ldots, |w^{(2^n-1)}\rangle \in \mathbb{C}^d$ and integer d. It is immediate to verify that using such a vector realization for the matrices $V^{(1)}, \ldots, V^{(n)}$ leads to the conclusion of the lemma. □

The dimension d of a vector solution can always be chosen as 2^n (for instance, by taking the column vectors of the Cholesky decompositions of the matrices $V^{(1)}, \ldots, V^{(n)}$). However, the smaller d is, the less memory the quantum algorithm will require.

The algorithm is going to determine the value of $f(x)$ based on the outcome of a quantum phase estimation procedure. The estimated eigenphase will be close to 0 when $f(x) = 1$ and at least $1/(2T)$ when $f(x) = 0$. The eigenphase gap between the two cases ensures that running phase estimation with a precision of $O(1/T)$ is sufficient.

We now define the input unitary R_x used in the phase estimation procedure. It is given by the product of two reflection operators acting on three registers, over a Hilbert space $\mathcal{H}$ of dimension $2nd + 1$ (in particular, the algorithm is not memoryless). The first two registers are the index and value registers. The third register has d dimensions and is used to encode the given vector realization. The extra dimension allows us to define a special state $|\star\rangle$, which will serve as the guiding state for phase estimation.

Definition 58 (Unitaries R_x associated with a vector realization)**.** Fix two integers n, d. For each $x \in \{0,1\}^n$, let $\mathcal{H}_x$ be the Hilbert space of dimension $2nd + 1$ defined as,

$$\mathcal{H}_x = \operatorname{span}\big\{|i, x_i\rangle \otimes |w\rangle : i \in \{1,\ldots,n\}, w \in \{1,\ldots,d\}\big\} \oplus \operatorname{span}\{|1,0\rangle \otimes |d+1\rangle\}.$$

Let $\mathcal{H} = \sum_x \mathcal{H}_x$ be the Hilbert space spanned by the union of the sets. Define $|\star\rangle = |1,0\rangle\otimes|d+1\rangle$.

Fix a vector realization $|w^{(x,i)}\rangle \in \operatorname{span}\{|1\rangle, \ldots, |d\rangle\}$ as described in Lemma 57. For each $x \in \{0,1\}^n$, let $|t^+_x\rangle \in \mathcal{H}_x$ and $|t^-_x\rangle \in \mathcal{H}_{\bar{x}}$ be the two (unnormalized) states defined as,

$$|t^+_x\rangle = |\star\rangle + \frac{1}{\sqrt{4T}} \sum_{i\in\{1,\ldots,n\}} |i, x_i\rangle \otimes |w^{(x,i)}\rangle \quad \text{and} \quad |t^-_x\rangle = |\star\rangle - \sqrt{4T} \sum_{i\in\{1,\ldots,n\}} |i, \overline{x_i}\rangle \otimes |w^{(x,i)}\rangle,$$

where $\overline{x_i} = 1 - x_i$. Finally, define the unitary R_x acting on $\mathcal{H}$ as the product,

$$R_x = (2\Pi_x - \mathrm{Id})(2\Delta - \mathrm{Id}).$$

where Π_x and Δ are the orthogonal projectors onto $\mathcal{H}_x$ and $\operatorname{span}\big\{|t^-_y\rangle : f(y) = 1\big\}$ respectively.

Before analyzing the spectral properties of R_x, note that it can be implemented using only two queries to a quantum oracle for x.

Lemma 59 (Query implementation of R_x)**.** *There exist two unitary operators U_0, U_1 such that, for any $x \in \{0,1\}^n$, the operation R_x given in Definition 58 can be decomposed as the product $R_x = (\mathcal{O}_x \otimes \mathrm{Id})U_1(\mathcal{O}_x \otimes \mathrm{Id})U_0$, where $\mathcal{O}_x$ is the quantum binary oracle to x.*

Proof. The first unitary can be chosen as the reflection $U_0 = 2\Delta - \mathrm{Id}$, since the projector Δ does not depend on x. By definition, the second reflection $2\Pi_x - \mathrm{Id}$ must flip the sign of the basis states $|i, \overline{x_i}\rangle \otimes |w\rangle$ when $w \neq d + 1$, and keep the other states the same. Omitting the last register, it suffices to use the Pauli-Z gate to obtain $\mathcal{O}_x(\mathrm{Id} \otimes Z)\mathcal{O}_x|i, \overline{x_i}\rangle = -|i, \overline{x_i}\rangle$ and $\mathcal{O}_x(\mathrm{Id} \otimes Z)\mathcal{O}_x|i, x_i\rangle = |i, x_i\rangle$. The unitary U_1 applies the same Pauli-Z transformation, but conditioned on the last register being not in the state $|d + 1\rangle$. □

