Quantum lower bounds in the black-box model and quantum complexity of graph properties

Nemo Fournier, Jérémy Petithomme, Ugo Giocanti

We present in this report some results from [BBC+98] [Amb00] and [SYZ04].

Contents

Introduction

An interesting measure of the complexity of a boolean function $f : \{0,1\}^N \to \{0,1\}$ in computer science is its query complexity, which corresponds to the minimum number $D(f)$ such that for any input $x \in \{0,1\}^N$, we can determine the value of $f(x)$ knowing at most only $D(f)$ well chosen bits of x. In other words, if we have to compute $f(x)$ for an unknown value of $x \in \{0,1\}^N$, and $D(f,x)$ is the minimum number of bits x_i of x that we we have to query in order to do so, then $D(f)$ is defined as the maximum value that $D(f,x)$ can take for $x \in \{0,1\}^N$. We call this model the black-box model, as we can imagine that x his hidden in a black-box, to which we do our queries.

This model has been studied in many domains of computer science (communication, cryptography, graph theory ...), and a lot of results have been proved since the 1960's, especially when it comes to lower bounding $D(f)$. Black-box model have interesting declinations like randomised Black-box model, or quantum Black-box model. As we can expect, we can achieve an advantage in terms of query complexity when we work in quantum. A famous and historical example of this is the Grover algorithm [Gro96] which finds a marked element in a list of N elements with $\mathcal{O}(\sqrt{N})$ queries to the elements of the list. A natural question is then: how large can gaps of complexities between quantum models and other models be? We will see here how we can bound (asymptotically) those gaps in 1. In 1 and 2, some techniques will be exposed, to find lower bounds of quantum query complexities. We then show one application of them in 3.3 from [SYZ04] to estimate the asymptotic query complexity of some well-known class of functions, and will see what they imply for the special case of graph properties.

Notations

We will use the following notations for the rest of the report:

- 2^X: Power set of X.
- For a set X we denote by $|X|$ its cardinality.
- For a bitstring or a binary tuple $X = (x_1, \ldots, x_N)$ we denote by $|X|$ the number of x_i's equal to 1.
- $\binom{X}{k}$: Set of finite subsets of X of cardinality k.
- $\binom{n}{k}$: Cardinality of $\binom{X}{k}$ when $|X| = n$.
- $\mathfrak{S}_X$: Set of permutations of the set X
- $\mathfrak{S}_N$: Set of permutations of the set $\{1, ..., N\}$

1 Nemo: Quantum Lower Bounds by Polynomials [BBC$^+$98]

1.1 Context and Goals of the Article

This article is about query complexity of problems involving black-boxes and quantum networks. Published in 1998, it follows the study of the Deutsch's and Simon's problem which led to the design of quantum algorithms achieving an exponential quantum advantage ([DJ92] and [Sim97]). The authors of the paper propose a general framework for such black-box problems, and prove that in the general case of total functions, one cannot hope to achieve more than a polynomial quantum-advantage. This article use the *Polynomial Method*, that had already proven useful to derive lower-bounds in classical complexity theory.

In this review, I will introduce the framework used by the authors, explain the ideas that link polynomials and query problems and provide the proof of a general bound on the query complexity achieved using the polynomial method. I will then only state other general bounds without their proof before dwelling on the link between *query complexity* and the *block sensitivity*, that allows to show that the quantum speed-up is at most polynomial.

1.2 Black-box Model

Definition (*Black-box*)**.** A black-box is a N-tuple of Boolean variable $X = (x_0, x_1, \ldots, x_{N-1})$, such that it outputs x_i on input i.

We can notice that it is technically easy to encode a function $\tilde{X} : \{0,1\}^n \to \{0,1\}^m$ as a black box $X = (x_0, \ldots, x_{N-1})$ where $N = m2^n$: on input $i \in \{0,1\}^n$ seen as an n-bits number, we output $x_i x_{i+1} \ldots x_{i+m-1}$.

Definition (*Property*)**.** A property is a function $f : \{0,1\}^N \to \{0,1\}$.

The idea is that we can ask whether a black-box X satisfies or not a given property, and try to answer this question with as few queries to X as possible. For instance, we can consider the property "*The x_i's are all equal to 1*", which is encoded by the function $\mathrm{AND}(X) = x_0 \wedge x_1 \wedge \cdots \wedge x_{N-1}$, or the property "*There is an odd number of x_i's equal to* 1" using the PARITY function: $\mathrm{PARITY}(X) = (-1)^{|X|}$, where $|X|$ denotes the number of Boolean variables of the tuple X equal to 1.

Those functions (AND and PARITY) are said to be *total*, since they are defined for any tuple X. We can also consider properties defined only on some subset of $\{0,1\}^n$, that we will call *partial* functions. This is the case when we study a property under some *promise* on the black-box: that is we disregard a whole class of Boolean tuples. For instance, formalizing the Deutsch's problem in this formalism boils down to stuying a property only defined for black-boxes that are either constant or balanced.

1.3 Decision Trees and Quantum Network

Definition (*Decision Tree*)**.** A decision tree is a classical algorithm that computes a property f of some black-box X by performing (adaptative) queries on X. We can represent such an algorithm as a tree whose nodes are queries giving birth to two children (one for each possible outcome of the query) and whose leaves answers whether $f(X) = 0$ of $f(X) = 1$. The cost of such an algorithm is the depth of the associated decision tree. The *decision-tree-complexity* of f (denoted $D(f)$) is thus the cost of the best decision tree computing f.

Definition (*Quantum Oracle*)**.** A quantum oracle O on m qubits implementing the black-box $X = (x_0, \ldots, x_{N-1})$ is a unitary transformation that maps the basis states $|i, b, z\rangle$ to $|i, b \oplus x_i, z\rangle$ for $i \in \{0, \ldots, N-1\}$ seen as a bitstring of length $\lceil \log N \rceil$, $b \in \{0,1\}$ and z an arbitrary bitstring of length $m - \lceil \log N \rceil - 1$.

Definition (*Quantum Network*)**.** A quantum network is the quantum analog of a decision tree. A quantum network performing T queries is represented as a sequence $U_0, O_1, U_1, \ldots, U_{T-1}, O_T, U_T$, where the U_i's represent unitary transformations, and O_i's represent queries to a quantum oracle implementing X. A computation corresponds to applying successively those transformations from a fixed initial state (*e.g.* $|0\rangle^{\otimes m}$). At the end of the computation, the final state is measured and its rightmost bit is considered as the output (*i.e.* as the value of $f(X)$).

A quantum network is said to compute f *exactly* if, for every X, it outputs $f(X)$ with certainty, and we denote $Q_E(f)$ the minimum number of query for a quantum network computing f exactly. If the output of the network is correct with probability at least $2/3$ for every X, we say that the network computes f with *bounded error-probability*, and denotes $Q_2(f)$ the minimum number of query of such a network. We also introduce the *zero-error* setting where we allow the network to be inconclusive (this is controlled by the second rightmost bit of the final measurement), but whenever it is conclusive (which has to happen with probability at least $1/2$ for every X) the answer (given by the rightmost bit of the final measurement) should be correct.

1.4 Polynomial and Boolean Functions

The core of the proofs presented by the authors are based upon a correspondance that we can make between Boolean functions and polynomials.

Definition (*Representation and Approximation by Polynomials*)**.** Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. A N-variate polynomial p is said to represent f if for all $X \in \{0,1\}^N$, $p(X) = f(X)$. A N-variate polynomial p is said to approximate f if for all $X \in \{0,1\}^N$, $|p(X) - f(X)| \leq 1/3$. We denote by $\deg(f)$ the smallest degree of a polynomial representing f, and by $\widetilde{\deg}(f)$ the smallest degree of a polynomial approximating f.

For instance $p(X) = 1 - (1 - x_0)(1 - x_1)\dots(1 - x_{N-1})$ represents the function OR, while $p(X) = \frac{1}{3}x_0 + \frac{1}{3}x_1$ approximates the function AND for 2 variables. One can notice that since for any $k > 0$ and $x \in \{0,1\}$ we have $x^k = x$, we can restrict ourselves to multilinear p.

