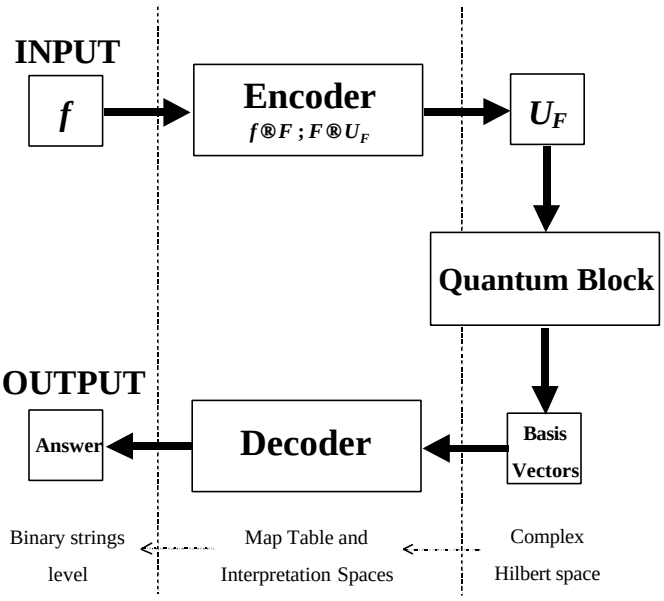
SERGUEI V. ULYANOV – FABIO GHISI

SERGUEI A. PANFILOV – VIKTOR S. ULYANOV

ICHIRO KURAWAKI – LUDMILA LITVINTSEVA

SIMULATIONS OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS



UNDER THE SCIENTIFIC SUPERVISION OF:

RYUICHI YAMASHITA

GIOVANNI DEGLI ANTONI

KAZUO YAMAFUJI

Yamaha Motor Europe N. V. R&D Office, Japan
Università degli Studi di Milano – Polo Didattico e di Ricerca di Crema
University of Electro-Communications, Japan

Via Bramante, 65 – 26013 CREMA (CR), Italy – 1999

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Contents

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Introduction

We will describe the situation from computer science viewpoint: how a Quantum Algorithm, written as a Quantum Circuit, can be automatically translated into the corresponding Programmable Quantum Gate. This gate is represented as a matrix operator such that, when it is applied to the vector representation of the quantum register input state, the produced result is the vector representation of the required register output state (see Fig.1).

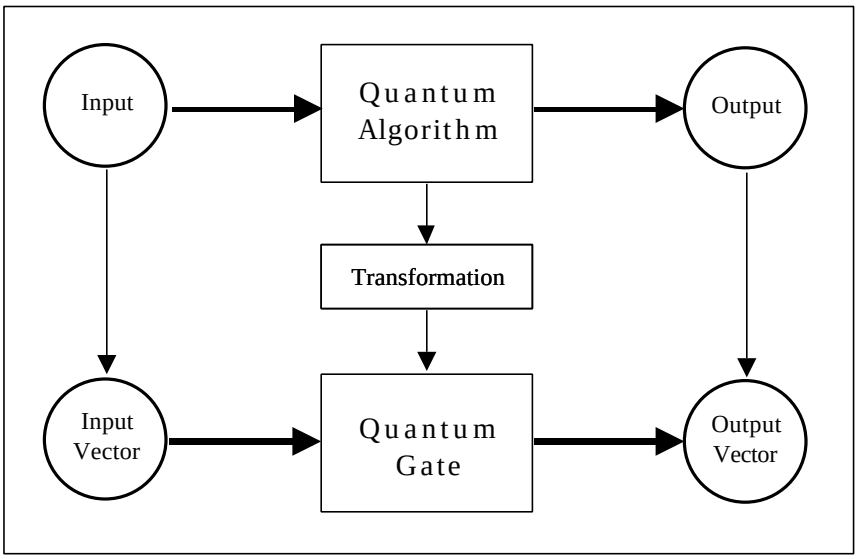


Figure 1: The Gate Approach for Simulation of Quantum Algorithms using Classical Computers

Bases of quantum computation are three operators on quantum coherent states: superposition, entanglement and interference.
The coherent states are described as those solutions of the corresponding Schrödinger equation that represent the evolution states with minimum of uncertainty (in Heisenberg sentence they are those quantum states with "maximum classical properties"). The Hadamard Transform creates the superposition on classical states, and quantum operators as CNOT create robust entangled states. Quantum Fast Fourier Transform carries on interference. The efficient implementations of a number of operations for quantum computation include controlled phase adjustment of the amplitudes in superposition, permutation, approximation of transformations and generalizations of the phase adjustments to block matrix transformations. These operations generalize those used in quantum search algorithms that realized on classical computer. We demonstrate the application of this approach to the simulation on classical computers of the Benchmarks as Deutsch's, Deutsch–Jozsa's, Simon's, Shor's and Grover`s algorithms.

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 1: General Outline of Quantum Algorithms

1. AIM

We discuss in this introductory part the general outline of the quantum algorithms we are going to deal with.

2. GENERAL STRUCTURE OF QUANTUM ALGORITHMS

The problems solved by the quantum algorithms we will describe can all be so stated:

Input	A function $f:\{0,1\}^n \rightarrow \{0,1\}^m$
Problem	Find a certain property of f

The structure of a quantum algorithm is outlined, with a high level representation, in the scheme diagram of fig.1.

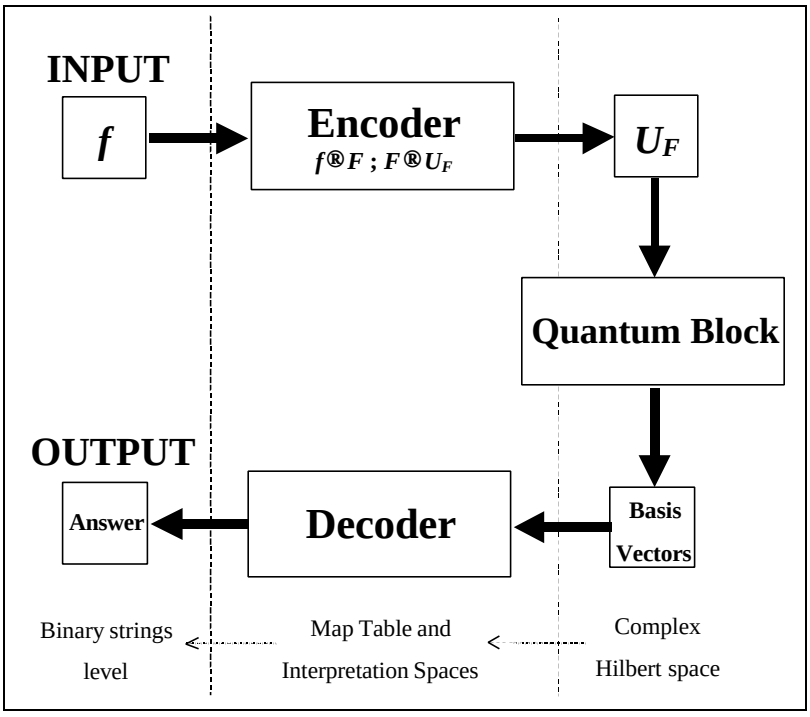


Figure 1: Scheme Diagram of Quantum Algorithms

The input of a quantum algorithm is always a function f from binary strings into binary strings. This function is represented as a map table, defining for every string its image.

Function *f* is firstly encoded into a unitary matrix operator U_F depending on *f* properties. In some sense, this operator calculates *f* when its input and output strings are encoded into canonical basis vectors of a Complex Hilbert Space: U_F maps the vector code of every string into the vector code of its image by *f*.

BOX 1: UNITARY MATRIX U_F

A squared matrix U_F on the complex field is *unitary* iff its inverse matrix coincides with its conjugate transpose:

$$U_F^{-1} = U_F^{\dagger}$$

A unitary matrix is always reversible and preserves the norm of vectors.

When the matrix operator U_F has been generated, it is embedded into a quantum gate *G*, a unitary matrix whose structure depends on the form of matrix U_F and on the problem we want to solve. The quantum gate is the heart of a quantum algorithm. In every quantum algorithm, the quantum gate acts on an initial canonical basis vector (we can always choose the same vector) in order to generate a complex linear combination (let's call it superposition) of basis vectors as output. This superposition contains all the information to answer the initial problem.
After this superposition has been created, measurement takes place in order to extract this information. In quantum mechanics, measurement is a non-deterministic operation that produces as output only one of the basis vectors in the entering superposition. The probability of every basis vector of being the output of measurement depends on its complex coefficient (probability amplitude) in the entering complex linear combination.
The segmental action of the quantum gate and of measurement constitutes the quantum block. The quantum block is repeated *k* times in order to produce a collection of *k* basis vectors. Being measurement a non-deterministic operation, these basic vectors won't be necessarily identical and each one of them will encode a peace of the information needed to solve the problem.
The last part of the algorithm consists into the interpretation of the collected basis vectors in order to get the right answer for the initial problem with a certain probability.

3. ENCODER

The behaviour of the encoder block is described in the detailed scheme diagram of fig.2.
Function *f* is encoded into matrix U_F in three steps.

Step 1

The map table of function $f:\{0,1\}^n \rightarrow \{0,1\}^m$ is transformed into the map table of the injective function $F:\{0,1\}^{n+m} \rightarrow \{0,1\}^{n+m}$ such that:

$$F(x_0, .., x_{n-1}, y_0, .., y_{m-1}) = (x_0, .., x_{n-1}, f(x_0, .., x_{n-1}) \oplus (y_0, .., y_{m-1}))$$

The need to deal with an injective function comes from the requirement that U_F is unitary. A unitary operator is reversible, so it can't map 2 different inputs in the same output. Since U_F

will be the matrix representation of F, F is supposed to be injective. If we directly employed the matrix representation of function f, we could obtain a non-unitary matrix, since f could be non-injective. So, injectivity is fulfilled by increasing the number of bits and considering function F instead of function f. Anyway, function f can always be calculated from F by putting $(y_0, .., y_{m-1})=(0,..,0)$ in the input string and reading the last m values of the output string.

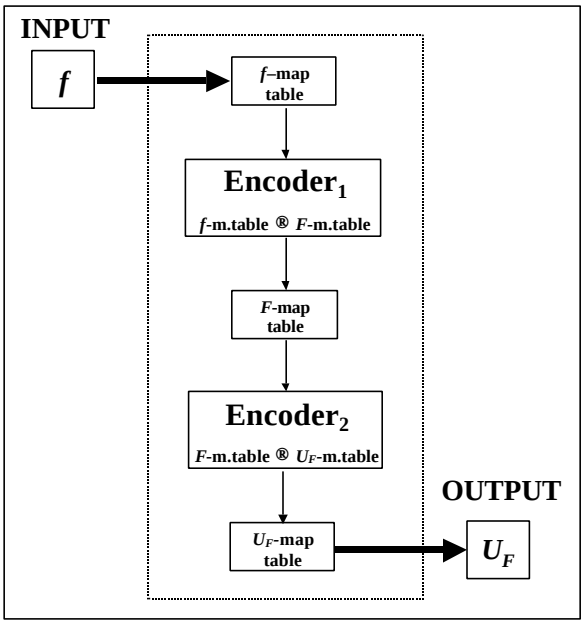


Figure 2: The Encoder Block Scheme Diagram

BOX 2: XOR OPERATOR Å

The XOR operator between two binary strings p and q of length m is a string s of length m such that the i-th digit of s is calculated as the exclusive OR between the i-th digits of p and q:

$$p=(p_0, .., p_{n-1})$$
$$q=(q_0, .., q_{n-1})$$
$$s = p \oplus q = ((p_0+q_0) \bmod 2, .., (p_{n-1}+q_{n-1}) \bmod 2))$$

Step 2

Function F map table is transformed into U_F map table, following the following constraint:

$$\forall s \in \{0,1\}^{n+m}: U_F[t(s)]= t\,[F(s)]$$

The code map $t : \{0,1\}^{n+m} \rightarrow \mathbf{C}^{2^{n+m}}$ ($\mathbf{C}^{2^{n+m}}$ is the target Complex Hilbert Space) is such that:

$$t(0)=\begin{pmatrix}1\\0\end{pmatrix}=|0\rangle \quad t(1)=\begin{pmatrix}0\\1\end{pmatrix}=|1\rangle$$
$$t(x_0,..,x_{n+m-1})=t(x_0)\otimes..\otimes t(x_{n+m-1})=|x_0..x_{n+m-1}\rangle$$

Code *t* maps bit values into complex vectors of dimension 2 belonging to the canonical basis of $\mathbf{C}^2$. Besides, using tensor product, *t* maps the general state of a binary string of dimension *n* into a vector of dimension 2^n, reducing this state to the joint state of the *n* bits composing the register. Every bit state is transformed into the corresponding 2-dimesional basis vector and then the string state is mapped into the corresponding 2^n-dimesional basis vector by composing all bit-vectors through tensor product. In this sense tensor product is the vector counterpart of state conjunction.

BOX 3: VECTOR TENSOR PRODUCT Ä

The tensor product between two vectors of dimensions *h* and *k* is a tensor product of dimension *h·k*, such that:

$$\begin{pmatrix}x_1\\...\\x_h\end{pmatrix}\otimes\begin{pmatrix}y_1\\...\\y_k\end{pmatrix}=\begin{pmatrix}x_1y_1\\...\\x_1y_k\\...\\x_hy_1\\...\\x_hy_k\end{pmatrix}$$

If a component of a complex vector is interpreted as the probability amplitude of a system of being in a given state (indexed by the component number), the tensor product between two vectors describes the joint probability amplitude of two systems of being in a joint state.

Examples: Vector Tensor Products

$$(0,0)\xrightarrow{t}\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}=\begin{pmatrix}1\\0\\0\\0\end{pmatrix}=|00\rangle \qquad (0,1)\xrightarrow{t}\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}=\begin{pmatrix}0\\1\\0\\0\end{pmatrix}=|01\rangle$$

$$(1,0)\xrightarrow{t}\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}=\begin{pmatrix}0\\0\\1\\0\end{pmatrix}=|10\rangle \qquad (1,1)\xrightarrow{t}\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}=\begin{pmatrix}0\\0\\0\\1\end{pmatrix}=|11\rangle$$

Basis vectors are denoted using the *ket* notation |*i*>. This notation is taken from Dirac description of quantum mechanics.

Step 3

U_F map table is transformed into U_F using the following transformation rule:

$$[U_F]_{ij} = 1 \Leftrightarrow U_F|j\rangle = |i\rangle$$

This rule can easily be understood considering vectors $|i\rangle$ and $|j\rangle$ as column vectors. Belonging these vectors to the canonical basis, U_F defines a permutation map of the identity matrix rows. In general, row $|j\rangle$ is mapped into row $|i\rangle$.
This rule will be illustrated in detail in part 2, where we face the first example of quantum algorithm: Deutsch's algorithm.

4. QUANTUM BLOCK

The heart of the quantum block is the quantum gate, which depends on the properties of matrix U_F. The scheme in fig.3 gives a more detailed description of the quantum block.

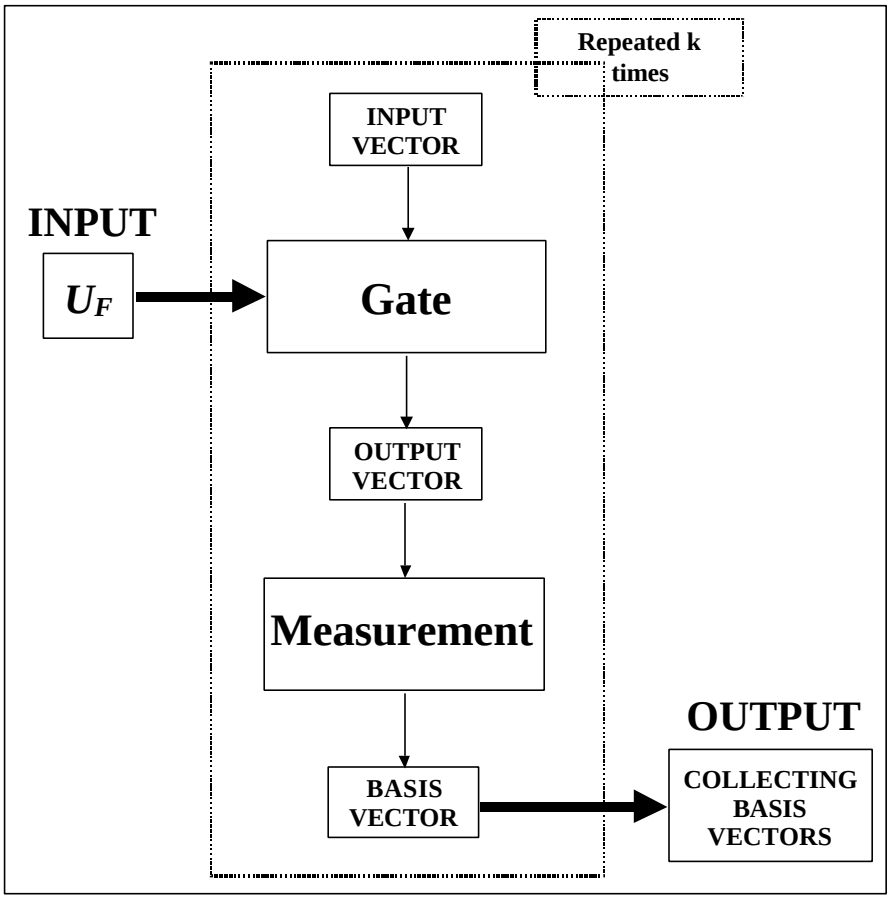


Figure 3: Structure of Quantum Block in Fig.1

Matrix operator U_F in fig.3 is the output of the encoder block represented in fig.2. Here, it becomes the input for the quantum block.
This matrix operator is firstly embedded into a more complex gate: the quantum gate G. Unitary matrix G is applied k times to an initial canonical basis vector $|i\rangle$ of dimension 2^{n+m}. Every time, the resulting complex superposition $G|0..01..1\rangle$ of basis vectors is measured,

producing one basis vector $|x_i\rangle$ as result. All the measured basis vectors $\{|x_1\rangle,..,|x_k\rangle\}$ are collected together. This collection is the output of the quantum block.
The "intelligence" of our algorithms is in the ability to build a quantum gate that is able to extract the information necessary to find the required property of *f* and to store it into the output vector collection.
We will discuss in detail the structure of the quantum gate for every quantum algorithm, observing that it can be described in a general way.

In order to represent quantum gates we are going to employ some special diagrams called quantum circuits.
An example of quantum circuit is reported in fig.4:

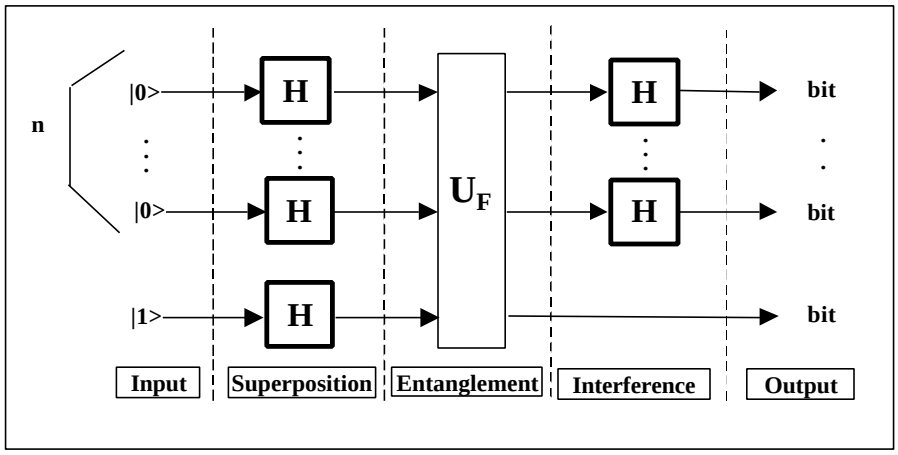


Figure 4: Example of Quantum Circuit

Every rectangle is associated to a matrix $2^n \times 2^n$, where *n* is the number of lines entering and leaving the rectangle. For example, the rectangle marked U_F is associated to matrix U_F.
Quantum circuits let us give a high-level description of the gate and, using some transformation rules, we can easily compile them into the corresponding gate-matrix. These rules are listed in fig.5:

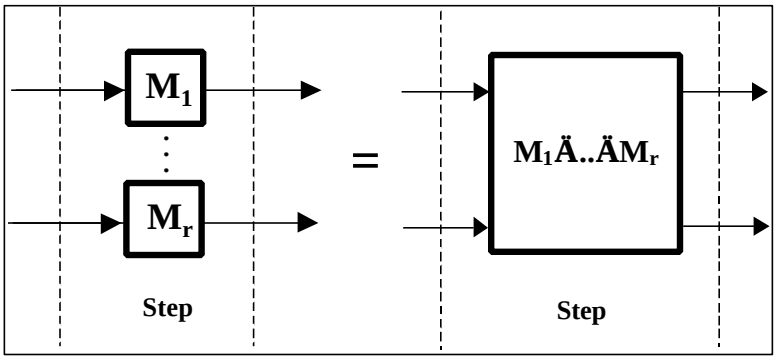


Figure 5.a: Rule 1 – Tensor Product Transformation

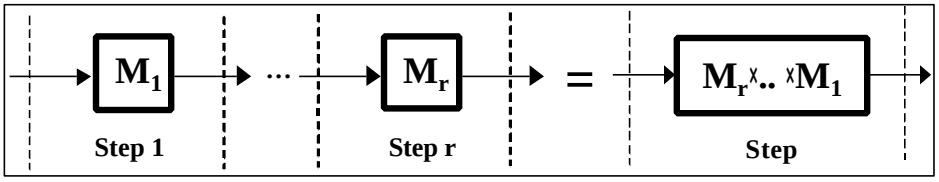


Figure 5.b: Rule 2 – Dot Product Transformation

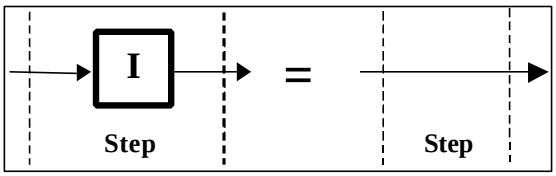


Figure 5.c: Rule 3 – Identity Transformation

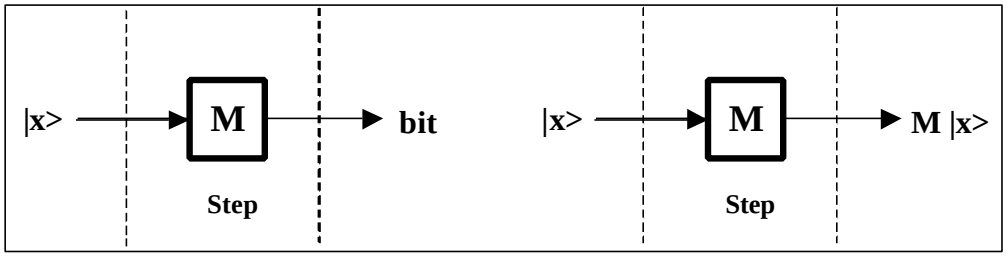


Figure 5.d: Rule 4 – Propagation Rule

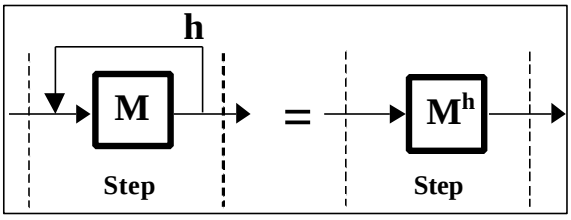


Figure 5.e: Rule 5 – Iteration Rule

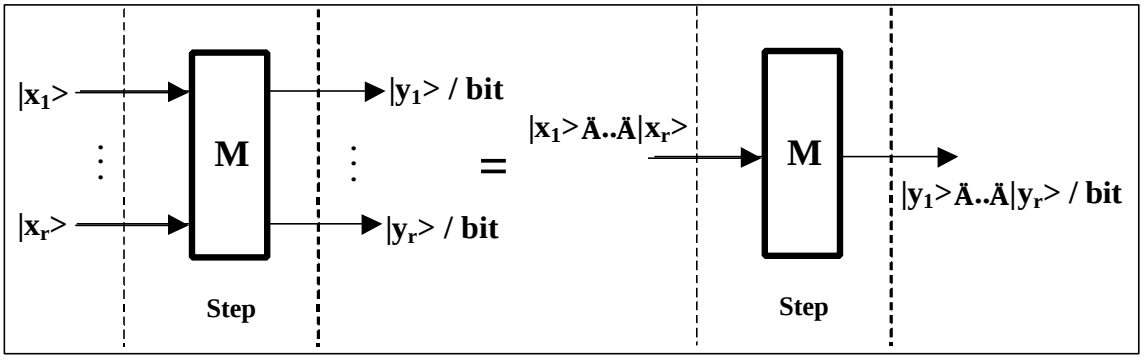


Figure 5.f: Rule 6 – Input/Output Tensor Rule

It will be clearer how to use these rules when we afford the first examples of quantum algorithm.