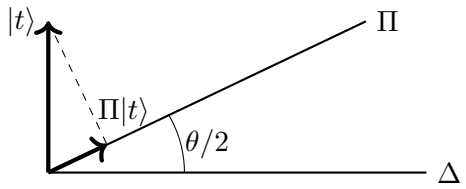

Figure 7: The orthogonal projection of the vector $|t\rangle$ on Π has norm $\|\Pi|t\rangle\| = \sin(\theta/2)\||t\rangle\|$.

We now state the central result regarding the spectral properties of R_x. If $f(x) = 1$ then it is shown that the state $|\star\rangle$ lies close the eigenspaces of R_x with eigenphase 0. On the other hand, if $f(x) = 0$ then the state $|\star\rangle$ principally belongs to the eigenspaces with larger eigenphases $\Omega(1/T)$.

Proposition 60 (Phase gap)**.** *Let $\Lambda_{x,\theta}$ be the projector onto the eigenspaces of R_x with eigenvalues in the set $\{e^{i\varphi} : |\varphi| \leq \theta\}$. Then, $\|\Lambda_{x,0}|\star\rangle\|^2 \geq 3/4$ if $f(x) = 1$ and $\|\Lambda_{x,\frac{1}{3T}}|\star\rangle\|^2 \leq 2/9$ if $f(x) = 0$.*

Proof. First, we consider the case $f(x) = 1$. The state $|t_x^+\rangle$ is both in the support of Δ and Π_x. Hence, $R_x|t_x^+\rangle = |t_x^+\rangle$, meaning that it is in the support of $\Lambda_{x,0}$ as well. The lemma follows by observing that the distance with the state $|\star\rangle$ is at most $\||\star\rangle - |t_x^+\rangle\|^2 = \frac{1}{4T}\|\sum_{i\in\{1,\ldots,n\}}|i, x_i\rangle \otimes |w^{(x,i)}\rangle\|^2 = \frac{1}{4T}\sum_{i\in\{1,\ldots,n\}}\||w^{(x,i)}\rangle\|^2 \leq \frac{1}{4T} \cdot T \leq \frac{1}{4}$.

Next, we consider the case $f(x) = 0$. We let the reader verifies that the state $|t_x^-\rangle$ is orthogonal to the support of Δ, i.e., $\Delta|t_x^-\rangle = 0$, and its projection onto the support of Π_x is $|\star\rangle = \Pi_x|t_x^-\rangle$ (the former equality makes use of the condition $\sum_{i:x_i\neq y_i}\langle w^{(x,i)} \,|\, w^{(y,i)}\rangle = 1$ satisfied by the vector realization when $f(x) \neq f(y)$). The proof will conclude by using general results about operators that are a *product of two reflections* (as is R_x). If Π and Δ were rank-one projectors, then the eigenvalues $e^{\pm i\theta}$ of the rotation $(2\Pi - \mathrm{Id})(2\Delta - \mathrm{Id})$ would be dictated by the angle θ between the two projectors. Moreover, the norm of a state $|t\rangle$ orthogonal to Δ would decrease by a factor $\sin(\theta/2) \leq \theta/2$ when projected on Π, as shown in Figure 7.

For higher-rank projectors, a similar argument applies to the norm of a state orthogonal to Δ that is projected on the small-phase eigenspaces.

Lemma 61 (Effective spectral gap lemma [LMR+11])**.** *Let Π and Δ be two projectors and set $R = (2\Pi - \mathrm{Id})(2\Delta - \mathrm{Id})$. Let Λ_θ be the projector onto the eigenspaces of R with eigenvalues in $\{e^{i\varphi} : |\varphi| \leq \theta\}$. Then, for any state $|t\rangle$ with $\Delta|t\rangle = 0$, we have $\|\Lambda_\theta\Pi|t\rangle\| \leq \frac{\theta}{2}\||t\rangle\|$.*

By applying this lemma to the state $|t_x^-\rangle$, we conclude $\|\Lambda_{x,1/3T}|\star\rangle\|^2 = \|\Lambda_{x,1/3T}\Pi_x|t_x^-\rangle\|^2 \leq \frac{1}{36T^2}\||t_x^-\rangle\|^2 = \frac{1}{36T^2}(1 + 4T\sum_{i\in\{1,\ldots,n\}}\||w^{(x,i)}\rangle\|^2) \leq \frac{1}{36T^2}(1 + 4T^2) \leq \frac{2}{9}$. □

The last ingredient of the construction is the well-known quantum phase estimation algorithm, which allows estimating (in superposition) the eigenphase of a guiding eigenvector.

