Int J Theor Phys (2009) 48: 2412–2426
DOI 10.1007/s10773-009-0031-0

The 50% Advanced Information Rule of the Quantum Algorithms

Giuseppe Castagnoli

Received: 16 February 2009 / Accepted: 6 May 2009 / Published online: 15 May 2009

Abstract The oracle chooses a function out of a known set of functions and gives to the player a black box that, given an argument, evaluates the function. The player should find out a certain character of the function (e.g. its period) through function evaluation. This is the typical problem addressed by the quantum algorithms. In former theoretical work, we showed that a quantum algorithm requires the number of function evaluations of a classical algorithm that knows in advance 50% of the information that specifies the solution of the problem. This requires representing physically, besides the solution algorithm, the possible choices of the oracle.

Here we check that this 50% *rule* holds for the main quantum algorithms. In structured problems, a classical algorithm with the advanced information, to identify the missing information should perform one function evaluation. The speed up is exponential since a classical algorithm without advanced information should perform an exponential number of function evaluations. In unstructured database search, a classical algorithm that knows in advance $n/2$ bits of the database location, to identify the $n/2$ missing bits should perform $O(2^{n/2})$ function evaluations. The speed up is quadratic since a classical algorithm without advanced information should perform $O(2^n)$ function evaluations. The 50% rule allows to identify in an entirely classical way the problems solvable with a quantum sped up.

The advanced information classical algorithm also defines the quantum algorithm that solves the problem. Each classical history, corresponding to a possible way of getting the advanced information and a possible result of computing the missing information, is represented in quantum notation as a sequence of sharp states. The sum of the histories yields the function evaluation stage of the quantum algorithm. Function evaluation entangles the oracle's choice register (containing the function chosen by the oracle) and the solution register (in which to read the solution at the end of the algorithm). Information about the oracle's choice propagates from the former to the latter register. Then the basis of the solution register should be rotated to make this information readable. This defines the quantum algorithm, or its iterate and the number of iterations.

G. Castagnoli (✉)
Pieve Ligure, Genova, Italy
e-mail: giuseppe.castagnoli@gmail.com

Keywords Quantum computation · Quantum speed up · Quantum algorithm · Advanced information · 50% rule

1 Introduction

We provide some context.

The problem typically addressed by a quantum algorithm can be seen as a competition between two players. There is a set of functions known to both players, for example the set of the periodic functions $f_{\mathbf{k}}(x) : \{0,1\}^n \to \{0,1\}^{n-1}$. The first player (the oracle) chooses one of these functions and gives to the second player a black box (i.e. non-inspectable inside) hardwired for the computation of that function. The second player should find out a certain character of the function (for example its period) by computing $f_{\mathbf{k}}(x)$ for different values of x. As well known, the quantum algorithm requires a substantially lower number of function evaluations than the corresponding classical algorithm.

In [5–7], on the basis of theoretical considerations, we showed that the quantum algorithm requires the number of function evaluations of a classical algorithm that knows in advance 50% of the information that specifies the solution of the problem (see also [4]).

To see this, the key step (common to the present work) is representing physically the interdependence between the problem and the solution. One should represent, together with the algorithm that solves the problem, the production of the problem on the part of the oracle. To this end, we replace the black box hardwired for the computation of $f_{\mathbf{k}}(x)$ by a general purpose black box, that, given the inputs x (the argument of the function) and $\mathbf{k}$ (the table of the function $f_{\mathbf{k}}(x)$—each row being the pair "value of the argument"/"value of the function"), computes $f(\mathbf{k}, x) = f_{\mathbf{k}}(x)$. We need an auxiliary input register K that contains $\mathbf{k}$. If the function is binary, $\mathbf{k}$ is an unstructured bit string (the sequence of function values for increasing arguments), otherwise it is a structured string. Register K, just a conceptual reference, is ideally added to the usual input register X (containing the value of x to query the black box with) and output register V (hosting the result of function evaluation $f(\mathbf{k}, x)$). K is prepared in the even weighted superposition of all the possible valuations of $\mathbf{k}$, X and V, as usual, respectively in the even weighted superposition of all the possible values of x and in an initial state depending on the algorithm. Each function evaluation entangles the oracle's choice register K and the solution register X. Correspondingly, information about the oracle's choice propagates from the former to the latter register. Eventually, we rotate the basis of X to make this information readable. Function evaluation and rotation of the X basis is done once (for just one function evaluation) in Deutsch's, Deutsch&Jozsa's, and Simon's algorithm, iteratively (for a number of iterations $O(2^{n/2})$) in Grover's algorithm. Measuring the content of K and X at the end of the algorithm, induces state reduction on both the function chosen by the oracle (a valuation of $\mathbf{k}$ hosted in register K) and the solution produced by the algorithm (hosted in register X).[1]

In this picture, quantum computation is reduction on the solution of the problem under a relation (or correlation) representing problem-solution interdependence—the correlation is between the content of register K (the oracle's choice) and the content of register X (the solution) measured at the end of the algorithm. Backdating in time, to before running the algorithm, a time symmetric part of this reduction shows that the quantum algorithm requires

[1] Backdating to before running the algorithm the reduction induced by measuring the content of K yields the usual quantum algorithm.

the number of function evaluations of a classical algorithm that knows in advance 50% of the information about the solution of the problem. We call this the 50% *rule*.

In this context, the information that specifies the solution of the problem is the information that specifies both the solution and the problem, this latter being the information in the string **k**. Since the solution is a function of **k**, the information that specifies the solution is redundant with respect to the information in **k**. Knowing in advance 50% of the information that specifies the solution of the problem amounts to knowing in advance 50% of the information in **k**.

The main objective of this work is checking that the 50% rule holds for the main quantum algorithms: Deutsch, Deutsch&Jozsa, Simon (where the analysis extends by similarity to the hidden subgroup algorithms, thus in particular to the quantum part of Shor's algorithm), and Grover. First we review the extended representation of the algorithm (extended to the oracle's choice), then we show that the number of function evaluations required by the quantum algorithm is the same of a classical algorithm that knows in advance 50% of the information in **k**.

In the case of the structured problems, the classical algorithm knowing in advance 50% of the rows of the table (excluding those half tables that already specify the solution of the problem), in order to identify the solution of the problem should compute one (any one) of the missing rows—i.e. perform one function evaluation for any value of x outside the advanced information. The speed up is exponential since a classical algorithm without advanced information should compute an exponential number of rows.