BOX 4: MATRIX TENSOR PRODUCT Ä

The tensor product between two matrices $X_{n\times m}$ and $Y_{h\times k}$ is a (block) matrix $(n\cdot h)\times(m\cdot k)$ such that:

$$X \otimes Y = \begin{bmatrix} x_{11}Y & .. & x_{1m}Y \\ .. & .. & .. \\ x_{n1}Y & .. & x_{nm}Y \end{bmatrix} \qquad with \quad X = \begin{bmatrix} x_{11} & .. & x_{1m} \\ .. & .. & .. \\ x_{n1} & .. & x_{nm} \end{bmatrix}$$

Example: Matrix Tensor Product

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \otimes \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1\cdot\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} & 2\cdot\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \\ 3\cdot\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} & 4\cdot\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 5 & 6 & 10 & 12 \\ 7 & 8 & 14 & 16 \\ 15 & 18 & 20 & 24 \\ 21 & 24 & 28 & 32 \end{bmatrix}$$

5. DECODER

The decoder block has the function to interpret the basis vectors collected after the iterated execution of the quantum block. Decoding these vectors means to retranslate them into binary strings and interpreting them directly if they already contain the answer to the starting problem or use them, for instance as coefficients vectors for some equation system, in order to get the searched solution. We shall not investigate this part in detail since it is a non-interesting easy classical part.

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 2: Deutsch's Algorithm

1. AIM

In order to illustrate the general method to synthesise a quantum algorithm and the quantum gate implementing it, we deal with a very simple pedagogical example: Deutsch's algorithm. We shall point out the role of superposition, entanglement and parallel quantum massive calculation.

2. DEUTSCH'S PROBLEM

A function $f:\{0,1\}\rightarrow\{0,1\}$ is said constant if and only if $\exists y\in\{0,1\}:\forall x\in\{0,1\}: f(x)=y$. It is said balanced if and only if $|\{x\in\{0,1\}: f(x)=0\}| = |\{x\in\{0,1\}: f(x)=1\}|$.

Deutsch's problem is so stated:

Input	A balanced or constant function f
Problem	Decide if f is constant or balanced

We distinguish 4 possible functions $f_i:\{0,1\}\rightarrow\{0,1\}$. They are defined by the following map tables:

Constant Functions

x	$f_1(x)$
0	0
1	0

x	$f_2(x)$
0	1
1	1

Balanced Functions

x	$f_3(x)$
0	0
1	1

x	$f_4(x)$
0	1
1	0

The set $\{f_i\}_{i\in\{1,2,3,4\}}$ is the input set for our algorithm. Every function f_i is represented by its map table.

3. ENCODER

The encoder block encodes input function f into matrix U_F. Suppose the function we are going to investigate is $f=f_3$. Its map table is the following:

x	$f_3(x)$
0	0
1	1

Step 1

Function f is firstly transformed into function $F:\{0,1\}^2 \rightarrow \{0,1\}^2$ such that

$$F(x_0, y_0) = (x_0, f(x_0) \oplus y_0)$$

In logic representation this means:

y_0	$F(x_0, y_0)$
0	$(x_0, f(x_0))$
1	$(x_0, \neg f(x_0))$

BOX 1: NOT OPERATOR Ø

The NOT operator acting on a binary string flips the value of every digit in the string.

$$p=(p_0, .., p_{n-1})$$
$$\neg p=((p_0+1)\text{mod}2, .., (p_{n-1}+1)\text{mod}2)$$

Therefore, if $f=f_3$, F map table is the following:

(x_0, y_0)	$F(x_0, y_0)$
(0,0)	(0,0)
(0,1)	(0,1)
(1,0)	(1,1)
(1,1)	(1,0)

Step 2

In this step, the map table of F is transformed into the map table of U_F.
The transformation rule is the following:

$$\forall s \in \{0,1\}^2: U_F[t(s)] = t\,[F(s)]$$

So, U_F map table is:

$\|x_0\, y_0\rangle$	$U_F \|x_0\, y_0\rangle$
|00>	|00>
|01>	|01>
|10>	|11>
|11>	|10>

or, writing basis vectors as column vectors:

v	U_F **v**
$(1,0,0,0)^T$	$(1,0,0,0)^T$
$(0,1,0,0)^T$	$(0,1,0,0)^T$
$(0,0,1,0)^T$	$(0,0,0,1)^T$
$(0,0,0,1)^T$	$(0,0,1,0)^T$

BOX 2: TRANSPOSE OPERATOR (..)^T

The TRANSPOSE operator acting on a row or column vector transforms the vector into its corresponding column or, respectively, row vector:

$$(x_1 \quad \ldots \quad x_n)^T = \begin{pmatrix} x_1 \\ \ldots \\ x_n \end{pmatrix} \qquad \begin{pmatrix} x_1 \\ \ldots \\ x_n \end{pmatrix}^T = (x_1 \quad \ldots \quad x_n)$$

Step 3

The matrix associated to such a map table is obtained from the identity matrix 4×4 by a permutation of its rows: the first and the second rows are mapped into themselves, whereas the third row is mapped into the fourth one and the fourth row into the third one:

$$U_F = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Remark

A general way to build U_F is to express every vector U_F (|s>) as a linear combination of all the basis vectors.

The co-ordinates of this combination are all 0, unless for one basis vector corresponding to the image of |s> by U_F:

$$U_F|00\rangle = 1|00\rangle + 0|01\rangle + 0|10\rangle + 0|10\rangle$$
$$U_F|01\rangle = 0|00\rangle + 1|01\rangle + 0|10\rangle + 0|10\rangle$$
$$U_F|10\rangle = 0|00\rangle + 0|01\rangle + 0|10\rangle + 1|10\rangle$$
$$U_F|11\rangle = 0|00\rangle + 0|01\rangle + 1|10\rangle + 0|10\rangle$$

We calculate $[U_F]_{ij}$ as the co-ordinate of vector U_F (|*j*>) with respect to vector |*i*>, where *i* and *j* are binary sequences. This means:

$$[U_F]_{ij} = 1 \Leftrightarrow U_F|j\rangle = |i\rangle$$

Value $[U_F]_{ij}$ is called the probability amplitude of |*j*> of being mapped into |*i*> by U_F.
The probability amplitude of |00> of being mapped into |00> is, for instance, 1, since U_F|00>=1|00>, whereas its probability amplitude of being mapped into |01> is 0, since U_F|00>=0|01>.
Using this technique, the following unitary matrix is built:

U_F	|00>	|01>	|10>	|11>
|00>	1	0	0	0
|01>	0	1	0	0
|10>	0	0	0	1
|11>	0	0	1	0

5. QUANTUM BLOCK

The encoder block has generated matrix U_F. This matrix is now embedded into the quantum gate that will act on the input vector |00>. We describe this gate using a quantum circuit.

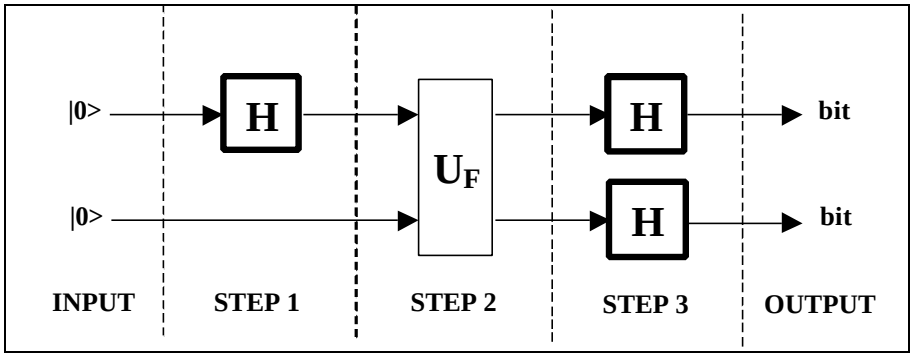


Figure 1: Deutsch's Quantum Gate – Circuit Representation

Every thin rectangle represents a classical matrix operator *n*×*n*, where *n* is the number of lines entering and leaving the rectangle. A matrix operator is said classical, when it maps every basis vector into another basis vector. For example, operator U_F is classical. A thick rectangle

stands for a non-classical matrix operator. A non-classical matrix operator maps at least one basis vector into a superposition of basis vectors.

Examples: Classical and Non-Classical Matrix operators

Classical Matrix Operator U_F

U_F	|00>	|01>	|10>	|11>
|00>	1	0	0	0
|01>	0	1	0	0
|10>	0	0	0	1
|11>	0	0	1	0

Non-Classical Matrix Operator H

H	|0>	|1>
|0>	$1/2^{1/2}$	$1/2^{1/2}$
|1>	$1/2^{1/2}$	$-1/2^{1/2}$

We want now to compile the circuit above into the corresponding computable gate. The first passage consists into completing the circuit making some operators explicit. Consider, for instance, step 1. The second line is in this step empty. This means that the second entering basis vector is left unchanged. We say that on this vector acts the identity matrix operator and we complete the circuit. This is rule 3 described in Part 1.

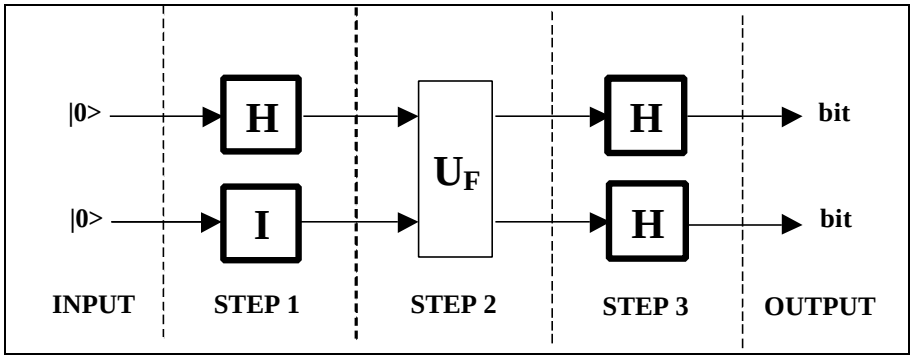


Figure 2: Deutsch's Quantum Gate – Second Circuit Representation

The identity matrix operator is classical and it is so defined:

I	|0>	|1>
|0>	1	0
|1>	0	1

At this point we should build a matrix operator corresponding to every step whose action corresponds to the concurrent action of the matrix operators acting on parallel lines. We use rules 1 and 6 obtaining the quantum circuit of fig.3.
Finally, we build a unique matrix operator that is equivalent to the sequential application of the operators in step 1, step 2 and step 3. This is operator composition and it is obtained with the dot product among matrices in the reverse order of application, as rule 2 states. Applying rule 2 to the circuit, we obtain the quantum circuit of fig.4, namely the programmable gate implementing Deutsch's algorithm.
Let's compute this gate. Firstly, we calculate ($H \otimes I$). The output matrix is 4×4. We label each column and row with the corresponding basis vector. We calculate the amplitude probability

for each basis vector of being mapped into another basis vector using *H* and *I*. Take for instance vector |00>: its probability amplitude of being transformed into |01> is the product between the probability amplitude of |0> of being mapped into |0> by *H* and the probability amplitude of |0> of being transformed into |1> by *I*. This is tensor product.

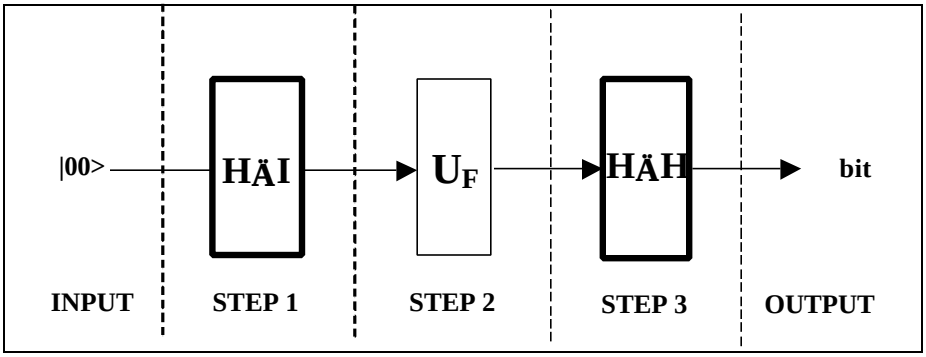


Figure 3: Deutsch's Quantum Gate – Third Circuit Representation

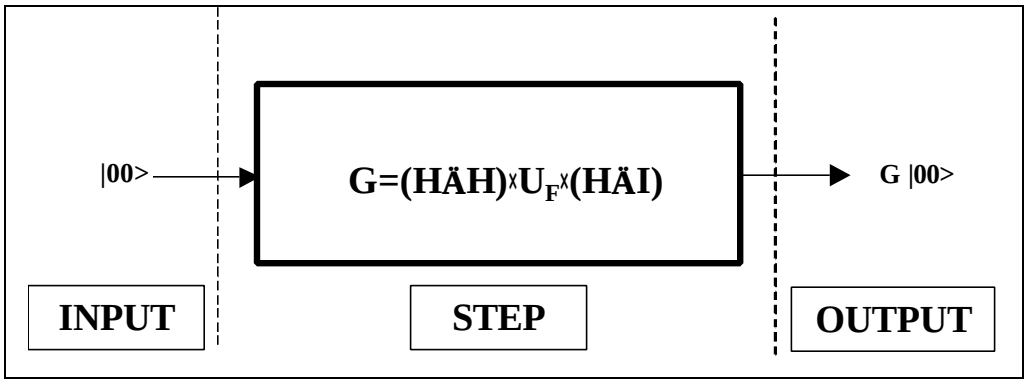


Figure 4: Deutsch's Quantum Gate – Final Representation

Therefore, being:

H	|0>	|1>
|0>	$1/2^{1/2}$	$1/2^{1/2}$
|1>	$1/2^{1/2}$	$-1/2^{1/2}$

I	|0>	|1>
|0>	1	0
|1>	0	1

we can automatically calculate $H \otimes I$ and $H \otimes H$:

HÄI	|00>	|01>	|10>	|11>
|00>	$1/2^{1/2}$	0	$1/2^{1/2}$	0
|01>	0	$1/2^{1/2}$	0	$1/2^{1/2}$
|10>	$1/2^{1/2}$	0	$-1/2^{1/2}$	0
|11>	0	$1/2^{1/2}$	0	$-1/2^{1/2}$

HÄH	|00>	|01>	|10>	|11>
|00>	1/2	1/2	1/2	1/2
|01>	1/2	-1/2	1/2	-1/2
|10>	1/2	1/2	-1/2	-1/2
|11>	1/2	-1/2	-1/2	1/2

We rewrite U_F when $f=f_3$:

U_{F_3}	|00>	|01>	|10>	|11>
|00>	1	0	0	0
|01>	0	1	0	0
|10>	0	0	0	1
|11>	0	0	1	0

The final programmable gate $G_3=(H\otimes H)\cdot(U_{F_3}\cdot(H\otimes I))$ is so obtained:

U_{F_3}x(***HÄI***)	|00>	|01>	|10>	|11>
|00>	$1/2^{1/2}$	0	$1/2^{1/2}$	0
|01>	0	$1/2^{1/2}$	0	$1/2^{1/2}$
|10>	0	$1/2^{1/2}$	0	$-1/2^{1/2}$
|11>	$1/2^{1/2}$	0	$-1/2^{1/2}$	0

G_3	|00>	|01>	|10>	|11>
|00>	$1/2^{1/2}$	$1/2^{1/2}$	0	0
|01>	0	0	$1/2^{1/2}$	$-1/2^{1/2}$
|10>	0	0	$1/2^{1/2}$	$1/2^{1/2}$
|11>	$1/2^{1/2}$	$-1/2^{1/2}$	0	0

Let's calculate the programmable gates for the other possible input functions. Here are the map tables.

x	$f_1(x)$
0	0
1	0

→

(x_0, y_0)	$F_1(x_0, y_0)$
(0,0)	(0,0)
(0,1)	(0,1)
(1,0)	(1,0)
(1,1)	(1,1)

x	$f_2(x)$
0	1
1	1

→

(x_0, y_0)	$F_2(x_0, y_0)$
(0,0)	(0,1)
(0,1)	(0,0)
(1,0)	(1,1)
(1,1)	(1,0)

x	$f_4(x)$
0	1
1	0

→

(x_0, y_0)	$F_4(x_0, y_0)$
(0,0)	(0,1)
(0,1)	(0,0)
(1,0)	(1,0)
(1,1)	(1,1)

From every table, it is easy to calculate the matrix operator:

$\|x_0\ y_0\rangle$	$U_{F_1}\|x_0\ y_0\rangle$
|00>	|00>
|01>	|01>
|10>	|10>
|11>	|11>

→

U_{F_1}	|00>	|01>	|10>	|11>
|00>	1	0	0	0
|01>	0	1	0	0
|10>	0	0	1	0
|11>	0	0	0	1

$\|x_0\ y_0\rangle$	$U_{F_2}\|x_0\ y_0\rangle$
|00>	|01>
|01>	|00>
|10>	|11>
|11>	|10>

→

U_{F_2}	|00>	|01>	|10>	|11>
|00>	0	1	0	0
|01>	1	0	0	0
|10>	0	0	0	1
|11>	0	0	1	0

(x_0, y_0)	$U_{F_4}\|x_0\ y_0\rangle$
|00>	|01>
|01>	|00>
|10>	|10>
|11>	|11>

→

U_{F_4}	|00>	|01>	|10>	|11>
|00>	0	1	0	0
|01>	1	0	0	0
|10>	0	0	1	0
|11>	0	0	0	1

Different U_{F_i} (i=1,2,4) generate different programmable gates $G_i=(H\otimes H)\cdot U_{F_i}\cdot(H\otimes I)$:

G_1	|00>	|01>	|10>	|11>
|00>	$1/2^{1/2}$	$1/2^{1/2}$	0	0
|01>	$1/2^{1/2}$	$-1/2^{1/2}$	0	0
|10>	0	0	$1/2^{1/2}$	$1/2^{1/2}$
|11>	0	0	$1/2^{1/2}$	$-1/2^{1/2}$

G_2	|00>	|01>	|10>	|11>
|00>	$1/2^{1/2}$	$1/2^{1/2}$	0	0
|01>	$-1/2^{1/2}$	$1/2^{1/2}$	0	0
|10>	0	0	$1/2^{1/2}$	$1/2^{1/2}$

|11>	0	0	$-1/2^{1/2}$	$1/2^{1/2}$

G_4	|00>	|01>	|10>	|11>
|00>	$1/2^{1/2}$	$1/2^{1/2}$	0	0
|01>	0	0	$-1/2^{1/2}$	$1/2^{1/2}$
|10>	0	0	$1/2^{1/2}$	$1/2^{1/2}$
|11>	$-1/2^{1/2}$	$1/2^{1/2}$	0	0

Finally, different programmable gates, generate different superposition states:

G_1|00>	=	$1/2^{1/2}$	|00>	+	$1/2^{1/2}$	|01>
G_2|00>	=	$1/2^{1/2}$	|00>	–	$1/2^{1/2}$	|01>
G_3|00>	=	$1/2^{1/2}$	|00>	+	$1/2^{1/2}$	|11>
G_4|00>	=	$1/2^{1/2}$	|00>	–	$1/2^{1/2}$	|11>

Observe that $G_1|00>$ and $G_2|00>$ can be written as the tensor products of two simpler vectors:

G_1|00>	=	$1/2^{1/2}$	|0>	$\otimes$	(|0>	+	|1>)
G_2|00>	=	$1/2^{1/2}$	|0>	$\otimes$	(|0>	–	|1>)

This is not possible for $G_3|00>$ and $G_4|00>$. We say that these two vectors constitute two entangled states.

BOX 3: ENTANGLED STATES

A vector $\underline{v}$ of dimension 2^n is said to represent an *entangled state* if and only if it can't be written as the tensor product of n vector of dimension 2. Mathematically, the entanglement condition is so written:

$$\neg\exists \underline{v}_1,...,\underline{v}_n : \underline{v} = \underline{v}_1 \otimes ... \otimes \underline{v}_n$$

When the quantum gate has generated the output vector, which is a linear complex superposition of basis vectors, measurement takes place.
We assume that measurement is a non-deterministic operation whose input is the linear superposition of basis vectors and whose output is only one of these basis vectors. The probability of a basis vector of being the result of measurement is given by the squared modulus of its complex co-ordinate in the starting superposition.
This description of measurement is taken from quantum mechanics and it constitutes the main constraint on the access one has to the results of our quantum gate. The non-deterministic evolution of a quantum system by measurement constitutes the true qualitative difference

between a quantum computation and a simple parallel computation. This is the price we pay to Nature.

BOX 4: QUANTUM MEASUREMENT

In quantum mechanics *measurement* is a non-deterministic operator. Writing a vector $\underline{v}$ as the complex linear combination of n basis vector $\underline{v}_i$ (i=1,…,n), the probability to *observe* $\underline{v}_i$ when $\underline{v}$ is measured is given by the squared modulus of the *complex co-ordinate* of $\underline{v}_i$ in $\underline{v}$.

$$\underline{v} = a_1\underline{v}_1 + a_2\underline{v}_2 + ... + a_n\underline{v}_n \xrightarrow{\textit{Measurement}}$$

Basis vector	Probability
$\underline{v}_1$	$\|\|a_1\|\|^2$
$\underline{v}_1$	$\|\|a_2\|\|^2$
…	…
$\underline{v}_n$	$\|\|a_n\|\|^2$

When we apply measurement to the superposition of basis vectors resulting from one of our 4 gates, we obtain:

Superposition of Basis Vectors before Measurement	Result of Measurement	
	Vector	Probability
$\boldsymbol{G_1}\|00\rangle = 1/\sqrt{2}\ \|00\rangle + 1/\sqrt{2}\ \|01\rangle$	|00> |01>	$\|\|1/\sqrt{2}\|\|^2=0.5$ $\|\|1/\sqrt{2}\|\|^2=0.5$
$\boldsymbol{G_2}\|00\rangle = 1/\sqrt{2}\ \|00\rangle - 1/\sqrt{2}\ \|01\rangle$	|00> |01>	$\|\|1/\sqrt{2}\|\|^2=0.5$ $\|\|1/\sqrt{2}\|\|^2=0.5$
$\boldsymbol{G_3}\|00\rangle = 1/\sqrt{2}\ \|00\rangle + 1/\sqrt{2}\ \|11\rangle$	|00> |11>	$\|\|1/\sqrt{2}\|\|^2=0.5$ $\|\|1/\sqrt{2}\|\|^2=0.5$
$\boldsymbol{G_4}\|00\rangle = 1/\sqrt{2}\ \|00\rangle - 1/\sqrt{2}\ \|11\rangle$	|00> |11>	$\|\|1/\sqrt{2}\|\|^2=0.5$ $\|\|1/\sqrt{2}\|\|^2=0.5$

With measurement, the quantum block ends. In Deutsch's algorithm the quantum block is repeated only one time, so only one resulting basis vector is collected.

6. DECODER

When the final basis vector has been produced, we must interpret the information it carries in order to establish if f is constant or balanced.
If the resulting vector is |00> nothing can be said about which function was encoded in U_F. But if the result is |01> or |11>, we know that the function was f_1 or f_2 in the first case, f_3 or f_4

in the second. In fact only gates G_1 and G_2 may produce a vector such that, when it is measured, basis vector |01> has a non-null probability of being observed. Similarly, only gates G_3 and G_4 may produce a superposition of basis vectors where vector |11> has non-null probability amplitude. Since f_1 and f_2 are constant, whereas f_3 and f_4 are balanced, the resulting vector is easily decoded in order to answer Deutsch's problem:

Resulting Vector after Measurement	**Answer**
|00>	Nothing can be said
|01>	*f* is constant
|11>	*f* is balanced

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 3: Deutsch-Jozsa's Algorithm

1. AIM

The aim of this part is to show that Deutsch-Jozsa's algorithm is based on the special form of its quantum gate. This gate is implemented according to the technique developed in Part 1. Here, we stress the importance of the structure of matrix operator U_F.