Lemma 62 (Phase estimation)**.** *Let R be a unitary operator and $|\psi\rangle$ be an eigenvector with eigenphase $\varphi \in (-\pi, \pi]$, i.e., $R|\psi\rangle = e^{i\varphi}|\psi\rangle$. Given a precision parameter $\epsilon \in (0, 1)$, the quantum phase estimation algorithm implements a unitary QPE_R that uses the controlled-R operation $O(1/\epsilon)$ times and computes a superposition $\mathrm{QPE}_R(|\psi\rangle \otimes |0\rangle) = |\psi\rangle \otimes (\sum_{\widetilde{\varphi}} \alpha_{\widetilde{\varphi}}|\widetilde{\varphi}\rangle)$ of phase estimates $\widetilde{\varphi}$ such that the probability of measuring an estimate with error $|\widetilde{\varphi} - \varphi| < \epsilon$ is at least $\sum_{\widetilde{\varphi}:|\widetilde{\varphi}-\varphi|\leq\epsilon}|\alpha_{\widetilde{\varphi}}|^2 \geq 8/9$.*

We now state the main theorem about converting the vector realization of a dual solution into a quantum algorithm.

Theorem 63 (Dual algorithm)**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function and $\{w^{(x,i)}\}_{x\in\{0,1\}^n, i\in\{1,\dots,n\}}$ be a set of complex vectors $w^{(x,i)} \in \mathbb{C}^d$, for some integer d, satisfying the condition*

$$\sum_{i:x_i\neq y_i} \langle w^{(x,i)} \,|\, w^{(y,i)}\rangle = 1 \quad \text{for all } x, y \text{ such that } f(x) \neq f(y).$$

Then, there exists a quantum algorithm that computes f with query complexity,

$$T = O\left(\max_{x\in\{0,1\}^n} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2\right).$$

Moreover, there exists at least one such set of vectors that satisfies $T = \Theta(\mathrm{Adv}(f))$.

Proof. The algorithm simply consists of using phase estimation (Lemma 62) on the unitary R_x and guiding state $|\star\rangle$ (Definition 58) with precision $\epsilon = 1/(6T)$, and measuring a value $\widetilde{\varphi}$ in the phase estimate register. If $|\widetilde{\varphi}| \leq 1/(6T)$ then it outputs 1, otherwise it outputs 0.

The query complexity is $O(T)$ by Lemmas 59 and 62. It remains to show that the output is equal to $f(x)$ with probability at least $2/3$.

If $f(x) = 0$, the probability of measuring $|\widetilde{\varphi}| \leq 1/(6T)$ is at least $\|\Lambda_{x,0}|\star\rangle\|^2 \cdot 8/9 \geq 3/4 \cdot 8/9 = 2/3$, which is the squared norm of the component in $|\star\rangle$ with eigenphase 0 (Proposition 60) multiplied by the probability of measuring an estimate less than $1/(6T)$ for such a component (Lemma 62).

If $f(x) = 1$, the probability of measuring $|\widetilde{\varphi}| \leq 1/(6T)$ is at most $\|\Lambda_{x,\frac{1}{3T}}|\star\rangle\|^2 + 1/9 \leq 2/9 + 1/9 = 1/3$, which is the squared norm of the component in $|\star\rangle$ with eigenphase at most $1/(3T)$ (Proposition 60), added to the probability of measuring an estimate less than $1/(6T)$ for the component with eigenphase at least $1/(3T)$ (Lemma 62).

Finally, the algorithm can be made to work with the optimal value $T = \Theta(\mathrm{Adv}(f))$ by using a vector realization (Lemma 57) for an optimal solution to the dual from Proposition 56. □

Together with Corollary 48, this result shows that the adversary value is a tight characterization of the quantum query complexity, up to constant factors.

Corollary 64. *There exist two universal constant $c_1 < c_2$ such that the quantum query complexity of any function $f : \{0,1\}^n \to \{0,1\}$ satisfies $c_1\mathrm{Adv}(f) \leq Q(f) \leq c_2\mathrm{Adv}(f)$.*

6.2 Applications

The dual of the adversary method has proven surprisingly useful in the design and analysis of new quantum algorithms. Several algorithmic frameworks have been developed based on the result established in Theorem 63, such as span programs [Rei09], learning graphs [Bel12], and, more recently, transducers [BJY24]. These frameworks can simplify the search for dual solutions and the study of their properties.