In unstructured database search, a classical algorithm that knows in advance $n/2$ bits of the database location, to identify the $n/2$ missing bits should perform $O(2^{n/2})$ function evaluations. The speed up is quadratic since a classical algorithm without advanced information should perform $O(2^n)$ function evaluations.

Thus the speed up comes from comparing two classical algorithms, with and without advanced information. The 50% rule brings the identification of the problems liable of being solved with a quantum speed up to an entirely classical framework.

We also pursue a secondary objective that comes natural in this context. We show that the advanced information classical algorithm defines the quantum algorithm. To this end we should consider the "skeleton" of the classical algorithm. This, given the advanced information, performs the function evaluations required to identify the solution of the problem (identifying does not mean computing)—this is what is needed to build the quantum algorithm out of it.

We consider all the possible histories of this classical algorithm. Each history—corresponding to a possible way of getting the advanced information and a possible result of computing the missing information—is represented in quantum notation as a sequence of sharp states. The sum of the histories yields the function evaluation stage of the quantum algorithm, namely the initial preparation and the outcome of function evaluation; the initial phase of each history should be chosen in such a way that the transfer of information from the classical to the quantum algorithm is maximized—this yields the initial state of register V for the various algorithms.

As already said, each function evaluation entangles the oracle's choice register K and the solution register X and information about the oracle's choice propagates from the former to the latter register. Which is this information is clear when the entanglement produced by function evaluation is maximal, which is the case in the algorithms of Deutsch, Deutsch&Jozsa (where the only function evaluation produces a maximally entangled state), and Grover (where each function evaluation feeds the amplitude a maximally entangled state at the expense of the amplitude of an unentangled state). In the maximally entangled state,

each orthogonal state of K (possibly itself a quantum superposition) corresponds to a solution of the problem and is correlated with an orthogonal state of X. This allows to define in a constructive way the rotation of the X basis that makes the solution readable, which completes the definition of the quantum algorithm (of the algorithm's iterate in Grover's case). Which is the information propagated to register X is less clear in the case of Simon's and the hidden subgroup algorithms, where the entanglement produced by function evaluation is not maximal; the final rotation of the X basis (here the Hadamard transform on X) can still be defined as the transformation that maximizes the information about the oracle's choice readable in it; this helps in circumscribing the problem but is no more a constructive definition.

Summing up, the 50% rule brings the search of the speed ups to an entirely classical framework. Once identified a problem liable of speed up, the same rule could be used a second time for searching the quantum algorithm that solves the problem. In fact the advanced information classical algorithm defines the quantum algorithm.

2 Deutsch's Algorithm

2.1 Reviewing and Extending the Algorithm

We review Deutsch's algorithm [9] as revised by Cleve et al. [8]. The problem is as follows. An oracle chooses at random one of the four binary functions $f_{\mathbf{k}} : \{0, 1\} \to \{0, 1\}$, see Table 1. $\mathbf{k} \equiv k_0, k_1$ is a two-bit string belonging to $\{0, 1\}^2$.

Note that $k_0 = f_{\mathbf{k}}(0)$ and $k_1 = f_{\mathbf{k}}(1)$—the string $\mathbf{k}$ is both the subfix of f and, clockwise rotated, the table of function values ordered for increasing values of the argument. Then the oracle gives to the second player a black box that, given the inputs $\mathbf{k}$ and x, computes $f_{\mathbf{k}}(x)$. The second player, by trying function evaluation for different values of x, must find whether the function is balanced (i.e. f_{01} or f_{10}, with an even number of zeroes and ones) or constant (i.e. f_{00} or f_{11}). This requires two function evaluations in the classical case (for $x = 0$ and $x = 1$), just one in the quantum case.

Deutsch's algorithm is the root of all subsequent quantum algorithms, for what concerns both the speed up and the representation of quantum computation. Bits are replaced by qubits [11, 12] and reversible logical operations [2, 3, 13] by unitary transformations [1, 11, 12].

The original algorithm uses a one qubit input register X, containing the value of x chosen by the second player to query the black box with, and a one qubit output register V, initially containing v and eventually the result of function evaluation. Function evaluation leaves X unaltered and puts $v \oplus f_{\mathbf{k}}(x)$ in V—the result of function evaluation is module 2 added to the former content of V for logical reversibility. The quantum algorithm consists of three steps: (0) preparing register X in an even weighted superposition of all the possible values of x and register V in the antisymmetric state, (1) performing function evaluation, and (2) applying the Hadamard transform to register X.

Table 1 Set of functions known to both players in Deutsch's algorithm

x	$f_{00}(x)$	$f_{01}(x)$	$f_{10}(x)$	$f_{11}(x)$
0	0	0	1	1
1	0	1	0	1

We give directly Deutsch's algorithm extended to the representation of the random choice of the function on the part of the oracle [5–7]. To this end we add to the description an auxiliary input register K containing the oracle's choice $\mathbf{k}$. The initial state is:

$$\Psi_0 = \frac{1}{4}(|00\rangle_K + |01\rangle_K + |10\rangle_K + |11\rangle_K)(|0\rangle_X + |1\rangle_X)(|0\rangle_Y - |1\rangle_Y), \tag{1}$$

where the superposition hosted in register K is indifferently coherent or incoherent (in the latter case each ket should be multiplied by a random phase factor, e.g. $|00\rangle_K$ would become $e^{i\delta_{00}}|00\rangle_K$, with δ_{00} a random angle with uniform distribution in $[0, 2\pi]$ etc.). This superposition represents the initial ignorance of the second player about the oracle's choice.