Definition (*Symmetrization*)**.** The symmetrization p^{sym} of a polynomial $p : \mathbf{R}^n \to \mathbf{R}$ averages p over all the permutations of its input. It is defined as

$$p^{sym}(X) = \frac{\sum_{\pi \in \mathfrak{S}_N} p(\pi(X))}{N!} \qquad \begin{array}{c}\text{where for } \pi \in \mathfrak{S}_N \text{ and } X = (x_0, \dots x_{N-1}) \\ \pi(X) = (x_{\pi(0)}, \dots, x_{\pi(N-1)})\end{array}$$

One of the polynomial method's strength is that the space of polynomials is a very well studied space, and a quantity of powerful theorems have already been proved. We will now state some of the theorems used by the authors in their paper.

Theorem 1.1 (Nisan, Szegedy [NS94]) If f is a Boolean function that depends on N variables (that is for every $i < N$, there exists an $X = (x_0, \dots, x_{N-1})$ such that $f(x_0, \dots, x_i, \dots, x_{N-1}) \neq f(x_0, \dots, \overline{x_i}, \dots, x_{N-1})$, where $\overline{x_i}$ denotes the binary negation of x_i or, said differently, f depends on each of its variables), then

$$\deg(f) \geq \log N - \mathcal{O}(\log \log N)$$

Theorem 1.2 (Minsky, Papert [MP88]) If $p : \mathbf{R}^n \to \mathbf{R}$ is a multilinear polynomial of degree d, then there exists a polynomial $q : \mathbf{R} \to \mathbf{R}$ of degree at most d such that for all $X \in \{0,1\}^n, p^{sym}(X) = q(|X|)$.

This last result is rather intuitive, and states that a symmetrized polynomial does not depend on the arrangement of the input, but only on the weight of the input (as in the number of bits set to 1). This notion can be extended to Boolean function as follows.

Definition (*Symmetric Boolean function*)**.** We say that a Boolean function $f : \{0,1\}^N \to \{0,1\}$ is symmetric if it is invariant under permuation of its input X, and thus only depends on the weight $|X|$ of its input. For such an f, we denote f_k the value of $f(X)$ taken on an input X such that $|X| = k$.

We introduce the quantity $\Gamma(f) = \min\{|2k - N + 1| : 0 \leq k < N \text{ and } f_k \neq f_{k+1}\}$. Intuitively this is low if f "jumps" near the middle, *i.e.* around some $k \approx N/2$.

Theorem 1.3 (Paturi [Pat92]) If $f : \{0,1\}^n \to \{0,1\}$ is a non-constant symmetric Boolean function, then

$$\widetilde{\deg}(f) \in \Theta\left(\sqrt{N(N - \Gamma(f))}\right)$$

1.5 A bound on the zero-error setting

I will present in this section a lower bound derived in the setting of zero-error, that will unveil how we can link polynomial representation of Boolean functions with quantum query complexity.

Lemma 1.1 Let $\mathcal{N}$ be a quantum network on m qubits that makes T queries to a black-box X of size N. Then there exist 2^m complex-valued N-variate multilinear polynomials $p_0, \ldots, p_{2^m-1}$, each of degree at most T, such that for any black-box X, the final state of the computation can be expressed as

$$\sum_{k \in \{0,1\}^m} p_k(X) \, |k\rangle$$

Proof. This result can be proved by induction on the number of queries T.

- ▷ Suppose that a black box performs $T = 0$ queries. The network is thus only composed of a unitary U_0 acting on the initial state. The amplitudes of the final state thus depend only on U_0 and on the initial state, but not on X, and are thus polynomials of degree 0 in X.

- ▷ Suppose that the network performs $T + 1$ queries. It can be written as the successive application of U_0, O_1, U_1, …, U_{T-1}, O_T, U_T, O_{T+1}, U_{T+1}. By the induction hypothesis, we know that the state obtained after performing the first T queries on X and applying U_T can be written as $|\phi\rangle = \sum_{k \in \{0,1\}^n} p_k(X) \, |k\rangle$ with each p_k of degree at most T. The final state is $|\psi\rangle = U_{T+1} O_{T+1} \, |\phi\rangle$. Let $k \in \{0,1\}^m$ (*i.e.* such that $|k\rangle$ is a basis state). There are some bitstrings i, b, z of length respectively $\lceil \log N \rceil$, 1, $m - \lceil \log N \rceil - 1$ such that $|k\rangle = |i, b, z\rangle$. We denote k' such that $|k'\rangle = |i, \bar{b}, z\rangle$. By definition of a quantum oracle, the final query maps the basis state $|i, b, z\rangle$ to $|i, b \oplus x_i, z\rangle$. Thus the amplitude of $|k\rangle$ after this query is $(1 - x_i)p_{k'}(X) + x_i p_k(X)$ (if the value of x_i is 1, $|k\rangle$ is mapped to $|k'\rangle$, and remains the same otherwise), which is a polynomial in X of degree at most $T + 1$ (since $p_{k'}$ and p_k are of degree at most T). Hence the amplitude of each basis state after the $(T + 1)$-th query is obtained as a polynomials in X of degree at most $T + 1$. Applying U_{T+1} only leads to linear combinations of the amplitudes, which thus remain polynomial in X of degree at most $T + 1$. Finally, we can make all of those polynomials multilinear by replacing x_i^k by x_i.

■

Lemma 1.2 Let $\mathcal{N}$ be a quantum network that makes T queries to a black-box X and B be a set of basis states. Then there exists a real-valued multilinear polynomial $P : \mathbf{R}^n \to \mathbf{R}$ of degree $2T$, which equals the probability that the final measurement of the computation yields a state from B.

Proof. This directly follows from the previous lemma: the polynomial that corresponds to this probability is simply $P(X) = \sum_{k \in B} |p_k(X)|^2$, that we make multilinear by replacing x_i^k by x_i. ■

Theorem 1.4 If f is a Boolean function, then $Q_E(f) \geq \deg(f)/2$.

Proof. In the zero-error setting, we know that the probability that the final measurement yields a state with a righmost bit equal to 1 should be 1 when $f(X) = 1$, and 0 if $f(X) = 0$ (otherwise an error may occur). Thus applying the previous lemma with the network achieving $Q_E(f)$ queries and the set B of the basis states having a rightmost bit equal to 1 yields a polynomial P of degree at most $2Q_E(f)$ that represents f exactly. Hence $\deg(f) \leq \deg P \leq 2Q_E(f)$. ■

Combining this with the aforementionned Nisan and Szegedy's theorem gives the following theorem.

Theorem 1.5 If f is a Boolean function that depends on N variables, then

$$Q_E(f) \geq (\log N)/2 - \mathcal{O}(\log \log N)$$

1.6 Other bounds derived by the authors

The authors derive other bounds, some for the special case where f is a symmetric Boolean function as well as some general bounds for the zero-error and bounded-error setting. I will only states those results here.

Theorem 1.6 If f is a non-constant and symmetric Boolean function, then $Q_0(f) \geq (N + 1)/4$

Theorem 1.7 If f is a non-constant and symmetric Boolean function, then $Q_E(f) \geq N/2 - \mathcal{O}(N^{0.548})$. Moreover if $N+1$ is prime, then $Q_E(f) \geq N/2$.

Theorem 1.8 If f is a Boolean function, then $Q_2(f) \geq \widetilde{\deg}(f)/2$.

Theorem 1.9 If f is a non-constant and symmetric Boolean function, then $Q_2(f) \in \Theta\left(\sqrt{N(N-\Gamma(f))}\right)$.

Now that we have derived such lower bounds on the quantum-query complexity, we would like to assess the speedup that quantum algorithm could provide. More precisely, using the notation that we have introduced, we would like to compare $D(f)$ and $Q_{\{E,0,2\}}(f)$.

Using some previous results from [NS94] and the previous bounds we could already conclude that $D(f) \in \mathcal{O}(Q_2(f)^8)$, that is the speedup is at most polynomial, with an exponent of 8. But, introducing the notion of *block-sensitivity* and of *certificate-complexity*, we will be able to show that we can lower this exponent to 6.

1.7 Block sensitivity

Let $f : \{0,1\}^N \to \{0,1\}$ be a function, $X \in \{0,1\}^N$ a vector and $B \subseteq \{0,\ldots,N-1\}$ a set of indices. We will denote by X^B the vector obtained by flipping the bits of X corresponding to indices in B. We say that f is sensitive to B on X if $f(X) \neq f(X^B)$.