In this section, we explicitly design the vector realizations of dual solutions that lead to optimal quantum algorithms for the OR and Connectivity functions. We also use the composition property of the adversary value for solving the AND-OR tree problem optimally.

Application 1: The OR function. We describe an optimal solution to the dual adversary proving that the lower bound $Q(\mathrm{OR}) = \Omega(\sqrt{n})$ is indeed optimal. We proceed by exhibiting a vector realization $w^{(x,i)}$ satisfying the conditions stated in Lemma 57 with value $\max_{x\in\{0,1\}^n} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2 = O(\sqrt{n})$.

For convenience in the proof, we first establish a general result stating that the value of a vector realization can be balanced between the 0-inputs and 1-inputs, such that the maximum of $\max_{x\in\{0,1\}^n} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2$ can be replaced with the geometric mean of the maximum evaluated on the 0-inputs and 1-inputs separately.

Lemma 65 (Balanced vector realization)**.** *Suppose that $\{|w^{(x,i)}\rangle\}$ is a vector realization of a solution with value $T = \max_{x\in\{0,1\}^n} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2$, as described in Lemma 57. Define,*

$$T_0 = \max_{x:f(x)=0} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2 \quad \text{and} \quad T_1 = \max_{x:f(x)=1} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2.$$

Then there exists a vector realization of a solution with value $\sqrt{T_0 T_1}$.

Proof. It suffices to define the vectors $v^{(x,i)} = (T_1/T_0)^{1/4} \cdot w^{(x,i)}$ when $f(x) = 0$ and $v^{(x,i)} = (T_0/T_1)^{1/4} \cdot w^{(x,i)}$ when $f(x) = 1$. When $f(x) \neq f(y)$, the condition $\sum_{i:x_i\neq y_i} \langle v^{(x,i)} \,|\, v^{(y,i)}\rangle = 1$ is satisfied since the factors are cancelling out: $(T_1/T_0)^{1/4}(T_0/T_1)^{1/4} = 1$. This new solution has value $\max_{x\in\{0,1\}^n} \sum_i \|v^{(x,i)}\|^2 \leq \max\{\sqrt{T_1/T_0} \cdot T_0, \sqrt{T_0/T_1} \cdot T_1\} = \sqrt{T_0 T_1}$. □

We now describe the algorithm for OR.

Proposition 66 (Dual algorithm applied to OR)**.** *There exists a quantum algorithm for the* OR *function with query complexity $Q(\mathrm{OR}) = O(\sqrt{n})$.*

Proof. We describe a feasible vector realization with dimension $d = 1$ (i.e., the vectors are scalar numbers). For each $x \in \{0,1\}^n$ and $i \in \{1,\dots,n\}$, set:

$$w^{(x,i)} = \begin{cases} 1 & \text{if } x \text{ is the all-0 input,} \\ 1 & \text{if } x_i = 1 \text{ and } x_j = 0 \text{ for all } j < i, \\ 0 & \text{otherwise.} \end{cases}$$

The condition $\sum_{i:x_i\neq y_i} \langle w^{(x,i)} \,|\, w^{(y,i)}\rangle = 1$ is easily verified when $\mathrm{OR}(x) \neq \mathrm{OR}(y)$, hence it is a valid vector realization.

The all-0 input gives the value of $T_0 = \max_{x:f(x)=0} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2 = n$, and the other inputs gives the value of $T_1 = \max_{x:f(x)=1} \sum_{i\in\{1,\dots,n\}} \|w^{(x,i)}\|^2 = 1$. Hence, by rebalancing the vector realization using Lemma 65 (which amounts to multiplying $w^{(x,i)}$ with $1/n^{1/4}$ when x is the all-0 input, and $n^{1/4}$ otherwise) and applying Theorem 63, we obtain a quantum algorithm with query complexity $O(\sqrt{T_0 T_1}) = O(\sqrt{n})$. □

Application 2: AND-OR TREE. A *read-once formula* is a Boolean function that can be represented as the evaluation of a rooted tree whose nodes are labeled with OR, AND and NOT gates, and where each input bit appears exactly once in the leaves. The composition property of the adversary value (Proposition 49) and Theorem 63 enable the design of optimal quantum algorithms for such functions, achieving a query complexity of $\Theta(\sqrt{n})$ [Rei11]. We outline the argument for the balanced AND-OR TREE, a depth-2 read-once formula with an AND gate at the root and OR gates at the first level (each with input size $\sqrt{n}$).