Function evaluation transforms Ψ_0 into:

$$\Psi_1 = \frac{1}{4}\big[(|00\rangle_K - |11\rangle_K)(|0\rangle_X + |1\rangle_X) + (|01\rangle_K - |10\rangle_K)(|0\rangle_X - |1\rangle_X)\big](|0\rangle_Y - |1\rangle_Y). \tag{2}$$

Applying the Hadamard transform to register X yields

$$\Psi_2 = \frac{1}{2\sqrt{2}}\big[(|00\rangle_K - |11\rangle_K)|0\rangle_X + (|01\rangle_K - |10\rangle_K)|1\rangle_X\big](|0\rangle_Y - |1\rangle_Y). \tag{3}$$

Let us denote by $[K]$ the content of register K, by $[X]$ the content of X, etc. Measuring $[K]$ and $[X]$ in (3) determines the moves of both players, namely the oracle's choice $\mathbf{k} \equiv k_0, k_1$ (in register K) and the solution found by the second player, namely $k_0 \oplus k_1$ (in register X). Backdating to before running the algorithm the state reduction induced by measuring $[K]$ gives the original Deutsch's algorithm—it generates at random the value of $\mathbf{k}$ hosted in the black box.

2.2 Checking the 50% Rule

The information acquired by measuring $[K]$ and $[X]$ in (3) is 2 bits—the two bits of register K; in fact the content of X is a function of the content of K, therefore the information contained in X is redundant. The quantum algorithm requires the number of function evaluations of a classical algorithm working on a solution space reduced in size because one bit of information about the content of K, either $k_0 = f(\mathbf{k}, 0)$ or $k_1 = f(\mathbf{k}, 1)$, is known in advance. To identify the character of the function, this algorithm must acquire the other bit of information by computing, respectively, either $k_1 = f(\mathbf{k}, 1)$ or $k_0 = f(\mathbf{k}, 0)$. Thus the classical algorithm has to perform just one function evaluation like the quantum algorithm. This verifies the 50% rule in the case of Deutsch's algorithm.

Interestingly, this rule shows that Deutsch's problem is liable of a quantum speed up independently of our knowledge of the quantum algorithm. In fact the speed up comes from comparing two classical algorithms, with and without the advanced information.

2.3 Building the Quantum Algorithm out of the Advanced Information Classical Algorithm

Let us build the function evaluation stage of the quantum algorithm out of the corresponding stage of a classical algorithm that knows in advance 50% of $\mathbf{k}$. We should combine all the possible ways of getting the advanced information and all the possible results of computing the missing information, see Table 2.

We represent the possible histories in quantum notation. Since we are dealing with classical computations, we require that the input and the output of each history (before and after function evaluation) is a sharp quantum state. There are sixteen possible histories:

Table 2 Combinations

#	Advanced information	Result of function evaluation	Character of the function
1	$k_0 = 0$	$k_1 = f(\mathbf{k}, 1) = 0$	constant
2	$k_0 = 0$	$k_1 = f(\mathbf{k}, 1) = 1$	balanced
3	$k_0 = 1$	$k_1 = f(\mathbf{k}, 1) = 0$	balanced
4	$k_0 = 1$	$k_1 = f(\mathbf{k}, 1) = 1$	constant
5	$k_1 = 0$	$k_0 = f(\mathbf{k}, 0) = 0$	constant
6	$k_1 = 0$	$k_0 = f(\mathbf{k}, 0) = 1$	balanced
7	$k_1 = 1$	$k_0 = f(\mathbf{k}, 0) = 0$	balanced
8	$k_1 = 1$	$k_0 = f(\mathbf{k}, 0) = 1$	constant

- Row #1. The advanced information is $k_0 = 0$. The classical algorithm should compute $k_1 = f(\mathbf{k}, 1)$ that, for this row, is $k_1 = f(\mathbf{k}, 1) = 0$. The quantum representation of the oracle's choice is thus $|00\rangle_K$. Register X should be in $|1\rangle_X$, the state to query the black box with in order to compute $f(\mathbf{k}, 1)$. Since the result of this computation is module 2 added to the initial content of register V, we should split row #1 into two sub-rows: #1.1 with register V initially in $|0\rangle_V$ and #1.2 with register V initially in $|1\rangle_V$. The initial state of history #1.1 is thus $\Psi_0^{(1.1)} = |00\rangle_K|1\rangle_X|0\rangle_V$, that of history #1.2 is $\Psi_0^{(1.2)} = -|00\rangle_K|1\rangle_X|1\rangle_V$. These computation histories have to be added together and must be given an initial phase. For the time being, we set the initial phase of each history in such a way that, in the superposition of all histories, we obtain the initial state of the quantum algorithm; further below we justify this choice independently of our a priori knowledge of the quantum algorithm. To simplify the notation, we sum together the initial states of these two histories: $\Psi_0^{(1)} = \Psi_0^{(1.1)} + \Psi_0^{(1.2)} = |00\rangle_K|1\rangle_X(|0\rangle_V - |1\rangle_V)$. We take care of normalization at the end. Function evaluation transforms $\Psi_0^{(1)}$ into itself: $\Psi_1^{(1)} = \Psi_0^{(1)}$ (module 2 adding $f(00, 1) = 0$ to the former content of V leaves this content unaltered).
- Row #5. Advanced information $k_1 = 0$, result of function evaluation $k_0 = f(\mathbf{k}, 0) = 0$. Applying the same rationale of the above point, we obtain the initial state $\Psi_0^{(5)} = |00\rangle_K|0\rangle_X(|0\rangle_V - |1\rangle_V)$; function evaluation transforms this state into itself: $\Psi_1^{(5)} = \Psi_0^{(5)}$.
- The sum of the histories of rows #1 and #5 yields the transformation of $|00\rangle_K(|0\rangle_X + |1\rangle_X)(|0\rangle_V - |1\rangle_V)$ into itself, namely the function evaluation stage of Deutsch's algorithm when K is in $|00\rangle_K$.
- Row #2. Advanced information $k_0 = 0$; result of function evaluation $k_1 = f(\mathbf{k}, 1) = 1$; initial state $\Psi_0^{(2)} = |01\rangle_K|1\rangle_X(|0\rangle_V - |1\rangle_V)$; state after function evaluation $\Psi_1^{(2)} = -|01\rangle_K|1\rangle_X(|0\rangle_V - |1\rangle_V)$ (module 2 adding $f(01, 1) = 1$ to the former content of V swaps $|0\rangle_V$ and $|1\rangle_V$; the overall result is rotating the phase of the present pair of histories by π).
- Row #7. Advanced information $k_1 = 1$, result of function evaluation $k_0 = f(\mathbf{k}, 0) = 0$; initial state $\Psi_0^{(7)} = |01\rangle_K|0\rangle_X(|0\rangle_V - |1\rangle_V)$; state after function evaluation $\Psi_1^{(7)} = \Psi_0^{(7)}$(module 2 adding $f(01, 0) = 0$ to the former content of V leaves this content unaltered).
- The sum of the histories of rows #2 and #7 yields the transformation of $|01\rangle_K(|0\rangle_X + |1\rangle_X)(|0\rangle_V - |1\rangle_V)$ into $|01\rangle_K(|0\rangle_X - |1\rangle_X)(|0\rangle_V - |1\rangle_V)$, namely the function evaluation stage of Deutsch's algorithm when K is in $|01\rangle_K$.
- We proceed in a similar way for the other histories. Summing together this quantum representations of the 16 classical histories and normalizing yields the transformation of Ψ_0 (1) into Ψ_1 (2).