Definition (*Block-sensitivity*)**.** The block-sensitivity $bs_X(f)$ of f on X is the maximum number t for which there exist t disjoint sets of indices $B_1,\ldots,B_t$ such that f is sensitive to each B_i on X. The block-sensitivity $bs(f)$ of f is defined as $bs(f) = \max_{X\in\{0,1\}^N} bs_X(f)$.

For example $bs(\mathrm{OR}) = N$ (to see this take $X = (0,\ldots,0)$ and $B_i = \{i\}$ for $i \in \{0,\ldots,N-1\}$). Having introduced this notion we can state and prove the following theorem, which explicits the relation between block sensitivity and quantum query complexity.

Theorem 1.10 If f is a Boolean function, then $Q_E(f) \geq \sqrt{bs(f)/8}$ and $Q_2(f) \geq \sqrt{bs(f)/16}$.

Proof. We will only detail the proof of the inequality involving $Q_2(f)$, the other one is analog and fairly easier. During the proof, we will also need a theorem from [RC66], that states as follows

> **Theorem** (Rivlin, Cheney [RC66])**.** *If $p : \mathbf{R} \to \mathbf{R}$ is a polynomial such that there exists a, b and c such that for every $i \in \{0,\ldots,N\}$, $a \leq p(i) \leq b$ and $|p'(x)| \geq c$ for some real number $x \in [0,N]$, then* $\deg(p) \geq \sqrt{\frac{cN}{c+b-a}}$

Consider a network $\mathcal{N}$ that uses $T = Q_2(f)$ queries and computes f with error-probability $\leq 1/3$. From the second lemma we stated in section 1.5, their exists a polynomial $P : \mathbf{R}^N \to \mathbf{R}$ of degree at most $2T$ that equals the probability that the rightmost bit of the measured state is 1 (take B the set of basis states with rightmost bit equal to 1). Since having the rightmost bit of the final measurement equal to one means that we output 1, and that $\mathcal{N}$ computes f with error-probability $\leq 1/3$, we have that

- If $f(X) = 1$, then $P(X) \geq 2/3$
- If $f(X) = 0$, then $P(X) \leq 1/3$

Hence for all $X \in \{0,1\}^N$, $|P(X) - f(X)| \leq 1/3$, that is P approximates f. Also observe that $P(X) \leq 1$ for any $X \in \{0,1\}^N$ because P represents a probability on those inputs.

Let $b = bs(f)$ and X and $B_0,\ldots,B_{b-1}$ be the input and subset of indices achieving the block-sensitivity b for f. Let us consider a variable $Y = (y_0,\ldots y_{b-1}) \in \mathbf{R}^b$ and define a variable $Z = (z_0,\ldots,z_{N-1}) \in \mathbf{R}^N$ for some given Y (that we will set later on) as follows:

$$\forall j \in \{0,\ldots,N-1\}, z_j = \begin{cases} y_i \text{ if } x_j = 0 \text{ and } \exists i \in \{0,\ldots,b-1\} : j \in B_i \\ 1 - y_i \text{ if } x_j = 1 \text{ and } \exists i \in \{0,\ldots,b-1\} : j \in B_i \\ x_j \text{ if } \forall i \in \{0,\ldots,b-1\}, j \notin B_i \end{cases}$$

For the following, and without loss of generality, we assume that $f(X) = 0$.

Observe that Z is built such that if $Y = (0,\ldots,0)$, then $Z = X$, and if Y is such that it has only one non-null coordinate $y_i = 1$, then $Z = X^{B_i}$, and thus, since f is sensitive to B_i on X, $f(X^{B_i}) = 1 - f(X) = 1$. More generally, we can express Z as follows

$$Z = X^{B_Y} \text{ where } B_Y = \bigcup_{\substack{i \in \{0,\ldots,b-1\} \\ y_i = 1}} B_i$$

Since Z only depends on Y for X given, we denote $q : \mathbf{R}^b \to \mathbf{R}$ the polynomial $q(Y) = P(Z)$. From the definition of Z (involving only monomials of degree 1 in X and Y) and the degree of P, we have that $\deg q \leq 2T$. Moreover, from the previous observations, q satisfies the following properties:

- For all $Y\{0,1\}^d$, since P (and thus q) represents a probability, $0 \leq q(Y) \leq 1$.
- $|q((0,\ldots,0)) - 0| = |P(X) - f(X)| \leq 1/3$ hence $0 \leq q((0,\ldots,0)) \leq 1/3$.
- For Y such that $y_i = 1$ and for $j \neq i, y_j = 0$, $|q(Y) - 1| = \left|P\left(X^{B_i}\right) - f\left(X^{B_i}\right)\right| \leq 1/3$. Thus $2/3 \leq q(Y) \leq 1$ for $|Y| = 1$.

Let us now symmetrize q over $\{0,1\}^b$ into a single-valuate polynomial r of degree $\leq 2T$. Clearly, for every integer $i \in \{0,\ldots,b\}$, $0 \leq r(i) \leq 1$. Moreover, from the previous properties, $r(1) \geq 2/3$ and $r(0) \leq 1/3$. Hence, by the mean value theorem, there exist $x \in]0,1[$ such that $r'(x) = \frac{r(1)-r(0)}{1-0} \geq \frac{2/3-1/3}{1-0} \geq 1/3$. Thus applying Rivlin and Cheney's theorem gives

$$\deg(r) \geq \sqrt{\frac{1/3 \times b}{1/3+1}} \geq \sqrt{b/4}$$

Hence $2T \geq \sqrt{b/4}$ and thus $T \geq \sqrt{b/16}$, which concludes, since $T = Q_2(f)$ and $b = bs(f)$. ■

1.8 Certificate-complexity

We introduce a new notion of complexity that will allow to link quantum query complexity to classical query complexity.

Definition (*Certificate*)**.** Let f be a function $f : \{0,1\}^n \to \{0,1\}$. A 1-certificate (resp. 0-certificate) is an assignment $C : S \to \{0,1\}$ of a subset S of the inputs of f such that whenever X is consistent with C, $f(X) = 1$ (resp. $f(X) = 0$). The *size* of C is $|S|$.

Definition (*Certificate-complexity*)**.** The certificate complexity $C_X(f)$ of f on X is the size of a smallest $f(X)$-certificate that agrees with X, and the overall certificate complexity $C(f)$ of f is the maximum of $C_X(f)$ over $X \in \{0,1\}^n$. The 1-certificate complexity $C^{(1)}(f)$ of f is the maximum of $C_X(f)$ over the X's such that $f(X) = 1$.

For instance, with f being the OR function, we have $C^{(1)}(f) = 1$, because an input X such that $\mathrm{OR}(X) = 1$ has at least one bit set to 1, and it is sufficient to set this force this one bit to 1 to have a valid 1-certificate. But on the input $X = (0,\ldots,0)$, $f(X) = 0$, and it is easy to see that a 0-certificate for OR has to have a size of N.

Intuitively, this notion provides an insight as to "how many bits do we need to check in order to be sure that the value of f is either 0 or 1". The authors indeed prove the following theorems, that show how this quantity can be related to other notions of complexity on boolean functions.

Theorem 1.11 $C^{(1)}(f) \leq C(f) \leq bs(f)^2$

Theorem 1.12 $D(f) \leq C^{(1)}(f)bs(f)$

Hence we deduce the main theorem of the article from those two theorems and from the fact from section 1.7 stating that $Q_2(f) \geq \sqrt{bs(f)/16}$, *i.e.* $bs(f) \leq 16Q_2(f)^2$.

Theorem 1.13 $D(f) \leq 4096Q_2(f)^6$, *i.e.* the quantum speedup is at most polynomial of degree 6.

The authors then indeed caracterize in the remaining of the article the query complexity for some simple *total* functions, for which the quantum advantage is even smaller than a degree 6 order: for $f = \mathrm{AND}$ or $f = \mathrm{OR}$, $Q_2(f) = \Theta(\sqrt{N})$ (already known results), for $f = \mathrm{PARITY}$, $Q_2(f) = N/2$ (new result) and for $f = \mathrm{MAJORITY}$, $Q_2(f) = \Theta(N)$ (new result).