2. DEUTSCH-JOZSA'S PROBLEM

Deutsch-Jozsa's algorithm is so stated:

Input	A constant or balanced function $f:\{0,1\}^n \rightarrow \{0,1\}$
Problem	Decide if f is constant or balanced

This problem is very similar to Deutsch's problem, but it has been generalised to $n>1$.

3. ENCODER

We firstly deal with some special functions with $n=2$. This should help the reader to understand the main ideas of this algorithm. Then we discuss the general case with $n=2$ and finally we encode a balanced or constant function in the more general situation $n>0$.

A. Encoding a constant function with value 1

Let's consider the case:

$$n = 2$$
$$\forall x \in \{0,1\}^n : f(x) = 1$$

In this case f map table is so defined:

x	$f(x)$
00	1
01	1
10	1
11	1

The encoder block takes f map table as input and encodes it into matrix operator U_F, which acts inside of a complex Hilbert space.

Step 1
Function *f* is encoded into the injective function *F*, built according to the following statement:

$$F:\{0,1\}^{n+1}\to\{0,1\}^{n+1}:F(x_0,x_1,y_0)=(x_0,x_1,f(x_0,x_1)\oplus y_0)$$

Then *F* map table is:

(x_0, x_1, y_0)	$F(x_0, x_1, y_0)$
000	001
010	011
100	101
110	111
001	000
011	010
101	100
111	110

Step 2
Let's now encode *F* into U_F map table using the rule:

$$\forall t\in\{0,1\}^{n+1}:\ U_F\,[t(t)]=t[F(t)]$$

where *t* is the code map defined in Part 1. This means:

$\|x_0\ x_1\ y_0\rangle$	$U_F\ \|x_0\ x_1\ y_0\rangle$
|000>	|001>
|010>	|011>
|100>	|101>
|110>	|111>
|001>	|000>
|011>	|010>
|101>	|100>
|111>	|110>

Here, we used ket notation (see Part 1) to denote basis vectors.

Step 3
Starting from the map table of U_F, we calculate the corresponding matrix operator.
This matrix is obtained using the rule:

$$[U_F]_{ij}=1\Leftrightarrow U_F|j\rangle=|i\rangle$$

So, U_F is the following matrix:

U_F	|000>	|001>	|010>	|011>	|100>	|101>	|110>	|111>
|000>	0	1	0	0	0	0	0	0
|001>	1	0	0	0	0	0	0	0
|010>	0	0	0	1	0	0	0	0
|011>	0	0	1	0	0	0	0	0
|100>	0	0	0	0	0	1	0	0
|101>	0	0	0	0	1	0	0	0
|110>	0	0	0	0	0	0	0	1
|111>	0	0	0	0	0	0	1	0

Using matrix tensor product, U_F can be written as:

$$U_F = I \otimes I \otimes C$$

where $\otimes$ is the tensor product, I is the identity matrix of order 2 and C is the NOT-matrix so defined:

$$C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Matrix C flips a basis vector: in fact it transforms vector |0> into |1> and |1> into |0>.
If matrix U_F is applied to the tensor product of three vectors of dimension 2, the resulting vector is the tensor product of the three vectors obtained applying matrix I to the first two input vectors and matrix C to the third.

BOX 1: TENSOR PRODUCT AND ENTANGLEMENT

Given m vectors $\underline{v}_1,.., \underline{v}_m$ of dimension $2^{d_1},.., 2^{d_m}$ and m matrix operators $M_1,.., M_m$ of order $2^{d_1}\times 2^{d_1},.., 2^{d_m}\times 2^{d_m}$ the following property holds:

$$(M_1 \otimes .. \otimes M_m)\cdot(\underline{v}_1 \otimes .. \otimes \underline{v}_n) = M_1 \cdot \underline{v}_1 \otimes ... \otimes M_m \cdot \underline{v}_n$$

This means that, if a matrix operator can be written as the tensor product of m smaller matrix operator, the evolutions of the m vectors the operator is applied to are independent, namely no correlation is present among this vector. An important corollary is that if the initial state was not entangled, also the final state is not entangled.

The structure of U_F is such that the first two vectors in the input tensor product are preserved (action of I), whereas the third is flipped (action of C). We can easily verify that this action corresponds to the constraints stated by U_F map table.

B. Encoding a constant function with value 0

Let's now consider the case:

$$n = 2$$
$$\forall x \in \{0,1\}^n : f(x) = 0$$

In this case f map table is so defined:

x	$f(x)$
00	0
01	0
10	0
11	0

Step 1

F map table is:

(x_0, x_1, y_0)	$F(x_0, x_1, y_0)$
000	000
010	010
100	100
110	110
001	001
011	011
101	101
111	111

Step 2

F map table is encoded into U_F map table:

$\|x_0\ x_1\ y_0\rangle$	$U_F\ \|x_0\ x_1\ y_0\rangle$
|000>	|000>
|010>	|010>
|100>	|100>
|110>	|110>
|001>	|001>
|011>	|011>
|101>	|101>
|111>	|111>

Step 3

It is very easy to transform this map table into a matrix. In fact, we can observe that every vector is preserved.

Therefore the corresponding matrix is the identity matrix of order 2^3.

U_F	|000>	|001>	|010>	|011>	|100>	|101>	|110>	|111>
|000>	1	0	0	0	0	0	0	0
|001>	0	1	0	0	0	0	0	0
|010>	0	0	1	0	0	0	0	0
|011>	0	0	0	1	0	0	0	0
|100>	0	0	0	0	1	0	0	0
|101>	0	0	0	0	0	1	0	0
|110>	0	0	0	0	0	0	1	0
|111>	0	0	0	0	0	0	0	1

Using matrix tensor product, this matrix can be written as:

$$U_F = I \otimes I \otimes I$$

The structure of U_F is such that all basis vectors of dimension 2 in the input tensor product evolve independently. No vector controls any other vector.

C. Encoding a balanced function

Consider now the balanced function:

$$n = 2$$
$$\forall (x_1,..,x_n) \in \{0,1\}^n : f(x_1,..,x_n) = x_1 \oplus .. \oplus x_n$$

In this case *f* map table is the following:

x	$f(x)$
00	0
01	1
10	1
11	0

Step 1

The following map table calculated in the usual way represents the injective function *F* (where *f* is encoded into):

(x_0, x_1, y_0)	$F(x_0, x_1, y_0)$
000	000
010	011
100	101
110	110

(x_0, x_1, y_0)	$F(x_0, x_1, y_0)$
001	001
011	010
101	100
111	111

Step 2

Let's now encode F into U_F map table:

$\|x_0\ x_1\ y_0\rangle$	$U_F\ \|x_0\ x_1\ y_0\rangle$
|000>	|000>
|010>	|011>
|100>	|101>
|110>	|110>
|001>	|001>
|011>	|010>
|101>	|100>
|111>	|111>

Step 3

The matrix corresponding to U_F is:

U_F	|000>	|001>	|010>	|011>	|100>	|101>	|110>	|111>
|000>	1	0	0	0	0	0	0	0
|001>	0	1	0	0	0	0	0	0
|010>	0	0	0	1	0	0	0	0
|011>	0	0	1	0	0	0	0	0
|100>	0	0	0	0	0	1	0	0
|101>	0	0	0	0	1	0	0	0
|110>	0	0	0	0	0	0	1	0
|111>	0	0	0	0	0	0	0	1

This matrix can't be written as the tensor product of smaller matrices. In fact, if we write it as a block matrix we obtain:

U_F	|00>	|01>	|10>	|11>
|00>	I	0	0	0
|01>	0	C	0	0
|10>	0	0	C	0
|11>	0	0	0	I

This means that the matrix operator acting on the third vector in the input tensor product depends on the values of the first two vectors. If these vectors are |0> and |0>, for instance, the operator acting on the third vector is the identity matrix, if the first two vectors are |0> and |1> then the evolution of the third is determined by matrix *C*.

We say that this operator creates entanglement, namely correlation among the vectors in the tensor product.

D. General case with n=2

Consider now a general function with n=2.
In this general case f map table is the following:

x	$f(x)$
00	f_{00}
01	f_{01}
10	f_{10}
11	f_{11}

with $f_i \in \{0,1\}$, i=00,01,10,11. If f is constant then $\exists y \in \{0,1\} \forall x \in \{0,1\}^2 : f(x)=y$. If f is balanced then $|\{f_i: f_i = 0\}|=|\{f_i: f_i = 1\}|$

Step 1

Injective function F (where f is encoded) is represented by the following map table calculated in the usual way:

(x_0, x_1, y_0)	$F(x_0, x_1, y_0)$
000	0 0 f_{00}
010	0 1 f_{01}
100	1 0 f_{10}
110	1 1 f_{11}
001	0 0 $\neg f_{00}$
011	0 1 $\neg f_{01}$
101	1 0 $\neg f_{10}$
111	1 1 $\neg f_{11}$

Step 2

Let's now encode F into U_F map table:

$\|x_0\ x_1\ y_0\rangle$	$U_F\ \|x_0\ x_1\ y_0\rangle$
$\|000\rangle$	$\|0\ 0\ f_{00}\rangle$
$\|010\rangle$	$\|0\ 1\ f_{01}\rangle$
$\|100\rangle$	$\|1\ 0\ f_{10}\rangle$
$\|110\rangle$	$\|1\ 1\ f_{11}\rangle$
$\|001\rangle$	$\|0\ 0\ \neg f_{00}\rangle$
$\|011\rangle$	$\|0\ 1\ \neg f_{01}\rangle$
$\|101\rangle$	$\|1\ 0\ \neg f_{10}\rangle$
$\|111\rangle$	$\|1\ 1\ \neg f_{11}\rangle$

Step 3

The matrix corresponding to U_F can be written as a block matrix with the following general form:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_i=I$ if $f_i=0$ and $M_i=C$ if $f_i=1$, i=00,01,10,11. The structure of this matrix is such that, when the first two vectors are mapped into some other vectors, the null operator is applied to the third vector, generating a null probability amplitude for this transition. This means that the first two vectors are always left unchanged. On the contrary, operators $M_i \in \{I, C\}$ and they are applied to the third vector when the first two are mapped into themselves. If all M_i coincide, operator U_F encodes a constant function. Otherwise it encodes a non-constant function. If $|\{M_i: M_i = I\}|=|\{M_i: M_i = C\}|$ then f is balanced.

E. General case

Consider now the general case $n>0$. Input function f map table is the following:

$x \hat{I} \{0,1\}^n$	$f(x)$
0..0	$f_{0..0}$
0..1	$f_{0..1}$
…	…
1..1	$f_{1..1}$

with $f_i \in \{0,1\}$, $i \in \{0,1\}^n$. If f is constant then $\exists y \in \{0,1\} \forall x \in \{0,1\}^n : f(x)=y$. If f is balanced then $|\{f_i: f_i = 0\}|=|\{f_i: f_i = 1\}|$.

Step 1

The map table of the corresponding injective function F is:

$x \hat{I} \{0,1\}^{n+1}$	$F(x)$
0..00	0..0 $f_{0..0}$
…	…
1..10	1..1 $f_{1..1}$
0..01	0..0 $\neg f_{0..0}$
…	…
1..11	1..1 $\neg f_{1..1}$

Step 2

Let's now encode F into U_F map table:

$\|x\rangle$	$U_F \|x\rangle$
$\|0..00\rangle$	$\|0..0\ f_{0..0}\rangle$
…	…
$\|1..10\rangle$	$\|1..1\ f_{1..1}\rangle$
$\|0..01\rangle$	$\|0..0\ \neg f_{0..0}\rangle$
…	…
$\|1..11\rangle$	$\|1..1\ \neg f_{1..1}\rangle$

Step 3

The matrix corresponding to U_F can be written as a block matrix with the following general form:

U_F	$\|0..0\rangle$	$\|0..1\rangle$	…	$\|1..1\rangle$
$\|0..0\rangle$	$M_{0..0}$	0	0	0
$\|0..1\rangle$	0	$M_{0..1}$	0	0
…	…	…	…	…
$\|1..1\rangle$	0	0	0	$M_{1..1}$

where $M_i=I$ if $f_i=0$ and $M_i=C$ if $f_i=1$, $i\in\{0,1\}^n$.
This matrix leaves the first n vectors unchanged and applies operator $M_i\in\{I, C\}$ to the last vector.
If all M_i coincide with I or C, the matrix encodes a constant function and it can be written as ${}^nI \otimes I$ or ${}^nI\otimes C$. In this case no entanglement is generated. Otherwise, if the condition $|\{M_i: M_i = I\}|=|\{M_i: M_i = C\}|$ is fulfilled, then f is balanced and the operator creates correlation among vectors.

BOX 2: MATRIX TENSOR AND DOT POWER

Given a matrix M we denote its *k-power according to tensor product* as: :

$$^kM = M \otimes .. \otimes M \ (k \ times)$$

On the contrary *k-power according to dot product* is denoted as usually:

$$M^{\,k} = M \cdot .. \cdot M \ (k \ times)$$

4. QUANTUM BLOCK

Matrix U_F, the output of the encoder, is now embedded into the quantum gate of Deutsch-Jozsa's algorithm. As we did for Deutsch's algorithm, we describe this gate using a quantum circuit:

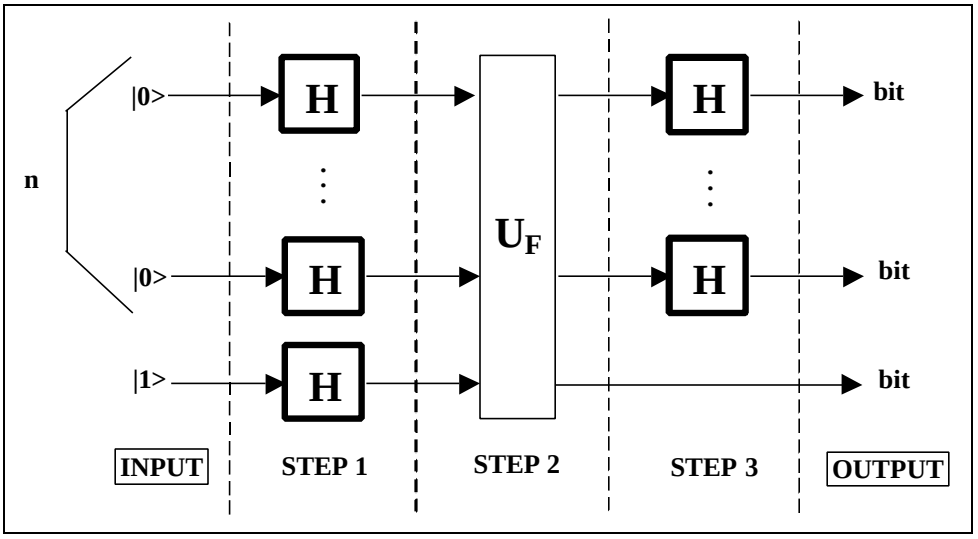


Figure 1: Circuit of Deutsch-Jozsa's Quantum Gate – First Representation

Using rule 3 (Part 1), we compile the previous circuit into the following:

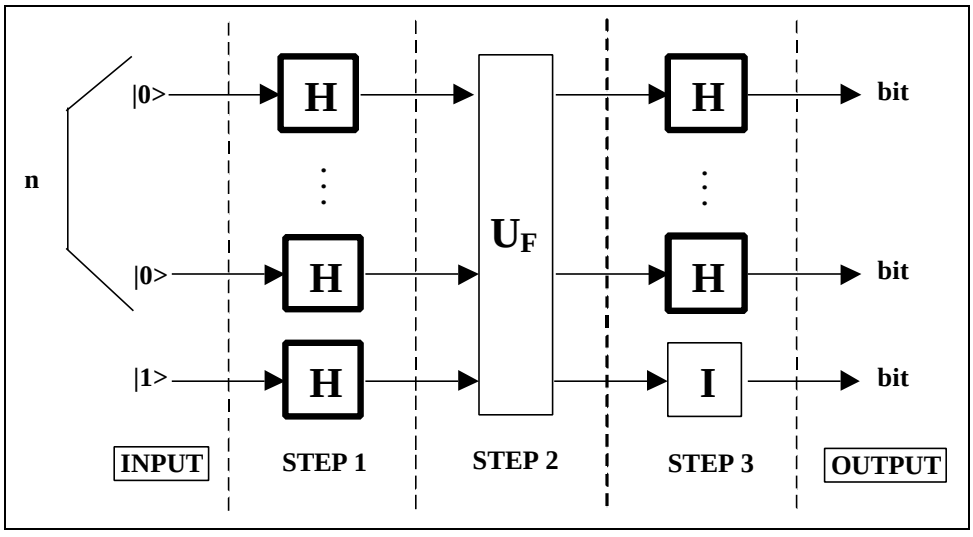


Figure 2: Circuit of Deutsch-Jozsa's Quantum Gate – Second Representation

Let's consider operator U_F in the case of constant and balanced functions. The structure of this operator strongly influences the structure of the whole gate. We shall analyse this structure in

the case f is 1 everywhere, f is 0 everywhere and in the general case with n=2. Finally, we propose the general form for our gate with n>0.

A. Constant function with value 1

If f is constant and its value is 1, matrix operator U_F can be written as ${}^nI\otimes C$. This means, as it is stated by rule 1 in Part 1, that U_F can be decomposed into n+1 smaller operators acting concurrently on the n+1 vectors of dimension 2 in the input tensor product.
The resulting circuit representation is reported in fig.3:

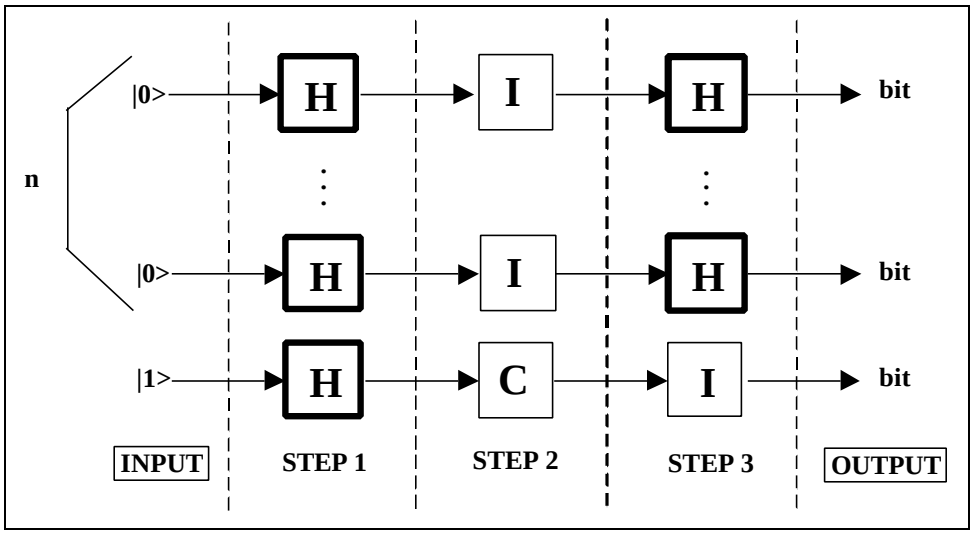


Figure 3: Constant Function with Value 1 – First Circuit

Let's now use rule number 2, finding the sub-gate acting on every vector of dimension 2 in input:

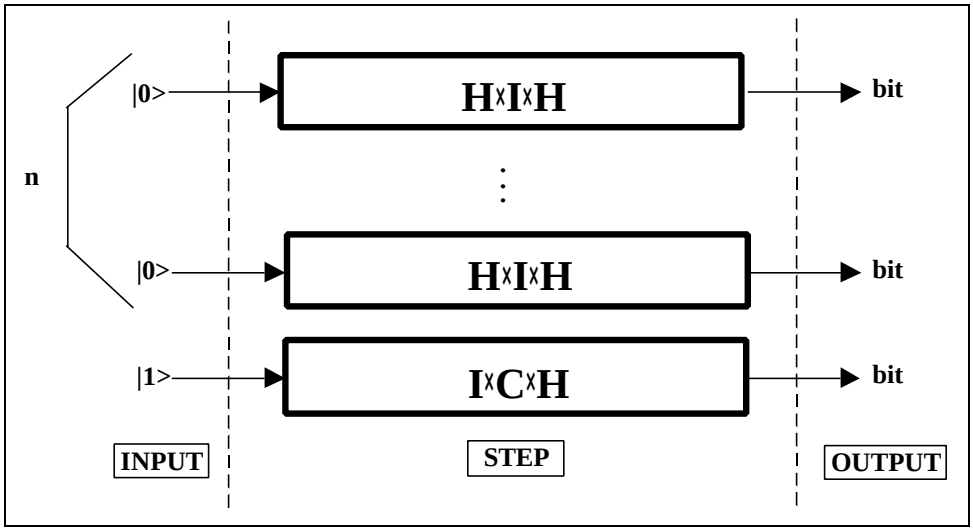


Figure 4 Constant Function with Value 1 – Second Circuit

Let's observe that every vector in input evolves independently from other vectors. This is because operator U_F doesn't create any correlation. So, the evolution of every input vector can be analysed separately.
This circuit can be written in a simpler way, observing that $M \cdot I = M$:

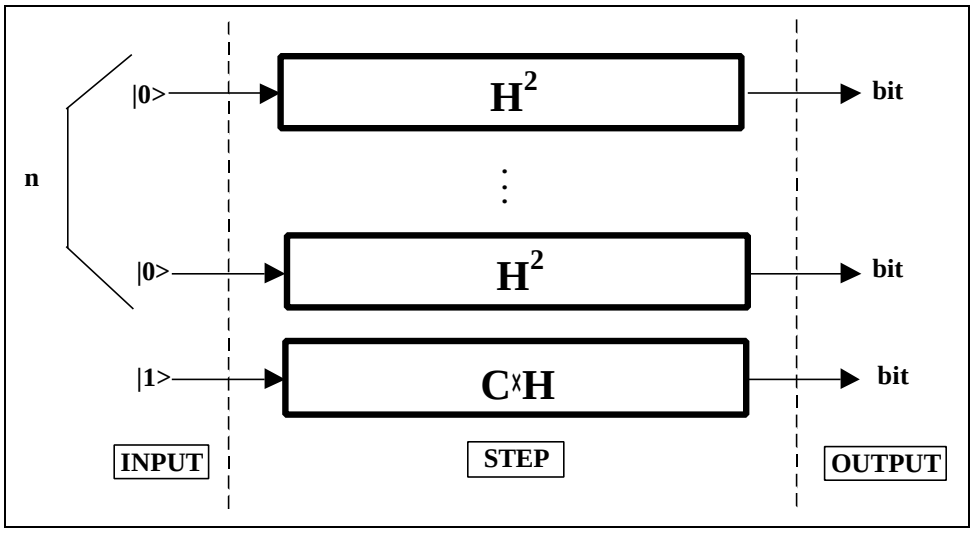


Figure 5: Constant Function with Value 1 – Third Circuit

We can easily show that:

$$H^2 = I$$

Therefore the circuit is rewritten in this way:

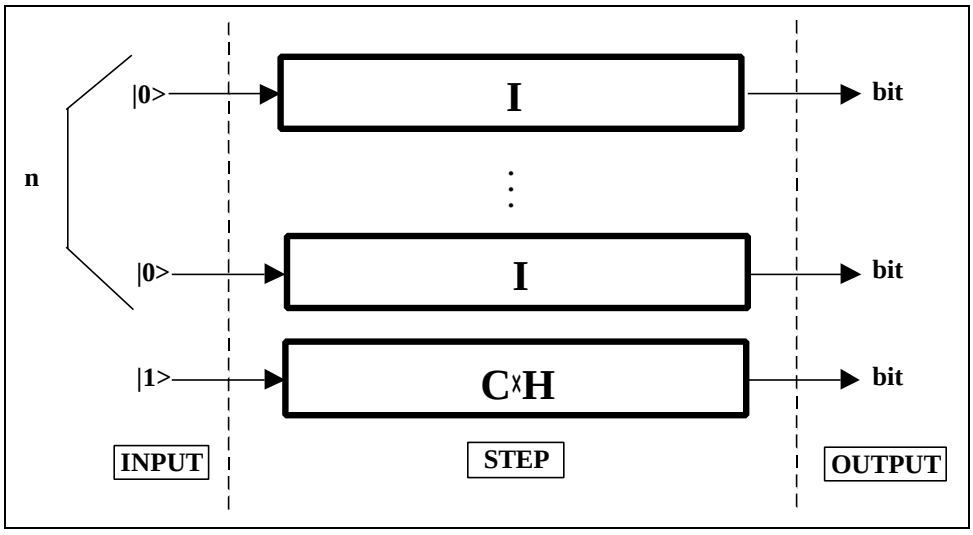


Figure 6: Constant Function with Value 1 – Fourth Circuit

Let's consider now the effect of the operators acting on every vector:

$$I|0\rangle = |0\rangle \qquad C \cdot H|1\rangle = -\frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Using these results in rule number 4 of Part 1 and applying rule number 3, we get the following circuit representation:

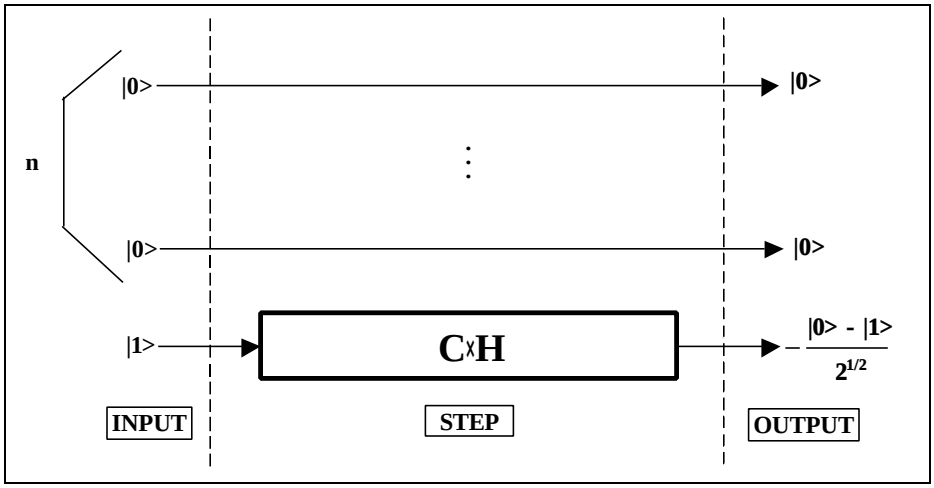


Figure 7: Constant Function with Value 1 – Fifth Circuit

You see that, if *f* is constant with value 1, the first *n* vectors are preserved.