In hindsight, we can see a shortcut. With reference to the above classical algorithm in quantum notation, for each $|\mathbf{k}\rangle_K$, we perform function evaluation not only for those values of x required to identify the solution of the problem, but also for all the other possible values of x; in other words we perform function evaluation for each product $|\mathbf{k}\rangle_K(|0\rangle_X + |1\rangle_X)(|0\rangle_V - |1\rangle_V)$; junk histories (for that $|\mathbf{k}\rangle_K$) do not harm, the important thing is performing function evaluation for the values of x required to identify the solution. As one can see, this yields directly the transformation of Ψ_0 (1) into Ψ_1 (2). Conversely, by simply inspecting the form of Ψ_0 in (1), one can see that each $|\mathbf{k}\rangle_K(|0\rangle_X + |1\rangle_X)(|0\rangle_V - |1\rangle_V)$ is the initial state of a bunch of histories as from the above shortcut. This holds in general for the function evaluation stage of all quantum algorithms.

Summing up, quantum parallel computation can be seen as the sum of the histories of a classical algorithm that, given the advanced information, computes the missing information required to identify the solution of the problem.

By considering the sum of the histories, we can justify the choice of the initial phase. We take the generic initial state of register V: $\alpha(|0\rangle_V + |1\rangle_V) + \beta(|0\rangle_V - |1\rangle_V)$; the initial phase of the histories with register V in $|0\rangle_V$ becomes $\alpha + \beta$, that of the histories with V in $|1\rangle_V$ becomes $\alpha - \beta$. Under the amplitude α, the computation performed by the reference classical algorithm gets lost in the quantum translation, since the overall initial state of the reference algorithm is transformed into itself. Under β we obtain the above development, where the transfer of information from the classical to the quantum algorithm is maximum; this justifies the choice $\alpha = 0$ and $\beta = 1$.

Now we look at the outcome of the second stage (2). Function evaluation has created a maximal entanglement between registers K and X, two orthogonal states of K, $|00\rangle_K - |11\rangle_K$ and $|01\rangle_K - |10\rangle_K$ (or indifferently $e^{i\delta_{00}}|00\rangle_K - e^{i\delta_{11}}|11\rangle_K$ and $e^{i\delta_{01}}|01\rangle_K - e^{i\delta_{10}}|10\rangle_K$) are correlated with two orthogonal states of X, respectively $|0\rangle_X + |1\rangle_X$ and $|0\rangle_X - |1\rangle_X$. This means that, after function evaluation, register X contains the information that discriminates between $|00\rangle_K - |11\rangle_K$ and $|01\rangle_K - |10\rangle_K$, namely between constant and balanced functions. Therefore we should rotate the X basis in such a way that this information becomes readable: $|0\rangle_X + |1\rangle_X$ should be transformed into $|0\rangle_X$, etc. This is a constructive definition of the Hadamard transform on register X. This completes the derivation of the quantum algorithm from the classical algorithm with the advanced information.

3 Deutsch&Jozsa Algorithm

3.1 Reviewing and Extending the Algorithm

Deutsch&Jozsa's algorithm is a generalization of Deutsch's algorithm that achieves an exponential speed up [10]). This time we deal with the set of the binary functions $f_{\mathbf{k}} : \{0,1\}^n \to \{0,1\}$ such that the function is either constant (all zeroes or all ones), or balanced (even number of zeroes and ones). $\mathbf{k} \equiv k_0, k_1, \ldots, k_{2^n-1}$ is a 2^n bit string belonging to $\{0,1\}^{2^n}$. Table 3 shows this set of functions for $n = 2$—we shall focus on this example. Note that $k_0 = f_{\mathbf{k}}(00)$, $k_1 = f_{\mathbf{k}}(01)$, $k_2 = f_{\mathbf{k}}(10)$ and $k_3 = f_{\mathbf{k}}(11)$. The string $\mathbf{k}$ is both the subfix of f and, clockwise rotated, the table of the function chosen by the oracle (the sequence of function values for increasing arguments).

The problem is as follows. An oracle chooses at random one of these functions and gives to the second player the black box hardwired for the computation of that function. The second player, by trying function evaluation for different values of x, must find whether the function is balanced or constant. In the worst case, this requires a number of function evaluations $\exp(n)$ in the classical case, just one in the quantum case.

Table 3 Set of functions known to both players in Deutsch&Jozsa's algorithm

x	$f_{0000}(x)$	$f_{1111}(x)$	$f_{0011}(x)$	$f_{1100}(x)$	$f_{0101}(x)$	$f_{1010}(x)$	$f_{0110}(x)$	$f_{1001}(x)$
00	0	1	0	1	0	1	0	1
01	0	1	0	1	1	0	1	0
10	0	1	1	0	0	1	1	0
11	0	1	1	0	1	0	0	1

We give directly the extended quantum algorithm—extended to the physical representation of the random choice of the function on the part of the oracle. This time register K is 2^n qubits (we should keep in mind that this register is just a conceptual reference), register X is n qubits and register V is 1 qubit. As before, register K contains the valuation of $\mathbf{k}$, register X contains the value of x chosen by the second player (to query the black box with), and register V, initially containing v, will contain the result of function evaluation. Function evaluation leaves the content of registers K and X unaltered and puts $v \oplus f_{\mathbf{k}}(x)$ in V. The algorithm consists of three steps: (0) preparing register K in an even weighted superposition of all the possible values of $\mathbf{k}$, register X in an even weighted superposition of all the possible values of x, and register V in the antisymmetric state, (1) performing function evaluation, and (2) applying the Hadamard transform to register X.