Proposition 67 (Dual algorithm applied to AND-OR TREE)**.** *The quantum query complexity of the balanced* AND-OR TREE *function is $Q(\mathrm{AND} \bullet \mathrm{OR}) = \Theta(\sqrt{n})$.*

Proof. The query complexities of the OR and AND functions over $m = \sqrt{n}$ bits are $Q(\mathrm{OR}) = Q(\mathrm{AND}) = \Theta(\sqrt{m})$, as established, for instance, in Propositions 50 and 66 (it is easy to see that OR and AND must have exactly the same query complexity). By the composition property of the adversary value (Proposition 49) and its characterization of quantum query complexity (Corollary 64), we deduce that the complexity of the AND-OR TREE function is $\Theta(\sqrt{m} \times \sqrt{m}) = \Theta(\sqrt{n})$. □

While the lower bound may not seem particularly surprising here, the upper bound falls below the natural complexity $O(\sqrt{n}\log n)$, which would be achieved using standard error reduction techniques when composing bounded-error algorithms. This errorless composition property is a striking feature of quantum query algorithms, allowing arbitrarily many levels of composition without any drift in the query complexity.

Application 3: The Connectivity function. We return to the Connectivity problem, for which a lower bound of $\Omega(n^{3/2})$ was established in Proposition 55. We complement this result with a matching upper bound due to Belovs and Reichardt [BR12], obtained by exhibiting the vector realization of an optimal solution to the dual adversary. These vectors will have a very compact description of dimension $d = n^2$, leading to a quantum algorithm with only $O(\log n)$ qubits of memory. This uses exponentially less space that an older quantum algorithm for Connectivity [DHHM06] that requires $O(n \log n)$ memory.

Proposition 68 (Dual algorithm applied to Connectivity)**.** *There exists a quantum algorithm for the* Connectivity *function with query complexity* $Q(\text{Connectivity}) = O(n^{3/2})$.

Proof. We first describe a vector realization for the problem of deciding if two vertices s and t are connected by a path (called st-Connectivity), and we adapt it next to Connectivity.

Recall that a graph over n vertices is represented by its adjacency matrix $x \in \{0,1\}^{\binom{n}{2}}$. We let $C_x(v) \subseteq \{1,\dots,n\}$ denote the set of vertices that belong to the same connected component as the vertex v. Given two vertices $s,t \in \{0,1\}^n$, we partition the set of graphs into two parts $\mathcal{G}_0^{st} \cup \mathcal{G}_1^{st} = \{0,1\}^{\binom{n}{2}}$, where $\mathcal{G}_0^{st}$ contains the graphs that are not st-connected, and $\mathcal{G}_1^{st}$ contains the graphs that are st-connected (i.e., $t \in C_x(s)$). For each input $x \in \{0,1\}^{\binom{n}{2}}$ and edge $\{i,j\} \subset \{1,\dots,n\}$ (representing a query index), we define the vector $|v_{st}^{(x,\{i,j\})}\rangle \in \text{span}\{|1\rangle,\dots,|n\rangle\}$ as,

- If $x \in \mathcal{G}_0^{st}$ then:

$$|v_{st}^{(x,\{i,j\})}\rangle = \begin{cases} |i\rangle - |j\rangle & \text{if } i \in C_x(s) \text{ and } j \notin C_x(s), \\ 0 & \text{otherwise.} \end{cases}$$

- If $x \in \mathcal{G}_1^{st}$ then fix a path *of shortest length* from s to t, and define:

$$|v_{st}^{(x,\{i,j\})}\rangle = \begin{cases} 0 & \text{if } \{i,j\} \text{ is not an edge on that path,} \\ |i\rangle & \text{if } \{i,j\} \text{ is an edge on the path and } i \text{ is visited first.} \end{cases}$$

We claim that this construction is a valid vector realization for the problem of deciding whether the vertices s and t are connected. Indeed, let $x \in \mathcal{G}_0^{st}$ and $y \in \mathcal{G}_1^{st}$. Then, the quantity $\sum_{\{i,j\}:x_{\{i,j\}}\neq y_{\{i,j\}}} \langle v_{st}^{(x,\{i,j\})} \,|\, v_{st}^{(y,\{i,j\})}\rangle$ counts the number of times the (oriented) st-path associated with y is leaving the connected component of s in x, minus the number of times it is entering this component. The former occurs exactly one more time than the latter (since the path starts at $s \in C_x(s)$ and ends at $t \notin C_x(s)$), hence $\sum_{\{i,j\}:x_{\{i,j\}}\neq y_{\{i,j\}}} \langle v_{st}^{(x,\{i,j\})} \,|\, v_{st}^{(y,\{i,j\})}\rangle = 1$.