2 Jérémy: Quantum Lower Bounds by Quantum Arguments [Amb00]

2.1 Introduction

The main goal of the article is to prove some lower bounds on query algorithms using some quantum arguments. In addition to prove some stronger bounds for some algorithms, the idea is to provide unified proofs for several known lower bounds, which have previously been proven with many different classical proofs, such as Grover's algorithm, or inverting a permutation. To do so, the article introduces the notion of "quantum adversary", which, instead of running the algorithm and slightly changing the input while running, runs the algorithm with a superposition of inputs. The idea is to use an algorithm and an oracle part, which are independant at the beginning, and that becomes entangled after some queries to the oracle. Then we can obtain lower bound on quantums algorithms by bounding the number of queries needed to entangle the alorithm and the oracle parts.

Here, I will first describe the new tools introduced in the article and the ideas behing it. I will then provide some ideas for the proofs using those tools, and state the new bounds proven in the paper.

2.2 Model and Ideas

2.2.1 The model

We consider a Boolean function (or property) $f : \{0,1\}^N \to \{0,1\}$, an oracle O and a quantum network:

$$U_0 \to O \to U_1 \to O \to ... \to U_{T-1} \to O \to U_T$$

where U_i's are unitary transformations not depending on the input and O are oracle transformations (queries).
We will denote O_x a oracle transformation corresponding to an input x. Also, we will use an alternative definition of a quantum oracle

Definition (*Quantum Oracle*)**.** Let $x = (x_1, ..., x_N)$, a quantum oracle O_x is a unitary transformation mapping the basis states $|i, b, z\rangle$ to $(-1)^{b \cdot x_i} |i, b, z\rangle$ for $i \in \{1, \ldots, N\}$.

This defintion is known to be equivalent to the one given in the previous section, up to a constant factor. We will use this one in the rest of this section for some technical convinience.

Definition (*Error of a quantum network*)**.** We say that a quantum network computes f with bounded error if, for every $x = (x_1, ...x_N)$, the probability that the rightmost bit of $U_T O U_{T-1} ... O U_0 |0\rangle$ equals $f(x_1, ..., x_N)$ is at least $1 - \epsilon$ for some $\epsilon > \frac{1}{2}$.

2.2.2 The idea

In this section we will describe more formally the idea behind the "quantum adversary".
Let S be a subset of $\{0,1\}^N$. We consider a bipartite system $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_I$ where $\mathcal{H}_A$ is the workspace of the algorithm A, and $\mathcal{H}_I$ is an "input space" spanned by basis vectors $|x\rangle$ corresponding to inputs $x \in S$. We transform the sequence $U_T O U_{T-1} ... O U_0$ of transformations on the algorithm A into a sequence $U'_T O' U'_{T-1} ... O' U'_0$ of transformations on $\mathcal{H}$, where $U'_i = U_i \otimes I$ and O' is simply O_x on $\mathcal{H} \otimes |x\rangle$. At the beginning, the algorithm is in state $|0\rangle$, so the initial state of the system is

$$|\psi_{start}\rangle = |0\rangle \otimes \sum_{x \in S} \alpha_x |x\rangle$$

And the final state is

$$|\psi_{end}\rangle = \sum_{x \in S} \alpha_x |\psi_x\rangle \otimes |x\rangle$$

where $|\psi_x\rangle$ is the final state of $U_T O U_{T-1} ... O U_0 |0\rangle$ on input x.

Let $m = |S|$, note that if $\alpha_x = \frac{1}{\sqrt{m}}$ for all $x \in S$, with bounded error, then the final state is

$$\frac{1}{\sqrt{m}} \sum_{x \in S} ((1 - \epsilon) |x\rangle |\varphi_x\rangle + |\psi'_x\rangle) \otimes |x\rangle$$

where $|\varphi_x\rangle$ are the algorithm workbits.
We see that this is a highly entangled state. Notice also that if we have probability error $\epsilon = 0$, then the algorithm always gives the right answer, then the final state is

$$\frac{1}{\sqrt{m}} \sum_{x \in S} |x\rangle |\varphi_x\rangle \otimes |x\rangle$$

which is fully entangled.

Now, we want to bound the entanglement, we trace out $\mathcal{H}_A$ from the states $|\psi_{start}\rangle$ and $|\psi_{end}\rangle$ to obtain mixed states over $\mathcal{H}_I$. We denote ρ_{start} and ρ_{end} the density matrices associated to these states. For the starting state $\sum_{x \in S} \alpha_x |x\rangle$, $(\rho_{start})_{xy} = \alpha_x^* \alpha_y$. For ρ_{end}, the following lemma states:

Lemma 2.1 Let A be an algorithm that computes f with probability at least $1 - \epsilon$. Let x, y be such that $f(x) \neq f(y)$. Then,

$$|(\rho_{end})_{xy}| \leq 2\sqrt{\epsilon(1-\epsilon)} |\alpha_x||\alpha_y|$$

The proof of this lemma introduces the main idea of the paper and gives intuitions about how the other proofs in the paper will work. I detail it here:

Proof. We denote $|\psi_x\rangle$ and $|\psi_y\rangle$ the final states of the algorithm A (over $\mathcal{H}_A$) on inputs x and y. We consider the two parts of the algorithm over $\mathcal{H}_A$, the answer bits (where the answer is measured at the end of the computation) and the workbits (used to compute the result but which are not considered in the output). We consider a basis $(|z\rangle |v\rangle)$ of $\mathcal{H}_A$ where $(|z\rangle)$ is a basis for the answer part, and $(|v\rangle)$ a basis for the working part. We can now express the states $|\psi_x\rangle$ and $|\psi_y\rangle$ in this basis and get:

$$|\psi_x\rangle = \sum_{z,v} a_{z,v} |z\rangle |v\rangle \text{ and } |\psi_y\rangle = \sum_{z,v} b_{z,v} |z\rangle |v\rangle$$

We have seen that the final state of the algorithm is

$$|\psi_{end}\rangle = \sum_{x \in S} \alpha_x |\psi_x\rangle \otimes |x\rangle$$

By tracing out $\mathcal{H}_A$ from this state, we get in the $|z\rangle |v\rangle$ basis:

$$(\rho_{end})_{xy} = \alpha_x \alpha_y \sum_{z,v} a_{z,v}^* b_{z,v}$$

We now want to bound $|\sum_{z,v} a_{z,v}^* b_{z,v}|$ so we will be done. We have:

$$\left| \sum_{z,v} a_{z,v}^* b_{z,v} \right| \leq \sum_{z,v} |a_{z,v}||b_{z,v}|$$

We now divide our sum in two parts, considering the cases $z = f(x)$ and $z \neq f(x)$ for our input x. We get then:

$$\sum_{z,v:z=f(x)} |a_{z,v}||b_{z,v}| + \sum_{z,v:z\neq f(x)} |a_{z,v}||b_{z,v}|$$
$$\leq \sqrt{\sum_{z,v:z=f(x)} |a_{z,v}|^2} \sqrt{\sum_{z,v:z=f(x)} |b_{z,v}|^2} + \sqrt{\sum_{z,v:z\neq f(x)} |a_{z,v}|^2} \sqrt{\sum_{z,v:z\neq f(x)} |b_{z,v}|^2}$$

Notice that the two quantities $\sum_{z,v:z\neq f(x)} |a_{z,v}|^2$ and $\sum_{z,v:z=f(x)} |b_{z,v}|^2$ are respectively the probabilities that makes an error on input x and on input y, hence they are both lower than ϵ. We will denote them ϵ_1 and ϵ_2. We now have a bound:

$$\sqrt{(1-\epsilon_1)\epsilon_2} + \sqrt{\epsilon_1(1-\epsilon_2)}$$

which is maximized when $\epsilon_1 = \epsilon_2 = \epsilon$, hence we can conclude:

$$|(\rho_{end})_{xy}| \leq 2\sqrt{\epsilon(1-\epsilon)|\alpha_x||\alpha_y|}$$

■

This lemma gives us an idea of how much the elements of ρ_{start} must be changed to become the corresponding element in ρ_{end}. Therefore, to lower bound the number of queries, we bound how much one query changes ρ_{xy}. Then we combine this bound with the one given by the Lemma to get a lower bound on the total number of queries needed to compute ρ_{end} from ρ_{start}.