The initial state is thus:

$$\Psi_0 = \frac{1}{8}(|0000\rangle_K + |1111\rangle_K + |0011\rangle_K + |1100\rangle_K + \cdots) \times (|00\rangle_X + |01\rangle_X + |10\rangle_X + |11\rangle_X)(|0\rangle_V - |1\rangle_V). \tag{4}$$

The black box, given the inputs $\mathbf{k}$ and x, computes $f(\mathbf{k}, x) \equiv f_{\mathbf{k}}(x)$ producing the outcome:

$$\Psi_1 = \frac{1}{4}\big[(|0000\rangle_K - |1111\rangle_K)(|00\rangle_X + |01\rangle_X + |10\rangle_X + |11\rangle_X) + (|0011\rangle_K - |1100\rangle_K)(|00\rangle_X + |01\rangle_X - |10\rangle_X - |11\rangle_X) + \cdots\big](|0\rangle_V - |1\rangle_V). \tag{5}$$

Applying the Hadamard transform to register X yields

$$\Psi_2 = \frac{1}{4}\big[(|0000\rangle_K - |1111\rangle_K)|00\rangle_X + (|0011\rangle_K - |1100\rangle_K)|10\rangle_X + \cdots\big](|0\rangle_V - |1\rangle_V). \tag{6}$$

Measuring $[K]$ and $[X]$ in (6) determines the moves of both players, namely the random choice of the function (a valuation of the string $\mathbf{k}$) on the part of the oracle and the solution provided by the second player (the content of X): all zeroes if the function is constant, not so if the function is balanced. Backdating to before running the algorithm the state reduction induced by measuring $[K]$ gives the original Deutsch&Jozsa's algorithm—it generates at random the value of $\mathbf{k}$ hosted in the black box.

3.2 Checking the 50% Rule

The information acquired by measuring $[K]$ and $[X]$ in (6) is the 2^n bits of the string $\mathbf{k}$ (the 2^n rows of the table of the function), the content of X is a function of the content of K, therefore the information contained in X is redundant. The quantum algorithm requires the number of function evaluations of a classical algorithm working on a solution space reduced

in size because 2^{n-1} bits of $\mathbf{k}$, 50% of the rows of the table, are known in advance. We should exclude the half tables containing different values of the function, which already identify the solution. Since all the bits are the same, the solution is always identified by computing an extra row, namely by performing just one function evaluation, for any value of x outside the advanced information (if the function has the same value as in the former rows it is constant, otherwise it is balanced). This demonstrates the 50% rule for Deutsch&Jozsa algorithm.

The 50% rule shows that Deutsch&Jozsa's problem is liable of an exponential speed up independently of our knowledge of the quantum algorithm—the speed up comes from comparing two classical algorithms, with and without the advanced information (we recall that this latter requires, in the worst case, an exponential number of function evaluations).

3.3 Building the Quantum Algorithm out of the Advanced Information Classical Algorithm

The function evaluation stage of the quantum algorithm—namely the transformation of Ψ_0 (4) into Ψ_1 (5)—is the sum of the histories of the advanced information classical algorithm[2] (see Sect. 2.3).

The choice of the initial phase of each history is justified as in Deutsch's algorithm.

We examine the outcome of function evaluation, namely Ψ_1 (5). There is a maximal entanglement between registers K and X. Orthogonal states of K, discriminating between constant and balanced functions (also between different kinds of balanced functions, but this is not relevant here), are correlated with orthogonal states of X. This means that the information whether the function is constant or balanced has propagated to register X. To read this information, we should rotate the X basis in such a way that $(|0000\rangle_K - |1111\rangle_K)(|00\rangle_X + |01\rangle_X + |10\rangle_X + |11\rangle_X$ goes into $(|0000\rangle_K - |1111\rangle_K)|00\rangle_X$, etc. This is a constructive definition of the Hadamard transform on register X. This completes the derivation of the quantum algorithm from the classical algorithm with the advanced information.

4 Simon's Algorithm

4.1 Reviewing and Extending the Algorithm

This time we deal with the set of the "periodic" functions $f_{\mathbf{k}} : \{0,1\}^n \to \{0,1\}^{n-1}$. The "periodic" function $f_{\mathbf{k}}(x)$ is such that $f_{\mathbf{k}}(x) = f_{\mathbf{k}}(y)$ if and only if $x = y$ or $x = y \oplus \mathbf{h^{(k)}}$, where: $\mathbf{k} \equiv k_0, k_1, \ldots, k_{2^n-1}$ is a 2^n bit string belonging to $\{0,1\}^{2^n}$; $\mathbf{h^{(k)}} \equiv h_0^{(\mathbf{k})}, h_1^{(\mathbf{k})}, \ldots, h_{n-1}^{(\mathbf{k})}$ is an n bit string (depending on the value of $\mathbf{k}$) belonging to $\{0,1\}^n$ with the exclusion of the all zeroes string; x and y are variables belonging to $\{0,1\}^n$ also represented as n bit strings; $\oplus$ denotes bit by bit module 2 addition. Thus, the string $\mathbf{h^{(k)}}$, also called the *hidden string*, is a sort of period of the function $f_{\mathbf{k}}(x)$. Since the bit by bit module 2 addition of the period with itself is zero, each value of the function appears exactly twice in the table. Thus, 50% of the rows plus one surely contain a same value of the function twice, which identifies the period.

[2]This is clear by looking at the form of Ψ_0 with the shortcut of Sect. 2.3 in mind. Without shortcut, we obtain the same result but with an extra rule: in the case that the advanced information contains both zeroes and ones (which implies that the function is balanced), no function evaluation is needed and there is no classical history in such a case.

Table 4 Set of functions known to both players in Simon's algorithm

x	$\mathbf{h}^{(0011)}=01$ $f_{0011}(x)$	$\mathbf{h}^{(1100)}=01$ $f_{1100}(x)$	$\mathbf{h}^{(0101)}=10$ $f_{0101}(x)$	$\mathbf{h}^{(1010)}=10$ $f_{1010}(x)$	$\mathbf{h}^{(0110)}=11$ $f_{0110}(x)$	$\mathbf{h}^{(1001)}=11$ $f_{1001}(x)$
00	0	1	0	1	0	1
01	0	1	1	0	1	0
10	1	0	0	1	1	0
11	1	0	1	0	0	1

By way of exemplification, Table 4 gives the set of the periodic functions for $n=2$. Note that $k_0=f_{\mathbf{k}}(00)$, $k_1=f_{\mathbf{k}}(01)$, $k_2=f_{\mathbf{k}}(10)$, and $k_3=f_{\mathbf{k}}(11)$. In fact the string $\mathbf{k}$ is both the subfix of f and, clockwise rotated, the table of function values for increasing values of the argument. When $n>2$, this table is a structured string—the sequence of 2^n fields of $n-1$ bits—each field yields a value of the function.