B. Constant function with value 0

A similar analysis can be repeated for a constant function with value 0. In this situation U_F can be written as ${}^nI \otimes I$ and the final circuit is:

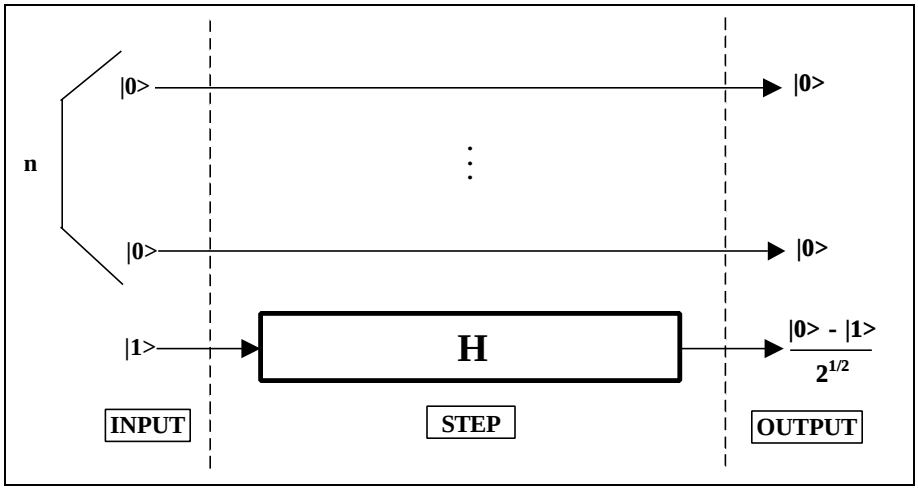


Figure 8: Constant Function with Value 0 – Final Circuit

In this case too, the first *n* input vectors are preserved. So, their output values after the quantum gate has acted are still |0>.

C. General case (*n*=2)

The gate implementing Deutsch-Jozsa's algorithm in the general case is obtained operating on the circuit of fig.2 with rules 1 and 2 defined in Part 1.
This is the circuit evolution:

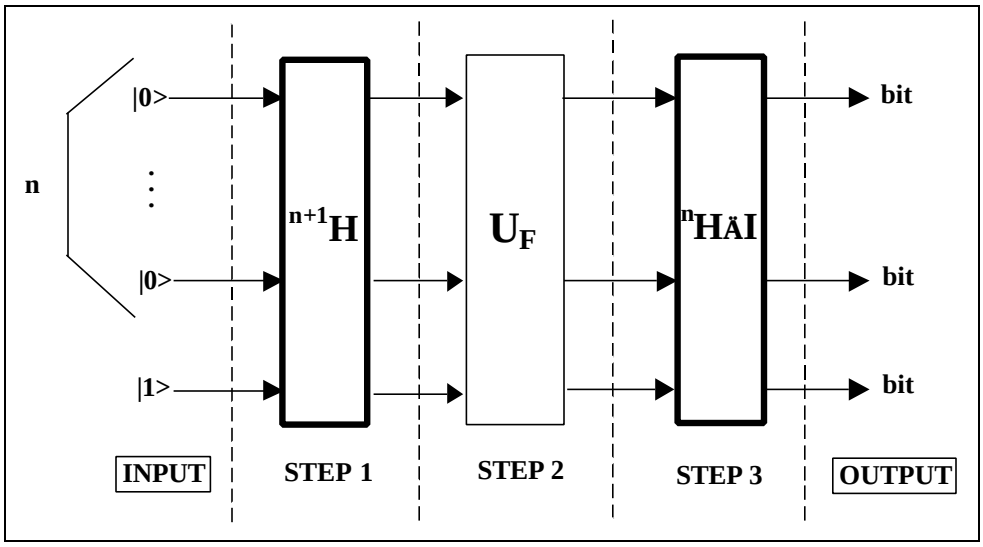


Figure 9: Evolution of the Circuit in fig.2

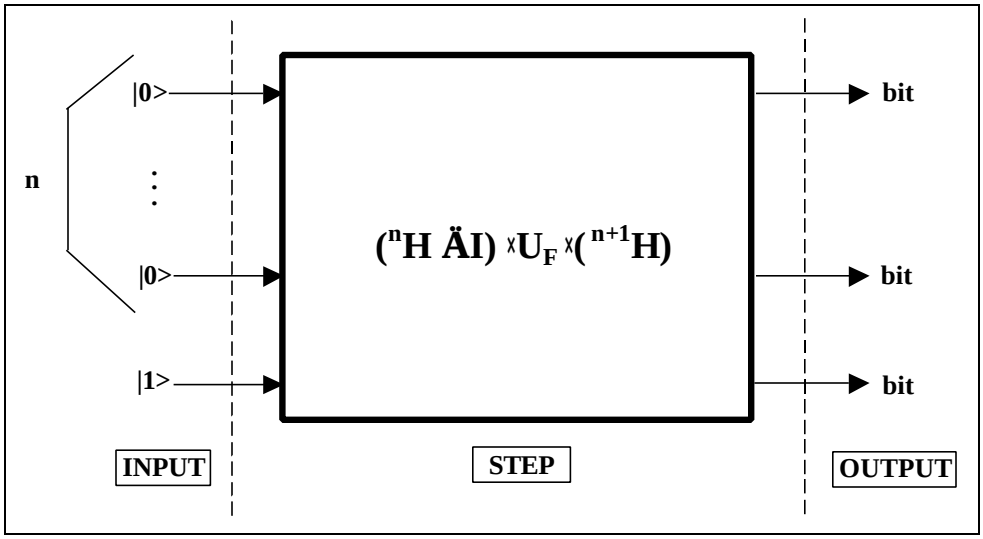


Figure 10: Deutsch-Jozsa's Quantum Gate

If *n*=2, U_F has the following form:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_i \in \{I, C\}$, i=00,01,10,11.
Let's calculate the quantum gate $G=({}^{2}H \otimes I) \cdot U_F \cdot ({}^{2+1}H)$ in this case:

${}^{3}H$	|00>	|01>	|10>	|11>
|00>	$H/2$	$H/2$	$H/2$	$H/2$
|01>	$H/2$	$-H/2$	$H/2$	$-H/2$
|10>	$H/2$	$H/2$	$-H/2$	$-H/2$
|11>	$H/2$	$-H/2$	$-H/2$	$H/2$

${}^{2}H$ÄI	|00>	|01>	|10>	|11>
|00>	$I/2$	$I/2$	$I/2$	$I/2$
|01>	$I/2$	$-I/2$	$I/2$	$-I/2$
|10>	$I/2$	$I/2$	$-I/2$	$-I/2$
|11>	$I/2$	$-I/2$	$-I/2$	$I/2$

U_F ¤ ${}^{3}H$	|00>	|01>	|10>	|11>
|00>	$M_{00}H/2$	$M_{00}H/2$	$M_{00}H/2$	$M_{00}H/2$
|01>	$M_{01}H/2$	$-M_{01}H/2$	$M_{01}H/2$	$-M_{01}H/2$
|10>	$M_{10}H/2$	$M_{10}H/2$	$-M_{10}H/2$	$-M_{10}H/2$
|11>	$M_{11}H/2$	$-M_{11}H/2$	$-M_{11}H/2$	$M_{11}H/2$

G	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{01}+M_{10}+M_{11})H/4$	$(M_{00}-M_{01}+M_{10}-M_{11})H/4$	$(M_{00}+M_{01}-M_{10}-M_{11})H/4$	$(M_{00}-M_{01}-M_{10}+M_{11})H/4$
|01>	$(M_{00}-M_{01}+M_{10}-M_{11})H/4$	$(M_{00}+M_{01}+M_{10}+M_{11})H/4$	$(M_{00}-M_{01}-M_{10}+M_{11})H/4$	$(M_{00}+M_{01}-M_{10}-M_{11})H/4$
|10>	$(M_{00}+M_{01}-M_{10}-M_{11})H/4$	$(M_{00}-M_{01}-M_{10}+M_{11})H/4$	$(M_{00}+M_{01}+M_{10}+M_{11})H/4$	$(M_{00}-M_{01}+M_{10}-M_{11})H/4$
|11>	$(M_{00}-M_{01}-M_{10}+M_{11})H/4$	$(M_{00}+M_{01}-M_{10}-M_{11})H/4$	$(M_{00}-M_{01}+M_{10}-M_{11})H/4$	$(M_{00}+M_{01}+M_{10}+M_{11})H/4$

Now, consider the application of G to vector |001>:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (M_{00} + M_{01} + M_{10} + M_{11})H|1\rangle + \frac{1}{4}|01\rangle \otimes (M_{00} - M_{01} + M_{10} - M_{11})H|1\rangle +$$
$$\frac{1}{4}|10\rangle \otimes (M_{00} + M_{01} - M_{10} - M_{11})H|1\rangle + \frac{1}{4}|11\rangle \otimes (M_{00} - M_{01} - M_{10} + M_{11})H|1\rangle$$

Consider the operator $(M_{00}+M_{01}+M_{10}+M_{11})H$ under the hypotheses of balanced functions $M_i \in \{I, C\}$ and $|\{M_i: M_i = I\}|=|\{M_i: M_i = C\}|$. Then:

$\boldsymbol{M_{00}+M_{01}+M_{10}+M_{11}}$	|0>	|1>
|0>	2	2
|1>	2	2

$\boldsymbol{(M_{00}+M_{01}+M_{10}+M_{11})H/4}$	|0>	|1>
|0>	$1/2^{1/2}$	0
|1>	$1/2^{1/2}$	0

Thus:

$$\frac{1}{4}(M_{00}+M_{01}+M_{10}+M_{11})H|1\rangle = 0$$

This means that the probability amplitude of vector |001> of being mapped into a vector |000> or |001> is null.
Consider now the operators:

$$(M_{00}+M_{01}+M_{10}+M_{11})H$$
$$(M_{00}-M_{01}+M_{10}-M_{11})H$$
$$(M_{00}+M_{01}-M_{10}-M_{11})H$$
$$(M_{00}-M_{01}-M_{10}+M_{11})H$$

under the hypotheses $\forall i$: $M_i=I$, which holds for constant functions with values 0:

$\boldsymbol{M_{00}+M_{01}+M_{10}+M_{11}}$	|0>	|1>
|0>	4	0
|1>	0	4

$\boldsymbol{(M_{00}+M_{01}+M_{10}+M_{11})H/4}$	|0>	|1>
|0>	$1/2^{1/2}$	$1/2^{1/2}$
|1>	$1/2^{1/2}$	$-1/2^{1/2}$

$\boldsymbol{M_{00}-M_{01}+M_{10}-M_{11}}$	|0>	|1>
|0>	0	0
|1>	0	0

$\boldsymbol{M_{00}+M_{01}-M_{10}-M_{11}}$	|0>	|1>
|0>	0	0
|1>	0	0

$\boldsymbol{M_{00}-M_{01}-M_{10}+M_{11}}$	|0>	|1>
|0>	0	0
|1>	0	0

Using these calculations, we obtain the following results:

$$\frac{1}{4}(M_{00} - M_{01} + M_{10} - M_{11})H|1\rangle = 0$$
$$\frac{1}{4}(M_{00} + M_{01} - M_{10} - M_{11})H|1\rangle = 0$$
$$\frac{1}{4}(M_{00} - M_{01} - M_{10} + M_{11})H|1\rangle = 0$$

This means that the probability amplitude of vector |001> of being mapped into a superposition of vectors |010>, |011>, |100>, |101>, |110>, |111> is null. The only possible output is a superposition of vectors |000> and |001>, as we showed before using circuits. A similar analysis can be developed under the hypotheses $\forall i$: M_i=C.

It is useful to outline the evolution of the probability amplitudes of every basis vector while operator ^3H, U_F and $^2H \otimes I$ are applied in sequence, for instance when f has constant value 1. This is done in fig.11:

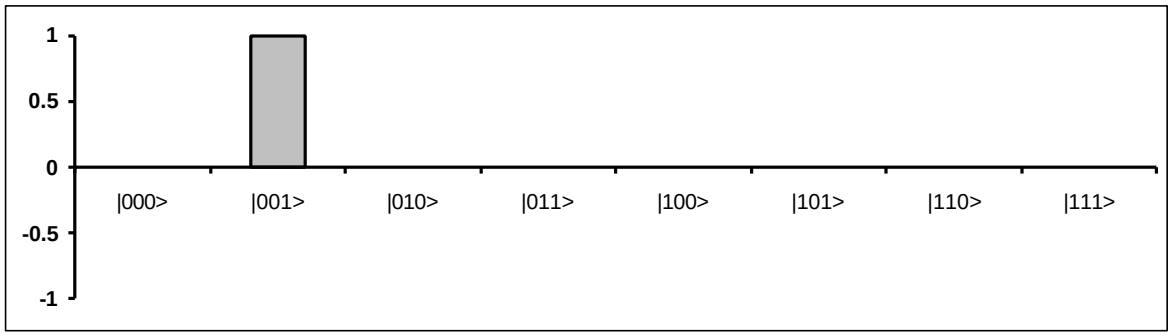


Figure 11.a: Input Probability Amplitudes

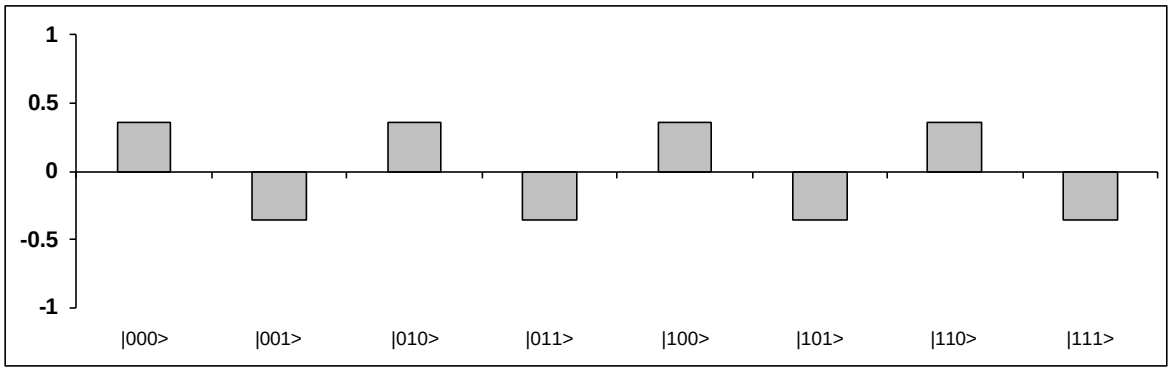


Figure 11.b: Probability Amplitudes after Step 1 (Fig. 1)

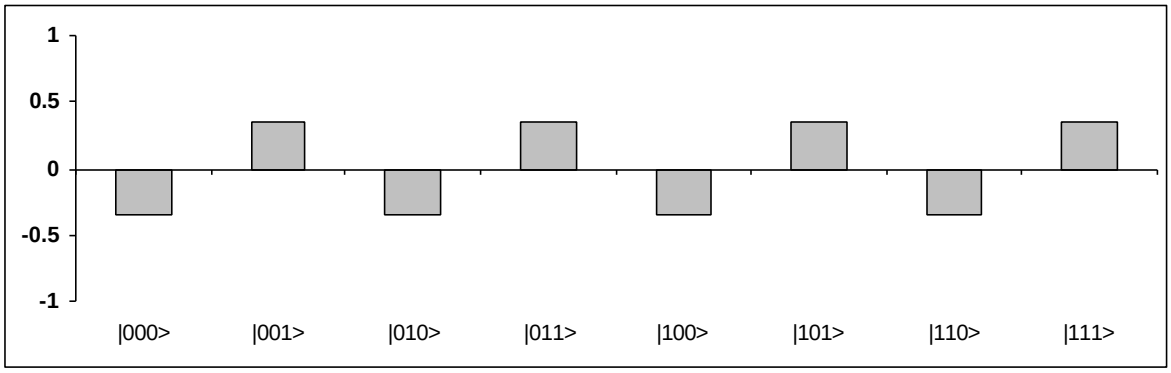


Figure 11.c: Probability Amplitudes after Step 2 (Fig. 1)

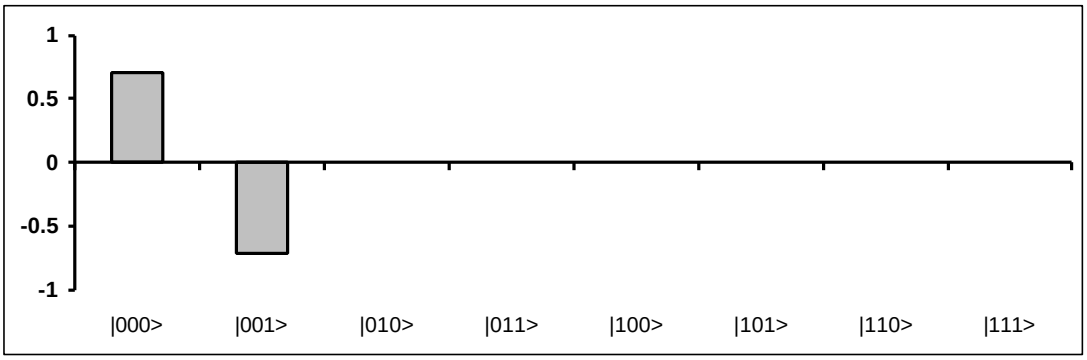


Figure 11.d: Probability Amplitudes after Step 3 (Fig. 1)

Operator ${}^3H$ puts the initial canonical basis vector |001> into a superposition of all basis vectors with the same (real) coefficients in modulus, but with positive sign if the last vector is |0>, negative otherwise.

Operator U_F in this case doesn't create correlation: it flips the third vector independently from the values of the first two vectors.

Finally, ${}^2H \otimes I$ produces interference: for every basis vector $|x_0 x_1 y_0\rangle$ it calculates its output probability amplitude $a'_{x_0 x_1 y_0}$ as the summation of the probability amplitudes of all basis vectors in the form $|x_0 x_1 y_0\rangle$ in the input superposition, all with the same sign if $|x_0 x_1\rangle = |00\rangle$, otherwise changing the sign of exactly the middle of the probability amplitudes.
Since, in this case, the vectors in the form $|x_0 x_1 0\rangle$ have the same (negative real) probability amplitude and vectors in the form $|x_0 x_1 1\rangle$ have the same (positive real) probability amplitude, when $|x_0 x_1\rangle = |00\rangle$, probability amplitudes interfere positively. Otherwise the terms in the summation interfere destructively annihilating the result.

D. General case (n>0)

In the general case n>0, U_F has the following form:

U_F	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	0	0
|0..1>	0	$M_{0..1}$	0	0
…	…	…	…	…
|1..1>	0	0	0	$M_{1..1}$

where $M_i \in \{I, C\}$, $i \in \{0,1\}^n$.
Let's calculate the quantum gate $G=({}^nH \otimes I) \cdot U_F \cdot ({}^{n+1}H)$:

${}^{n+1}H$	|0..0>	…	|j>	…	|1..1>
|0..0>	$H/2^{n/2}$	…	$H/2^{n/2}$	…	$H/2^{n/2}$
…	…	…	…	…	…
|i>	$H/2^{n/2}$	…	$(-1)^{i \cdot j}H/2^{n/2}$	…	$(-1)^{i \cdot (1..1)}H/2^{n/2}$
…	…	…	…	…	…
|11>	$H/2^{n/2}$	…	$(-1)^{(1..1) \cdot j}H/2^{n/2}$	…	$(-1)^{(1..1) \cdot (1..1)}H/2^{n/2}$

Here we employed binary string operator $\barwedge$, which represents the parity of the AND bit per bit between two strings.

BOX 3: PARITY OF BIT PER BIT AND $\barwedge$

Given two binary strings x and y of length n, we define:

$$x \cdot y = x_1 \cdot y_1 \oplus x_2 \cdot y_2 \oplus .. \oplus x_n \cdot y_n$$

The symbol $\cdot$ used between two bits is interpreted as the logical AND operator.

We shall prove that matrix ${}^{n+1}H$ really has the described form. We show that:

$$\left[{}^nH\right]_{ij} = \frac{(-1)^{i \cdot j}}{2^{n/2}}$$

The proof is by induction:

- n=1:

$$\left[{}^1H\right]_{0,0} = \frac{1}{2^{1/2}} = \frac{(-1)^{(0)\cdot(0)}}{2^{1/2}} \quad \left[{}^1H\right]_{0,1} = \frac{1}{2^{1/2}} = \frac{(-1)^{(0)\cdot(1)}}{2^{1/2}}$$
$$\left[{}^1H\right]_{1,0} = \frac{1}{2^{1/2}} = \frac{(-1)^{(1)\cdot(0)}}{2^{1/2}} \quad \left[{}^1H\right]_{1,1} = \frac{-1}{2^{1/2}} = \frac{(-1)^{(1)\cdot(1)}}{2^{1/2}}$$

- $n>1$:

$$\left[{}^{n}H\right]_{i0,j0} = \frac{1}{2^{1/2}}\left[{}^{n-1}H\right]_{i,j} = \frac{1}{2^{1/2}}\frac{(-1)^{i\cdot j}}{2^{(n-1)/2}} = \frac{(-1)^{(i0)\cdot(j0)}}{2^{n/2}}$$

$$\left[{}^{n}H\right]_{i0,j1} = \frac{1}{2^{1/2}}\left[{}^{n-1}H\right]_{i,j} = \frac{1}{2^{1/2}}\frac{(-1)^{i\cdot j}}{2^{(n-1)/2}} = \frac{(-1)^{(i0)\cdot(j1)}}{2^{n/2}}$$

$$\left[{}^{n}H\right]_{i1,j0} = \frac{1}{2^{1/2}}\left[{}^{n-1}H\right]_{i,j} = \frac{1}{2^{1/2}}\frac{(-1)^{i\cdot j}}{2^{(n-1)/2}} = \frac{(-1)^{(i1)\cdot(j0)}}{2^{n/2}}$$

$$\left[{}^{n}H\right]_{i1,j1} = -\frac{1}{2^{1/2}}\left[{}^{n-1}H\right]_{i,j} = -\frac{1}{2^{1/2}}\frac{(-1)^{i\cdot j}}{2^{(n-1)/2}} = \frac{(-1)^{(i1)\cdot(j1)}}{2^{n/2}}$$

Matrix ${}^{n+1}H$ is obtained from ${}^{n}H$ by tensor product. Similarly, matrix ${}^{n}H\otimes I$ is calculated:

n*HÄ*I	|0..0>	…	|*j*>	…	|1..1>
|0..0>	$I/2^{n/2}$	…	$I/2^{n/2}$	…	$I/2^{n/2}$
…	…	…	…	…	…
|*i*>	$I/2^{n/2}$	…	$(-1)^{i\cdot j}I/2^{n/2}$	…	$(-1)^{i\cdot(1..1)}I/2^{n/2}$
…	…	…	…	…	…
|11>	$I/2^{n/2}$	…	$(-1)^{(1..1)\cdot j}I/2^{n/2}$	…	$(-1)^{(1..1)\cdot(1..1)}I/2^{n/2}$

U_F ¤ ***n+1*H**	|0..0>	…	|*j*>	…	|1..1>
|0..0>	$M_{0..0}H/2^{n/2}$	…	$M_{0..0}H/2^{n/2}$	…	$M_{0..0}H/2^{n/2}$
…	…	…	…	…	…
|*i*>	$M_iH/2^{n/2}$	…	$(-1)^{i\cdot j}M_iH/2^{n/2}$	…	$(-1)^{i\cdot(1..1)}M_iH/2^{n/2}$
…	…	…	…	…	…
|1..1>	$M_{1..1}H/2^{n/2}$	…	$(-1)^{(1..1)\cdot j}M_{1..1}H/2^{n/2}$	…	$(-1)^{(1..1)\cdot(1..1)}M_{1..1}H/2^{n/2}$

We calculated only the first column of gate *G* since this operator is applied exclusively to input vector |0..01> and so only the first column is involved.