We now adapt this vector realization to the Connectivity problem. We use the basic observation that a graph is connected if and only if it is st-connected for $s = 1$ and all $t \in \{2,\dots,n\}$. Let $\mathcal{G}_0 \subset \{0,1\}^{\binom{n}{2}}$ be the set of all graphs that are not connected, and $\mathcal{G}_1$ those that are connected. For each $x \in \{0,1\}^{\binom{n}{2}}$ and $\{i,j\} \subset \{1,\dots,n\}$, we define $|w^{(x,\{i,j\})}\rangle \in \text{span}\{|k\rangle|t\rangle : k \in \{1,\dots,n\}, t \in \{2,\dots,n\}\}$ as,

- If $x \in \mathcal{G}_0$ then $|w^{(x,\{i,j\})}\rangle = \frac{1}{n-|C_x(1)|}\sum_{t\notin C_x(1)} |v_{1t}^{(x,\{i,j\})}\rangle|t\rangle$.

- If $x \in \mathcal{G}_1$ then $|w^{(x,\{i,j\})}\rangle = \sum_{t\in\{2,\dots,n\}} |v_{1t}^{(x,\{i,j\})}\rangle|t\rangle$.

Given $x \in \mathcal{G}_0$ and $y \in \mathcal{G}_1$, we have $\sum_{\{i,j\}:x_{\{i,j\}} \neq y_{\{i,j\}}} \langle w^{(x,\{i,j\})} \mid w^{(y,\{i,j\})} \rangle = \sum_{t \notin C_x(1)} \frac{1}{n-|C_x(1)|} \cdot 1 = 1$, hence it is a valid vector realization for the CONNECTIVITY problem.

Finally we show that the 0-inputs have a value of at most $T_0 = 2(n-1)$ and the 1-inputs have a value of at most $T_1 = (n-1)(n-2)$. Indeed, for all $x \in \mathcal{G}_0$, we have $\sum_{\{i,j\}} \| |w^{(x,\{i,j\})}\rangle \|^2 = \frac{1}{(n-|C_x(1)|)^2} \sum_{t \notin C_x(1)} \sum_{\{i,j\}} \| |v_{1t}^{(x,\{i,j\})}\rangle \|^2 = \frac{1}{(n-|C_x(1)|)^2} \sum_{t \notin C_x(1)} 2|C_x(1)|(n - |C_x(1)|) = 2|C_x(1)| \leq 2(n-1)$. For all $x \in \mathcal{G}_1$, we have $\sum_{\{i,j\}} \|w^{(x,\{i,j\})}\|^2 = \sum_{t \in \{2,\ldots,n\}} \sum_{\{i,j\}} \| |v_{1t}^{(x,\{i,j\})}\rangle \|^2 \leq \sum_{t \in \{2,\ldots,n\}} (n-1) = (n-1)(n-2)$, where the inequality uses the fact that the shortest path between $s = 1$ and t must be of size at most $n-1$. By Lemma 65 and Theorem 63, we can convert this vector realization into a quantum algorithm with query complexity $O(\sqrt{T_0 T_1}) = O(n^{3/2})$. □

References

[ABB+17] A. Ambainis, K. Balodis, A. Belovs, T. Lee, M. Santha, and J. Smotrovs. "Separations in Query Complexity Based on Pointer Functions". In: *Journal of the ACM* 64.5 (2017) (cit. on p. 7).

[ABK+21] S. Aaronson, S. Ben-David, R. Kothari, S. Rao, and A. Tal. "Degree vs. Approximate Degree and Quantum Implications of Huang's Sensitivity Theorem". In: *Proceedings of the 53rd Symposium on Theory of Computing (STOC)*. 2021, pp. 1330–1342 (cit. on pp. 7, 12, 34).

[ABP19] S. Arunachalam, J. Briët, and C. Palazuelos. "Quantum Query Algorithms Are Completely Bounded Forms". In: *SIAM Journal on Computing* 48.3 (2019), 903–)925 (cit. on pp. 11, 34).

[Amb02] A. Ambainis. "Quantum Lower Bounds by Quantum Arguments". In: *Journal of Computer and System Sciences* 64.4 (2002), pp. 750–767 (cit. on p. 32).

