Proc. Int. Cong. of Math. – 2018
Rio de Janeiro, Vol. 4 (3283–3304)

UNDERSTANDING QUANTUM ALGORITHMS VIA QUERY COMPLEXITY

Andris Ambainis

Abstract

Query complexity is a model of computation in which we have to compute a function $f(x_1, \ldots, x_N)$ of variables x_i which can be accessed via queries. The complexity of an algorithm is measured by the number of queries that it makes. Query complexity is widely used for studying quantum algorithms, for two reasons. First, it includes many of the known quantum algorithms (including Grover's quantum search and a key subroutine of Shor's factoring algorithm). Second, one can prove lower bounds on the query complexity, bounding the possible quantum advantage. In the last few years, there have been major advances on several longstanding problems in the query complexity. In this talk, we survey these results and related work, including:

- the biggest quantum-vs-classical gap for partial functions (a problem solvable with 1 query quantumly but requiring $\Omega(\sqrt{N})$ queries classically);
- the biggest quantum-vs-determistic and quantum-vs-probabilistic gaps for total functions (for example, a problem solvable with M queries quantumly but requiring $\tilde{\Omega}(M^{2.5})$ queries probabilistically);
- the biggest probabilistic-vs-deterministic gap for total functions (a problem solvable with M queries probabilistically but requiring $\tilde{\Omega}(M^2)$ queries deterministically);
- the bounds on the gap that can be achieved for subclasses of functions (for example, symmetric functions);
- the connections between query algorithms and approximations by low-degree polynomials.

MSC2010: primary 68Q12; secondary 81P68, 68Q10, 68Q17, 41A10.

1 Introduction

Quantum computers open new possibilities for computing, by being able to solve problems that are considered intractable classically. The most famous example is factoring large numbers which is thought to require $\Omega(2^{n^c})$ time classically but is efficiently solvable by a quantum computer, due to Shor's quantum algorithm Shor [1997]. Another example is simulating quantum physical systems which is thought to require $\Omega(2^n)$ time classically but is also solvable in polynomial time quantumly Cirac and Zoller [2012] and Georgescu, Ashhab, and Nori [2014].

This naturally leads to a question: how large is the advantage of quantum computers? Can we put limits on it?

In the Turing machine model, we have $BQTIME(f(n)) \subseteq \cup_c TIME(2^{cf(n)})$ where $TIME$ and $BQTIME$ denote the classes of problems that are solvable by deterministic or quantum Turing machines within the respective time bound. However, it is very difficult to prove unconditional separations between complexity classes and we cannot even show that $BQTIME(f(n))$ is larger than $TIME(f(n))$.

For this reason, the power of quantum computers is often studied in the query model (also known as the decision tree model Buhrman and de Wolf [2002]). In this model, we have to compute a function $f(x_1, \ldots, x_N)$ of an input $(x_1, \ldots, x_N)$, with x_i accessible via queries to a black box that, given i, outputs x_i. The complexity is measured by the number of queries that an algorithm makes.

The query model is very interesting in the quantum case because it captures most of the known quantum algorithms. Some of the problems that can be described in it are:

Search. Given black box access to $x_1, \ldots, x_N \in \{0, 1\}$, determine whether there exists $i : x_i = 1$ (or find such i).

Search requires N queries classically but can be solved with $O(\sqrt{N})$ queries quantumly Grover [1996]. It can be viewed as a black box model for a generic exhaustive search problem where one has to check N possibilities (without any information which of those N possibilities are more likely) and implies quantum speedups for a variety of problems (for example, a quadratic quantum speedup over the best probabilistic algorithm for 3-SAT Ambainis [2004]).

Period-finding. Given black box access to $x_1, \ldots, x_N \in [M]$, determine the smallest r such that $x_i = x_{i+r}$ for all i (and $x_i \neq x_{i+q}$ for all i and $q < r$), under a promise that such r exists and is smaller than $c\sqrt{N}$ for some $c > 0$.

Period-finding is solvable with $O(1)$ queries quantumly and requires $\Omega(\frac{N^{1/4}}{\log N})$ queries classically Shor [1997] and Chakraborty, Fischer, Matsliah, and de Wolf [2010]. It is at the heart of Shor's factoring algorithm Shor [1997] which consists of a classical reduction from factoring to period-finding and a quantum algorithm for period-finding.

Element distinctness. Given black box access to $x_1, \ldots, x_N \in [M]$, determine if there are $i, j : i \neq j$ such that $x_i = x_j$.

Element distinctness requires N queries classically and $\Theta(N^{2/3})$ queries quantumly Ambainis [2007] and Aaronson and Shi [2004]. It is related to black box models of algorithms for breaking collision-resistant hash functions (an important cryptographic primitive). The quantum algorithm for element disticntness is also useful as a subroutine for other quantum algorithms, from checking matrix products Buhrman and Špalek [2006] to solving typical instances of subset sum (which is also important for cryptography) Bernstein, Jeffery, Lange, and Meurer [2013].

Many other quantum query algorithms are known, as can be seen from Quantum Algorithms Zoo (a website collecting information about all quantum algorithms Jordan [n.d.]). From a complexity-theoretic perspective, the query model is very interesting because it allows to prove lower bounds on quantum algorithms and it is often possible to characterize the quantum advantage within a big-O factor.

The current survey is focused on characterizing the maximum possible quantum advantage in the query model, for different types of computational tasks. Let $Q(f)$ and $R(f)$ denote the number of queries for the best quantum and randomized algorithm, respectively. For partial Boolean functions, we describe a gap of $Q(f) = 1$ vs $R(f) = \Omega(\sqrt{N}/\log N)$ Aaronson and Ambainis [2015]. For total functions, the biggest known gap is much smaller: $R(f) = \tilde{\Omega}(Q^{2.5}(f))$ Aaronson, Ben-David, and Kothari [2016]. Imposing symmetry constraints on f also decreases the maximum possible gap.

As a side result, this research has lead to new results on classical query algorithms. This includes solutions to two well known problems in the classical query complexity which had been open for about 30 years (such as determining the maximum gap between randomized and deterministic query complexities Saks and Wigderson [1986] and Arunachalam, Briët, and Palazuelos [2017]). We describe those developments, as well.

2 Computational Models

2.1 Deterministic, randomized and quantum query algorithms. We now formally define the models of query complexity that we use. We consider computing a function $f(x_1, \ldots, x_N)$ of variables x_i. By default, we assume that the variables x_i are $\{0, 1\}$-valued. (If we consider x_i with values in a larger set, this is explicitly indicated.) The function $f(x_1, \ldots, x_N)$ can be either a total function (defined on the entire $\{0, 1\}^N$) or a partial function (defined on a subset of $\{0, 1\}^N$).

Deterministic algorithms. Deterministic query algorithms are often called *decision trees*, because they can be described by a tree (as in Figure 1). At each node of this tree, we have the name of a variable that is asked if the algorithm gets to this node. Depending

on the outcome of the query, the algorithm proceeds to the $x_i = 0$ child or to the $x_i = 1$ child of the node. If the algorithm gets to a leaf of the tree, it outputs the value of the function listed at this leaf.

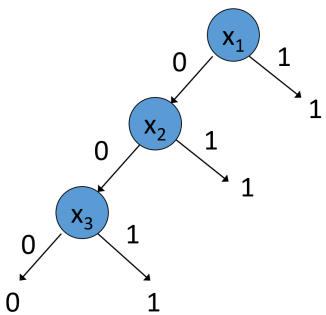

Figure 1: Example of a decision tree

The complexity of an algorithm $\mathcal{A}$ is the maximum number of queries that it can make. Deterministic query complexity $D(f)$ is the smallest complexity of a deterministic $\mathcal{A}$ which outputs $f(x_1, \ldots, x_N)$ if the queries are answered according to $(x_1, \ldots, x_N)$, whenever $f(x_1, \ldots, x_N)$ is defined.

Randomized algorithms. In a randomized query algorithm, the algorithm may choose the variable x_i for the next query randomly from some probability distribution.