Since by definition each value of the function appears twice in the table of the function, 50% of the rows plus one surely contain a same value of the function twice, which identifies the period.

In the original algorithm, a first player (the oracle) chooses at random a function $f_{\mathbf{k}}(x)$, then he gives to the second player the black box hardwired for the computation of that function. The second player should find the string $\mathbf{h}^{(\mathbf{k})}$ hidden in $f_{\mathbf{k}}(x)$ (should find the "period" of the function) by performing function evaluation for different values of x. To find the value of $\mathbf{h}^{(\mathbf{k})}$ with probability, say, $\frac{2}{3}$, $f_{\mathbf{k}}(x)$ must be computed the order of $2^{\frac{n}{3}}$ times in the classical case, the order of $3n$ times in the quantum case (e.g. Kaye et al. [15]). There is an exponential speed up.

The original algorithm [17] consists of three steps: (0) preparing register X in an even weighted superposition of all the possible values of x and register V in $|0\rangle_V$, (1) performing function evaluation, and (2) applying the Hadamard transform to register X.

In the extended algorithm there are three registers. The $2^n(n-1)$ qubit register K contains the value of $\mathbf{k}$ chosen by the oracle (the table of the function). The n qubit register X contains the value of x chosen by the second player to query the black box with. The $n-1$ qubit register V, initially containing v (an $n-1$ bit string), will contain the result of function evaluation. Function evaluation leaves registers K and X unaltered and puts $v\oplus f_{\mathbf{k}}(x)$ in V, $\oplus$ denotes bit by bit module 2 addition. The initial state of the extended algorithm is:

$$\Psi_0=\frac{1}{4\sqrt{3}}(|0011\rangle_K+|1100\rangle_K+|0101\rangle_K+|1010\rangle_K+\cdots)\times(|00\rangle_X+|01\rangle_X+|10\rangle_X+|11\rangle_X)|0\rangle_V. \tag{7}$$

The black box, given the inputs $\mathbf{k}$ and x, computes $f(\mathbf{k},x)\equiv f_{\mathbf{k}}(x)$, producing the outcome:

$$\Psi_1=\frac{1}{4\sqrt{3}}\big[(|0011\rangle_K+|1100\rangle_K)[(|00\rangle_X+|01\rangle_X)|0\rangle_V+(|10\rangle_X+|11\rangle_X)|1\rangle_V]+(|0101\rangle_K+|1010\rangle_K)[(|00\rangle_X+|10\rangle_X)|0\rangle_V+(|01\rangle_X+|11\rangle_X)|1\rangle_V]+\cdots\big]. \tag{8}$$

Applying the Hadamard transform to register X yields

$$\Psi_2=\frac{1}{4}\big[(|0011\rangle_K+|1100\rangle_K)[(|00\rangle_X+|10\rangle_X)|0\rangle_V+(|00\rangle_X-|10\rangle_X)|1\rangle_V]+(|0101\rangle_K+|1010\rangle_K)[(|00\rangle_X+|01\rangle_X)|0\rangle_V+(|00\rangle_X-|01\rangle_X)|1\rangle_V]+\cdots\big]. \tag{9}$$

Backdating to before running the algorithm the state reduction induced by measuring $[K]$ gives the original Simon's algorithm—it generates at random the value of $\mathbf{k}$ hosted in the black box.

As one can see (9), for each pair of complementary values of the oracle's choice (e.g. for register K in $|0011\rangle_K + |1100\rangle_K$) and for each value of $f_{\mathbf{k}}(x)$ (e.g. for register V in $|0\rangle_V$), register X hosts an even weighted superposition (e.g. $|00\rangle_X + |10\rangle_X$) of the 2^{n-1} strings $\mathbf{s}_j^{(\mathbf{k})}$ ($j = 1, \ldots, 2^{n-1}$) "orthogonal" to the hidden string $\mathbf{h}^{(\mathbf{k})}$ (if we multiply bit by bit two orthogonal strings and take the module 2 addition of the product bits, the result is zero)—00 and 10 are the two strings orthogonal to the hidden strings $\mathbf{h}^{(0011)} = \mathbf{h}^{(1100)} \equiv 01$. Note that, in (9), only the phase of the terms of this superposition depend on the value of $f_{\mathbf{k}}(x)$. Therefore, by measuring $[K]$ and $[X]$ in (9), we obtain at random the oracle's choice $\mathbf{k}$ and one of the $\mathbf{s}_j^{(\mathbf{k})}$ orthogonal to $\mathbf{h}^{(\mathbf{k})}$.

At this point, we leave register K in its after-measurement state, so that the value of $\mathbf{k}$ remains fixed, and iterate the right part of the algorithm (initial preparation of registers X and V, function evaluation, Hadamard transform on X, and measurement of $[X]$) until obtaining $n-1$ different $\mathbf{s}_j^{(\mathbf{k})}$. This allows to find $\mathbf{h}^{(\mathbf{k})}$ by solving a system of $n-1$ module 2 linear equations. If the algorithm is iterated, say, $3n$ times, the probability of obtaining $n-1$ different $\mathbf{h}_j^{(\mathbf{k})}$, thus of finding the solution, is $\frac{2}{3}$. The probability of not finding the solution goes down exponentially with the number of iterations.

4.2 Checking the 50% Rule

For simplicity, we reformulate Simon's problem as the problem of generating at random a string orthogonal to the hidden string. Any such string is thus a "solution". For what concerns the character of the speed up, the two formulations are equivalent, an exponential speed up in the former implies an exponential speed up in the latter and vice-versa.