G	|0..0>	…
|0..0>	$(M_{0..0}+..+M_i+..+M_{1..1})H/2^n$	…
…	…	…
|*i*>	$(\Sigma_{j\in\{0,1\}^n}(-1)^{i\cdot j}M_j)H/2^n$	…
…	…	…
|1..1>	$(\Sigma_{j\in\{0,1\}^n}(-1)^{(1..1)\cdot j}M_j)H/2^n$	…

Now consider the case of *f* constant. We saw that this means that all matrices M_i are identical.

This implies:

$$\frac{1}{2^n}\left(\sum_j (-1)^{i\cdot j} M_j\right)H = 0$$

since in this summation the number of +1 equals the number of –1. Therefore, the input vector |0..01> is mapped into a superposition of vectors |0..00> and |0..01> as we showed using circuits.
If *f* is balanced, the number of $M_i = I$ equals the number of $M_i = C$. This implies:

$$\frac{1}{2^n}\left(\sum_j M_j\right)H = \frac{1}{2^n}\left(2^{n-1}I + 2^{n-1}C\right)H = \frac{1}{2}\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}H = \frac{1}{2\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 0\\ 1 & 0\end{bmatrix}$$

And therefore:

$$\frac{1}{2^n}\left(\sum_j M_j\right)H|1\rangle = 0$$

This means that input vector |0..01>, in the case of balanced functions, can't be mapped by the quantum gate into a superposition containing vectors |0..00> or |0..01>.

The quantum block terminates with measurement. Considering the results showed till now, we can determine the possible outputs of measurement and their probabilities:

Superposition of Basis Vectors Before Measurement	**Result of Measurement**	
	Vector	**Probability**
Constant functions: $\boldsymbol{G}\|0..01\rangle = \|0..0\rangle \otimes (a_0\|0\rangle + a_1\|1\rangle)$	|0..00> |0..01>	$\|\|a_0\|\|^2$ $\|\|a_1\|\|^2$
Balanced functions: $\boldsymbol{G}\|0..01\rangle = \Sigma_{i\in\{0,1\}^n - \{0..00,\ 0..01\}}\ a_i\|i\rangle$	$\forall i \in \{0,1\}^n - \{0..00,\ 0..01\}: \|i\rangle$	$\|\|a_i\|\|^2$

The set $A–B$ is given by all elements of A, unless those elements belonging to B too. This set is sometimes denoted as A/B. The quantum block is repeated only one time in Deutsch-Jozsa's algorithm. So, the final collection is made only by one vector.

5. DECODER

As in Deutsch's algorithm, when the final basis vector has been measured, we must interpret it in order to decide if *f* is constant or balanced.
If the resulting vector is |0..0> we know that the function was constant, otherwise we decide that it is balanced. In fact gate *G* produces a vector such that, when it is measured, only basis vectors |0..00> and |0..01> have a non-null probability amplitude exclusively in the case *f* is

constant. Besides, if *f* is balanced, these two vectors have null coefficients in the linear combination of basis vectors generated by *G*. In this way, the resulting vector is easily decoded in order to answer Deutsch-Jozsa's problem:

Resulting Vector after Measurement	**Answer**
|0..00>	*f* is constant
|0..01>	*f* is constant
otherwise	*f* is balanced

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 4: Simon's Algorithm

1. AIM

In this part we are going to illustrate Simon's algorithm using circuits and pointing out the role of interference.

2. SIMON'S PROBLEM

Simon's problem is so stated:

Input	$f:\{0,1\}^n \rightarrow \{0,1\}^n$: $\exists s \in \{0,1\}^n - \{0..0\}: \forall x,y \in \{0,1\}^n$: $f(x)=f(y) \Leftrightarrow (x=y \vee x=y \oplus s)$
Problem	Find s

3. ENCODER

As we did for Deutsch-Jozsa's algorithm, we firstly consider some special cases.

A. Introductory example

Let's consider the case:

$$n = 2$$
$$f(00) = 00,\ f(01) = 01$$
$$s = 11$$

Then, f map table is:

(x_0,x_1)	$f(x_0,x_1)$
00	00
01	01
10	01
11	00

Step1

Function *f* is encoded into the injective function *F* built in the usual way:

$$F:\{0,1\}^{n+n} \to \{0,1\}^{n+n} \quad \textit{such that}$$
$$F(x_0, x_1,.., x_{n-1}, y_0, y_1,.., y_{n-1}) = (x_0, x_1,.., x_{n-1}, f(x_0, x_1,.., x_{n-1}) \oplus (y_0, y_1,.., y_{n-1}))$$

This is *F* map table:

$(x_0,.., x_{n-1}, y_0,.., y_{n-1})$	$F(x_0,.., x_{n-1}, y_0,.., y_{n-1})$
0000	0000
0100	0101
1000	1001
1100	1100
0001	0001
0101	0100
1001	1000
1101	1101

$(x_0,.., x_{n-1}, y_0,.., y_{n-1})$	$F(x_0,.., x_{n-1}, y_0,.., y_{n-1})$
0010	0010
0110	0111
1010	1011
1110	1110
0011	0011
0111	0110
1011	1010
1111	1111

Step2

Let's now encode *F* map table into U_F map table. As usually, the rule is:

$$\forall t \in \{0,1\}^{n+n}: U_F[t(t)] = t[F(t)]$$

where t is the code map defined in Part 1. This means:

$\|x_0.. x_{n-1} y_0.. y_{n-1}\rangle$	$U_F\|x_0.. x_{n-1} y_0.. y_{n-1}\rangle$
|0000>	|0000>
|0100>	|0101>
|1000>	|1001>
|1100>	|1100>
|0001>	|0001>
|0101>	|0100>
|1001>	|1000>
|1101>	|1101>

$\|x_0.. x_{n-1} y_0.. y_{n-1}\rangle$	$U_F\|x_0.. x_{n-1} y_0.. y_{n-1}\rangle$
|0010>	|0010>
|0110>	|0111>
|1010>	|1011>
|1110>	|1110>
|0011>	|0011>
|0111>	|0110>
|1011>	|1010>
|1111>	|1111>

Step3

Using the rule:

$$[U_F]_{ij} = 1 \Leftrightarrow U_F|j\rangle = |i\rangle$$

we calculate U_F as a block matrix:

U_F	|00>	|01>	|10>	|11>
|00>	$I\otimes I$	0	0	0
|01>	0	$I\otimes C$	0	0
|10>	0	0	$I\otimes C$	0
|11>	0	0	0	$I\otimes I$

This matrix preserves the first two vectors in the input tensor product vector and:

- it preserves the last two too when the first two vectors are |0> and |0> or |1> and |1>;
- it preserves the third vector, but it flips the fourth, when the first two vectors are |0> and |1> or |1> and |0>.

Observe that the block matrix in cell (i,i) is identical to the block matrix in cell ($i\oplus s$, $i\oplus s$), where i is the binary label of the vector marking the matrix row and column of the cell.

B. General case with n=2

In general, if n=2, repeating steps 1, 2 and 3 as we did for Deutsch-Jozsa's algorithm, we obtain the general operator U_F in the following form:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_i\in\{I\otimes I, I\otimes C, C\otimes I, C\otimes C\}$ and $M_i=M_j \Leftrightarrow (j=i\vee j=i\oplus s)$.

C. General case

Generalising the results obtained in the previous examples and reasoning like in Deutsch-Jozsa's algorithm, we can find the structure of U_F for Simon's algorithm too. The final matrix is:

U_F	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	…	0
|0..1>	0	$M_{0..1}$	…	0
…	…	…	…	...
|1..1>	0	0	0	$M_{1..1}$

where $M_i= P_1 \otimes..\otimes P_n$, $P_k\in\{I, C\}$, k=1,..,n and $M_i = M_j \Leftrightarrow (j=i\vee j=i\oplus s)$.

Note that the column labels are basis vectors of dimension *n* (not 2*n*).

4. QUANTUM BLOCK

The following circuit describes Simon's quantum gate:

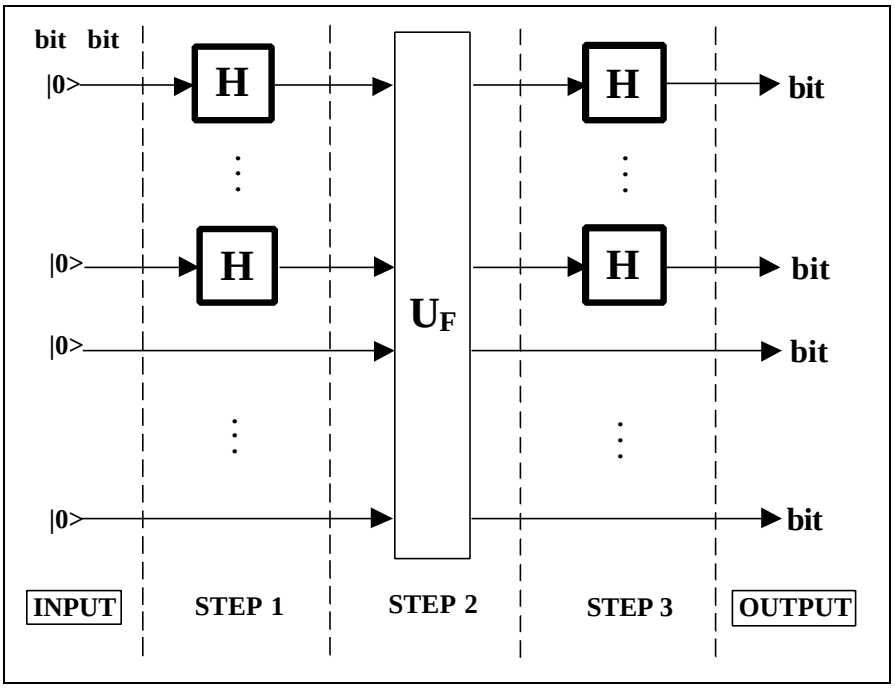


Figure 1: Circuit of Simon's Quantum Gate – First Representation

Using the transformation rules defined in Part 1, we can easily compile this circuit into the corresponding gate:

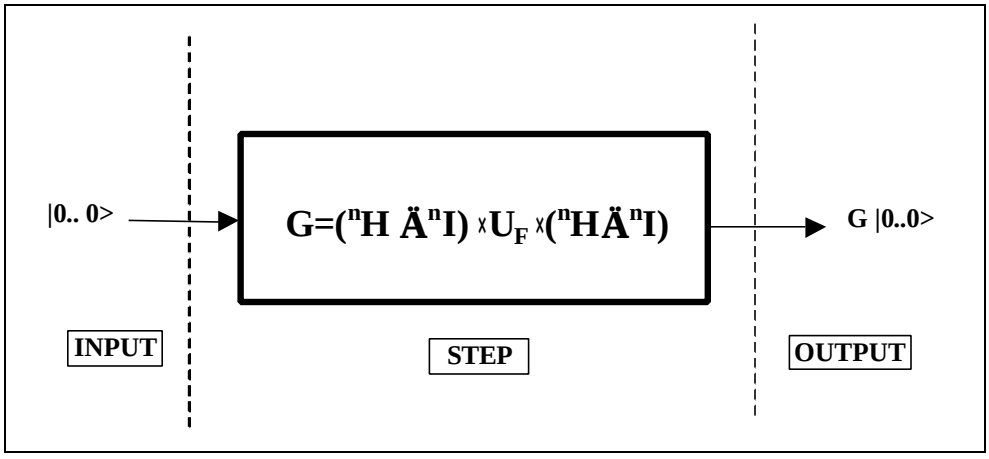


Figure 2: Simon's Quantum Gate

Let's calculate this gate and establish what output vector it produces. We firstly deal with the introductory example of Section 1, passing then to the general case with n=2. Finally we describe the gate structure in a general situation (n>0).

A. Introductory example

In the case considered before (n=2, f(00)=00, f(01)=01, s=11), the quantum gate assumes this form:

$$G=({}^{n}H \otimes {}^{n}I) \cdot U_F \cdot ({}^{n}H \otimes {}^{n}I)$$

where U_F has been calculated in Section 1.
Let's start finding matrix ${}^{2}H \otimes {}^{2}I$, using the results about the tensor power of matrix H obtained in Part 3:

${}^{2}H$ Ä ${}^{2}I$	|00>	|01>	|10>	|11>
|00>	${}^{2}I/2$	${}^{2}I/2$	${}^{2}I/2$	${}^{2}I/2$
|01>	${}^{2}I/2$	- ${}^{2}I/2$	${}^{2}I/2$	- ${}^{2}I/2$
|10>	${}^{2}I/2$	${}^{2}I/2$	- ${}^{2}I/2$	- ${}^{2}I/2$
|11>	${}^{2}I/2$	- ${}^{2}I/2$	- ${}^{2}I/2$	${}^{2}I/2$

We recall matrix U_F and calculate G:

U_F	|00>	|01>	|10>	|11>
|00>	${}^{2}I$	0	0	0
|01>	0	$I \otimes C$	0	0
|10>	0	0	$I \otimes C$	0
|11>	0	0	0	${}^{2}I$

U_F x (^{2}H Ä ^{2}I)	|00>	|01>	|10>	|11>
|00>	${}^{2}I/2$	${}^{2}I/2$	${}^{2}I/2$	${}^{2}I/2$
|01>	$I \otimes C/2$	- $I \otimes C/2$	$I \otimes C/2$	- $I \otimes C/2$
|10>	$I \otimes C/2$	$I \otimes C/2$	- $I \otimes C/2$	- $I \otimes C/2$
|11>	${}^{2}I/2$	- ${}^{2}I/2$	- ${}^{2}I/2$	${}^{2}I/2$

G	|00>	|01>	|10>	|11>
|00>	$({}^{2}I+I \otimes C)/2$	0	0	$({}^{2}I-I \otimes C)/2$
|01>	0	$({}^{2}I+I \otimes C)/2$	$({}^{2}I-I \otimes C)/2$	0
|10>	0	$({}^{2}I-I \otimes C)/2$	$({}^{2}I+I \otimes C)/2$	0
|11>	$({}^{2}I-I \otimes C)/2$	0	0	$({}^{2}I+I \otimes C)/2$

Having G this structure, when we apply it to vector |0000> we obtain the following result:

$$G|0000\rangle = |00\rangle \frac{\left({}^2I + I \otimes C\right)}{2}|00\rangle + |11\rangle \frac{\left({}^2I - I \otimes C\right)}{2}|00\rangle$$

This means:

$$G|0000\rangle = \frac{1}{2}|00\rangle\left(|00\rangle + |01\rangle\right) + \frac{1}{2}|11\rangle\left(|00\rangle - |01\rangle\right)$$

If we measure the output vector we can obtain only 4 possible results: |0000>, |0001>, |1100> and |1101>. Encode back into their binary labels the values of the first two basis vectors of dimension 2 in the output tensor product: these labels are 00 or 11. Let's solve the system:

$$\begin{cases} (00)\cdot(t_1 t_2) = 0 \\ (11)\cdot(t_1 t_2) = 0 \\ t_1 \neq 0,\ t_2 \neq 0 \end{cases} \Rightarrow \begin{cases} 0\cdot t_1 \oplus 0\cdot t_2 = 0 \\ 1\cdot t_1 \oplus 1\cdot t_2 = 0 \\ t_1 \neq 0,\ t_2 \neq 0 \end{cases} \Rightarrow \begin{cases} 0 \oplus 0 = 0 \\ t_1 \oplus t_2 = 0 \\ t_1 \neq 0,\ t_2 \neq 0 \end{cases} \Rightarrow \begin{cases} t_1 \oplus t_2 = 0 \\ t_1 \neq 0,\ t_2 \neq 0 \end{cases} \Rightarrow \begin{cases} t_1 = 1 \\ t_2 = 1 \end{cases}$$

Since s=(11), then s=(t_1 , t_2). Therefore s can be calculated as the solution of the system:

$$\begin{cases} (00)\cdot s = 0 \\ (11)\cdot s = 0 \\ s \neq (0,0) \end{cases}$$

B. General case with *n*=2

In the general case with n=2, matrix U_F has the form:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_i \in \{I\otimes I, I\otimes C, C\otimes I, C\otimes C\}$ and $M_i = M_j \Leftrightarrow (j=i \vee j=i\oplus s)$.

Using matrix ${}^2H \otimes {}^2I$ (calculated above), we obtain:

U_F × (${}^2H$ Ä ${}^2I$)	|00>	|01>	|10>	|11>
|00>	$M_{00}/2$	$M_{00}/2$	$M_{00}/2$	$M_{00}/2$
|01>	$M_{01}/2$	$-M_{01}/2$	$M_{01}/2$	$-M_{01}/2$
|10>	$M_{10}/2$	$M_{10}/2$	$-M_{10}/2$	$-M_{10}/2$
|11>	$M_{11}/2$	$-M_{11}/2$	$-M_{11}/2$	$M_{11}/2$

G	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{01}+M_{10}+M_{11})/4$	$(M_{00}-M_{01}+M_{10}-M_{11})/4$	$(M_{00}+M_{01}-M_{10}-M_{11})/4$	$(M_{00}-M_{01}-M_{10}+M_{11})/4$
|01>	$(M_{00}-M_{01}+M_{10}-M_{11})/4$	$(M_{00}+M_{01}+M_{10}+M_{11})/4$	$(M_{00}-M_{01}-M_{10}+M_{11})/4$	$(M_{00}+M_{01}-M_{10}-M_{11})/4$
|10>	$(M_{00}+M_{01}-M_{10}-M_{11})/4$	$(M_{00}-M_{01}-M_{10}+M_{11})/4$	$(M_{00}+M_{01}+M_{10}+M_{11})/4$	$(M_{00}-M_{01}+M_{10}-M_{11})/4$
|11>	$(M_{00}-M_{01}-M_{10}+M_{11})/4$	$(M_{00}+M_{01}-M_{10}-M_{11})/4$	$(M_{00}-M_{01}+M_{10}-M_{11})/4$	$(M_{00}+M_{01}+M_{10}+M_{11})/4$

Now, consider the following cases:

1) s=01; 2) s=10; 3) s=11

In the first case $M_{00}=M_{01}\neq M_{10}=M_{11}$. This means:

G_{01}	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{10})/2$	0	$(M_{00}-M_{10})/2$	0
|01>	0	$(M_{00}+M_{10})/2$	0	$(M_{00}-M_{10})/2$
|10>	$(M_{00}-M_{10})/2$	0	$(M_{00}+M_{10})/2$	0
|11>	0	$(M_{00}-M_{10})/2$	0	$(M_{00}+M_{10})/2$

In the second case $M_{00}=M_{10}\neq M_{01}=M_{11}$. This means:

G_{10}	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{01})/2$	$(M_{00}-M_{01})/2$	0	0
|01>	$(M_{00}-M_{01})/2$	$(M_{00}+M_{01})/2$	0	0
|10>	0	0	$(M_{00}+M_{01})/2$	$(M_{00}-M_{01})/2$
|11>	0	0	$(M_{00}-M_{01})/2$	$(M_{00}+M_{01})/2$

Finally, in the third case $M_{00}=M_{11}\neq M_{01}=M_{10}$. This means:

G_{11}	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{01})/2$	0	0	$(M_{00}-M_{01})/2$
|01>	0	$(M_{00}+M_{01})/2$	$(M_{00}-M_{01})/2$	0
|10>	0	$(M_{00}-M_{01})/2$	$(M_{00}+M_{01})/2$	0
|11>	$(M_{00}-M_{01})/2$	0	0	$(M_{00}+M_{01})/2$

Consider the application of G_{01}, G_{10} and G_{11} to vector |0000> in the three cases:

Case	s	**Output vector: G_s|0000>**
1	01	G_{01}|0000>=1/2 |00>$(M_{00}+M_{10})$|00> + 1/2 |10>$(M_{00}-M_{10})$|00>
2	10	G_{10}|0000>=1/2 |00>$(M_{00}+M_{01})$|00> + 1/2 |01>$(M_{00}-M_{01})$|00>
3	11	G_{11}|0000>=1/2 |00>$(M_{00}+M_{01})$|00> + 1/2 |11>$(M_{00}-M_{01})$|00>

If we measure the output vector in these three cases and we encode back into binary values the first two basis vectors in the tensor product, we obtain the following result:

Case	***s***	**Binary Values** (From The First Two Vectors)	**Probabilities**
1	01	(a, b)=(0,0) (a, b)=(1,0)	0.5 0.5
2	10	(a, b)=(0,0) (a ,b)=(0,1)	0.5 0.5
3	11	(a, b)=(0,1) (a, b)=(1,1)	0.5 0.5

Let's note that:

$$(a, b) \cdot s = 0$$

where a and b are the binary values from the first two vectors. The equations so generated let us find s as the solution of the corresponding system.