Randomized algorithms are usually studied either in the zero-error setting or in the bounded error setting. In the zero-error setting, the algorithm is required to output $f(x_1, \ldots, x_N)$ with probability at least 1/2 and may output "don't know" otherwise but must not output a value that is different from $f(x_1, \ldots, x_N)$. In the bounded-error setting, algorithm is required to output $f(x_1, \ldots, x_N)$ with probability at least 2/3 and may output anything otherwise. In both cases, the requirement has to be satisfied for every $(x_1, \ldots, x_N)$ for which $f(x_1, \ldots, x_N)$ is defined.

The complexity of an algorithm $\mathcal{A}$ is measured by the largest number of queries that is made by $\mathcal{A}$, for the worst choice of $(x_1, \ldots, x_N)$ and the worst random choices of $\mathcal{A}$. $R_0(f)$ and $R_2(f)$ are the smallest complexities of a zero-error randomized and a bounded error randomized algorithm for f, respectively. (Alternatively, one can define randomized query complexity via the expected number of queries for the worst case $(x_1, \ldots, x_N)$ but this changes the complexities R_0 and R_2 by at most a constant factor.)

Quantum algorithms. Unlike in the probabilistic case, different branches of a quantum algorithm can recombine at a later stage. For this reason, a quantum query algorithm cannot be described by a tree.

Instead, a quantum query algorithm is defined by an initial state $|\psi_{start}\rangle$ and transformations $U_0, Q, U_1, \ldots, Q, U_T$. The initial state $|\psi_{start}\rangle$ and transformations U_i are independent of $x_1, \ldots, x_N$. Q are the queries - transformations of a fixed form that depend on x_i's. The algorithm consists of performing $U_0, Q, U_1, \ldots, Q, U_T$ on $|\psi_{start}\rangle$ and measuring the result (as shown in Figure 2). The algorithm computes f if, for every $(x_1, \ldots, x_N)$ for which $f(x_1, \ldots, x_N)$ is defined, this measurement produces $f(x_1, \ldots, x_N)$.

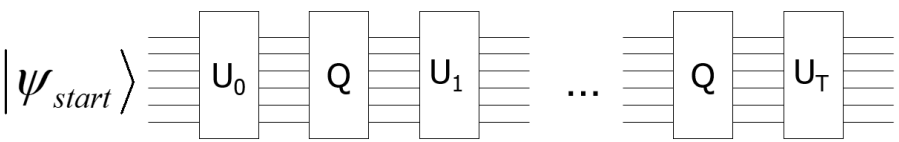

Figure 2: Structure of a quantum query algorithm

To define the model more precisely, we must define the notions of a quantum state, a transformation, and a measurement. (For more details on these notions, we refer the reader to the book Nielsen and Chuang [2000].) The state space of a quantum algorithm is a complex vector space of dimension d (where d can be choosen by the designer of the algorithm). Let $|1\rangle, \ldots, |d\rangle$ be an orthonormal basis for this vector space. A quantum state is a vector

$$|\psi\rangle = \alpha_1|1\rangle + \ldots + \alpha_d|d\rangle = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \ldots \\ \alpha_d \end{pmatrix}$$

of unit length (i.e. satisfying $\sum_i |\alpha_i|^2 = 1$). A unitary transformation is a linear transformation on $|\psi\rangle$ that preserves the length of $|\psi\rangle$. The principles of quantum mechanics allow to perform any unitary U on a quantum state.

A measurement is the way of obtaining information from a quantum state. Measuring a state $|\psi\rangle$ with respect to $|1\rangle, \ldots, |d\rangle$ yields the result i with probability $|\alpha_i|^2$.

To define a quantum query algorithm, we allow the starting state $|\psi_{start}\rangle$ to be an arbitrary quantum state. U_i's can be arbitrary unitary transformations that do not depend

on $x_1, \ldots, x_N$. Q is the query transformation, defined in a following way[1]. We rename the basis states from $|1\rangle, \ldots, |d\rangle$ to $|i, j\rangle$ with $i \in \{0, 1, \ldots, N\}$ and $j \in [d_i]$ for some d_i and define

$$Q|0, j\rangle = |0, j\rangle \text{ for all } j,$$

$$Q|i, j\rangle = \begin{cases} |i, j\rangle & \text{if } x_i = 0 \\ -|i, j\rangle & \text{if } x_i = 1 \end{cases}.$$

It can be argued that this is a natural quantum conterpart of a probabilistic query in which we choose i according to a probability distribution and get the corresponding x_i.

After the last transformation, the state of the algorithm is measured w.r.t. $|1\rangle, \ldots, |d\rangle$ and the result is transformed into the answer of the algorithm according to a predefined rule. (For example, if the answer should be $\{0, 1\}$-valued, we could take the first bit of the measurement result i as the answer.)

Two most frequently considered types of quantum query algorithms are *exact* and *bounded error* algorithms. A quantum query algorithm computes f exactly if its answer is always the same as $f(x_1, \ldots, x_N)$, whenever f is defined. A quantum query algorithm $\mathcal{A}$ computes f with bounded error, if for every $(x_1, \ldots, x_N)$, for which $f(x_1, \ldots, x_N)$ is defined, the probability that $\mathcal{A}$ outputs $f(x_1, \ldots, x_N)$ as the answer is at least 2/3. $Q_E(f)$ and $Q_2(f)$ are the smallest numbers of queries in quantum algorithms that compute f exactly and with bounded error, respectively.

2.2 Quantities that are related to query complexity. In this section, we define several quantities that provide upper and lower bounds on different query complexities. Using them, we can prove bounds on the maximum gaps between query complexity measures (for example, that $D(f) = O(R_2^3(f))$ Nisan [1991] and $D(f) = O(Q_2^6(f))$ Beals, Buhrman, Cleve, Mosca, and de Wolf [2001] for any total Boolean function f).

Block sensitivity. For an input $x \in \{0, 1\}^N$ and a subset of variables $S \subseteq [N]$, $x^{(S)}$ is the input obtained from x by changing all $x_i, i \in S$ to opposite values. The block sensitivity $bs(f)$ is the maximum k for which there is an input $x \in \{0, 1\}^N$ and pairwise disjoint subsets $S_1, \ldots, S_k \subseteq [N]$ with $f(x) \neq f(x^{(S_i)})$ for all $i \in [k]$.

Block sensitivity is a lower bound on all the query complexity measures: $D(f) \geq bs(f)$, $R(f) = \Omega(bs(f))$ Nisan [1991] and $Q(f) = \Omega(\sqrt{bs(f)})$ Beals, Buhrman, Cleve, Mosca, and de Wolf [2001]. It also provides an upper bound on $D(f)$ for total Boolean functions f: $D(f) = O(bs^3(f))$ Nisan [1991]. Combining these relations yields $D(f) = O(R_2^3(f))$ and $D(f) = O(Q_2^6(f))$ - the best upper bounds on the gap between $D(f)$ and $R_2(f)$ or $Q_2(f)$.

[1] Since most of this survey considers functions $f(x_1, \ldots, x_N)$ of variables $x_i \in \{0, 1\}$, we only give the definition of a query for this case.

Certificate complexity. For an input $x \in \{0,1\}^N$, a certificate is a set $S \subseteq [N]$ with the property that the variables $x_i, i \in S$ determine the value of $f(x)$. (More precisely, $S \subseteq [N]$ is a certificate on an input x if, for any $y \in \{0,1\}^N$ such that $x_i = y_i$ for all $i \in [S]$, we have $f(x) = f(y)$.) $C_x(f)$ is the minimum size $|S|$ of a certificate S on the input x. The certificate complexity $C(f)$ is the maximum of $C_x(f)$ over all $x \in \{0,1\}^N$.

Certificate complexity provides a better upper bound on $D(f)$ for total f: $D(f) \leq C^2(f)$ Nisan [ibid.]. If one could show that $Q_2(f) = \Omega(\sqrt{C(f)})$, this would imply $D(f) = O(Q_2^4(f))$, improving the best known relation between $D(f)$ and $Q_2(f)$.

Randomized certificate complexity Aaronson [2008]. For an input x, $RC_x(f)$ is the minimum number of queries in a bounded-error randomized query algorithm that accepts x and rejects all $y : f(x) \neq f(y)$. The randomized certificate complexity $RC(f)$ is the maximum of $RC_x(f)$ over all $x \in \{0,1\}^N$.