The information acquired in the final measurement of $[K]$ and $[X]$ is the information contained in the string $\mathbf{k}$, namely the table of the function chosen by the oracle (a four bit string when $n = 2$). The information read in register X—$\mathbf{s}_j^{(\mathbf{k})}$—is redundant, since all $\mathbf{s}_j^{(\mathbf{k})}$ are function of $\mathbf{k}$ (which j sorts out in reading register X is not a function of $\mathbf{k}$, but this is a random event with no information in it). The quantum algorithm requires the number of function evaluations of a classical algorithm working on a solution space reduced in size because 50% of the rows of the table are known in advance. As before, we should discard the half tables that already identify the solution. The solution is always identified by computing an extra row, namely by performing just one function evaluation for any argument outside the advanced information. This demonstrates the 50% rule for Simon's algorithm.

The 50% rule says that Simon's problem is liable of an exponential speed up through the comparison of two classical algorithms, with and without the advanced information (we recall that this latter algorithm, to solve the problem with probability $\frac{2}{3}$ must perform $2^{\frac{n}{3}}$ function evaluations).

One can readily see that the same holds by similarity for the generalized Simon's algorithm, thus for the hidden subgroup algorithms [16], like finding orders, finding the period of a function (the quantum part of Shor's factorization algorithm), finding discrete logarithms, etc. (e.g., Kaye et al. [15]).

4.3 Building the Quantum Algorithm out of the Advanced Information Classical Algorithm

The function evaluation stage of the quantum algorithm—namely the transformation of Ψ_0 (7) into Ψ_1 (8)—is the sum of the histories of the advanced information classical algorithm (see Sect. 2.3).

We justify the choice of the initial state of V. We start with the generic initial state $\alpha|0\rangle_V + \beta|1\rangle_V$ (α is thus the initial phase of the histories beginning with $|0\rangle_V$, etc.). Under α, we obtain the transformation of Ψ_0 (7) into Ψ_1 (8). Under β, we obtain the same result with $|0\rangle_V$ and $|1\rangle_V$ interchanged. Since we are interested in the superposition hosted in register X, which is the same in either case, we can suppress either α or β.

We examine the outcome of function evaluation, namely Ψ_1 (8). This time the entanglement between registers K and X is not maximal. We know that one function evaluation moves to register X information about the oracle's choice, but we do not know which is this information. The Hadamard transform on register X can still be defined as the rotation of the X basis that maximizes the information about the oracle's choice readable in it. However, this is no more a constructive definition, we are left with the problem of discovering that this information is a string orthogonal to $\mathbf{h}^{(\mathbf{k})}$.

5 Grover's Algorithm

5.1 Reviewing and Extending the Algorithm

The problem addressed by Grover's algorithm [14] is database search. It can be seen as a game between two players with a chest of N drawers; the first player (the oracle) hides a ball in drawer number $\mathbf{k}$; $\mathbf{k} \equiv k_0, k_1, \ldots, k_{n-1}$ is an n bit string belonging to $\{0, 1\}^n$ with $n = \log_2 N$ (for simplicity, we can assume that N is a power of 2). The second player must find where the ball is. Opening drawer x to check whether the ball is in it amounts to computing the Kronecker function $\delta(\mathbf{k}, x)$, which is 1 if $\mathbf{k} = x$ and 0 otherwise.

The value of $\mathbf{k}$ chosen by the first player is hardwired inside a black box that, for each input x, computes $\delta(\mathbf{k}, x)$. The black box is given to the second player, who has to find the value of $\mathbf{k}$ by computing $\delta(\mathbf{k}, x)$ for different values of x (i.e., by opening different drawers to check whether the ball is in it). In the classical case, to find the value of $\mathbf{k}$, $\delta(\mathbf{k}, x)$ must be computed the order of N times, in the quantum case the order of $\sqrt{N}$ times—there is a quadratic speed up.

The original algorithm consists of three steps: (0) preparing register X in an even weighted superposition of all the possible values of x and register V in the antisymmetric state, (1) performing function evaluation, and (2) applying the transformation U to register X (see further below).

We review the extension of Grover's algorithm for $n = 2$ (further below we generalize to $n > 2$). As usual we have three registers: the n qubit oracle's choice register K containing the database location chosen by the oracle,[3] the n qubit query register X, and the 1 qubit output register V.

The initial state is:

$$\Psi_0 = \frac{1}{4\sqrt{2}}(|00\rangle_K + |01\rangle_K + |10\rangle_K + |11\rangle_K)(|00\rangle_X + |01\rangle_X + |10\rangle_X + |11\rangle_X)(|0\rangle_V - |1\rangle_V), \tag{10}$$

[3] We could have chosen a 2^n qubit register containing the table of the binary function $\delta(\mathbf{k}, x)$, but this register would contain exactly the same information as the n qubit register, namely the database location chosen by the oracle. The present definition of K simplifies the notation.

Function evaluation, namely computing $\delta(\mathbf{k}, x)$, yields:

$$\begin{aligned}\Psi_1 = \frac{1}{4\sqrt{2}}\big[&|00\rangle_K(-|00\rangle_X + |01\rangle_X + |10\rangle_X + |11\rangle_X)\\ &+ |01\rangle_K(|00\rangle_X - |01\rangle_X + |10\rangle_X + |11\rangle_X)\\ &+ |10\rangle_K(|00\rangle_X + |01\rangle_X - |10\rangle_X + |11\rangle_X)\\ &+ |11\rangle_K(|00\rangle_X + |01\rangle_X + |10\rangle_X - |11\rangle_X)\big](|0\rangle_V - |1\rangle_V).\end{aligned} \tag{11}$$

This is four orthogonal states of K correlated with four orthogonal states of X. Applying to register X the Hadamard transform, then the transformation obtained by computing $\delta(0, x)$, then another time the Hadamard transform (for short, applying the transformation U) yields:

$$\Psi_2 = \frac{1}{2\sqrt{2}}(|00\rangle_K|00\rangle_X + |01\rangle_K|01\rangle_X + |10\rangle_K|10\rangle_X + |11\rangle_K|11\rangle_X)(|0\rangle_V - |1\rangle_V), \tag{12}$$

namely an entangled state where each value of k is correlated with the corresponding solution found by the second player (the same value of k but in register X). The final measurement of $[K]$ and $[X]$ in state (12) determines the moves of both players. The reduction induced by measuring $[K]$, backdated to before running the algorithm, yields the original Grover's algorithm.