The next sections of the paper are theorems giving some lower bounds for different algorithm. Therefore, the next parts of this section will be kind of a survey of the theorems presented. For each of them, I will brievely decribe how to use the technique presented before to prove the statement.

2.3 Some lower bounds

In this section I give some lower bounds for different problems. For each case, I will give a short description of the problem before the thorem, and provide an idea of proof.
But before doing so, we will prove a general lower bound theorem, which we will apply to all the next problems. This theorem is the most important of the whole paper because it is the one which unifies many lower bounds proofs.

2.3.1 The general lower bound

To prove the lower bound theorem, we will before proove a lower bound on unordered search. This is because the proofs are similar but the proof of Theorem 2.1 is much simpler than the one of Theorem 2.2.

Problem Given N Boolean variables $x_1, ...x_N \in 0, 1$, we want to find i such that $x_i = 1$

Theorem 2.1 Any quantum algorithm that finds i with probability $1 - \epsilon$ uses $\Omega(\sqrt{N})$ queries.

Proof. We define S the set of inputs with one x_i equal to 1 and the rest to 0. We have then that $|S| = N$ and $\mathcal{H}_I$ is a N-dimensional space.
We denote ρ_k the density matrix of $\mathcal{H}_I$ after k queries. Then we consider

$$S_k = \sum_{x,y,x\neq y} |(\rho_k)_{xy}|$$

As the idea we described at the end of Section 2.2.2 we will show that :

1. $S_0 = N - 1$
2. $S_T \leq 2\sqrt{\epsilon(1-\epsilon)}(N-1)$
3. $S_{k-1} - S_k \leq 2\sqrt{N-1}$ for all $k \in \{1, ..., T\}$.

with T the number of queries.
If we have those three points, then we can lower bound the number of queries T by $(1-2\sqrt{\epsilon(1-\epsilon)})\frac{\sqrt{N-1}}{2}$.
The fisrt point comes from the fact that $\rho_0 = \rho_{start}$ and that each non-diagonal entry of this $N \times N$ density matrix is $\frac{1}{N}$. And the second point come directly from Lemma 2.1. To show the third point, first notice that

$$S_{k-1} - S_k \leq \sum_{x,y,x\neq y} |(\rho_{k-1})_{xy} - (\rho_k)_{xy}|$$

We now just need to bound $\sum_{x,y,x\neq y} |(\rho_{k-1})_{xy} - (\rho_k)_{xy}|$. This can be done studying the behavior of the (ρ_k) and of the states $|\psi_k\rangle$.
Finally we end with

$$\sum_{x,y,x\neq y} |(\rho_{k-1})_{xy} - (\rho_k)_{xy}| \leq 2\sqrt{N-1}$$

which completes the proof. ■

Now here is the general lower bound theorem.

Theorem 2.2 Let $f(x_1, ..., x_N)$ be a function of n $\{0,1\}$-valued variables and X,Y be two sets of inputs such that $f(x) \neq f(y)$ if $x \in X$ and $y \in Y$. Let $R \subset X \times Y$ be such that:

1. For every $x \in X$ there exist at least m different $y \in Y$ such that $(x, y) \in R$.
2. For every $y \in Y$ there exist at least m' different $x \in X$ such that $(x, y) \in R$.
3. For every $x \in X$ and $i \in \{1, ..., n\}$ there are at most l different $y \in Y$ such that $(x, y) \in R$ and $x_i \neq y_i$.
4. For every $y \in Y$ and $i \in \{1, ..., n\}$ there are at most l' different $x \in X$ such that $(x, y) \in R$ and $x_i \neq y_i$.

Then any algorithm computing f uses $\Omega\left(\sqrt{\frac{mm'}{ll'}}\right)$

Proof. The proof of this theorem is the same idea as the previous pne.
We define a set $S = X \cup Y$ of inputs, and we consider the superposition:

$$\frac{1}{\sqrt{2|X|}} \sum_{x \in X} |x\rangle + \frac{1}{\sqrt{2|Y|}} \sum_{y \in Y} |y\rangle$$

over the sets of inputs. We define

$$S_i = \sum_{x,y,(x,y) \in R} |(\rho_k)_{xy}|$$

and we show:

1. $S_0 - S_T \geq (1 - 2\sqrt{\epsilon(1-\epsilon)}\sqrt{mm'})$
2. $S_{k-1} - S_k \leq \sqrt{ll'}$

where T is the number of queries.
Again the first point follows directly from Lemma 2.1, and it remains to show the second point with some manipulations over the states. ■

2.3.2 Block sensitivity

Theorem 2.2 generalizes the block sensitivity bound described is Section 1.7. Indeed, the following theorem:

Theorem 2.3 Let f be any Boolean function (or property). Then, any quantum algorithm computing f uses $\Omega(\sqrt{bs(f)})$ queries.

can be seen as a particular case of Theorem 2.2. Indeed, let x be the input on which f achieves $bs(f)$, and let $X = \{x\}$, $Y = \{x^{(S_1)}, ..., x^{(S_{bf(s)})}\}$. Finally let $R = \{(x, x^{(S_1)}), (x, x^{(S_2)}), ..., (x, x^{(S_{bs(f)})})\}$. We see that the conditions of Theorem 2 are verified, indeed we have $m = bs(f)$, $m' = 1$, $l = 1$ and $l' = 1$.
Apply the theorem and get the following lower bound:

$$\Omega\left(\sqrt{\frac{mm'}{ll'}}\right) = \Omega(\sqrt{bs(f)})$$

2.3.3 AND and OR's

Let $x_1, ..., x_N$ be N boolean variables, we consider a function AND and ORs:

$$f(x_1, ..., x_N) = (x_1 OR x_2 ... OR x_{\sqrt{N}}) AND ... AND (x_{N-\sqrt{N}+1} OR ... OR x_N)$$

Theorem 2.4 Any quantum algorithm computing AND and ORs uses $\Omega(\sqrt{N})$ queries.

Proof. Application of Theorem 2.2 ■

This theorem gives us a better bound than the bound given by some polynomials because the block sensitivity of f is $\Theta(\sqrt{bs(f)}) = \Theta(\sqrt{N})$. Hence this method gives a better bound than block-sensitivity method.

2.3.4 Theorem of Nayak and Wu

Theorem 2 can also give a better bound than the block sensitivity method for the theorem of Nayak and Wu, which statement is the following:

Theorem 2.5 Let $f : \{1, ..., n\} \to \{0, 1\}$ be a Boolean function that is equal to 1 eithet at exactly $\frac{n}{2}$ ponts of the domain or at exactly $(1 + \epsilon)\frac{n}{2}$ points. Then, any quantum algorithm that determines whether the number of points is $\frac{n}{2}$ or $(1 + \epsilon)\frac{n}{2}$ uses $\Omega(\frac{1}{\epsilon})$ queries.

2.3.5 More general result

Note that for some other problems, we would need a more general theorem than Theorem 2.2. For example, we need this extention of the theorem to prove some bounds like the one of inverting a permutation. Here is the statement of the most general theorem of the paper:

Theorem 2.6 Let $f(x_1, ..., x_N)$ be a function of n $\{0, 1\}$-valued variables and X,Y be two sets of inputs such that $f(x) \neq f(y)$ if $x \in X$ and $y \in Y$. Let $R \subset X \times Y$ be such that:

1. For every $x \in X$ there exist at least m different $y \in Y$ such that $(x, y) \in R$.
2. For every $y \in Y$ there exist at least m' different $x \in X$ such that $(x, y) \in R$.

- Let $l_{x,i}$ be the number of $y \in Y$ such that $(x, y) \in R$ and $x_i \neq y_i$
- Let $l_{y,i}$ be the number of $x \in X$ such that $(x, y) \in R$ and $x_i \neq y_i$
- Let l_{max} be the maximum of $l_{x,i}ly, i$ over all $(x, y) \in R$ and $i \in \{1, ..., N\}$ such that $x_i \neq y_i$.