Unlike for the standard certificate complexity, it is known that $Q_2(f) = \Omega(\sqrt{RC(f)})$ Aaronson [ibid.]. Proving $D(f) = O(RC^2(f))$ for total f (which is not known) would also imply $D(f) = O(Q_2^4(f))$.

Polynomial degree. The exact degree, $\deg(f)$, is the degree of the multilinear polynomial $p(x_1, \ldots, x_N)$ which satisfies $f(x_1, \ldots, x_N) = p(x_1, \ldots, x_N)$ for all $(x_1, \ldots, x_N)$. The approximate degree, $\widetilde{deg}(f)$, is the smallest degree of a multilinear polynomial $p(x_1, \ldots, x_N)$ $\{0,1\}^N$ which satisfies $|f(x_1, \ldots, x_N) - p(x_1, \ldots, x_N)| \leq \frac{1}{3}$ for all $(x_1, \ldots, x_N) \in \{0,1\}^N$.

Both of these measures also provide lower bounds on quantum query complexity: $Q_E(f) \geq \frac{\deg(f)}{2}$ and $Q_2(f) = \Omega(\sqrt{\widetilde{deg}(f)})$ Beals, Buhrman, Cleve, Mosca, and de Wolf [2001].

3 Maximum quantum-classical gap for partial functions

In this section, we consider the question: what is the maximum possible gap between $Q_2(f)$ and the most general of classical complexities $R_2(f)$, for a partial function $f(x_1, \ldots, x_N)$ if we do not place any constraints on f?

As we already mentioned, period finding has $Q_2(f) = O(1)$ and $R_2(f) = \tilde{\Omega}(\sqrt[4]{N})$. In the form defined in Section 1, period-finding is not a Boolean function (it has variables $x_i \in [M]$ instead of Boolean variables). While it is possible to define a Boolean version of period-finding with almost the same gap, there is Boolean function with an even bigger gap:

Theorem 1. *Aaronson and Ambainis [2015] There exists f with $Q_2(f) = 1$ and $R_2(f) = \Omega(\sqrt{N}/\log N)$.*

The function f is defined as follows Aaronson and Ambainis [ibid.]. We have $N = 2^{n+1}$ variables. For technical convenience, we denote variables $x_0, \ldots, x_{2^n-1}$,

$y_0, \ldots, y_{2^n-1}$ and assume that the possible values for variables are ± 1 (instead of 0 and 1). Let F be the $2^n \times 2^n$ matrix (with rows and columns indexed by $a, b \in [0, 2^n - 1]$) defined by $F_{a,b} = \frac{1}{2^{n/2}}(-1)^{a \cdot b}$ where $a \cdot b = \sum_i a_i b_i$ is the inner product between a and b interpreted as n-bit strings $a_{n-1} \ldots a_0$ and $b_{n-1} \ldots b_0$. (In terms of quantum computing, $F = H^{\otimes n}$ where H is the standard 2×2 Hadamard matrix.) We define

$$f(x_0, \ldots, y_{2^n-1}) = \begin{cases} 1 & \text{if } \sum_{a,b} F_{a,b} x_a y_b \geq \frac{3}{5} 2^n \\ 0 & \text{if } \sum_{a,b} F_{a,b} x_a y_b \leq \frac{1}{100} 2^n \end{cases}.$$

The thresholds $\frac{3}{5}$ and $\frac{1}{100}$ are chosen so that:

- if we choose $x_i \in \{-1, 1\}$ for $i \in \{0, \ldots, 2^n - 1\}$ uniformly at random and then choose $y_i = sgn((Fx)_i)$, we get $f = 1$ with a high probability;
- if we choose both x_i and y_j uniformly at random form $\{-1, 1\}$, we get $f = 0$ with a high probability.

Thus, by solving f, we are effectively distinguishing between $\vec{y} = (y_i)_{i \in [0, 2^n-1]}$ being the vector of signs of $F\vec{x}$ where $\vec{x} = (x_i)_{i \in [0, 2^n-1]}$ and $\vec{y}$ being independently random.

$Q_2(f) = 1$ is shown by a quantum algorithm that generates a quantum state

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \left(\frac{x_i}{\sqrt{2^n}} |0, i\rangle + \frac{y_i}{\sqrt{2^n}} |1, i\rangle \right).$$

This quantum state can be generated by just 1 query. We then apply the transformation F to basis states $|0, i\rangle$, transforming the state to

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \left(\frac{(Fx)_i}{\sqrt{2^n}} |0, i\rangle + \frac{y_i}{\sqrt{2^n}} |1, i\rangle \right).$$

We then use the SWAP test Buhrman, Cleve, Watrous, and De Wolf [2002], a well known test for testing similarity of coefficient vectors of two parts of a quantum state.

The proof of the lower bound, $R_2(f) = \Omega(\sqrt{N}/\log N)$, is quite intricate. We define a corresponding problem (which we call REAL FORRELATION) with real valued variables $x_0, \ldots, x_{2^n-1}, y_0, \ldots, y_{2^n-1}$ in which we have to distinguish between two cases:

(a) all x_i and y_i are i.i.d. random with Gaussian distribution $\mathfrak{N}(0, 1)$;

(b) x_i's are i.i.d random with Gaussian distribution $\mathfrak{N}(0, 1)$ and y_i are obtained by applying Fourier transform to a vector consisting of x_i's: $y_i = ((Fx)_i)$.

In Aaronson and Ambainis [2015], we show that any algorithm for FORRELATION implies an algorithm for REAL FORRELATION with a similar complexity. Thus, it suffices to show a classical lower bound on REAL FORRELATION.

REAL FORRELATION is, in turn, a special case of a more general problem, GAUSSIAN DISTINGUISHING, in which we have to determine whether a set of real-valued variables $x_1, \ldots, x_M$ has a hidden structure. Let $\vec{v_1}, \ldots, \vec{v_M}$ be a set of vectors in $\mathbb{R}^d$ for some d. We have to distinguish between two cases:

(a) all x_i are i.i.d. random with Gaussian distribution $\mathfrak{N}(0,1)$;

(b) $x_1, \ldots, x_M$ are generated by choosing a random $\vec{u} \in \mathbb{R}^d$ (whose entries are i.i.d. $\mathfrak{N}(0,1)$ random variables) and taking $x_i = (\vec{u}, \vec{v_i})$.

The lower bound on REAL FORRELATION is a special case of

Theorem 2. *Let $\vec{v_i}$ be such that $|(\vec{v_i}, \vec{v_j})| \leq \epsilon$ for all $i \neq j$. Then, GAUSSIAN DISTINGUISHING requires $\Omega(\frac{1/\epsilon}{\log(M/\epsilon)})$ queries.*

In the case of REAL FORRELATION, $M = 2^{n+1}$, $d = 2^n$, $\vec{v_1}, \ldots, \vec{v_{2^n}}$ are the computational basis states $|0\rangle, \ldots, |2^n - 1\rangle$ and $\vec{v_{2^n+1}}, \ldots, \vec{v_{2^{n+1}}}$ are $F|0\rangle, \ldots, F|2^n - 1\rangle$. Then, $\epsilon = \frac{1}{\sqrt{2^n}} = \frac{1}{\sqrt{N/2}}$, implying a lower bound of $\Omega(\sqrt{N}/\log N)$ on REAL FORRELATION. This bound is nearly tight, as shown by

Theorem 3. *Let $\mathfrak{A}$ be a 1-query quantum algorithm. There is a probabilistic algorithm $\mathfrak{A}'$ that makes $O(\sqrt{N})$ queries and, on every input $(x_1, \ldots, x_N)$, outputs an estimate $\tilde{p}$ such that $|p - \tilde{p}| \leq \epsilon$ (where p is the accepting probability of $\mathfrak{A}$ on $(x_1, \ldots, x_N)$) with a high probability.*

The simulation makes use of the connection between quantum algorithms and polynomials:

Lemma 1. *Beals, Buhrman, Cleve, Mosca, and de Wolf [2001] Let $\mathfrak{A}$ be a quantum algorithm that makes k queries to an input $(x_1, \ldots, x_N)$, $x_i \in \{0,1\}$. The accepting probability of $\mathfrak{A}$ can be expressed as a polynomial $p(x_1, \ldots, x_N)$ in variables $x_1, \ldots, x_N$ of degree at most $2k$.*