C. General case (n>0)

Let's consider a general positive value for number n.
We saw that operator U_F is:

$\boldsymbol{U_F}$	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	…	0
|0..1>	0	$M_{0..1}$	…	0
…	…	…	…	…
|1..1>	0	0	0	$M_{1..1}$

where $M_i = P_1 \otimes .. \otimes P_n$, $P_k \in \{I, C\}$, k=1,..,n and and $M_i = M_j \Leftrightarrow (j=i \vee j=i \oplus s)$.
Operator ${}^nH \otimes {}^nI$ is easily built from operator ${}^nH$ (already calculated in Part 3):

${}^n\boldsymbol{H}$ Ä ${}^n\boldsymbol{I}$	|0..0>	|0..1>	…	|j>	…	|1..1>
|0..0>	${}^nI/2^{n/2}$	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2}$	…	${}^nI/2^{n/2}$
|0..1>	${}^nI/2^{n/2}$	$-{}^nI/2^{n/2}$	…	$(-1)^{(0..1)\cdot j}({}^nI/2^{n/2})$	…	$-{}^nI/2^{n/2}$
…	…	…	…	…	…	…
|i>	${}^nI/2^{n/2}$	$(-1)^{i\cdot(0..1)}({}^nI/2^{n/2})$	…	$(-1)^{i\cdot j}({}^nI/2^{n/2})$	…	$(-1)^{i\cdot(1..1)}({}^nI/2^{n/2})$
…	…	…	…	…	…	…
|1..1>	${}^nI/2^{n/2}$	$-{}^nI/2^{n/2}$	…	$(-1)^{(1..1)\cdot j}({}^nI/2^{n/2})$	…	$(-1)^{(1..1)\cdot(1..1)}({}^nI/2^{n/2})$

U_F ⊗ (H^n ⊗ I^n)	|0..0>	…	|*j*>	…	|1..1>
|0..0>	$M_{0..0}/2^{n/2}$	…	$M_{0..0}/2^{n/2}$	…	$M_{0..0}/2^{n/2}$
…	…	…	…	…	…
|*i*>	$M_i/2^{n/2}$	…	$(-1)^{i\cdot j} M_i/2^{n/2}$	…	$(-1)^{i\cdot(1..1)} M_i/2^{n/2}$
…	…	…	…	…	…
|1..1>	$M_{1..1}/2^{n/2}$	…	$(-1)^{(1..1)\cdot j} M_{1..1}/2^{n/2}$	…	$(-1)^{(1..1)\cdot(1..1)} M_{1..1}/2^{n/2}$

The first column of the final gate has the following form:

G	|0..0>	…
|0..0>	$(M_{0..0}+..+M_i+..+M_{1..1})/2^n$	…
…	…	…
|*i*>	$(\Sigma_{j\in\{0,1\}^n}(-1)^{i\cdot j}M_j)/2^n$	…
…	…	…
|1..1>	$(\Sigma_{j\in\{0,1\}^n}(-1)^{(1..1)\cdot j}M_j)/2^n$	…

Consider the term:

$$\frac{1}{2^n}\sum_{j\in\{0,1\}^n}(-1)^{i\cdot j}M_j$$

Since $M_h=M_k \Leftrightarrow (h=k \vee h=k\oplus s)$, then this term may be written as:

$$\frac{1}{2^n}\sum_{j\in\{0,1\}^n}(-1)^{i\cdot j}M_j=\frac{1}{2^n}\sum_{k\in S}\left[(-1)^{i\cdot k}+(-1)^{i\cdot(k\oplus s)}\right]M_k=\frac{1}{2^n}\sum_{k\in S}(-1)^{i\cdot k}\left[1+(-1)^{i\oplus s}\right]M_k$$

where *S* is such that:

$$\neg\exists x,y\in S: x\oplus s=y \quad \neg\exists x,y\in\{0,1\}^n-S: x\oplus s=y$$

The gate can be rewritten in this way:

G	|0..0>	…
|0..0>	$\Sigma_{k\in S}(-1)^{(0..0)\cdot k}[1+(-1)^{(0..0)\cdot s}]M_k/2^n$	…
…	…	…
|*i*>	$\Sigma_{k\in S}(-1)^{i\cdot k}[1+(-1)^{i\cdot s}]M_k/2^n$	…
…	…	…
|1..1>	$\Sigma_{k\in S}(-1)^{(1..1)\cdot k}[1+(-1)^{(1..1)\cdot s}]M_k/2^n$	…

You see that the term $[1+(-1)^{i\cdot s}]$ is 0 if and only if $i\cdot s=1$. So, only those cells in the column that are labelled by \|*i*> such that $i\cdot s=0$ are non-null. This means that:

$$G|0..00..0\rangle = \frac{1}{2^{n-1}} \sum_{i \in \{0,1\}^n : i \cdot s = 0} |i\rangle$$

The quantum block ends with measurement, which therefore produces a basis vector $|i\rangle$ such that $i \cdot s = 0$.
How many times should the quantum block be repeated? A number that is sufficient to get enough information to determine s. Since every vector will constitute a coefficient vector for an equation where s is the variable vector, this number depends on how many different equations we need in order to find s. Since s has length n, in general we will need a number n of different equations. This requires, in general, a linear number of measurements (more precisely the probability to get n different vectors in the final collection increases linearly with n).

5. DECODER

The quantum block is repeated $O(n)$ times till a collection of n different vectors have been generated. As we did for the case $n=2$, for every vector in this collection, the first n basis vectors of dimension 2 composing it through tensor product are encoded back into their binary values. In this way they can be used as coefficients for building an equation whose variables are the bits of s. By solving the system made of this equations, we can find s.

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 5: Shor's Algorithm

1. AIM

In this part we are going to illustrate Shor's algorithm as a beefed-up version of Simon's algorithm.

2. SHOR'S PROBLEM

Shor's problem is so stated:

Given an integer number N, find a factor p for N

This problem seems to be different from the other problems we analysed till now. Nevertheless, it can be reduced to an equivalent problem with the same form as the other quantum problems. This reduction is made possible by a result of number theory that relates the period *r* of a special periodic function to the factors of an integer *N*. This function is:

$$f_{N,a} : \aleph \to \aleph \text{ such that } f_{N,a}(x) = a^x \bmod N$$

where *a* is a random number coprime to *N*, namely:

$$\gcd(a, N) = 1$$

where gcd(*x*, *y*) is the greatest common divisor between *x* and *y*.

This function is periodical (the period is at most *N*). Let the period be *r*. Then:

$$f_{N,a}(0) = f_{N,a}(r)$$

namely:

$$a^r \equiv 1 \bmod N$$

If the period is even, this equation can be rewritten as:

$$\left(a^{r/2}\right)^r \equiv 1 \bmod N \Leftrightarrow \left(a^{r/2}\right)^r - 1 \equiv 0 \bmod N \Leftrightarrow \left(a^{r/2} - 1\right)\left(a^{r/2} + 1\right) \equiv 0 \bmod N$$

This means:

$$\exists h \in \aleph : \left(a^{r/2} - 1\right)\left(a^{r/2} + 1\right) = hN$$

So, unless $(a^{r/2}-1) \equiv 0 \bmod N$ or $(a^{r/2}+1) \equiv 0 \bmod N$, namely $a^{r/2} \equiv \pm 1 \bmod N$, at least one of $a^{r/2}+1$ or $a^{r/2}-1$ should have a non-trivial factor in common with N. We find it calculating:

$$gcd\left(a^{r/2}-1, N\right) \qquad gcd\left(a^{r/2}+1, N\right)$$

Using this reduction, the true question becomes: "What is the period of f?" Since the period of this function is less than N, we can restrict it to the interval [0, 1, .., N-1]. We code every input value as a binary string. We need $n=[\log N]$ (eventually $[\log N]+1$) bits in order to code all the N possible input values.

Therefore, Shor's problem is translated into the following standard quantum problem:

Input	$f:\{0,1\}^n \rightarrow \{0,1\}^n$ with period r
Problem	Find r

3. ENCODER

We firstly deal with a simple example. Then we generalise our conclusions.

A. Introductory example

Let's consider the case:

$$N = 4 \Rightarrow n = 2; \qquad a = 3$$

Then, f map table is:

(x_0,x_1)	$f(x_0,x_1)$
00	01
01	10
10	01
11	10

The period of this function is r=2.

Step1

Function f is encoded into the injective function F built in the same way as in Simon's algorithm. This is F map table:

$(x_0,.., x_{n-1}, y_0,.., y_{n-1})$	$F(x_0,.., x_{n-1}, y_0,.., y_{n-1})$
0000	0001
0100	0110
1000	1001
1100	1110
0001	0000
0101	0111
1001	1000
1101	1111

$(x_0,.., x_{n-1}, y_0,.., y_{n-1})$	$F(x_0,.., x_{n-1}, y_0,.., y_{n-1})$
0010	0011
0110	0100
1010	1011
1110	1100
0011	0010
0111	0101
1011	1010
1111	1101

Step 2

Let's now encode F into the map table of operator U_F :

$\|x_0.. x_{n-1}\, y_0.. y_{n-1}\rangle$	$U_F\|x_0.. x_{n-1}\, y_0.. y_{n-1}\rangle$
|0000>	|0001>
|0100>	|0110>
|1000>	|1001>
|1100>	|1110>
|0001>	|0000>
|0101>	|0111>
|1001>	|1000>
|1101>	|1111>

$\|x_0.. x_{n-1}\, y_0.. y_{n-1}\rangle$	$U_F\|x_0.. x_{n-1}\, y_0.. y_{n-1}\rangle$
|0010>	|0011>
|0110>	|0100>
|1010>	|1011>
|1110>	|1100>
|0011>	|0010>
|0111>	|0101>
|1011>	|1010>
|1111>	|1101>

Step 3

The matrix corresponding to U_F is obtained using the rule:

$$\left[U_F\right]_{ij} = 1 \Leftrightarrow U_F \left| j \right\rangle = \left| i \right\rangle$$

or more simply observing that the first two vectors in the input tensor product are left unchanged, whereas the operator acting on the last two is chosen inside of the set {$I\otimes I$, $I\otimes C$, $C\otimes I$, $C\otimes C$} depending on the values of the first two vectors:

U_F	|00>	|01>	|10>	|11>
|00>	$I\otimes C$	0	0	0
|01>	0	$C\otimes I$	0	0
|10>	0	0	$I\otimes C$	0
|11>	0	0	0	$C\otimes I$

This matrix preserves the first two vectors and:

- it preserves the third, flipping the fourth when the first two vectors are |0> and |0> or |1> and |0>;
- it flips the third vector, preserving the fourth when the first two vectors are |0> and |1> or |1> and |1>.

Let's observe that the block matrix in cell (i,i) is identical to the block matrix in cell $((i+r)\text{mod}N, (i+r)\text{mod}N)$ where i is the binary label of the vector marking the matrix row and column of the cell.

B. General case with n=2

In general, if n=2, taking a different value for a and so a different period r, our operator still maps the first n vectors into themselves, but the block matrix pattern on the main diagonal changes.
Matrix U_F has the following form:

$\boldsymbol{U_F}$	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_i \in \{I\otimes I, I\otimes C, C\otimes I, C\otimes C\}$ and $M_i=M_j \Leftrightarrow (j=i \vee j=(i+r)\text{mod}N)$.

C. General case

Reasoning in the same way, we propose the following general form for matrix U_F when n>0:

$\boldsymbol{U_F}$	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	…	0
|0..1>	0	$M_{0..1}$	…	0
...	...	…	…	...
|1..1>	0	0	0	$M_{1..1}$

where $M_i = P_1 \otimes..\otimes P_n$, $P_k \in \{I, C\}$, k=1,..,n and and $M_i = M_j \Leftrightarrow (j=i \vee j=(i+r)\text{mod}N)$.
Like in Simon's algorithm, the column labels are basis vectors of dimension n (not 2n).

4. QUANTUM BLOCK

As we did fo Simon's algorithm we employ a quantum circuit in order to describe Shor's quantum gate. This circuit is reported in fig.1.

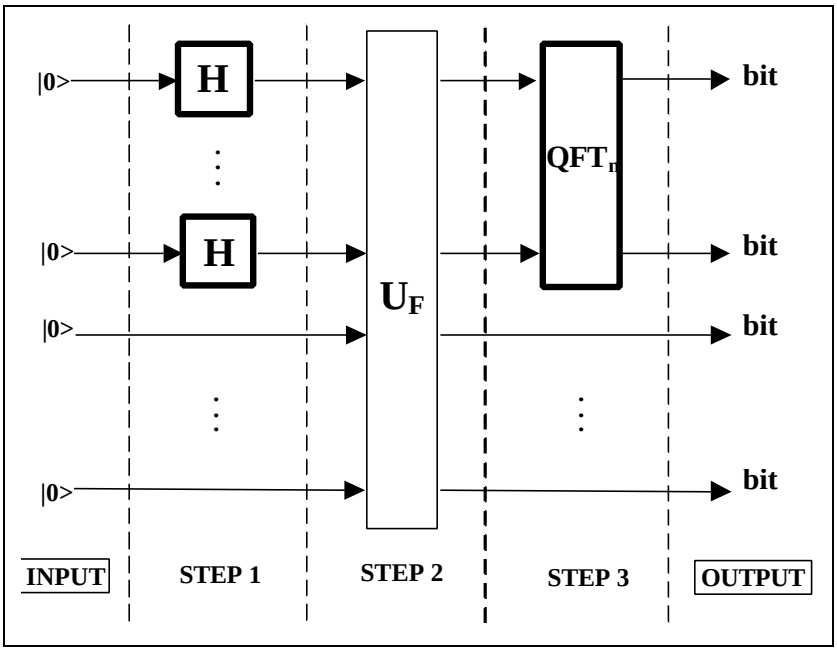


Figure 1: Circuit of Shor's Quantum Gate

Shor's quantum gate circuit is very similar to Simon's quantum gate circuit, unless for the last operator. In Shor algorithm we applied operator ${}^nH$ to the first n vectors exiting from U_F. This operator was responsible of the annihilation of some cells in the first column of the final gate and this effect produced a special superposition of basis vectors as output. Measuring this superposition we obtained some coefficient vectors and we built an equation system in order to find s. We said that the final operator produced interference among different basis vectors.
In Shor's algorithm the interference operator isn't ${}^nH$ any more, since the main diagonal pattern of U_F is different and we need a different interference operator in order to extract information. This operator is represented by matrix QFT_n, called Quantum Fourier Transform of order n. This operator is non-classical since it maps a basis vector into a complex linear combination of basis vector. In general, a basis vector $|i\rangle$ is mapped into the linear combination $a_1y_1 + ..+a_{2^n}y_{2^n}$ where a_i and a_{i+1} have the same modulus $1/2^{n/2}$ but they are shifted in the phase of $i\cdot(2\pi/2^n)$ starting with $a_1=1/2^{n/2}$. The operator is so defined

QFT_n	$f=0$	$f=2p/2^n$	…	$f=(2^n-1)2p/2^n$
	|0..0>	|0..1>	…	|1..1>
|0..0>	$1/2^{n/2}$	$1/2^{n/2}$	…	$1/2^{n/2}$
|0..1>	$1/2^{n/2}$	$1/2^{n/2}\, e^{J2\pi/2^n}$	…	$1/2^{n/2}\, e^{J(2^n-1)\,2\pi/2^n}$
…	...	...	…	…
|1..1>	$1/2^{n/2}$	$1/2^{n/2}\, e^{J(2^n-1)\,2\pi/2^n}$	…	$1/2^{n/2}\, e^{J(2^n-1)^2\,2\pi/2^n}$

where J is the imaginary unit. Using the rules defined in Part 1 on the circuit of fig.1, we obtain the final general quantum gate in fig.2. We are going to discuss the form of these gate in some special case and then we shall generalise our observations.

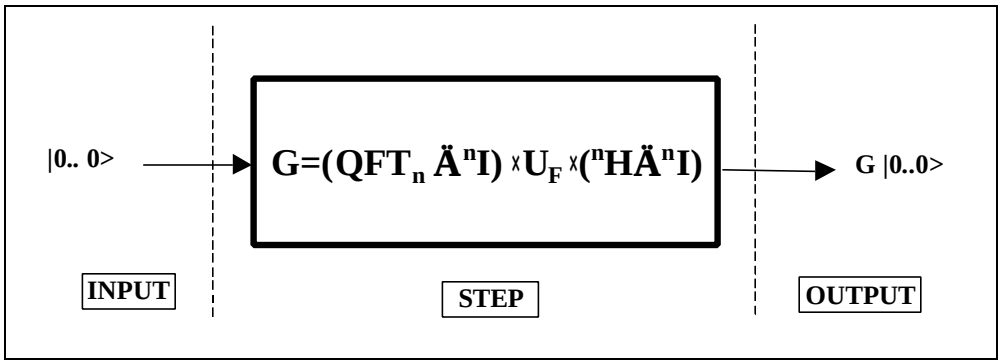


Figure 2: Shor's Quantum Gate

A. Introductory example

In the case n=2, the quantum gate has the following form:

$$G=(QFT_2 \otimes {}^2I) \cdot U_F \cdot ({}^2H \otimes {}^2I)$$

Let's calculate this gate for our introductory example:

$^2H Ä^2 I$	|00>	|01>	|10>	|11>
|00>	$^2I/2$	$^2I/2$	$^2I/2$	$^2I/2$
|01>	$^2I/2$	$-^2I/2$	$^2I/2$	$-^2I/2$
|10>	$^2I/2$	$^2I/2$	$-^2I/2$	$-^2I/2$
|11>	$^2I/2$	$-^2I/2$	$-^2I/2$	$^2I/2$

U_F	|00>	|01>	|10>	|11>
|00>	$I \otimes C$	0	0	0
|01>	0	$C \otimes I$	0	0
|10>	0	0	$I \otimes C$	0
|11>	0	0	0	$C \otimes I$

If n=2, QFT_2 is so calculated:

QFT_2	f=0 |00>	f=p/2 |01>	f=p |10>	f=3p/2 |11>
|00>	1/2	1/2	1/2	1/2
|01>	1/2	J/2	-1/2	-J/2
|10>	1/2	-1/2	1/2	-1/2
|11>	1/2	-J/2	-1/2	J/2

U_Fx(^2H Ä ^2I)	|00>	|01>	|10>	|11>
|00>	$I\otimes C/2$	$I\otimes C/2$	$I\otimes C/2$	$I\otimes C/2$
|01>	$C\otimes I/2$	$-C\otimes I/2$	$C\otimes I/2$	$-C\otimes I/2$
|10>	$I\otimes C/2$	$I\otimes C/2$	$-I\otimes C/2$	$-I\otimes C/2$
|11>	$C\otimes I/2$	$-C\otimes I/2$	$-C\otimes I/2$	$C\otimes I/2$

QFT_2Ä ^2I	|00>	|01>	|10>	|11>
|00>	$^2I/2$	$^2I/2$	$^2I/2$	$^2I/2$
|01>	$^2I/2$	J $^2I/2$	$-^2I/2$	-J $^2I/2$
|10>	$^2I/2$	$-^2I/2$	$^2I/2$	$-^2I/2$
|11>	$^2I/2$	-J $^2I/2$	$-^2I/2$	J $^2I/2$

G	|00>	|01>	|10>	|11>
|00>	$(I\otimes C+C\otimes I)/2$	$(I\otimes C-C\otimes I)/2$	0	0
|01>	0	0	$(I\otimes C+JC\otimes I)/2$	$(I\otimes C-JC\otimes I)/2$
|10>	$(I\otimes C-C\otimes I)/2$	$(I\otimes C+C\otimes I)/2$	0	0
|11>	0	0	$(I\otimes C-JC\otimes I)/2$	$(I\otimes C+JC\otimes I)/2$

Consider the application of G to vector |0000>:

$$G|0000\rangle = |00\rangle\frac{1}{2}(I\otimes C + C\otimes I)|00\rangle + |10\rangle\frac{1}{2}(I\otimes C - C\otimes I)|00\rangle$$

Therefore:

$$G|0000\rangle = \frac{1}{2}(|00\rangle + |10\rangle)I\otimes C|00\rangle + \frac{1}{2}(|00\rangle - |10\rangle)C\otimes I|00\rangle =$$
$$\frac{1}{2}(|00\rangle + |10\rangle)|01\rangle + \frac{1}{2}(|00\rangle - |10\rangle)|10\rangle$$

If we do a measurement of this vector and encode back the first two vectors of dimension 2 in the resulting tensor product vector, the possible results are:

00 with probability 0.5
10 with probability 0.5

The distance between this values is $d=[|10-00|]_{10}=[10]_{10}=2$, where $[s]_{10}$ is the decimal representation of the binary string s. Observe that $N/r=4/2=2$. Therefore $d=N/r$. If we don't know r, then we can calculate it as:

$$r=N/d$$

It might be useful to picture the evolution of the probability amplitude of every basis vector while operator $^2H\otimes^2I$, U_F and $QFT_2\otimes^2I$ are applied in sequence. This is done in fig.3:

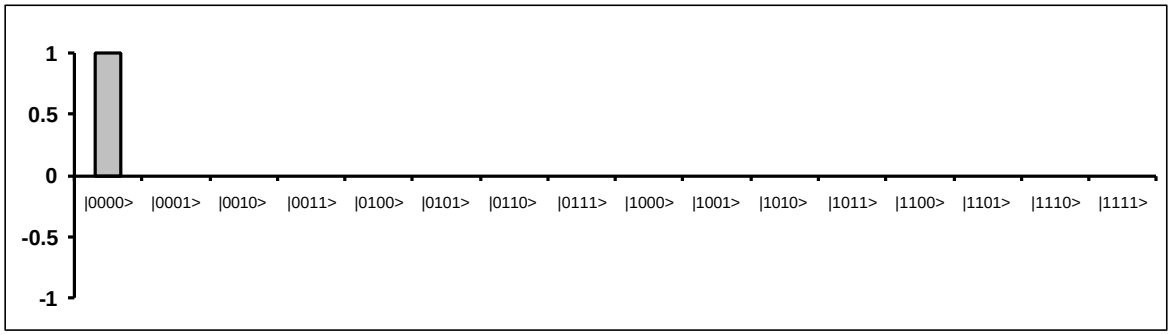


Figure 3.a: Input Probability Amplitudes

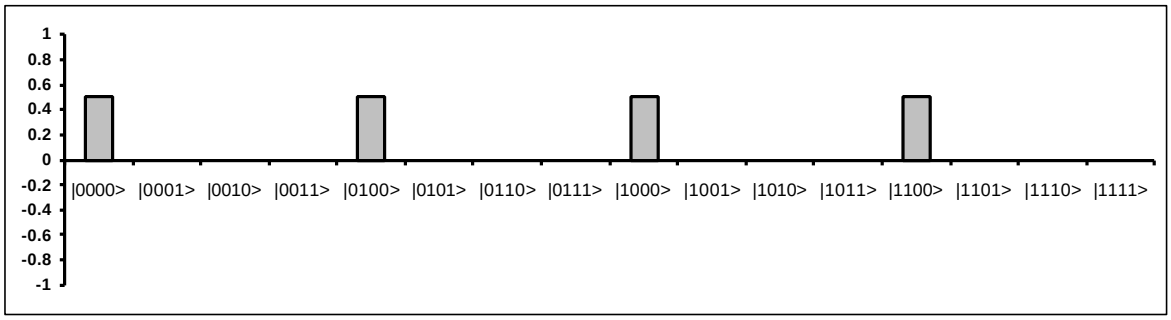


Figure 3.b: Probability Amplitudes after Step 1 (Fig. 1)

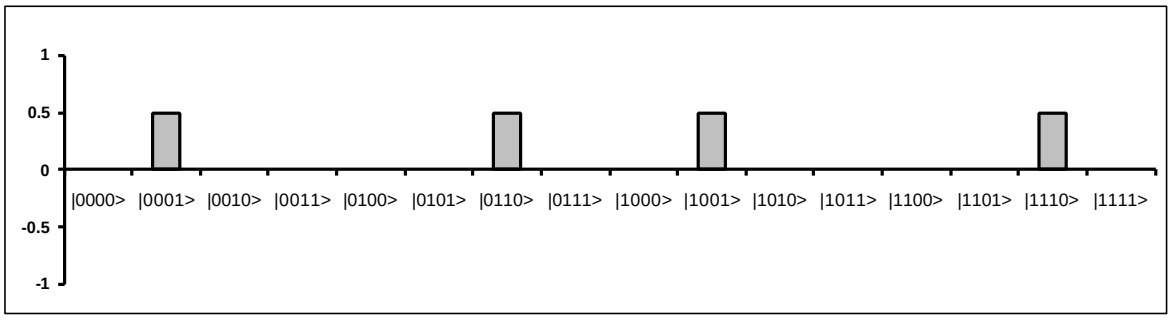


Figure 3.c: Probability Amplitudes after Step 2 (Fig. 1)

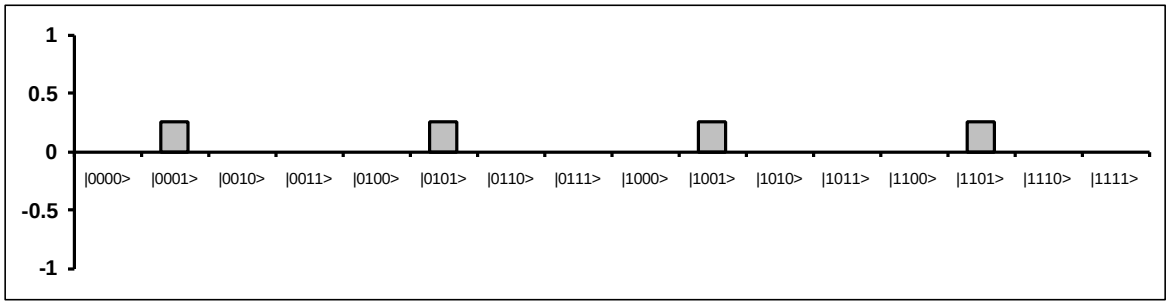


Figure 3.d: Contribute of |0001> through QFT (Real Part)

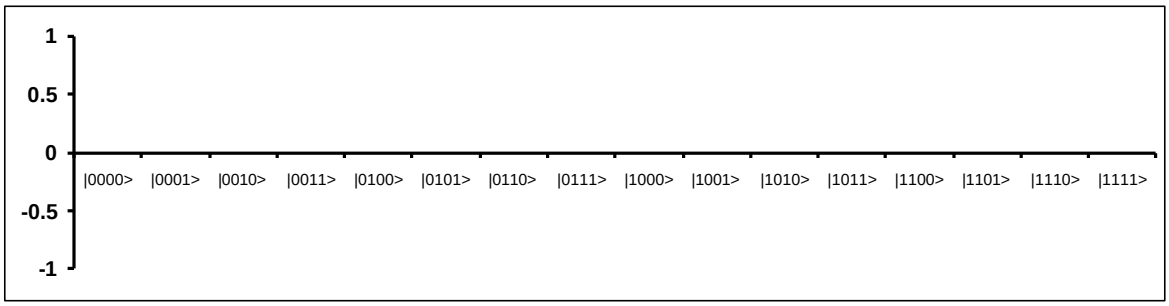


Figure 3.e: Contribute of |0001> through QFT (Imaginary Part)