[Amb07] A. Ambainis. "Quantum Walk Algorithm for Element Distinctness". In: *SIAM Journal on Computing* 37.1 (2007), pp. 210–239 (cit. on pp. 27, 28).

[AS04] S. Aaronson and Y. Shi. "Quantum Lower Bounds for the Collision and the Element Distinctness Problems". In: *Journal of the ACM* 51.4 (2004), pp. 595–605 (cit. on pp. 11, 17).

[BBBV97] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. "Strengths and Weaknesses of Quantum Computing". In: *SIAM Journal on Computing* 26.5 (1997), pp. 1510–1523 (cit. on pp. 1, 7).

[BBC+01] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. "Quantum Lower Bounds by Polynomials". In: *Journal of the ACM* 48.4 (2001), pp. 778–797 (cit. on p. 11).

[Bel12] A. Belovs. "Span Programs for Functions with Constant-Sized 1-Certificates". In: *Proceedings of the 44th Symposium on Theory of Computing (STOC)*. 2012, pp. 77–84 (cit. on p. 37).

[Bel14] A. Belovs. *Applications of the Adversary Method in Quantum Query Algorithms*. arXiv:1402.3858 [quant-ph]. 2014 (cit. on pp. 2, 34).

[Ben20] S. Ben-David. *Quantum Lower Bounds*. Available at https://cs.uwaterloo.ca/~s4bendav/CS860S20.html. 2020 (cit. on p. 2).

[BHT98] G. Brassard, P. Høyer, and A. Tapp. "Quantum Cryptanalysis of Hash and Claw-Free Functions". In: *Proceedings of the 3rd Latin American Symposium on Theoretical Informatics (LATIN)*. 1998, pp. 163–169 (cit. on pp. 27, 28).

[BJY24] A. Belovs, S. Jeffery, and D. Yolcu. "Taming Quantum Time Complexity". In: *Quantum* 8 (2024), p. 1444 (cit. on p. 37).

[BR12] A. Belovs and B. W. Reichardt. "Span Programs and Quantum Algorithms for st-Connectivity and Claw Detection". In: *Proceedings of the 20th European Symposium on Algorithms (ESA)*. 2012, pp. 193–204 (cit. on p. 39).

[BS21] N. Bansal and M. Sinha. "k-Forrelation Optimally Separates Quantum and Classical Query Complexity". In: *Proceedings of the 53rd Symposium on Theory of Computing (STOC)*. 2021, pp. 1303–1316 (cit. on pp. 1, 7).

[BT22] M. Bun and J. Thaler. "Approximate Degree in Classical and Quantum Computing". In: *Foundations and Trends in Theoretical Computer Science* 15.3-4 (2022), pp. 229–423 (cit. on pp. 12, 14, 18).

[BV97] E. Bernstein and U. V. Vazirani. "Quantum Complexity Theory". In: *SIAM Journal on Computing* 26.5 (1997), pp. 1411–1473 (cit. on p. 1).

[BW02] H. Buhrman and R. de Wolf. "Complexity Measures and Decision Tree Complexity: A Survey". In: *Theoretical Computer Science* 288.1 (2002), pp. 21–43 (cit. on p. 2).

[CCD+03] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman. "Exponential Algorithmic Speedup by a Quantum Walk". In: *Proceedings of the 35th Symposium on Theory of Computing (STOC)*. 2003, pp. 59–68 (cit. on pp. 1, 7).

[Chi17] A. M. Childs. *Lecture Notes on Quantum Algorithms*. Available at https://www.cs.umd.edu/~amchilds/qa/. 2017 (cit. on p. 2).

[Deu85] D. E. Deutsch. "Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer". In: *Proceedings of the Royal Society of London Series A* 400.1818 (1985), pp. 97–117 (cit. on p. 16).

[DHHM06] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. "Quantum Query Complexity of Some Graph Problems". In: *SIAM Journal on Computing* 35.6 (2006), pp. 1310–1328 (cit. on pp. 33, 39).

[DJ92] D. Deutsch and R. Jozsa. "Rapid Solution of Problems by Quantum Computation". In: *Proceedings of the Royal Society of London Series A* 439.1907 (1992), pp. 553–558 (cit. on pp. 1, 7).

[Gro97] L. K. Grover. "Quantum Mechanics Helps in Searching for a Needle in a Haystack". In: *Physical Review Letters* 79.2 (1997), pp. 325–328 (cit. on pp. 1, 6, 7, 9).