Since the accepting probability of an algorithm must be between 0 and 1, we have $0 \leq p(x_1, \ldots, x_N) \leq 1$ whenever $x_1, \ldots, x_N \in \{0,1\}$. Theorem 3 then follows from a more general result about estimating bounded polynomials:

Lemma 2. *Aaronson and Ambainis [2015] For every polynomial $p(x_1, \ldots, x_N)$ with $\deg p \leq 2$ and $0 \leq p(x_1, \ldots, x_N) \leq 1$ for any $x_1, \ldots, x_N \in \{0,1\}$, there is a probabilistic algorithm $\mathfrak{A}'$ that makes $O(\sqrt{N})$ queries and outputs an estimate $\tilde{p}$ such that $|p(x_1, \ldots, x_N) - \tilde{p}| \leq \epsilon$ with a high probability, for every input $(x_1, \ldots, x_N) \in \{0,1\}^N$.*

More generally, if we have a bounded polynomial $p(x_1, \ldots, x_N)$ with $\deg p \leq k$, its value can be estimated with $O(N^{1-1/k})$ queries. Together with Lemma 1, this implies a probabilistic simulation of t query quantum algorithms with $O(N^{1-1/2t})$ queries. Unlike for $t = 1$, we do not know whether this is optimal.

Open Problem 1. *Let $t \geq 2$. Is there a partial function $g(x_1, \ldots, x_N)$ with $Q_2(g) = t$ and $R_2(g) = \tilde{\Omega}(N^{1-1/2t})$?*

The problem remains open, if instead of $\tilde{\Omega}(N^{1-1/2t})$ (which matches our upper bound), we ask for a weaker lower bound of $\Omega(N^c)$, $c > 1/2$ and, even, if instead of a constant t, we allow $t = O(\log^c N)$.

Open Problem 2. *Is there a partial function $g(x_1, \ldots, x_N)$ with $Q_2(g) = O(\log^c N)$ for some c and $R_2(g) = \Omega(N^d)$ for $d > 1/2$?*

The well known examples of problems with a large quantum-classical gap (such as Simon's problem Simon [1997] or period-finding) typically have $R_2(g) = O(\sqrt{N})$. In Aaronson and Ambainis [2015], we give a candidate problem, k-FOLD FORRELATION for which we conjecture that bounds of Open Problem 1 hold. This is, however, the only candidate problem that we know.

4 Total functions: pointer function method

For total functions f, the possible gaps between $Q(f)$, $R(f)$ and $D(f)$ are much smaller: all of these complexity measures are polynomially related.

It is well known that $D(f) = O(Q_2^6(f))$ Beals, Buhrman, Cleve, Mosca, and de Wolf [2001] and $D(f) = O(R_2^3(f))$ Nisan [1991]. For exact/zero error algorithms we know that $D(f) = O(Q_E^3(f))$ Midrijanis [2004] and $D(f) = O(R_0^2(f))$ Saks and Wigderson [1986]. The question is: how tight are these bounds?

For a very long time, the best separations were:

- Quantum vs. probabilistic/deterministic: $OR(x_1, \ldots, x_N)$ has $Q_2(OR) = O(\sqrt{N})$ due to Grover's quantum search algorithm and $R_2(f) = \Omega(N)$.
- Probabilistic vs. deterministic: the binary AND–OR tree function of depth d has $D(f) = 2^d$ and $R_0(f) = O((\frac{1+\sqrt{33}}{4})^d)$, thus implying that $R_0(f) = O(D^{0.753\ldots}(f))$ Saks and Wigderson [ibid.].

Both of these separations were conjectured to be optimal by a substantial part of the respective research community.

For exact quantum query complexity, the best separation was $Q_2(XOR) = N/2$ vs. $D(XOR) = R_2(XOR) = N$ for the N-bit XOR function Beals, Buhrman, Cleve, Mosca,

and de Wolf [2001] until 2013 when an example with $Q_E(f) = O(R_2^{0.86\cdots}(f))$ was discovered Ambainis [2016].

In 2015, major improvements to all of these bounds were achieved, via two new methods. The first of them, the *pointer function* method was first invented by Göös, Göös, Pitassi, and Watson [2015] for solving the communication vs. partition number problem in classical communication complexity. It was then quickly adapted to separating query complexity measures by Ambainis, Balodis, Belovs, Lee, Santha, and Smotrovs [2017]:

Theorem 4. *Ambainis, Balodis, Belovs, Lee, Santha, and Smotrovs [ibid.]*

1. *There exists a total Boolean function f with $Q_2(f) = \tilde{O}(D^{1/4}(f))$.*
2. *There exists a total Boolean function f with $R_0(f) = \tilde{O}(D^{1/2}(f))$.*
3. *There exists a total Boolean function f with $R_2(f) = \tilde{O}(R_0^{1/2}(f))$.*

The first two results provide major improvements over the previously known results mentioned at the beginning of this section. The third result is the first ever superlinear gap between $R_0(f)$ and $R_2(f)$ for a total f.

We now illustrate the method by describing the simplest function by Göös, Pitassi, and Watson [2015] and sketch a proof that it achieves $R_2(f) = \tilde{O}(D^{1/2}(f))$, a slightly weaker result than the second item above. Consider $f(x_{ij}, y_{ij}, z_{ij})$, with variables $x_{ij} \in \{0,1\}$, $y_{ij} \in [0,N]$, $z_{ij} \in [0,M]$ indexed by $i \in [N]$, $j \in [M]$. The variables x_{ij} are interpreted as elements of an $N \times M$ table and pairs of variables (y_{ij}, z_{ij}) are interpreted as pointers to entries in this table[2] .

We define that $f = 1$ if the following conditions are satisfied:

1. the $N \times M$ table has a unique column i in which all entries x_{ij} are 1;
2. in this column, there is exactly one j for which $(y_{ij}, z_{ij}) \neq (0,0)$;
3. if we start at this (i,j) and repeatedly follow the pointers (that is, consider the sequence (i_k, j_k) defined by $(i_0, j_0) = (i,j)$ and $(i_k, j_k) = (y_{i_{k-1}j_{k-1}}, z_{i_{k-1}j_{k-1}})$ for $k > 0$), then:
   (a) for each $i' \neq i$, there is a unique $k \in [N-1]$ with $i_k = i'$,
   (b) $(i_N, j_N) = (0,0)$,
   (c) $x_{i_k j_k} = 0$ for all $k \in [N-1]$.

This function f has the following properties:

[2] As described, this is a function of variables with a larger set of values but it can be converted into a function with $\{0,1\}$-valued variables, with complexities changing by at most a logarithmic factor.

1. $D(f) = NM$: for any deterministic algorithm, an adversary may choose the values for variables so that at least one of x_{ij}, y_{ij}, z_{ij} needs to be queried for each ij.

2. If $f(x_{ij}, y_{ij}, z_{ij}) = 1$, this can be certified by showing variables x_{ij}, y_{ij}, z_{ij} for $N + M - 1$ different (i, j): the all-1 column and the cells (i_k, j_k) in the sequence of pointers. Moreover, there is one and only one way to certify this.

To show a gap between $D(f)$ and $R_2(f)$, it suffices to show that a randomized algorithm can find this certificate faster than a deterministic algorithm. For that, we set $N = M$ and consider the following randomized algorithm (due to Mukhopadhyay and Sanyal [2015]):

1. $\Theta(N \log N)$ times repeat:

   (a) Choose a random entry (i, j) of the table in a column that has not been eliminated yet.

   (b) While $x_{ij} = 0$, $y_{ij} \neq 0$, $z_{ij} \neq 0$ and i is not a column that has been already eliminated:

      - eliminate column i;
      - set $i = y_{ij}$ and $j = z_{ij}$.

   (c) If $x_{ij} = 0$ but $y_{ij} = 0$ or $z_{ij} = 0$, eliminate column i.