Then any algorithm computing f uses $\Omega\left(\sqrt{\frac{mm'}{l_{max}}}\right)$

Proof. The proof of this theorem is very similar to the proof of Theorem 2.2.
Indeed, we define a set $S = X \cup Y$ of inputs, and we consider the superposition:

$$\frac{1}{\sqrt{2|X|}} \sum_{x \in X} |x\rangle + \frac{1}{\sqrt{2|Y|}} \sum_{y \in Y} |y\rangle$$

over the sets of inputs. We define

$$S_i = \sum_{x,y,(x,y) \in R} |(\rho_k)_{xy}|$$

and we show:

1. $S_0 - S_T \geq (1 - 2\sqrt{\epsilon(1 - \epsilon)}\sqrt{mm'})$
2. $S_{k-1} - S_k \leq \sqrt{l_{max}}$

■

We can easily see that this is a generalization of Theorem 2.2, because we just have

$$l = \max_{x \in X, i} l_{i,x} \qquad \text{and} \qquad l' = \max_{y \in Y, i} l_{i,y}$$

and it is obvious that

$$\max_{(x,y) \in R, x_i \neq y_i, i} l_{i,x} l_{i,y} \leq \max_{x \in X, i} l_{i,x} \max_{y \in Y, i} l_{i,y}$$

This paper gives a nice method to prove some lower bounds on quantum algorithm numer of queries. All the bounds presented in the paper are shown in a very similar way. Moreover, this method have permits to find some stronger bounds fro the AND and ORs and the inverting a permutation problems. The authors conclude giving some open problems which could maybe also be proven with the same kind of proof, as the collision problem or the binary search problem.

3 Ugo: Graph Properties and Query Complexity[SYZ04]

3.1 Some introductory words and context of the work

3.1.1 A general overview of the article

This article deals with (asymptotic) lower bounds on the quantum query complexity of some properties having special features; more precisely, the authors focused on boolean properties invariant under transitive group actions. Upper bounds are given first for different properties of these types, namely the ones corresponding to graph properties, directed graph properties, and functions invariants under cyclic shifts. Then a lower bound is given which holds for all properties invariant under some transitive group action. We will take time to explain this last bound, where results from 1 and 2 are used.
As this survey is only a conference article, and no detailed version was published, reasonnings are uncomplete, some arguments are missing, and sometimes even wrong or given in a wrong way. That is why we won't enter in details for the main part of the proofs, and sometimes only give general main ideas without any detail.
Before presenting this article, we will first give some introductory words about properties invariant under transitive group actions, and more precisely about graph properties. The motivation of working on such objects is that in the deterministic model, the evasiveness conjecture (also called Karp conjecture) is open since 1973. This conjecture states that for monotone graph properties f, $D(f)$ is maximal. This is why it is interesting to be able to see if in the quantum model, query complexity of such functions are still "high".

3.1.2 Invariance and graph properties

We keep here notations used in 1. As we just stated, we will now focus on some specific class of properties, invariants under some group actions:

Definition (Invariance)**.** Let $f : \{0,1\}^N \to \{0,1\}$ be a property, and Γ a subgroup of $\mathfrak{S}_N$. We say that f is invariant under the action of Γ iff :

$$\forall\ (x_1,...,x_N) \in \{0,1\}^N,\ \forall\ \sigma \in \Gamma,\ f(x_{\sigma(1)},...,x_{\sigma(N)}) = f(x_1,...,x_N)$$

This definition is very general; we will focus on a class of groups which have the property of transitivity:

Definition (Transitivity)**.** A subgroup Γ of $\mathfrak{S}_N$ is said to have a transitive action on $\{1,...,N\}$ if forall $i,j \in \{1,...N\}$, there exists $\sigma \in \Gamma$ such that: $\sigma(i) = j$. In other words, the action of Γ on $\{1,..,N\}$ has only one orbit.

Here are some examples of properties invariant under some transitive group actions:

Example 3.1

- One can see that symmetric boolean functions $f : \{0,1\}^N \to \{0,1\}$ that were defined in 1 are exactly properties invariant under $\mathfrak{S}_N$. The action is transitive.
- For $\Gamma :=< (1\ 2\ ...\ N) >$ the group composed of all cyclic shifts of $\{1,...,N\}$, if $f : \{0,1\}^N \to \{0,1\}$ is invariant under Γ, we call f a circular function. Such an action is transitive.
- The trivial properties are the two constant functions over $\{0,1\}^N$; they are of course symmetric, and have deterministic query complexity 0.

Two other classes of properties, called graph properties and directed graph properties lead to open interesting questions in the classical deterministic model: we consider a fixed set of vertices V of size n. Then we can see that if $N = \binom{n}{2}$ (resp. $N = n(n-1)$), a boolean vector $(x_1,...,x_N)$ can be associated bijectively to a graph G with edges in $\binom{V}{2}$ (resp. a directed graph G with edges in $V^2\backslash\{(v,v), v \in V\}$). Indeed, one have to see indices $i \in \{1,...,N\}$ as edges, and $x_i = 1$ iff i is an edge of $E(G)$. Then a graph property will be a property $f : \{0,1\}^N \to \{0,1\}$ which is "invariant under graph isomorphism", which means that two x's inducing two isomorphic graphs will have the same image under f. More formally, it simply corresponds to the following definition:

Definition (Graph property)**.** We assume that V is a fixed set of vertices such that $|V| = n$ for some nonnegative integer n. In the case $N = \binom{n}{2}$ (resp. $N = n(n-1)$), we associate implicitely $\{1, ..., N\}$ and $\binom{V}{2}$ (resp. $V^2 \backslash \{(v, v),\ v \in V\}$). Hence a relabelling $\sigma \in \mathfrak{S}_V$ of the vertices induces a bijection $\sigma \in \mathfrak{S}_N$ with: $\forall \{u, v\} \in \{1, ..., N\},\ \sigma \cdot \{u, v\} := \{\sigma u, \sigma v\}$. Let $\Gamma \simeq \mathfrak{S}_n$ be the subgroup of $\mathfrak{S}_N$ whose elements are all these induced bijections. We say that $f : \{0,1\}^N \to \{0,1\}$ is a graph property (resp. directed graph property) if f is invariant under the action of Γ.

One have to pay attention when talking about graph property that the underlying number of vertices n is always fixed. One can see that properties as "being connected", "having at most/least k edges" (for a fixed k) or "having a $k - clique$" correspond to graph properties. Deciding query complexity of such properties is equivalent to know how many bits of the upper triangle part of the adjacency matrix of an input graph G we will have to see see in the worst case before being able to decide whether G satisfies f or not.

An interesting case of properties is the one of monotone properties, which correspond to monotone functions $f : \{0,1\}^N \to \{0,1\}$ (where $\{0,1\}^N$ has to be considered with the product order). No result about them is given in the article, and we will just mention them there with a few words as they appear to be quite hard to decide. One have to see that a monotone graph property is a graph property f which is invariant under taking subgraph, which means that if x, x' corresponds to two graphs, with x a subgraph of x', then if $f(x') = 1$, we have $f(x) = 1$. Amongst our last examples, the property "having at most k edges" is monotone. It is known since 1976 that deciding monotone graph properties is "asymptoticaly hard"; more precisely, Rivest and Vuillemin showed in [LRV76] that for nontrivial monotone graph properties (directed or not), we have: $D(f) = \Omega(n^2) = \Omega(N)$. However the following more precise conjecture is still open today:

Conjecture 3.1 (Karp's Conjecture, 1973) Any non trivial monotone graph property $f : \{0,1\}^N \to \{0,1\}$ has deterministic query complexity $D(f) = \binom{n}{2} = N$.

3.1.3 The main results of the article

Important convention: From now until the end of the section, we will consider that all properties that we are considering are nontrivial, otherwise everything we will assert is be trivially wrong.
The authors give "almost tight" upper bounds on $Q_2(f)$ for different properties invariant under some transitive actions, and a general lower bound, using some results from 1 and 2:

Theorem 3.1 There exists a graph property $f : \{0,1\}^N \to \{0,1\}$ ($N = \binom{n}{2}$), such that forall $\epsilon > 0$:

$$Q_2(f) = \mathcal{O}(N^{\frac{1}{4}+\epsilon})$$

Theorem 3.2 There exists a directed graph property $f : \{0,1\}^N \to \{0,1\}$ ($N = \binom{n}{2}$), such that forall $\epsilon > 0$:

$$Q_2(f) = \mathcal{O}(N^{\frac{1}{4}+\epsilon})$$

The proof of Theorem 2 is not given in the paper, as a similar reasonning than the one of Theorem 1 is used.