[GSLW19] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe. "Quantum Singular Value Transformation and Beyond: Exponential Improvements for Quantum Matrix Arithmetics". In: *Proceedings of the 51st Symposium on Theory of Computing (STOC)*. 2019, pp. 193–204 (cit. on p. 11).

[HLŠ07] P. Høyer, T. Lee, and R. Špalek. "Negative Weights Make Adversaries Stronger". In: *Proceedings of the 39th Symposium on Theory of Computing (STOC)*. 2007, pp. 526–535 (cit. on pp. 28, 31).

[HM21] Y. Hamoudi and F. Magniez. "Quantum Time-Space Tradeoff for Finding Multiple Collision Pairs". In: *Proceedings of the 16th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC)*. 2021, 1:1–1:21 (cit. on p. 19).

[KK11] D. M. Kane and S. A. Kutin. "Quantum Interpolation of Polynomials". In: *Quantum Information & Computation* 11.1&2 (2011), pp. 95–103 (cit. on p. 15).

[Kre21] W. Kretschmer. "Lower Bounding the AND-OR Tree via Symmetrization". In: *ACM Transactions on Computation Theory* 13.1 (2021) (cit. on p. 17).

[LMR+11] T. Lee, R. Mittal, B. W. Reichardt, R. Špalek, and M. Szegedy. "Quantum Query Complexity of State Conversion". In: *Proceedings of the 52nd Symposium on Foundations of Computer Science (FOCS)*. 2011, pp. 344–353 (cit. on pp. 34, 36).

[MP69] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969 (cit. on p. 15).

[NC11] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. 10th ed. Cambridge University Press, 2011 (cit. on p. 2).

[Nis91] N. Nisan. "CREW PRAMs and Decision Trees". In: *SIAM Journal on Computing* 20.6 (1991), pp. 999–1007 (cit. on pp. 7, 10).

[NS94] N. Nisan and M. Szegedy. "On the Degree of Boolean Functions as Real Polynomials". In: *Computational Complexity* 4.4 (1994), pp. 301–313 (cit. on p. 12).

[Rei09] B. W. Reichardt. "Span Programs and Quantum Query Complexity: The General Adversary Bound Is Nearly Tight for Every Boolean Function". In: *Proceedings of the 50th Symposium on Foundations of Computer Science (FOCS)*. 2009, pp. 544–551 (cit. on pp. 31, 34, 37).

[Rei11] B. W. Reichardt. "Reflections for Quantum Query Algorithms". In: *Proceedings of the 22nd Symposium on Discrete Algorithms (SODA)*. 2011, pp. 560–569 (cit. on pp. 34, 38).

[Ros14] A. Rosmanis. "Lower Bounds on Quantum Query and Learning Graph Complexities". PhD thesis. University of Waterloo, 2014 (cit. on p. 2).

[Rub95] D. Rubinstein. "Sensitivity vs. Block Sensitivity of Boolean Functions". In: *Combinatorica* 15.2 (1995), pp. 297–299 (cit. on p. 10).

[Sim97] D. R. Simon. "On the Power of Quantum Computation". In: *SIAM Journal on Computing* 26.5 (1997), pp. 1474–1483 (cit. on pp. 1, 7).

[Špa06] R. Špalek. "Quantum Algorithms, Lower Bounds, and Time-Space Tradeoffs". PhD thesis. University of Amsterdam, 2006 (cit. on p. 2).

[ŠS06] R. Špalek and M. Szegedy. "All Quantum Adversary Methods are Equivalent". In: *Theory of Computing* 2.1 (2006), pp. 1–18 (cit. on pp. 31, 33).

[SSW23] A. A. Sherstov, A. A. Storozhenko, and P. Wu. "An Optimal Separation of Randomized and Quantum Query Complexity". In: *SIAM Journal on Computing* 52.2 (2023), pp. 525–567 (cit. on pp. 1, 7).

[Vaz98] U. Vazirani. "On the Power of Quantum Computation". In: *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 356.1743 (1998), pp. 1759–1768 (cit. on p. 9).

[Wol19] R. de Wolf. *Quantum Computing: Lecture Notes*. arXiv:1907.09415 [quant-ph]. 2019 (cit. on pp. 2, 34).

[YZ24] T. Yamakawa and M. Zhandry. "Verifiable Quantum Advantage without Structure". In: *Journal of the ACM* 71.3 (2024) (cit. on p. 1).

[Zha19] M. Zhandry. "How to Record Quantum Queries, and Applications to Quantum Indifferentiability". In: *Proceedings of the 39th International Cryptology Conference (CRYPTO)*. 2019, pp. 239–268 (cit. on p. 18).