2. If all columns are eliminated or more than 100 columns remain, output 0.

3. Otherwise, test each of remaining columns by checking whether it satisfies the conditions for a certificate.

If $f = 1$, each time when we choose a random entry in a column that is not the all-1 column, there is an $\frac{1}{N}$ probability of choosing the entry that is a part of the pointer chain. This means that, during $\Theta(N \log N)$ repetitions, this happens $\Theta(\log N)$ times. Each time, the columns that are after this entry in the pointer chain get eliminated. On average, half of remaining columns are after the entry that gets chosen. This means that, with a high probability, after $\Theta(\log N)$ times, only $O(1)$ columns are not eliminated. Then, one can test each of them with $O(N)$ queries.

This basic construction can be modified in several ways Ambainis, Balodis, Belovs, Lee, Santha, and Smotrovs [2017]. To separate two models of computation, we should make the certificate for $f = 1$ easy to find in one of them but difficult in the other model. (For example, hard to find by zero-error probabilistic algorithms but easy to find by bounded error probabilistic algorithms.) For different separations, the modifications include:

- Arranging the cells with pointers (in columns that are not the all-1 column) into a binary tree instead of a chain.
- Introducing back pointers at the end of the pointer chain or at the leaves of the tree, pointing back to the all-1 column.
- Having more than one all-1 column with pointers among the all-1 columns.

Besides the three major results in Theorem 4, this approach gives better-than-before separations between exact quantum query complexity and all classical complexity measures ($Q_E(f) = \tilde{O}(\sqrt{D(f)})$, $Q_E(f) = \tilde{O}(\sqrt{R_0(f)})$, and $Q_E(f) = \tilde{O}(R_2^{2/3}(f))$), between bounded-error quantum and zero-error probabilistic complexity ($Q_2(f) = \tilde{O}(\sqrt[3]{R_0(f)})$), and between polynomial degree and randomized query complexity ($\widetilde{deg}(f) = \tilde{O}(\sqrt[4]{R_2(f)})$) Ambainis, Balodis, Belovs, Lee, Santha, and Smotrovs [ibid.].

5 Total functions: cheat sheet method

5.1 Query complexity. After the developments described in the previous section, the biggest separation between quantum and randomized complexities still remained $Q(f) = O(\sqrt{R_2(f)})$. This was improved to $Q(f) = \tilde{O}(R_2(f)^{2/5})$ in a breakthrough paper by Aaronson, Ben-David, and Kothari [2016], using another new method, *cheat sheets*.

The key feature of *cheat sheet* method is that it takes separations for partial functions and transforms them into separations for total functions, by adding extra variables that allow to check that the input satisfies the promise for one of two cases when the partial function f is defined. The main result is

Theorem 5. *Aaronson, Ben-David, and Kothari [ibid.] Let $f(x_1, \ldots, x_N)$ be a partial function with $Q_2(f) = Q$, $R_2(f) = R$ and $C(f) = C$. Then, there exists a total function f_{CS} with $Q_2(f_{CS}) = \tilde{O}(Q + \sqrt{C})$ and $R_2(f_{CS}) = \Omega(R)$.*

Let $f(x_1, \ldots, x_{N^3})$ be the partial function $f = AND \circ OR \circ FORRELATION$ obtained by composing AND, OR and $FORRELATION$ on N variables each. From the complexities of AND, OR and $FORRELATION$ and composition properties of the complexity measures it follows that $Q_2(f) = O(N)$, $C(f) = O(N^2)$ and $R_2(f) = \tilde{\Omega}(N^{2.5})$, implying

Theorem 6. *Aaronson, Ben-David, and Kothari [ibid.] There exists a total Boolean function f_{CS} with $Q_2(f_{CS}) = \tilde{O}(R_2^{2/5}(f_{CS}))$.*

Moreover, if Open Problem 1 was resolved in affirmative, we could substitute the corresponding g instead of FORRELATION and Theorem 5 would imply $Q_2(g_{CS}) = \tilde{O}(R_2^{1/3+o(1)}(g_{CS}))$.

The cheat sheet method also gives new separations betwen $Q_2(f)$ and many of combinatorial complexity measures: $Q_2(f) = \tilde{\Omega}(C^2(f))$, $Q_2(f) = \tilde{\Omega}(\deg^2(f))$, and $Q_2(f) = \Omega(\widetilde{deg}^{4-o(1)}(f))$. Moreover, several of results proven via the pointer function method (for example, the $Q_E(f) = \tilde{O}(R_2^{2/3}(f))$ separation) can be reproven via cheat sheets Aaronson, Ben-David, and Kothari [2016].

To show Theorem 5, Aaronson, Ben-David, and Kothari [ibid.] define the *cheat-sheet* function f_{CS} in a following way. f_{CS} has $tN + 2^t tC\lceil \log N + 1\rceil$ variables (for an appropriately chosen $t = \Theta(\log N)$) which we denote $x_{11}, \ldots, x_{tN}$, $y_{11}, \ldots, y_{2^t M}$ (where $M = Ct\lceil \log N + 1\rceil$). We interpret the blocks of variables $x^{(i)} = (x_{i1}, \ldots, x_{iN})$ as inputs to the function f and and the blocks $y^{(i)} = (y_{i1}, \ldots, y_{iM})$ as descriptions for t certificates of function f, with the description containing both the set of variables $S \subseteq [N]$ and and the values that x_i, $i \in S$ must take. (We refer to those blocks as *cheat-sheets*, as they allow to verify the values of $f(x^{(1)}), \ldots, f(x^{(t)})$ with less queries than it takes to compute them.)

We interpret the t-bit string $s = s_1 \ldots s_t$, $s_i = f(x^{(i)})$ as an index for the block $y^{(s)}$. We define that $f_{CS} = 1$ if the block $y^{(s)}$ contains certificates for $f(x) = s_1, \ldots, f(x) = s_t$ and the values of corresponding input variables in inputs $x^{(1)}, \ldots, x^{(t)}$ match the ones specified by the corresponding certificate. Otherwise, $f_{CS} = 0$.

To compute f_{CS} by a quantum algorithm, we proceed as follows:

1. compute $f(x^{(1)}), \ldots, f(x^{(t)})$, repeating each computation $O(\log t)$ times, to make the error probability at most $1/(10t)$ for each $f(x^i)$ (then, the probability that all $f(x^{(i)})$ are all simultaneously correct is at least 9/10);

2. check whether the certificates in the block $y^{(s)}$ are satisfied by inputs $x^{(1)}, \ldots, x^{(t)}$, by using Grover's quantum search to search for a variable in one of $x^{(i)}$ which does not match the corresponding certificate.

The complexity of the 1st stage is $O(Qt \log t)$. The complexity of the 2nd stage is $O(\sqrt{Ct} \log N)$, since we have to search among tC variables $x_j^{(i)}$ (t certificates, each of which contains C variables), Grover's quantum search Grover [1996] allows to search among them by testing $O(\sqrt{tC})$ possibilities, and testing each possibility requires reading $O(\log N)$ variables in the block $y^{(s)}$. Thus, the overall complexity is $\tilde{O}(Q + \sqrt{C})$ quantum queries.

Classically, Rt queries are required to solve t instances of $f(x^{(i)})$. Moreover, if the number of queries is substantially smaller (of an order $o(Rt)$), then, with a high probability, most of $f(x^{(i)})$ are not solved yet and, at that point, a classical algorithm cannot make use of certificate descriptions in $y^{(j)}$ because there are too many possible $y^{j)}$. This suggests that $R_2(f_{CS}) = \Omega(Rt)$ and Aaronson, Ben-David, and Kothari [2016] show that this is indeed the case.

5.2 Communication complexity. The cheat sheet method has also found applications in a different domain, *communication complexity* Kushilevitz and Nisan [1997] and Lee and Shraibman [2007]. In the standard model of communication complexity, we have two parties, Alice and Bob, who want to compute a function $f(x, y)$, with Alice holding the input x and Bob holding the input y. The task is to compute $f(x, y)$ with the minimum amount of communication between Alice and Bob. Communication complexity has a number of applications, from designing efficient communication protocols for various tasks to proving lower bounds on other models of computation (for example, streaming algorithms).

If quantum communication is allowed, the communication complexity may decrease exponentially. Similarly to query complexity, let $Q_2(f)$, $Q_E(f)$ and $R_2(f)$ denote the bounded-error quantum, exact quantum and bounded error randomized communication complexity of f. A partial function with an exponential gap between $R_2(f)$ and $Q_2(f)$ was first constructed by Raz [1999] in 1999. In a later work, it was shown that quantum protocols can be exponentially more efficient even if the quantum protocol is restricted to one message from Klartag and Regev [2011] but it is compared against randomized protocols that can send an arbitrary number of messages back and forth.