Theorem 3.3 There exists a circular function $f : \{0,1\}^N \to \{0,1\}$, such that forall $\epsilon > 0$:

$$Q_2(f) = \mathcal{O}(N^{\frac{1}{4}+\epsilon})$$

The following theorem states that Theorems 1, 2 and 3 are "almost optimal":

Theorem 3.4 If there exists some subgroup Γ of $\mathfrak{S}_N$ acting transitively on $\{1, ..., N\}$such that the property $f : \{0,1\}^N \to \{0,1\}$ is invariant under the action of Γ, then :

$$Q_2(f) = \Omega(N^{\frac{1}{4}})$$

We will develop more precisely the proof of Theorem 4.

3.2 General idea about the proofs of the upper bounds

To prove Theorems 1 and 2, the authors considered properties whose query complexity is known to be "low" in the deterministic case, and showed that this is still the case in the quantum model. They designed efficient quantum algorithms to decide these properties, that use as subroutine variants of Grover's algorithm [Gro96]. These algorithmic tricks allow them to do fast computings on graph, namely find some vertex with a certain property amongst some subset of the vertex set. Some technical lemmas allow them to get a bounded complexity at each step. For Theorem 1, the graph property studied is the property of beeing a Scorpion: a graph G is a Scorpion iff there exist three distinct special vertices $B, T, S \in V$ called the Body, the Tail and the Sting, such that:

- S has degree 1 and its only neighbour is T,
- T has degree 2 and its two neighbors are S and B,
- B has degree $n-2$ (with $n = |V(G)|$).

It is known (see for example [dL12]) that for $f_{scorpion}$ corresponding to this property and $n \geq 5$, we have $D(f_{scorpion}) \leq 6n - 13$ (recall that $N = \binom{n}{2}$). This complexity is one of the lowest that a graph property can have in deterministic model.
For Theorem 2, the graph property studied is the property f_{sink} of being a sink: we say that a directed graph G is a sink when there exists a vertex $v \in V(G)$ with out-degree 0 and in-degree $n-1$. Again in deterministic case, the complexity of this property is linear (see for example [BMZ17]) : $D(f_{sink}) \leq 3n-4$.

3.2.1 Variants of Grover algorithm

We recall that Grover's algorithm return a marked element amongst a list of n elements, and an error message if such element does not exist in time $\mathcal{O}(\sqrt{n})$ (with a constant error probability, say $\frac{1}{3}$). Here are some more precises versions of this algorithm and of its complexity which are used in the article almost everywhere, and were presented in [HMdW03]:

Lemma 3.1 Assume we have a subroutine that decides if a given element is marked or not in time q. Then there exists a quantum algorithm that we will call $Algo1$ which takes as input n elements and returns a marked element or an error message if no marked element exists, with constant error probability (say $\frac{1}{3}$). Moreover this algorithm runs in expected time and query complexity $\mathcal{O}(\sqrt{\frac{n}{m}}q)$, where m is the number of marked elements.

From this Lemma, with some algorithmic tricks the authors show how to get the following result:

Lemma 3.2 Assume we have a subroutine that decides if a given element is marked or not in time q. Let k be a fixed integer and $\epsilon > 0$. Then there exists a quantum algorithm that we will call $Algo2$ which takes as input n elements, with m of them marked, and returns k marked element if there are at least k marked elements or the m marked elements otherwise, with error probability less than ϵ. Moreover this algorithm runs in expected time and query complexity $\mathcal{O}(q\sqrt{min(k,m)n} + q\sqrt{n}log(\frac{k}{\epsilon}))$.

We give here some examples of applications of these algorithms that are used in the article: let G be a graph with n vertices and $\epsilon > 0$.

- Given a vertex v, we can check in $\mathcal{O}(\sqrt{n}log(\frac{2}{\epsilon}))$ queries if it has degree 1 with error probability less than ϵ. For this apply $Algo2$ with $k = 2$, list of elements $V\backslash\{v\}$ and marked elements neighbours of v. Our subroutine consists here only in checking that a given vertex u is a neighbour of v, so $q = 1$. If the algorithm returns some solution, then v has degree at least 2 with error probability $\frac{\epsilon}{2}$. Apply $Algo1$ (or $Algo2$ with $k = 1$) then to check whether $deg(v) = 0$.
- With the same idea, by inverting marked and unmarked elements, we can check that a given v has degree $n-2$.
- Contrarily to what is written in the article, with the same method we need $\mathcal{O}(\sqrt{n}log(\frac{3}{\epsilon}))$ queries to check whether a given vertex has degree 2.
- If U is a subset of V with u vertices such that all vertices v in $V\backslash U$ with no neighbour in U have degree at most d in G, then we can find with $\mathcal{O}(\sqrt{n}\sqrt{\frac{u}{d}})$ queries such a vertex v. For this we

apply $Algo1$ with list of elements V and marked elements vertices v with no neighbour in U. The subroutine of $Algo1$ used here consists in checking that an input vertex v has no neighbour in U, which can be done by running $Algo2$ with list of elements U, marked elements neighbours of v (so $m \leq d$), and $q = 1$.

3.2.2 One of the algorithms (very informal version)

Two algorithms are given deciding $f_{scorpion}$, the second beeing a modified version of the first which has the desired complexity. Their goal is to find as quick as possible candidates for S, T and B, and to discard a maximum of candidates if it fails to do this. If we have three candidates vertices, we can then check with "low" complexity (namely $\mathcal{O}(\sqrt{n}log(\frac{1}{\epsilon}))$) whether the input graph G is a scorpion graph with these vertices as S, T and B (see Lemma 7 of the article). We just give here the idea of the first algorithm, and won't talk of its modified version, as its complexity analysis is quite unclear.
Recall that the algorithm takes as input a graph $G = (V, E)$ and outputs whether G is a scorpion or not with high probability. Here are its five steps:

1 We chose uniformly at random a subset U of V with only "few vertices": $|U| = 2\sqrt{nlog(n)}$. Then if G is a scorpion, T and S are not in U with "high" probability.

2 We find a vertex v which has no neighbour in U with tricks used in the last subsection; if G is a scorpion, we know that such a vertex exists with high probability, as S and T have this property. Some probability lemma (see Lemma 8 of the article) ensure us that if v exists, then it has "low" degree with high probability, namely: $deg(v) \leq \sqrt{nlog(n)}$.

3 If we are lucky, we found in step 3 a vertex v which is T or S. So we check whether G is a scorpion with v in the role of T or S. If the answer is 'Yes' we are done and return 'Scorpion graph'.

4 Otherwise, we know that if G is a scorpion, then B has to be a neighbour of v. Our objective now is then to find B amongst the neighbors of v. We saw that we can assume that v has at most $\sqrt{nlog(n)}$ neighbors, so it won't be too costly to look for B amongst them. We use $Algo2$ with $k = \sqrt{nlog(n)}$ to get the set W of these elements.

5 We look for a (unique) candidate for B in W with $Algo1$, i.e a vertex with degree $n-2$. If we don't find it, then we return 'Not a Scorpion graph'; otherwise we test whether G is a scorpion with this candidate as B, and return the answer of this check.

As we mentioned earlier, this algorithm is still too costly to achieve the bound of Theorem 1. This is why the authors transformed it with the following idea: instead of looking for a unique v in step 1, they now look for many $v_i's$ associated to many sets $U_i's$, and hope to find with high probability a good candidate for body in the intersection of their neighborhoods. However it is not so clear how they get all their complexity bounds and use $Algo1$ and $Algo2$...

We won't talk about the proof of Theorem 3 here, as the same kind of tricks and reasonnings are used.

3.3 Proof of the lower bound of Theorem 4

In the last section of the article, two proofs are given of two lower bounds: one for the classical model, and then the one of Theorem 4. We give here these two proofs; however we warn the reader that though the Lemma 14 which is given in the paper to give the first proof is true, it is too specific and unappropriate to prove the result which is discussed. The authors were more likely thinking in their proof about the more general version of the lemma that we can find in [LRV76], as it is the one we need to use to obtain the good equality. We will then use this formulation here instead of the one given in the article.

The first result discussed is a lower bound in the classical case; we take again here the notations of section 1 for certificate complexity:

Theorem 3.5 If a property $f : \{0,1\}^N \to \{0,1\}$ is invariant under some transitive group action Γ, then $C_0(f)C_1(f) \geq N$ and thus $D(f) \geq C(f) \geq \sqrt{N}$.