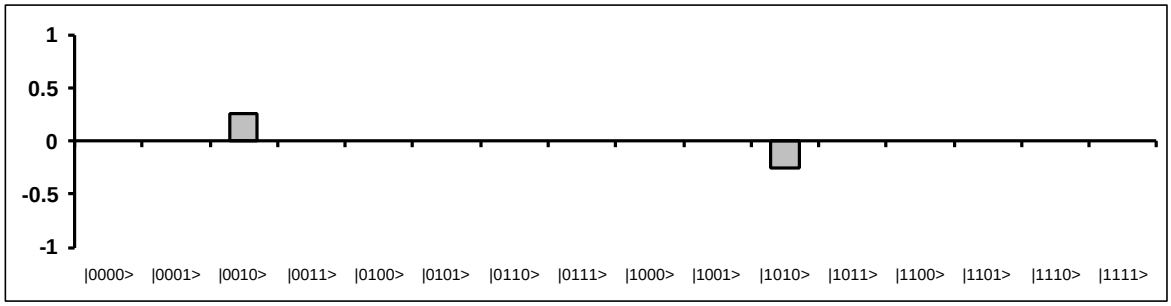


Figure 3.f: Contribute of |0110> through QFT (Real Part)

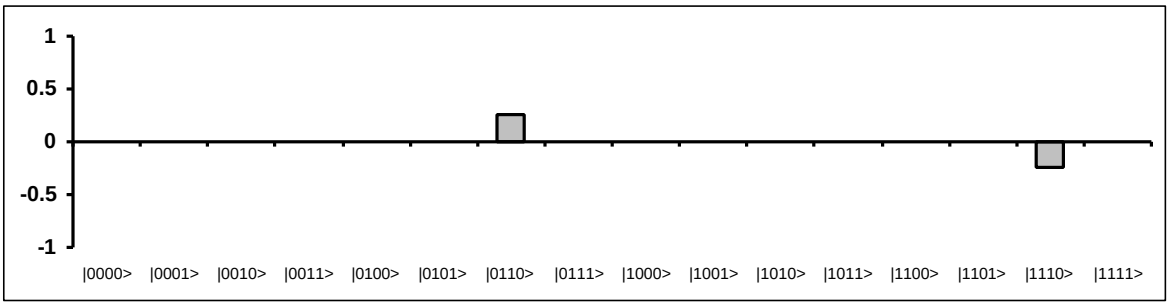


Figure 3.g: Contribute of |0110> through QFT (Imaginary Part)

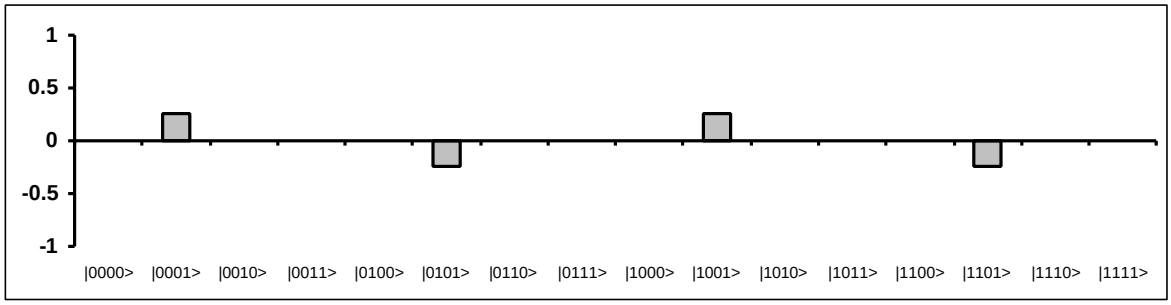


Figure 3.h: Contribute of |1001> through QFT (Real Part)

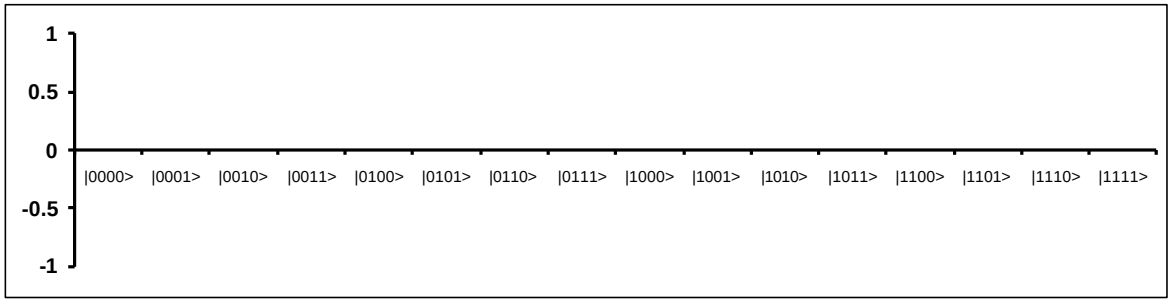


Figure 3.i: Contribute of |1001> through QFT (Imaginary Part)

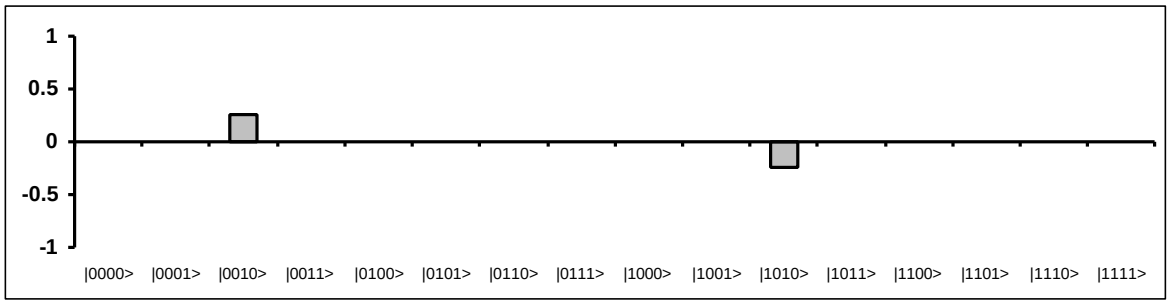


Figure 3.k: Contribute of |1110> through QFT (Real Part)

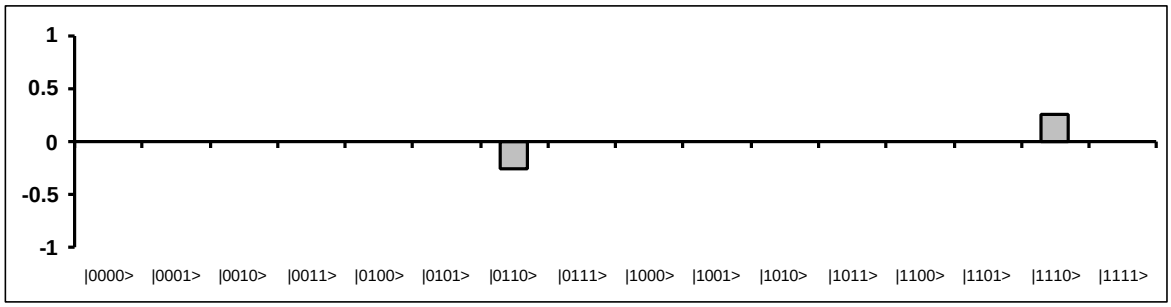


Figure 3.l: Contribute of |1110> through QFT (Imaginary Part)

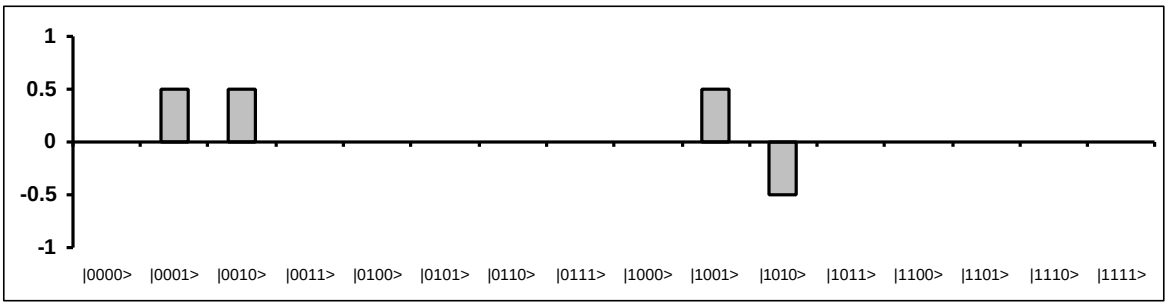


Figure 3.m: Input Probability Amplitudes (Real Part) after Step 3 (Fig. 1)

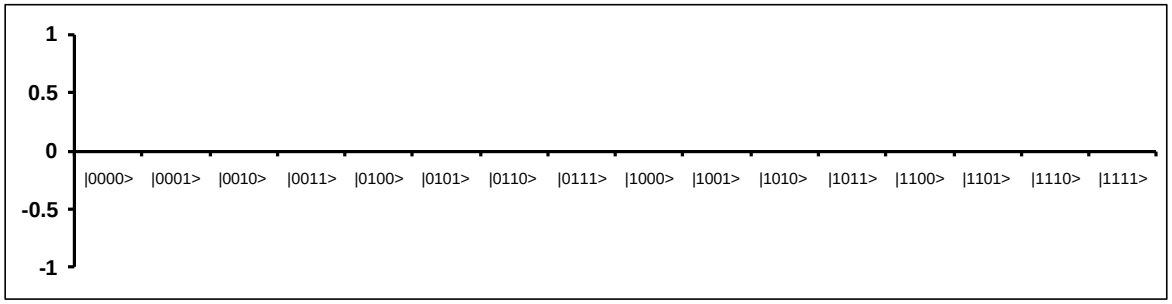


Figure 3.n: Input Probability Amplitudes (Imaginary Part) after Step 3 (Fig. 1)

These pictures should help the reader to understand the role of the operators composing the quantum gate. We have described the probability amplitude contribute of every basis vector in the supeposition obtained after the action of U_F through *QFT*. These contributes are then summed up and interference takes place. In order to understand every passage, it is useful to calculate our results applying matrix $QFT_2 \otimes^2 I$ to the vector superposition of fig.3.c.
We can note that the strength of the algorithm is in the use of *QFT* after correlation have been created by U_F.

B. General case with n=2, r=2

We saw that when n=2, the quantum gate has the following form:

$$G=(QFT_2 \otimes^2 I) \cdot U_F \cdot ({}^2H \otimes^2 I)$$

Using the matrices calculated in the introductory example and recalling U_F in this situation:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_i \in \{I\otimes I, I\otimes C, C\otimes I, C\otimes C\}$ and $M_i = M_j \Leftrightarrow (j=i \vee j=(i+r) \bmod N)$, we find the following generalised form for *G*:

U_F x(^2H Ä ^2I)	|00>	|01>	|10>	|11>
|00>	$M_{00}/2$	$M_{00}/2$	$M_{00}/2$	$M_{00}/2$
|01>	$M_{01}/2$	$-M_{01}/2$	$M_{01}/2$	$-M_{01}/2$
|10>	$M_{10}/2$	$M_{10}/2$	$-M_{10}/2$	$-M_{10}/2$
|11>	$M_{11}/2$	$-M_{11}/2$	$-M_{11}/2$	$M_{11}/2$

$QFT_2 \ddot{A}^2 I$	|00>	|01>	|10>	|11>
|00>	$^2I/2$	$^2I/2$	$^2I/2$	$^2I/2$
|01>	$^2I/2$	J $^2I/2$	$-^2I/2$	-J $^2I/2$
|10>	$^2I/2$	$-^2I/2$	$^2I/2$	$-^2I/2$
|11>	$^2I/2$	-J $^2I/2$	$-^2I/2$	J $^2I/2$

G	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{01}+M_{10}+M_{11})/4$	$(M_{00}-M_{01}+M_{10}-M_{11})/4$	$(M_{00}+M_{01}-M_{10}-M_{11})/4$	$(M_{00}-M_{01}-M_{10}+M_{11})/4$
|01>	$(M_{00}+JM_{01}-M_{10}-JM_{11})/4$	$(M_{00}-JM_{01}-M_{10}+JM_{11})/4$	$(M_{00}+JM_{01}+M_{10}+JM_{11})/4$	$(M_{00}-JM_{01}+M_{10}-JM_{11})/4$
|10>	$(M_{00}-M_{01}+M_{10}-M_{11})/4$	$(M_{00}+M_{01}+M_{10}+M_{11})/4$	$(M_{00}-M_{01}-M_{10}+M_{11})/4$	$(M_{00}+M_{01}-M_{10}-M_{11})/4$
|11>	$(M_{00}-JM_{01}-M_{10}+JM_{11})/4$	$(M_{00}+JM_{01}-M_{10}-JM_{11})/4$	$(M_{00}-JM_{01}+M_{10}-JM_{11})/4$	$(M_{00}+JM_{01}+M_{10}+JM_{11})/4$

If r=2, like in our introductory example, then $M_{00}=M_{10}\neq M_{01}=M_{11}$. This means:

G	|00>	|01>	|10>	|11>
|00>	$(M_{00}+M_{01})/2$	$(M_{00}-M_{01})/2$	0	0
|01>	0	0	$(M_{00}+JM_{01})/2$	$(M_{00}-JM_{01})/2$
|10>	$(M_{00}-M_{01})/2$	$(M_{00}+M_{01})/2$	0	0
|11>	0	0	$(M_{00}-JM_{01})/2$	$(M_{00}+JM_{01})/2$

Consider the application of G to vector |0000>:

$$G|0000\rangle = |00\rangle\frac{1}{2}(M_{00}+M_{01})|00\rangle + |10\rangle\frac{1}{2}(M_{00}-M_{01})|00\rangle$$

If we do a measurement of this vector and encode back the first two vectors of dimension 2 (in the resulting tensor product) into their binary labels, then the possible results are:

00 with probability 0.5
10 with probability 0.5

These are the same results we obtained for our introductory example and the same conclusions hold.

C. General case

We saw that, in the general case, operator U_F is defined as:

U_F	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	…	0
|0..1>	0	$M_{0..1}$	…	0
…	…	...	…	...
|1..1>	0	0	0	$M_{1..1}$

where $M_i = P_1 \otimes .. \otimes P_n$, $P_k \in \{I, C\}$, $k=1,..,n$ and $M_i = M_j \Leftrightarrow (j=i \vee j=(i+r) \bmod N)$.
Let's calculate the gate G in this general situation:

^nH Ä ^nI	|0..0>	|0..1>	…	|j>	…	|1..1>
|0..0>	$^nI/2^{n/2}$	$^nI/2^{n/2}$	…	$^nI/2^{n/2}$	…	$^nI/2^{n/2}$
|0..1>	$^nI/2^{n/2}$	$-^nI/2^{n/2}$	…	$(-1)^{(0..1)\cdot j}(^nI/2^{n/2})$	…	$-^nI/2^{n/2}$
…	…	…	…	…	…	…
|i>	$^nI/2^{n/2}$	$(-1)^{i\cdot(0..1)}(^nI/2^{n/2})$	…	$(-1)^{i\cdot j}(^nI/2^{n/2})$	…	$(-1)^{i\cdot(1..1)}(^nI/2^{n/2})$
…	…	…	…	…	…	…
|1..1>	$^nI/2^{n/2}$	$-^nI/2^{n/2}$	…	$(-1)^{(1..1)\cdot j}(^nI/2^{n/2})$	…	$(-1)^{(1..1)\cdot(1..1)}(^nI/2^{n/2})$

U_F ¤ (^nH Ä ^nI)	|0..0>	…	|j>	…	|1..1>
|0..0>	$M_{0..0}/2^{n/2}$	…	$M_{0..0}/2^{n/2}$	…	$M_{0..0}/2^{n/2}$
…	…	…	…	…	…
|i>	$M_i/2^{n/2}$	…	$(-1)^{i\cdot j} M_i/2^{n/2}$	…	$(-1)^{i\cdot(1..1)} M_i/2^{n/2}$
…	…	…	…	…	…
|1..1>	$M_{1..1}/2^{n/2}$	…	$(-1)^{(1..1)\cdot j} M_{1..1}/2^{n/2}$	…	$(-1)^{(1..1)\cdot(1..1)} M_{1..1}/2^{n/2}$

Observe that:

$$[QFT]_{i,j} = \frac{1}{2^{n/2}} e^{J[i]_{10}\cdot\frac{[j]_{10}\cdot 2p}{2^n}}$$

where $[i]_{10}$ and $[j]_{10}$ are the decimal representations of binary strings i and j. Therefore:

QFT_nÄ ^nI	|0..0>	…	|j>	…	|1..1>
|0..0>	$^nI/2^{n/2}$	…	$^nI/2^{n/2}$	…	$^nI/2^{n/2}$
…	...	…	…	…	…
|i>	$^nI/2^{n/2}$	…	$^nI/2^{n/2} e^{J[i]_{10}\cdot[j]_{10}2\pi/2^n}$	…	$^nI/2^{n/2} e^{J[i]_{10}\cdot(2^n-1)\,2\pi/2^n}$
…	…	…	…	…	…
|1..1>	$^nI/2^{n/2}$	…	$^nI/2^{n/2} e^{J(2^n-1)\cdot[j]_{10}2\pi/2^n}$	…	$^nI/2^{n/2} e^{J(2^n-1)^2\,2\pi/2^n}$

The final gate has the following form:

G	|0..0>	…
|0..0>	$1/2^n \sum_{k\in\{0,1\}^n} e^{J\pi\cdot 0\cdot[k]_{10}/2^{n-1}} M_k$	…
…	…	…
|i>	$1/2^n \sum_{k\in\{0,1\}^n} e^{J\pi\cdot[i]_{10}\cdot[k]_{10}/2^{n-1}} M_k$	…
…	…	...
|1..1>	$1/2^n \sum_{k\in\{0,1\}^n} e^{J\pi\cdot(2^n-1)\cdot[k]_{10}/2^{n-1}} M_k$	...

Consider the term:

$$1/2^n \sum_{k\in\{0,1\}^n} e^{J\pi\cdot[i]_{10}\cdot[k]_{10}/2^{n-1}} M_k$$

Since $M_i = P_1 \otimes .. \otimes P_n$, $P_k \in \{I, C\}$, $k=1,..,n$ and and $M_i = M_j \Leftrightarrow (j=i \vee j=(i+r) \bmod N)$ this term may be written as:

$$1/2^n \sum_{h\in R} [e^{J\pi\cdot[i]_{10}\cdot[h]_{10}/2^{n-1}} + e^{J\pi\cdot[i]_{10}\cdot([h]_{10}+1r)/2^{n-1}} + .. + e^{J\pi\cdot[i]_{10}\cdot([h]_{10}+(l_k-1)r)/2^{n-1}}] M_h$$

or:

$$1/2^n \sum_{h\in R} (-1)^{[h]_{10}[i]_{10}/2^{n-1}} [(-1)^{0r[i]_{10}/2^{n-1}} + (-1)^{1r[i]_{10}/2^{n-1}} + .. + (-1)^{(l_h-1)r[i]_{10}/2^{n-1}}] M_h$$

where $R=\{0..0,0..1, .., [r-1]_2\}$. Suppose N is a multiple of r, then $l_h=2^n/r=l$ for every h. Therefore, the previous term can be transformed into:

$$1/2^n \sum_{h\in R} (-1)^{[h]_{10}[i]_{10}/2^{n-1}} [(-1)^{2\cdot 0\cdot[i]_{10}/l} + (-1)^{2\cdot 1\cdot[i]_{10}/l} + .. + (-1)^{2\cdot(l-1)\cdot[i]_{10}/l}] M_h$$

and finally:

$$1/2^n \sum_{h\in R} (-1)^{[h]_{10}[i]_{10}/2^{n-1}} [e^{J\cdot 0\cdot(2\pi[i]_{10}/l)} + e^{J\cdot 1\cdot(2\pi[i]_{10}/l)} + .. + e^{J\cdot(l-1)(2\pi[i]_{10}/l)}] M_h$$

The term:

$$e^{J\cdot 0\cdot(2\pi[i]_{10}/l)} + e^{J\cdot 1\cdot(2\pi[i]_{10}/l)} + .. + e^{J\cdot(l-1)\cdot(2\pi[i]_{10}/l)}$$

is the summation of the l roots of order l of the unity, unless i is a multiple of l. The summation of the roots of a given order of the unity is always null.
So, in the first column of G only those cells whose row label is $|i\rangle$ with i multiple of l are non-null. This means that applying G to vector $|0..0\rangle$, measuring the result and encoding back into their binary values the first n basis vectors of dimension 2 in the resulting tensor product, we obtain only strings i such that $i=m\cdot l$ for some integer m. This means $l \equiv 0 \bmod i$.

5. DECODER

The quantum block, as we did for Simon's algorithm, is repeated several times in order to build a collection of vector $|i\rangle$ such that $l \equiv 0 \bmod i$. Putting these equations in a system and solving it, we obtain the value of l. Since $l=2^n/r$, we calculate $r=2^n/l$.
How many vectors do we need in order to get r? It depends on the technique we use to solve the system. In general, we need to repeat the quantum block a number of time that increases polynomially with n.
If 2^n is not a multiple of r, then $l_h=[2^n/r]$ for some h, $l_h=[2^n/r]+1$ for some other ones. The term $e^{J\cdot 0\,(2\pi[i]_{10}/l_h)} + e^{J\cdot 1(2\pi[i]_{10}/l_h)} + .. + e^{J\cdot(l_h-1)\,(2\pi[i]_{10}/l_h)}$ is not exactly 0 when i isn't a multiple of l_h, although it approximates 0. So, all possible strings may be found as result of measurement, but strings i that don't represent a multiple of $2^n/r$ are less likely to be observed. In order to decrease this probability (and increase the probability of $2^n/r$–multiples) we employ $2n$ input bits for encoding f. This means that more roots of the unity are involved and so a better approximation is reached.

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 6: Grover's Algorithm

1. AIM

As Shor's algorithm is a variant on Simon's algorithm, where the difference is played by the interference block, Grover's algorithm is described here as a variation on Deutsch-Jozsa's algorithm introduced in Part 3.

2. GROVER'S PROBLEM

Grover's problem is so stated:

Input	A function $f:\{0,1\}^n \rightarrow \{0,1\}$ such that $\exists x \in \{0,1\}^n: (f(x)=1 \wedge \forall y \in \{0,1\}^n: x \neq y \Rightarrow f(y)=0)$
Problem	Find x

In Deutsch-Jozsa's algorithm we distinguished two classes of input functions and we were supposed to decide what class the input function belonged to. In this case the problem is in some sense identical in its form, even if it is harder because now we are dealing with 2^n classes of input functions (each function of the kind described constitutes a class).

3. ENCODER

In order to make the discussion more comprehensible, we prefer firstly to consider a special function with $n=2$. Then we discuss the general case with $n=2$ and finally we analyse the general case with $n>0$.

A. Introductory example

Let's consider the case:

$$n = 2 \qquad f(01) = 1$$

In this case f map table is so defined:

x	$f(x)$
00	0
01	1
10	0
11	0

Step 1

Function f is encoded into injective function F, built according to the usual statement:

$$F:\{0,1\}^{n+1}\to\{0,1\}^{n+1}:F(x_0,x_1,y_0)=(x_0,x_1,f(x_0,x_1)\oplus y_0)$$

Then F map table is:

(x_0, x_1, y_0)	$F(x_0, x_1, y_0)$
000	000
010	011
100	100
110	110
001	001
011	010
101	101
111	111

Step 2

Let's now encode F into the map table of U_F using the usual rule:

$$\forall s\in\{0,1\}^{n+1}:\ U_F\,[t(s)]=t[F(s)]$$

where t is the code map defined in Part 1. This means:

$\|x_0\ x_1\ y_0\rangle$	$U_F\ \|x_0\ x_1\ y_0\rangle$
|000>	|000>
|010>	|011>
|100>	|100>
|110>	|110>
|001>	|001>
|011>	|011>
|101>	|101>
|111>	|111>

Step 3

From the map table of U_F we are supposed to calculate the corresponding matrix operator. This matrix is obtained using the rule:

$$[U_F]_{ij}=1\Leftrightarrow U_F|j\rangle=|i\rangle$$

U_F is so calculated:

U_F	|00>	|01>	|10>	|11>
|00>	I	0	0	0
|01>	0	C	0	0
|10>	0	0	I	0
|11>	0	0	0	I

The effect of this matrix is to leave unchanged the first and the second input basis vectors of the input tensor product, flipping the third one when the first vector is |0> and the second is |1>. This agrees with the constraints on U_F stated above.