However, similarly to query complexity, quantum advantages for total functions have been much more limited, with the best known separation of $Q(f) = O(\sqrt{R_2(f)})$ Buhrman, Cleve, and Wigderson [1999] and Aaronson and Ambainis [2005] for the set disjointness problem which is the natural communication counterpart of Grover's search. Anshu, Belovs, Ben-David, Göös, Jain, Kothari, Lee, and Santha [2016] have adapted the cheat sheet method to communication complexity, proving

Theorem 7. *Anshu, Belovs, Ben-David, Göös, Jain, Kothari, Lee, and Santha [ibid.]*

1. *There is a total function $f(x, y)$ with $Q_2(f) = \tilde{O}(R_2^{2/5}(f))$;*
2. *There is a total function $f(x, y)$ with $Q_E(f) = \tilde{O}(R_2^{2/3}(f))$;*

6 Quantum-classical separations on almost all inputs?

All known partial functions $f(x_1, \ldots, x_N)$ with a superpolynomial quantum advantage have the property that f takes one of values $f = 0$ and $f = 1$ on a very small subset of inputs. For example, for FORRELATION, the fraction of inputs with $f = 1$ is exponentially small in the number of variables N. This had led to a following conjecture (known as a folklore since about 1999):

Conjecture 1. *Aaronson and Ambainis [2014] Let $\mathbb{Q}$ be a quantum algorithm that makes T queries and let $\epsilon, \delta > 0$. There is a deterministic algorithm with a number of queries*

that is polynomial in T, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$ and approximates the probability of $\mathcal{Q}$ outputting 1 to an additive error ϵ on at least $1-\delta$ fraction of all inputs.

For total function, this conjecture implies that quantum and deterministic complexity are polynomially equivalent in the setting of approximately computing f. That is, for a total function f, let $D_\epsilon(f)$ and $Q_\epsilon(f)$ be the smallest number of queries for a (deterministic or quantum) algorithm that outputs the correct answer on at least $1-\epsilon$ fraction of inputs $(x_1,\ldots,x_N)$. Then, Conjecture 1 implies that $D_\epsilon(f)$ and $Q_{\epsilon'}(f)$ are polynomially related, for all constant ϵ,ϵ' with $\epsilon > \epsilon'$.

There is a natural path towards proving Conjecture 1. Due to Lemma 1, Conjecture 1 is implied by

Conjecture 2. *Aaronson and Ambainis [2014] Let $p(x_1,\ldots,x_N)$ be a polynomial of degree $2T$ which satisfies $|p(x_1,\ldots,x_N)| \leq 1$ for all $x_1,\ldots,x_N \in \{0,1\}$ and let $\epsilon,\delta > 0$. There is a deterministic algorithm with a number of queries that is polynomial in T, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$ and approximates $p(x_1,\ldots,x_N)$ to an additive error ϵ on at least $1-\delta$ fraction of all inputs.*

The natural way to design such a deterministic algorithm is by repeatedly choosing the variable x_i that has the biggest influence on the value of p (with the influence defined as $Inf_i(p) = E_x[|p(x)-p(x^{(\{i\})})|^2]$ with the expectation over a random choice of $x \in \{0,1\}^n$). To prove Conjecture 2, it suffices to show

Conjecture 3. *Aaronson and Ambainis [ibid.] Let $p(x_1,\ldots,x_N)$ be a polynomial of degree $2T$ which satisfies $|p(x_1,\ldots,x_N)| \leq 1$ for all $x_1,\ldots,x_N \in \{0,1\}$. Assume that*

$$E_{x\in\{0,1\}^n}\left[\left(p(x)-E[p(x)]\right)^2\right] \geq \epsilon.$$

Then, there is a variable i with $Inf_i[p] \geq \left(\frac{\epsilon}{T}\right)^c$ for some constant c.

Conjecture 3 connects with research in the analysis of Boolean functions. In particular, work of Dinur, Friedgut, Kindler, and O'Donnell [2006] implies a weaker form of the conjecture, with $Inf_i[p] \geq \frac{\epsilon^3}{2^{O(T)}}$. Improving it to $Inf_i[p] \geq \left(\frac{\epsilon}{T}\right)^c$ is a challenging open problem which is interesting for both analysis of Boolean functions and quantum query complexity.

7 Structure of quantum speedups?

Another related question is: when can we achieve large quantum speedups? From the known examples of exponential and superexponential speedups for partial functions, we

can observe that they are typically achieved for problems with an algebraic structure. For example, Simon [1997] showed an exponential speedup for the following problem:

Simon's problem. Let $N = 2^n$. We are promised that the input $(x_0, \dots, x_{N-1})$ (where $x_i \in [M]$) satisfies one of two promises:

(a) the mapping $i \to x_i$ is 2-to-1 with some $z \in [N], z \neq 0$ such that $x_y = x_{y \oplus z}$ for all $y \in [N]$, with $\oplus$ denoting bitwise addition modulo 2;

(b) the mapping $i \to x_i$ is 1-1.

As shown by Simon, $Q_2(f) = O(n)$ but $R_2(f) = \Omega(2^{n/2})$. However, randomly permuting inputs turns Simon's problem into the problem of distinguishing whether $i \to x_i$ is 2-1 or 1-1 for which it is known that $Q_2(f) = \Theta(2^{n/3})$ but $R_2(f) = \Theta(2^{n/2})$ Brassard, Høyer, and Tapp [1997] and Aaronson and Ambainis [2015], with the exponential quantum speedup disappearing. Similarly, permuting the input variables destroys the superexponential quantum speedup for the FORRELATION problem.

This leads to a question: can we show that quantum speedup is at most polynomial for any partial function that is symmetric with respect to permuting the input variables x_i? A positive answer would imply that large quantum speedups require problems with a structure (typically, of algebraic nature) that disappears if inputs are permuted.

For the case when x_i's are binary, evaluating a partial symmetric function essentially requires counting the number of $i : x_i = 1$ up to a certain precision (which is sufficient for distinguishing whether the input $x = (x_1, \dots, x_N)$ satisfies $f(x) = 0$ or $f(x) = 1$). Quantum algorithms can count $i : x_i = 1$ quadratically faster than classical algorithms Brassard, Høyer, and Tapp [1998] and it is easy to show that larger speedups cannot be obtained.

For non-binary inputs there are two possible ways of defining a "symmetric function":

(a) $f : [M]^N \to \{0, 1\}$ is symmetric, if $f(x_1, \dots, x_N) = f(x_{\pi(1)}, \dots, x_{\pi(N)})$ for any permutation π on $\{1, 2, \dots, N\}$;

(b) $f : [M]^N \to \{0, 1\}$ is symmetric, if $f(x_1, \dots, x_N) = f(\tau(x_{\pi(1)}), \dots, \tau(x_{\pi(N)}))$ for any permutations π on $\{1, 2, \dots, N\}$ and τ on $\{1, 2, \dots, M\}$.

For example, the property of being 1-1 or 2-1 is preserved both if $x_1, \dots, x_N$ are permuted and if the values for $x_1, \dots, x_N$ are permuted. Thus, it is symmetric in the second, stronger sense. Similarly, element distinctness (determining whether $x_1, \dots, x_N$ are all distinct) and other natural properties are symmetric in the second sense. For such properties, we have

Theorem 8. *Assume that a partial function $f : [M]^N \to \{0, 1\}$ is symmetric in the second sense. Then, $R_2(f) = O(Q_2^7(f) \log^c Q_2(f))$.*

It has been conjectured since about 2000 that a similar result also holds for f with a symmetry of the first type.

A related question has been studied by Aaronson and Ben-David [2016]: given a total function $f : \{0,1\}^N \to \{0,1\}$, can we define a subproblem f_P (f restricted to some subset $P \subseteq \{0,1\}^N$) for which $Q_2(f_P) = O(\log^c R_2(f_P))$?