5.2 Checking the 50% Rule

The information acquired in the final measurement of $[K]$ and $[X]$ is 2 bits. These are the two bits of register K; in fact the content of X is a function of the content of K, therefore the information contained in X is redundant. The quantum algorithm requires the number of function evaluations of a classical algorithm working on a solution space reduced in size because one bit of information about the content of K, either k_0 or k_1, is known in advance. The classical algorithm should perform function evaluation (compute $\delta(\mathbf{k}, x)$) for the missing bit. This verifies the 50% rule for $n = 2$.

More in general, a classical algorithm that knows in advance 50% of the n bits that specify the database location, in order to identify the $n/2$ missing bits should perform $O(2^{n/2} = \sqrt{N})$ function evaluations (computations of $\delta(\mathbf{k}, x)$), against the $O(2^n = N)$ of a classical algorithm without advanced information. This means a quadratic speed up. This verifies the 50% rule for $n > 2$. This rule says that unstructured database search is liable of a quadratic speed up on the basis of purely classical considerations.

5.3 Building the Quantum Algorithm out of the Advanced Information Classical Algorithm

The function evaluation stage of the quantum algorithm—namely the transformation of Ψ_0 (10) into Ψ_1 (11)—is the sum of the histories of the advanced information classical algorithm (see Sect. 2.3).

The choice of the initial phase of each history is justified as in Deutsch's algorithm.

We examine the outcome of the function evaluation stage, namely the Ψ_1 of (11). Registers K and X are maximally entangled, orthogonal states of K, each corresponding to a value of $\mathbf{k}$, are correlated with orthogonal states of X. This means that the value of $\mathbf{k}$ has propagated to register X. To read this value, we should rotate the X basis in such a way

that $-|00\rangle_X + |01\rangle_X + |10\rangle_X + |11\rangle_X$ (correlated with $|00\rangle_K$) goes into $|00\rangle_X$, etc. This is a constructive definition of the transformation U.

Generalizing to $n > 2$ is straightforward. Given the advanced knowledge of $n/2$ bits, in order to compute the missing $n/2$ bits we should perform function evaluation and rotate the basis of X an $O(2^{\frac{n}{2}})$ times. The first time we obtain a superposition of an unentangled state of the form (10) (the initial state transformed into itself with a slightly smaller amplitude) and a maximally entangled state of the form (12). At each successive iteration, the amplitude of the latter state is amplified at the expense of the amplitude of the former, until it becomes (about) 1 in $O(2^{\frac{n}{2}})$ iterations.

6 Conclusions

We have verified that the 50% rule, the fact that a quantum algorithm requires the number of function evaluations of a classical algorithm that knows in advance 50% of the information that specifies the solution of the problem, holds for the main quantum algorithms.

This rule, besides shading light on the nature of quantum computation, brings the search of the problems solvable with a quantum speed up to an entirely classical framework. The quantum speed up comes out by comparing two classical algorithms, with and without the advanced information. This rule can therefore be used for a systematic exploration of the possibility of achieving speed ups, perhaps to explain why the speed ups discovered until now are so few. Once identified a problem liable of speed up, the same rule can be used a second time to search the quantum algorithm that solves the problem. In fact the advanced information classical algorithm defines the quantum algorithm.

The 50% rule establishes a correspondence between quantum computation and classical computation with advanced information. It is natural to ask ourselves whether, in some more general sense, quantum mechanics is classical mechanics with advanced information. This question would deserve further investigation.

Acknowledgements Thanks are due, for encouragement and useful comments, to Vint Cerf, Artur Ekert, David Finkelstein, Lov Grover, Günter Mahler, and Hartmut Neven.

References

1. Benioff, P.: Quantum mechanical Hamiltonian models of Turing machines. J. Stat. Phys. **29**, 515 (1982)
2. Bennett, C.H.: Logical reversibility of computation. IBM J. Res. Develop. **6**, 525 (1973)
3. Bennett, C.H.: The thermodynamics of computation—a review. Int. J. Theor. Phys. **21**, 905 (1982)
4. Castagnoli, G., Finkelstein, D.: Theory of the quantum speed up. Proc. R. Soc. Lond. A **457**, 1799 (2001). quant-ph/0010081
5. Castagnoli, G.: The mechanism of quantum computation. Int. J. Theor. Phys. **47**(8), 2181–2194 (2008)
6. Castagnoli, G.: The quantum speed up as advanced cognition of the solution. Int. J. Theor. Phys. **48**(3), 857–873 (2009)
7. Castagnoli, G.: The quantum speed up as advanced knowledge of the solution. arXiv:0809.4545
8. Cleve, R., Ekert, A., Macchiavello, C., Mosca, M.: Quantum algorithms revisited. Proc. R. Soc. Lond. A **454**(1969), 339–354 (1998). quant-ph/9708016
9. Deutsch, D.: Quantum theory, the Church-Turing principle and the universal quantum computer. Proc. R. Soc. Lond. A **400**, 97 (1985)
10. Deutsch, D., Jozsa, R.: Rapid solution of problems by quantum computation. Proc. R. Soc. Lond. A **439**, 553 (1992)
11. Finkelstein, D.R.: Space–time structure in high energy interactions. In: Gudehus, T., Kaiser, G., Perlmutter, A. (eds.) Coral Gables Conference on Fundamental Interactions at High Energy. Center of Theoretical Studies January 22–24, 1969. University of Miami, pp. 324–343. Gordon and Breach, New York (1969)

12. Finkelstein, D.R.: Space–time code. Phys. Rev. **184**, 1261 (1969)
13. Fredkin, E., Toffoli, T.: Conservative logic. Int. J. Theor. Phys. **21**, 219 (1982)
14. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proc. 28th Ann. ACM Symp. Theory of Computing (1996)
15. Kaye, P., Laflamme, R., Mosca, M.: An Introduction to Quantum Computing. Oxford University Press, New York (2007)
16. Mosca, M., Ekert, A.: The Hidden Subgroup Problem and Eigenvalue Estimation on a Quantum Computer. Lecture Notes in Computer Science, vol. 1509. Springer, Berlin (1999)
17. Simon, D.: On the Power of Quantum Computation. In: Proc. 35th Ann. Symp. on Foundations of Comp. Sci., pp. 116–123 (1994)