Proof. We consider here $A, B \subseteq \{1, ..., N\}$ 1- and 0-certificates of size respectively $C_1(f)$ and $C_0(f)$, with associated a and $b \in \{0,1\}^N$ such that $f(a) = 1$ and $f(b) = 0$. If we show that $|A||B| \geq N$, then

we are done. One can observe that the group action of Γ on $\{1,...,N\}$ induces a group action of Γ on $2^{\{1,...,N\}}$ defined by:

$$\forall \sigma \in \Gamma,\ \forall C \subseteq \{1,...,N\},\ \sigma(C) := \{\sigma(i),\ i \in C\}$$

This action preserves cardinalities of subsets. We denote with $\Gamma(B) := \{\sigma(B), \sigma \in \Gamma\}$ the orbit of B by this action. One can observe by transitivity of Γ that, forall $i \in \{1,...,N\}$, the quantity $|\{\sigma(B), \sigma \in \Gamma, i \in \sigma(B)\}|$ is constant, i.e does not depend on i.
A first observation that can be done is that any $B' = \sigma(B) \in \Gamma(B)$ is a 0-certificate as well associated to $b' := \sigma(b)$; indeed, let x be such that: $x|_{B'} = b'|_{B'}$. Then if we note $\sigma(x) := (x_{\sigma(1)},...,x_{\sigma(N)})$ we have: $f(x) = f(\sigma^{-1}(x))$, and $f(b') = f(\sigma^{-1}(b'))$ as f is invariant under Γ action. But $\sigma^{-1}(x)|_B = \sigma^{-1}(b')|_B = b|_B$, so : $f(x) = f(b') = 0$, which prove that B' is a 0-certificate.
One can observe moreover that any $B' \in \Gamma(B)$ must intersect A: if there is $B' \in \Gamma(B)$ such that $A \cap B' = \emptyset$, then by flipping bits of a with indices in B' to those of $b' = \sigma(b)$, we get a a' valued 1 by f, as A is a 1-certificate, which contradicts our previous observation that B' is a 0-certificate. Hence: $A \cap B' \neq \emptyset$.
This result allows us to derive the following inequality:

$$\sum_{B' \in \Gamma(B)} |A \cap B'| \geq |\Gamma(B)| \tag{1}$$

Now we will use the following lemma stated in [LRV76]:

Lemma 3.3 Let E be a finite set, and $\mathcal{O}$ be a family of k-subsets of E. If Γ is a transitive group action on E such that $\mathcal{O}$ is an orbit of its induced action on 2^E, then for any $i \in E$:

$$k|\mathcal{O}| = |E|\ |\{B' \in \mathcal{O},\ i \in B'\}|$$

If we apply this lemma here with $E := \{1,...,N\}$, $\mathcal{O} := \Gamma(B)$ and any $i \in E$ then we get:

$$|B||\Gamma(B)| = N|\{\sigma(B),\ \sigma \in \Gamma,\ i \in \sigma(B)\}| \tag{2}$$

Hence we have :

$$\sum_{B' \in \Gamma(B)} |A \cap B'| = \sum_{\sigma \in \Gamma} \sum_{i \in A} 1_{i \in \sigma(B)} = \sum_{i \in A} \sum_{\sigma \in \Gamma} 1_{i \in \sigma(B)} = \sum_{i \in A} |\{\sigma(B),\ \sigma \in \Gamma,\ i \in \sigma(B)\}| = |A| \frac{|B||\Gamma(B)|}{N}$$

We then conclude with (1) that $|A||B| \geq N$, which is the desired result. ■

Now we have a bound in the classical model, we will conclude by giving the proof in the quantum model of Theorem 4; again, we will use the notations of section 1 for block sensitivity complexity, and of 2:

Proof (Theorem 4). Let f be a nontrivial function invariant under the transitive group action of some Γ. As query complexity of f is the same that the one of its complementary $1 - f$ in any model, we can consider wlog that $f((0,...,0)) = 0$. We consider $B \subseteq \{1,...,N\}$ some minimal by inclusion such that $f((0,...,0)^B) = 1$. Then we have: $|B| \leq bs(f)$, and from section 1, we know that $Q_2(f) = \Omega(\sqrt{bs(f)})$. Then if are able to show that $Q_2(f) = \Omega(\sqrt{\frac{N}{|B|}}) = \Omega(\sqrt{\frac{N}{bs(f)}})$, by combining the two bounds, we will get that $Q_2(f)^2 = \Omega(\sqrt{N})$, which gives us the desired result. We show this last bound from Ambaini's theorem presented in section 2:
we choose set of entries $X := \{(0,...,0)\}$ and $Y := \{\sigma((0,...,0)^B),\ \sigma \in \Gamma\}$. By invariance of f under Γ action, f is valued 1 on Y, and so $f(x) \neq f(y)$ for any $(x,y) \in X \times Y$. We choose the relation $R := X \times Y$, and the parameters of the theorem (there is a little mistake in the article: one have to exchange min and maxs):

1 $m := min_{x \in X}|\{y \in Y, (x,y) \in R\}| = |Y|$

2 $m' := min_{y \in Y}|\{x \in X, (x,y) \in R\}| = 1$

3 $l := max_{x \in X, i \in \{1,...,N\}}|\{y \in Y,\ (x,y) \in R,\ x_i \neq y_i\}| = |\{y \in Y,\ y_i = 1\}| = |\{\sigma((0,...,0)^B),\ \sigma \in \Gamma,\ \sigma((0,...,0)^B)_i = 1\}|$ for any $i \in \{1,...,N\}$.

4 $l' := max_{y \in Y, i \in \{1,...,N\}}|\{x \in X,\ (x,y) \in R,\ x_i \neq y_i\}| = 1$

Ambaini's theorem gives us $Q_2(f) = \Omega(\sqrt{\frac{mm'}{ll'}})$. Now we can see that if we apply Lemma 3 by identifying elements of $\{0,1\}^N$ with subsets of $\{1,...,N\}$, with $E := \{1,...,N\}$, $\mathcal{O} := Y$ the orbit of B by Γ, and k the hamming weight of $(0,...,0)^B$, so $k = |B|$, we have for any i:

$$|B||Y| = N|\{\sigma((0,...,0)^B),\ \sigma \in \Gamma,\ \sigma((0,...,0)^B)_i = 1\}|$$

Hence:

$$\frac{mm'}{ll'} = \frac{|Y|}{l}$$

and we can conclude the proof. ■

References

[Amb00] Andris Ambainis. Quantum lower bounds by quantum arguments, 2000.

[BBC+98] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials, 1998.

[BMZ17] Anders Björner, Jiří Matoušek, and Günter Ziegler. *Using Brouwer's Fixed Point Theorem*, pages 221–271. 05 2017.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1907):553–558, 1992.

[dL12] M. de Longueville. *A Course in Topological Combinatorics*. Universitext. Springer New York, 2012.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search, 1996.

[HMdW03] Peter Høyer, Michele Mosca, and Ronald de Wolf. Quantum search on bounded-error inputs. *Lecture Notes in Computer Science*, page 291–299, 2003.

[LRV76] Ronald L. Rivest and Jean Vuillemin. On recognizing graph properties from adjacency matrices. *Theoretical Computer Science*, 3:371–384, 12 1976.

[MP88] Marvin Minsky and Seymour Papert. Perceptrons. In *Neurocomputing: foundations of research*, pages 157–169. MIT Press, 1988.

[NS94] Noam Nisan and Mario Szegedy. On the degree of boolean functions as real polynomials. *Computational complexity*, 4(4):301–313, 1994.

[Pat92] Ramamohan Paturi. On the degree of polynomials that approximate symmetric boolean functions (preliminary version). In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 468–474. ACM, 1992.

[RC66] Theodore J Rivlin and Elliott Ward Cheney. A comparison of uniform approximations on an interval and a finite subset thereof. *SIAM Journal on numerical Analysis*, 3(2):311–320, 1966.

[Sim97] Daniel R Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.

[SYZ04] Xiaoming Sun, A.C. Yao, and Shengyu Zhang. Graph properties and circular functions: How low can quantum query complexity go? volume 19, pages 286– 293, 07 2004.