For example, if $f(x_1, \ldots, x_N) = x_1 OR \ldots OR x_N$, then, for any restriction, the quantum advantage is at most quadratic. (An intuitive explanation is that computing OR is essentially equivalent to finding $i : x_i = 1$ and, for search, the quantum advantage is quadratic whatever the number of $i : x_i = 1$ is.) In contrast, both MAJORITY and PARITY can be restricted so that quantum advantage becomes exponential.

The next theorem gives a full characterization when superpolynomial speedups can be achieved:

Theorem 9. *Aaronson and Ambainis [2014] A promise $P \subseteq \{0,1\}^N$ with $Q_2(f_P) = O(N^{o(1)})$ and $R_2(f_P) = \Omega(N^{\Omega(1)})$ exists if and only if, for some $c > 0$, there are 2^{N^c} inputs $x \in \{0,1\}^N$ with $C_x(f) \geq N^c$.*

8 From polynomials to quantum algorithms

As shown by Lemma 1, a quantum algorithm that makes k queries can be converted into a polynomial of degree at most $2k$. In the opposite direction, the existence of a polynomial of degree $2k$ does not imply the existence of a quantum algorithm that makes k queries. As mentioned in Section 5.1, there is a total f with $Q_2(f) = \Omega(\widetilde{deg}^{4-o(1)}(f))$ Aaronson, Ben-David, and Kothari [2016].

However, there is an interesting particular case in which polynomials and quantum algorithms are equivalent.

Theorem 10 (Aaronson, Ambainis, Iraids, Kokainis, and Smotrovs [2016])**.** *Let $f(x_1, \ldots, x_N)$ be a partial Boolean function. Assume that there is a polynomial $p(x_1, \ldots, x_N)$ of degree 2 with the following properties:*

- *for any $x_1, \ldots, x_N \in \{0,1\}$, $0 \leq p(x_1, \ldots, x_N) \leq 1$;*
- *if $f(x_1, \ldots, x_N) = 1$, $p(x_1, \ldots, x_N) \geq \frac{1}{2} + \delta$;*
- *if $f(x_1, \ldots, x_N) = 0$, $p(x_1, \ldots, x_N) \leq \frac{1}{2} - \delta$.*

Then, $f(x_1, \ldots, x_N)$ can be computed by a 1-query quantum algorithm with the probability of correct answer at least $\frac{1}{2} + \frac{\delta}{3(2K+1)}$ where K is the Groethendieck's constant Pisier [2012] (for which it is known that $1.5707... \leq K \leq 1.7822...$ Braverman, K. Makarychev, Y. Makarychev, and Naor [2011]).

The main ideas for the transformation from a polynomial to a quantum algorithm are as follows:

1. For technical convenience, we assume that x_i are $\{-1,1\}$-valued (instead of $\{0,1\}$-valued). We start by transforming a polynomial $p(x_1,\dots,x_N)$ into another polynomial
$$q(x_1,\dots,x_N,y_1,\dots,y_N)=\sum_{i,j} a_{i,j}x_i y_j$$
which satisfies $q(x_1,\dots,x_N,x_1,\dots,x_N)=p(x_1,\dots,x_N)$ for all $x_1,\dots,x_N\in\{-1,1\}$ and $|q(x_1,\dots,x_N,y_1,\dots,y_N)|\le 1$ for all $x_1,\dots,x_N,y_1,\dots,y_N\in\{-1,1\}$.

2. If the spectral norm $\|A\|$ of the matrix A is small, then the polynomial q can be transformed into a quantum algorithm:

   Lemma 3. *Let $A=(a_{ij})_{i\in[N],j\in[M]}$ with $\sqrt{NM}\|A\|\le C$ and let*
$$q(x_1,\dots,x_N,y_1,\dots,y_M)=\sum_{i=1}^{N}\sum_{j=1}^{M} a_{ij}x_i y_j.$$
   Then, there is a quantum algorithm that makes 1 query to $x_1,\dots,x_N$, $y_1,\dots,y_M$ and outputs 1 with probability
$$r=\frac{1}{2}\left(1+\frac{q(x_1,\dots,x_N,y_1,\dots,y_M)}{C}\right).$$
   The quantum algorithm consists of creating a combination of quantum states $|\psi\rangle=\sum_{i=1}^{N}\frac{x_i}{\sqrt{N}}|i\rangle$ and $|\phi\rangle=\sum_{j=1}^{M}\frac{y_i}{\sqrt{M}}|j\rangle$, applying $U=\sqrt{NM}\cdot A$ to $|\phi\rangle$ and then using the SWAP test Buhrman, Cleve, Watrous, and De Wolf [2002] to estimate the inner product of $|\psi\rangle$ and $U|\phi\rangle$ which happens to be equal to the desired quantity $\sum_{i=1}^{N}\sum_{j=1}^{M}a_{ij}x_i y_j$.

   If U is unitary, we can apply this procedure as described. If $\|U\|=C>1$, U is not unitary and cannot be applied directly. Instead, we design and apply a unitary transformation that is equal to $\frac{1}{C}U$ on a certain subspace.

3. For the general case, a corollary of Groethendieck's inequality Pisier [2012], Aaronson, Ambainis, Iraids, Kokainis, and Smotrovs [2016], and Arunachalam, Briët, and Palazuelos [2017] implies that, if a_{ij} are such that $|\sum_{i=1}^{N}\sum_{j=1}^{M}a_{ij}x_i y_j|\le 1$ for all choices of $x_i\in\{-1,1\}$ and $y_j\in\{-1,1\}$, there

exist $\vec{u} = (u_i)_{i \in N}$ and $\vec{v} = (v_j)_{j \in M}$ such that $\|\vec{u}\| = 1$, $\|\vec{v}\| = 1$, $a_{ij} = b_{ij} u_i v_j$ for all $i \in [N]$, $j \in [M]$ and $B = (b_{ij})_{i,j}$ satisfies $\|B\| \leq K$.

Then, we can perform a similar algorithm with quantum states $|\psi\rangle = \sum_{i=1}^{N} u_i x_i |i\rangle$ and $|\phi\rangle = \sum_{j=1}^{M} v_j y_j |j\rangle$.

Following this work, it was shown Arunachalam, Briët, and Palazuelos [2017] that quantum algorithms are equivalent to polynomial representations by polynomials of a particular type. Namely, the accepting probability of a t query quantum algorithm is equal to a completely bounded form of degree $2t$. For $t = 1$, representatios of f by a completely bounded forms are equivalent to representations by general polynomials (implying Theorem 10) but this does not hold for $t \geq 2$.

References

Scott Aaronson (2008). "Quantum certificate complexity". *J. Comput. System Sci.* 74.3, pp. 313–322. MR: 2384077 (cit. on p. 3289).

Scott Aaronson and Andris Ambainis (2005). "Quantum search of spatial regions". *Theory Comput.* 1, pp. 47–79. MR: 2322514 (cit. on p. 3297).

– (2014). "The need for structure in quantum speedups". *Theory Comput.* 10, pp. 133–166. MR: 3249097 (cit. on pp. 3297, 3298, 3300).

– (2015). "Forrelation: a problem that optimally separates quantum from classical computing". In: *STOC'15—Proceedings of the 2015 ACM Symposium on Theory of Computing*. ACM, New York, pp. 307–316. MR: 3388209 (cit. on pp. 3285, 3289, 3291, 3292, 3299).

Scott Aaronson, Andris Ambainis, Jānis Iraids, Martins Kokainis, and Juris Smotrovs (2016). "Polynomials, quantum query complexity, and Grothendieck's inequality". In: *31st Conference on Computational Complexity*. Vol. 50. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 25, 19. MR: 3540826 (cit. on pp. 3300, 3301).

Scott Aaronson and Shalev Ben-David (2016). "Sculpting quantum speedups". In: *31st Conference on Computational Complexity*. Vol. 50. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 26, 28. MR: 3540827 (cit. on p. 3300).

Scott Aaronson, Shalev Ben-David, and Robin Kothari (2016). "Separations in query complexity using cheat sheets". In: *STOC'16—Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, New York, pp. 863–876. MR: 3536620 (cit. on pp. 3285, 3295, 3296, 3300).

Scott Aaronson and Yaoyun Shi (2004). "Quantum lower bounds for the collision and the element distinctness problems". *J. ACM* 51.4, pp. 595–605. MR: 2147849 (cit. on p. 3285).

Andris Ambainis (2004). "Quantum search algorithms". *ACM SIGACT News* 35.2, pp. 22–35 (cit. on p. 3284).

– (2007). "Quantum walk algorithm for element distinctness". *SIAM J. Comput.* 37.1, pp. 210–239. MR: 2306290 (cit. on p. 3285).

– (2016). "Superlinear advantage for exact quantum algorithms". *SIAM J. Comput.* 45.2, pp. 617–631. MR: 3490898 (cit. on p. 3293).

Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs (2017). "Separations in query complexity based on pointer functions". *J. ACM* 64.5, Art. 32, 24. MR: 3716887 (cit. on pp. 3293–3295).

Anurag Anshu, Aleksandrs Belovs, Shalev Ben-David, Mika Göös, Rahul Jain, Robin Kothari, Troy Lee, and Miklos Santha (2016). "Separations in communication complexity using cheat sheets and information complexity [extended abstract]". In: *57th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2016*. IEEE Computer Soc., Los Alamitos, CA, pp. 555–564. MR: 3631018 (cit. on p. 3297).

Srinivasan Arunachalam, Jop Briët, and Carlos Palazuelos (2017). "Quantum Query Algorithms are Completely Bounded Forms". arXiv: 1711.07285 (cit. on pp. 3285, 3301, 3302).

Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf (2001). "Quantum lower bounds by polynomials". *J. ACM* 48.4, pp. 778–797. MR: 2144930 (cit. on pp. 3288, 3289, 3291, 3292).

Daniel J Bernstein, Stacey Jeffery, Tanja Lange, and Alexander Meurer (2013). "Quantum algorithms for the subset-sum problem". In: *International Workshop on Post-Quantum Cryptography*. Springer, pp. 16–33 (cit. on p. 3285).

Gilles Brassard, Peter Høyer, and Alain Tapp (1997). "Quantum cryptanalysis of hash and claw-free functions". *ACM Sigact News* 28.2, pp. 14–19 (cit. on p. 3299).

Gilles Brassard, Peter Høyer, and Alain Tapp (1998). "Quantum counting". In: *Automata, languages and programming (Aalborg, 1998)*. Vol. 1443. Lecture Notes in Comput. Sci. Springer, Berlin, pp. 820–831. MR: 1683527 (cit. on p. 3299).

Mark Braverman, Konstantin Makarychev, Yury Makarychev, and Assaf Naor (2011). "The Grothendieck constant is strictly smaller than Krivine's bound". In: *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science—FOCS 2011*. IEEE Computer Soc., Los Alamitos, CA, pp. 453–462. MR: 2932721 (cit. on p. 3300).

Harry Buhrman, Richard Cleve, John Watrous, and Ronald De Wolf (2002). "Quantum fingerprinting". *Physical Review Letters* 87.16, p. 167902 (cit. on pp. 3290, 3301).

Harry Buhrman, Richard Cleve, and Avi Wigderson (1999). "Quantum vs. classical communication and computation". In: *STOC '98 (Dallas, TX)*. ACM, New York, pp. 63–68. MR: 1731563 (cit. on p. 3297).

Harry Buhrman and Robert Špalek (2006). "Quantum verification of matrix products". In: *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, pp. 880–889. MR: 2373814 (cit. on p. 3285).

Harry Buhrman and Ronald de Wolf (2002). "Complexity measures and decision tree complexity: a survey". *Theoret. Comput. Sci.* 288.1. Complexity and logic (Vienna, 1998), pp. 21–43. MR: 1934888 (cit. on p. 3284).

Sourav Chakraborty, Eldar Fischer, Arie Matsliah, and Ronald de Wolf (2010). "New results on quantum property testing". In: *30th International Conference on Foundations of Software Technology and Theoretical Computer Science*. Vol. 8. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, pp. 145–156. MR: 2853831 (cit. on p. 3284).

J Ignacio Cirac and Peter Zoller (2012). "Goals and opportunities in quantum simulation". *Nature Physics* 8.4, pp. 264–266 (cit. on p. 3284).

Irit Dinur, Ehud Friedgut, Guy Kindler, and Ryan O'Donnell (2006). "On the Fourier tails of bounded functions over the discrete cube (extended abstract)". In: *STOC'06: Proceedings of the 38th Annual ACM Symposium on Theory of Computing*. ACM, New York, pp. 437–446. MR: 2277169 (cit. on p. 3298).

IM Georgescu, Sahel Ashhab, and Franco Nori (2014). "Quantum simulation". *Reviews of Modern Physics* 86.1, p. 153 (cit. on p. 3284).

Mika Göös, Toniann Pitassi, and Thomas Watson (2015). "Deterministic communication vs. partition number". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science—FOCS 2015*. IEEE Computer Soc., Los Alamitos, CA, pp. 1077–1088. MR: 3473358 (cit. on p. 3293).

Lov K. Grover (1996). "A fast quantum mechanical algorithm for database search". In: *Proceedings of the Twenty-eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*. ACM, New York, pp. 212–219. MR: 1427516 (cit. on pp. 3284, 3296).

Stephen Jordan (n.d.). "Quantum algorithm zoo" (cit. on p. 3285).

Bo'az Klartag and Oded Regev (2011). "Quantum one-way communication can be exponentially stronger than classical communication". In: *STOC'11—Proceedings of the 43rd ACM Symposium on Theory of Computing*. ACM, New York, pp. 31–40. MR: 2931952 (cit. on p. 3297).

Eyal Kushilevitz and Noam Nisan (1997). *Communication complexity*. Cambridge University Press, Cambridge, pp. xiv+189. MR: 1426129 (cit. on p. 3297).

Troy Lee and Adi Shraibman (2007). "Lower bounds in communication complexity". *Found. Trends Theor. Comput. Sci.* 3.4, front matter, 263–399 (2009). MR: 2558900 (cit. on p. 3297).

Gatis Midrijanis (2004). "Exact quantum query complexity for total Boolean functions". arXiv: quant-ph/0403168 (cit. on p. 3292).

Sagnik Mukhopadhyay and Swagato Sanyal (2015). "Towards better separation between deterministic and randomized query complexity". In: *35th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*. Vol. 45. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, pp. 206–220. MR: 3464466 (cit. on p. 3294).

Michael A. Nielsen and Isaac L. Chuang (2000). *Quantum computation and quantum information*. Cambridge University Press, Cambridge, pp. xxvi+676. MR: 1796805 (cit. on p. 3287).

Noam Nisan (1991). "CREW PRAMs and decision trees". *SIAM J. Comput.* 20.6, pp. 999–1007. MR: 1135744 (cit. on pp. 3288, 3289, 3292).

Noam Nisan and Márió Szegedy (1994). "On the degree of Boolean functions as real polynomials". *Comput. Complexity* 4.4. Special issue on circuit complexity (Barbados, 1992), pp. 301–313. MR: 1313531.

Gilles Pisier (2012). "Grothendieck's theorem, past and present". *Bull. Amer. Math. Soc. (N.S.)* 49.2, pp. 237–323. MR: 2888168 (cit. on pp. 3300, 3301).

Ran Raz (1999). "Exponential separation of quantum and classical communication complexity". In: *Annual ACM Symposium on Theory of Computing (Atlanta, GA, 1999)*. ACM, New York, pp. 358–367. MR: 1798056 (cit. on p. 3297).

Michael Saks and Avi Wigderson (1986). "Probabilistic Boolean decision trees and the complexity of evaluating game trees". In: *Foundations of Computer Science, 1986., 27th Annual Symposium on*. IEEE, pp. 29–38 (cit. on pp. 3285, 3292).

Peter W. Shor (1997). "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer". *SIAM J. Comput.* 26.5, pp. 1484–1509. MR: 1471990 (cit. on p. 3284).

Daniel R. Simon (1997). "On the power of quantum computation". *SIAM J. Comput.* 26.5, pp. 1474–1483. MR: 1471989 (cit. on pp. 3292, 3299).

Received 2017-12-09.

Andris Ambainis
Faculty of Computing
University of Latvia
Raina bulvāris 19
Riga, LV-1586
Latvia
andris.ambainis@lu.lv