B. General case with *n*=2

Let's now take into consideration the more general case:

$$n = 2 \qquad f(\underline{x}) = 1$$

The corresponding matrix operator is:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

with $M_{\underline{x}}=C \wedge \forall i \neq \underline{x}: M_i=I$.

C. General case

It is fairly natural to generalise operator U_F from the case n=2 to the case n>1. In fact, we always find operator C on the main diagonal of the block matrix, in correspondence of the celled labelled by vector $|\underline{x}>$, where $\underline{x}$ is the binary string having image one by f. Therefore:

U_F	|00>	|01>	…	|11>
|00>	M_{00}	0	…	0
|01>	0	M_{01}	…	0
…	…	…	…	…
|11>	0	0	…	M_{11}

with $M_{\underline{x}}=C \wedge \forall i \neq \underline{x}: M_i=I$.

4. QUANTUM BLOCK

Matrix U_F, the output of the encoder, is embedded into the quantum gate. We describe this gate using a quantum circuit:

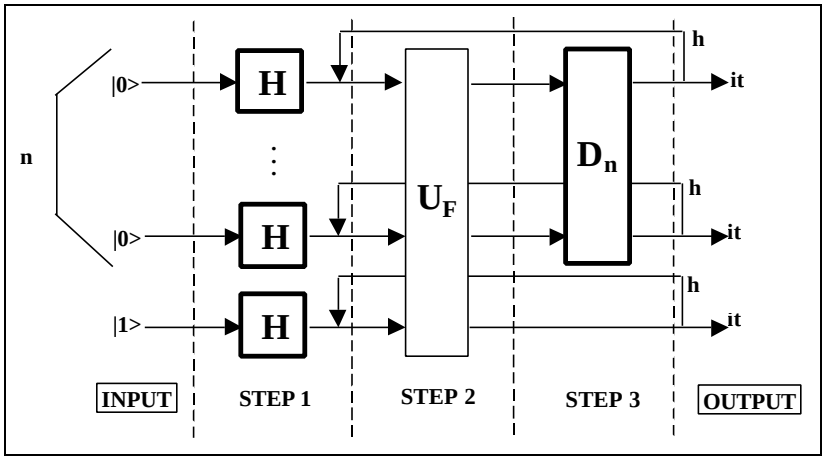


Figure 1: Circuit of Grover's Quantum Gate

Operator D_n is called diffusion matrix of order n and it is responsible of interference in this algorithm. It plays the same role as QFT_n in Shor's algorithm and of ${}^nH$ in Deutsch-Jozsa's and Simon's algorithms. This matrix is defined in this way:

D_n	|0..0>	|0..1>	…	|*i*>	…	|1..0>	|1..1>
|0..0>	$\mathbf{-1+1/2^{n-1}}$	$1/2^{n-1}$	…	$1/2^{n-1}$	…	$1/2^{n-1}$	$1/2^{n-1}$
|0..1>	$1/2^{n-1}$	$\mathbf{-1+1/2^{n-1}}$	…	$1/2^{n-1}$	…	$1/2^{n-1}$	$1/2^{n-1}$
…	…	…	…	…	…	…	…
|*i*>	$1/2^{n-1}$	$1/2^{n-1}$	…	$\mathbf{-1+1/2^{n-1}}$	…	$1/2^{n-1}$	$1/2^{n-1}$
…	…	…	…	…	…	…	…
|1..0>	$1/2^{n-1}$	$1/2^{n-1}$	…	$1/2^{n-1}$	…	$\mathbf{-1+1/2^{n-1}}$	$1/2^{n-1}$
|1..1>	$1/2^{n-1}$	$1/2^{n-1}$	…	$1/2^{n-1}$	…	$1/2^{n-1}$	$\mathbf{-1+1/2^{n-1}}$

Using rule 3 (Part 1), we compile the previous circuit into the following:

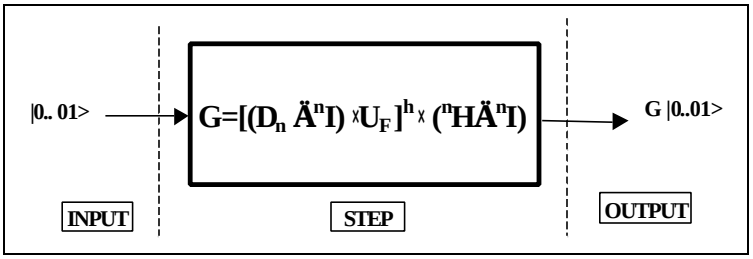


Figure 2: Grover's Quantum Gate

A. Introductory example

In the introductory example we dealt above, U_F had the following form:

U_F	|00>	|01>	|10>	|11>
|00>	I	0	0	0
|01>	0	C	0	0
|10>	0	0	I	0
|11>	0	0	0	I

Let's calculate the quantum gate $G=[(D_2\otimes I)\cdot U_F]^h \cdot ({}^{2+1}H)$ in this case:

${}^3H$	|00>	|01>	|10>	|11>
|00>	$H/2$	$H/2$	$H/2$	$H/2$
|01>	$H/2$	$-H/2$	$H/2$	$-H/2$
|10>	$H/2$	$H/2$	$-H/2$	$-H/2$
|11>	$H/2$	$-H/2$	$-H/2$	$H/2$

D_2ÄI	|00>	|01>	|10>	|11>
|00>	$-I/2$	$I/2$	$I/2$	$I/2$
|01>	$I/2$	$-I/2$	$I/2$	$I/2$
|10>	$I/2$	$I/2$	$-I/2$	$I/2$
|11>	$I/2$	$I/2$	$I/2$	$-I/2$

U_F ¤ ${}^3H$	|00>	|01>	|10>	|11>
|00>	$H/2$	$H/2$	$H/2$	$H/2$
|01>	$CH/2$	$-CH/2$	$CH/2$	$-CH/2$
|10>	$H/2$	$H/2$	$-H/2$	$-H/2$
|11>	$H/2$	$-H/2$	$-H/2$	$H/2$

Choosing h=1, we obtain:

G	|00>	|01>	|10>	|11>
|00>	$(C+I)H/4$	$(-C-I)H/4$	$(C-3I)H/4$	$(-C-I)H/4$
|01>	$(-C+3I)H/4$	$(C+I)H/4$	$(-C-I)H/4$	$(C+I)H/4$
|10>	$(C+I)H/4$	$(-C-I)H/4$	$(C+I)H/4$	$(-C+3I)H/4$
|11>	$(C+I)H/4$	$(-C+3I)H/4$	$(C+I)H/4$	$(-C-I)H/4$

Now, consider the application of G to vector |001>:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|01\rangle \otimes (-C+3I)H|1\rangle +$$
$$\frac{1}{4}|10\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|11\rangle \otimes (C+I)H|1\rangle$$

Let's calculate the operator (-*C*+3*I*)*H*/4. Then:

-C*+3*I	|0>	|1>
|0>	3	-1
|1>	-1	3

***(-C*+3*I*)*H*/4**	|0>	|1>
|0>	$1/2^{3/2}$	$1/2^{1/2}$
|1>	$1/2^{3/2}$	$-1/2^{1/2}$

Therefore:

$$\frac{1}{4}(-C+3I)H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Let's calculate the operator (*C*+*I*)*H*/4. Then:

C*+*I	|0>	|1>
|0>	1	1
|1>	1	1

***(C*+*I*)*H*/4**	|0>	|1>
|0>	$1/2^{3/2}$	0
|1>	$1/2^{3/2}$	0

Therefore:

$$\frac{1}{4}(C+I)H|1\rangle = 0$$

This means that |001> is mapped into vector |01>(|0>-|1>)/$2^{1/2}$. Taking the binary values of the first two vectors of dimension 2, we find $\underline{x}$.

It might be useful to picture the evolution of the probability amplitude of every basis vector while operator ^3H, U_F and $D_2 \otimes I$ are applied in sequence. This is done in fig.3:

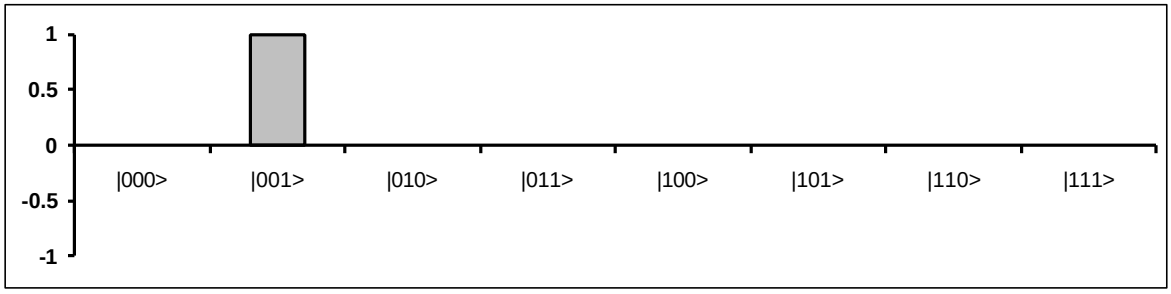


Figure 3.a: Input Probability Amplitudes

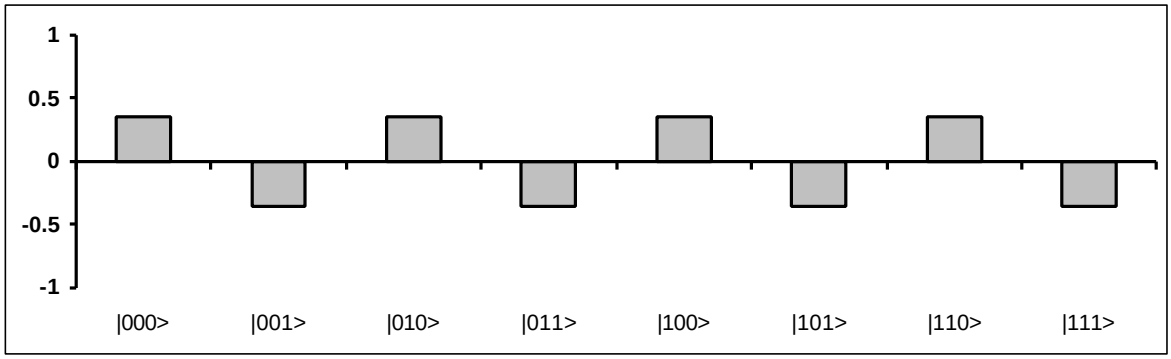


Figure 3.b: Probability Amplitudes after Step 1 (Fig. 1)

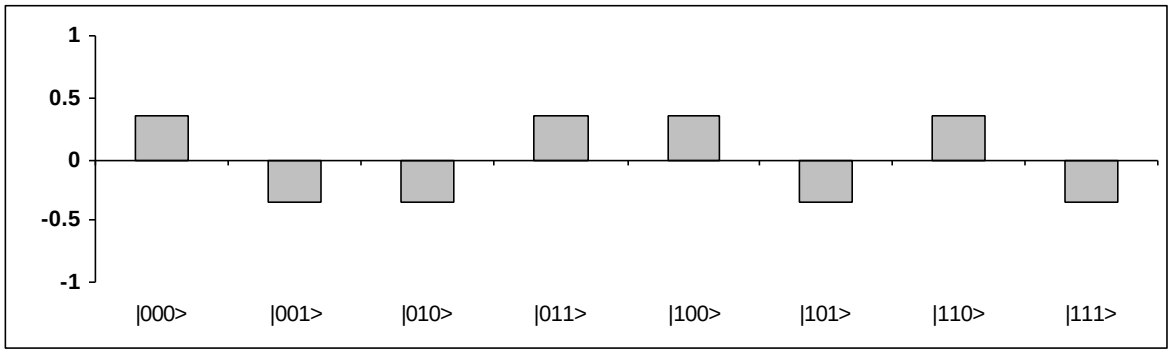


Figure 3.c: Probability Amplitudes after Step 2 (Fig. 1)

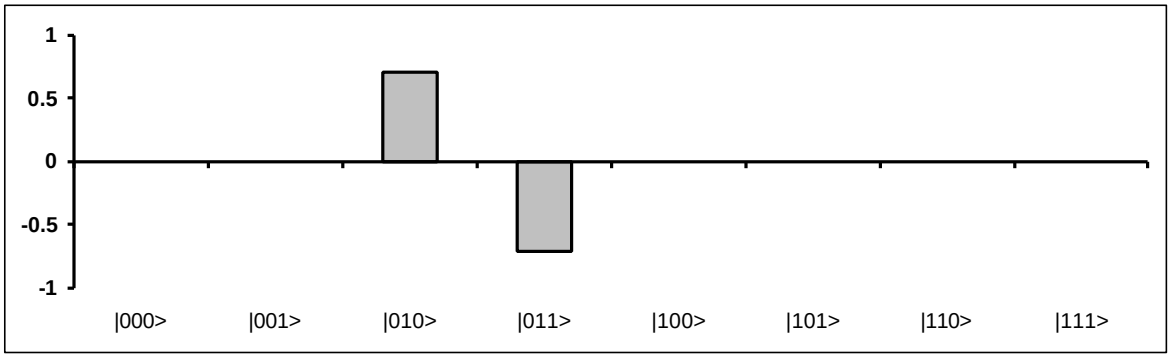


Figure 3.d: Probability Amplitudes after Step 3 (Fig. 1)

Operator ${}^3H$ puts the initial canonical basis vector |001> into a superposition of all basis vectors with the same (real) coefficients in modulus, but with positive sign if the last vector is |0>, negative otherwise. Operator U_F creates correlation: it flips the third vector if the first two vector are |0> and |1>. Finally, $D_2 \otimes I$ produces interference: for every basis vector $|\underline{x}_0 x_1 y_0\rangle$ it calculates its output probability amplitude $a'_{x_0 x_1 y_0}$ by inverting its initial probability amplitude $a_{x_0 x_1 y_0}$ and summing the double of the mean $\underline{a}_{y_0}$ of the probability amplitude of all vectors in the form $|x_0 x_1 \underline{y}_0\rangle$. In our example $\underline{a}_0 = 1/(4 \cdot 2^{1/2})$, $\underline{a}_1 = -1/(4 \cdot 2^{1/2})$. Take, for instance, basis vector |000>. Then $a'_{000} = -a_{000} + 2\underline{a}_0 = -1/(2 \cdot 2^{1/2}) + 2/(4 \cdot 2^{1/2}) = 0$.

B. General case with *n*=2

In general, if $n=2$, U_F has the following form:

U_F	|00>	|01>	|10>	|11>
|00>	M_{00}	0	0	0
|01>	0	M_{01}	0	0
|10>	0	0	M_{10}	0
|11>	0	0	0	M_{11}

where $M_{\underline{x}} = C \wedge \forall i \neq \underline{x} : M_i = I\ (\underline{x}, i \in : \{0,1\}^n\ )$.
Let's calculate the quantum gate $G = (D_2 \otimes I) \cdot U_F \cdot ({}^{2+1}H)$ in this general case:

$U_F \times {}^3H$	|00>	|01>	|10>	|11>
|00>	$M_{00}H/2$	$M_{00}H/2$	$M_{00}H/2$	$M_{00}H/2$
|01>	$M_{01}H/2$	$-M_{01}H/2$	$M_{01}H/2$	$-M_{01}H/2$
|10>	$M_{10}H/2$	$M_{10}H/2$	$-M_{10}H/2$	$-M_{10}H/2$
|11>	$M_{11}H/2$	$-M_{11}H/2$	$-M_{11}H/2$	$M_{11}H/2$

G	|00>	|01>	|10>	|11>
|00>	$(-M_{00}+M_{01}+M_{10}+M_{11})H/4$	$(-M_{00}-M_{01}+M_{10}-M_{11})H/4$	$(-M_{00}+M_{01}-M_{10}-M_{11})H/4$	$(-M_{00}-M_{01}-M_{10}+M_{11})H/4$
|01>	$(M_{00}-M_{01}+M_{10}+M_{11})H/4$	$(M_{00}+M_{01}+M_{10}-M_{11})H/4$	$(M_{00}-M_{01}-M_{10}-M_{11})H/4$	$(M_{00}+M_{01}-M_{10}+M_{11})H/4$
|10>	$(M_{00}+M_{01}-M_{10}+M_{11})H/4$	$(M_{00}-M_{01}-M_{10}-M_{11})H/4$	$(M_{00}+M_{01}+M_{10}-M_{11})H/4$	$(M_{00}-M_{01}+M_{10}+M_{11})H/4$
|11>	$(M_{00}+M_{01}+M_{10}-M_{11})H/4$	$(M_{00}-M_{01}+M_{10}+M_{11})H/4$	$(M_{00}+M_{01}-M_{10}+M_{11})H/4$	$(M_{00}-M_{01}-M_{10}-M_{11})H/4$

Now, consider the application of G to vector |001>:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (-M_{00} + M_{01} + M_{10} + M_{11})H|1\rangle + \frac{1}{4}|01\rangle \otimes (M_{00} - M_{01} + M_{10} + M_{11})H|1\rangle +$$
$$\frac{1}{4}|10\rangle \otimes (M_{00} + M_{01} - M_{10} + M_{11})H|1\rangle + \frac{1}{4}|11\rangle \otimes (M_{00} + M_{01} + M_{10} - M_{11})H|1\rangle$$

Consider the following cases:

$\underline{x}$=00:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (-C+3I)H|1\rangle + \frac{1}{4}|01\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|10\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|11\rangle \otimes (C+I)H|1\rangle = |00\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

$\underline{x}$=01:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|01\rangle \otimes (-C+3I)H|1\rangle + \frac{1}{4}|10\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|11\rangle \otimes (C+I)H|1\rangle = |01\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

$\underline{x}$=10:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|01\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|10\rangle \otimes (-C+3I)H|1\rangle + \frac{1}{4}|11\rangle \otimes (C+I)H|1\rangle = |10\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

$\underline{x}$=11:

$$G|001\rangle = \frac{1}{4}|00\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|01\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|10\rangle \otimes (C+I)H|1\rangle + \frac{1}{4}|11\rangle \otimes (-C+3I)H|1\rangle = |11\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

This means that if we measure the output vector and encode back the first two basis vectors of dimension 2 in the resulting tensor product, we get the following results:

$\underline{x}$	Result	Probability
00	00	1
01	01	1
10	10	1
11	11	1

D. General case (n>0)

In the general case n>0, U_F has the following form:

U_F	|0..0>	|0..1>	…	|1..1>
|0..0>	$M_{0..0}$	0	0	0
|0..1>	0	$M_{0..1}$	0	0
…	…	…	…	…
|1..1>	0	0	0	$M_{1..1}$

where $M_{\underline{x}} = C \wedge \forall i \neq \underline{x}: M_i = I\ (\underline{x}, i \in : \{0,1\}^n\ )$.
Let's calculate the quantum gate $G=(D_n \otimes I)^h \cdot U_F \cdot (^{n+1}H)$:

^{n+1}H	|0..0>	…	|j>	…	|1..1>
|0..0>	$H/2^{n/2}$	…	$H/2^{n/2}$	…	$H/2^{n/2}$
…	…	…	…	…	…
|i>	$H/2^{n/2}$	…	$(-1)^{i \cdot j} H/2^{n/2}$	…	$(-1)^{i \cdot (1..1)} H/2^{n/2}$
…	…	…	…	…	…
|11>	$H/2^{n/2}$	…	$(-1)^{(1..1) \cdot j} H/2^{n/2}$	…	$(-1)^{(1..1) \cdot (1..1)} H/2^{n/2}$

D_nÄI	|0..0>	|0..1>	…	|i>	…	|1..0>	|1..1>
|0..0>	$-I+I/2^{n-1}$	$I/2^{n-1}$	…	$I/2^{n-1}$	…	$I/2^{n-1}$	$I/2^{n-1}$
|0..1>	$I/2^{n-1}$	$-I+I/2^{n-1}$	…	$I/2^{n-1}$	…	$I/2^{n-1}$	$I/2^{n-1}$
…	…	…	…	…	…	…	…
|i>	$I/2^{n-1}$	$I/2^{n-1}$	…	$-I+I/2^{n-1}$	…	$I/2^{n-1}$	$I/2^{n-1}$
…	…	…	…	…	…	…	…
|1..0>	$I/2^{n-1}$	$I/2^{n-1}$	…	$I/2^{n-1}$	…	$-I+I/2^{n-1}$	$I/2^{n-1}$
|1..1>	$I/2^{n-1}$	$I/2^{n-1}$	…	$I/2^{n-1}$	…	$I/2^{n-1}$	$-I+I/2^{n-1}$

U_F x ^{n+1}H	|0..0>	…	|j>	…	|1..1>
|0..0>	$M_{0..0} H/2^{n/2}$	…	$M_{0..0} H/2^{n/2}$	…	$M_{0..0} H/2^{n/2}$
…	…	…	…	…	…
|i>	$M_i H/2^{n/2}$	…	$(-1)^{i \cdot j} M_i H/2^{n/2}$	…	$(-1)^{i \cdot (1..1)} M_i H/2^{n/2}$
…	…	…	…	…	…
|1..1>	$M_{1..1} H/2^{n/2}$	…	$(-1)^{(1..1) \cdot j} M_{1..1} H/2^{n/2}$	…	$(-1)^{(1..1) \cdot (1..1)} M_{1..1} H/2^{n/2}$

Now, suppose h=1. Then:

$G_{h=1}$	|0..0>	…
|0..0>	$(-M_{0..0} + \Sigma_{j \in \{0,1\}^n} M_i/2^{n-1}) H/2^{n/2}$	…
…	…	…
|i>	$(-M_i + \Sigma_{j \in \{0,1\}^n} M_i/2^{n-1}) H/2^{n/2}$	…
…	…	…
|1..1>	$(-M_{1..1} + \Sigma_{j \in \{0,1\}^n} M_i/2^{n-1}) H/2^{n/2}$	…

Being $M_{\underline{x}} = C$ and $\forall i \neq \underline{x}: M_i = I$, this column may be written as:

$G_{h=1}$	$\|0..0\rangle$	…
$\|0..0\rangle$	$(-I+\Sigma_{j\in\{0,1\}^n-\{\underline{x}\}} I/2^{n-1} + C/2^{n-1})H/2^{n/2}$	…
…	…	…
$\|\underline{x}\rangle$	$(-C+\Sigma_{j\in\{0,1\}^n-\{\underline{x}\}} I/2^{n-1} + C/2^{n-1})H/2^{n/2}$	…
…	…	…
$\|1..1\rangle$	$(-I+\Sigma_{j\in\{0,1\}^n-\{\underline{x}\}} I/2^{n-1} + C/2^{n-1})H/2^{n/2}$	…

and so:

$G_{h=1}$	$\|0..0\rangle$	…
$\|0..0\rangle$	$\{[-1+(2^n-1)/2^{n-1}]I + C/2^{n-1}\}H/2^{n/2}$	…
…	…	…
$\|\underline{x}\rangle$	$\{(2^n-1)/2^{n-1}I + [-1+1/2^{n-1}]C\}H/2^{n/2}$	…
…	…	…
$\|1..1\rangle$	$\{[-1+(2^n-1)/2^{n-1}]I + C/2^{n-1}\}H/2^{n/2}$	…

Now, consider to apply matrix operator $\{[-1+(2^n-1)/2^{n-1}]I + C/2^{n-1}\}H/2^{n/2}$ and matrix operator $\{(2^n-1)/2^{n-1}I + [-1+1/2^{n-1}]C\}H/2^{n/2}$ to vector $|1\rangle$:

$$\frac{1}{2^{n/2}}\left\{\left[-1+\frac{2^n-1}{2^{n-1}}\right]I+\frac{1}{2^{n-1}}C\right\}H|1\rangle=\left(-1+\frac{2^n-2}{2^{n-1}}\right)\frac{|0\rangle-|1\rangle}{2^{\frac{(n+1)}{2}}}$$

$$\frac{1}{2^{n/2}}\left\{\frac{2^n-1}{2^{n-1}}I+\left[-1+\frac{1}{2^{n-1}}\right]C\right\}H|1\rangle=\left(+1+\frac{2^n-2}{2^{n-1}}\right)\frac{|0\rangle-|1\rangle}{2^{\frac{(n+1)}{2}}}$$

This means:

$$G_{h=1}|0..01\rangle =$$
$$\left[\left(-1+\frac{2^n-2}{2^{n-1}}\right)|0..0\rangle+\left(-1+\frac{2^n-2}{2^{n-1}}\right)|0..1\rangle+..+\left(+1+\frac{2^n-2}{2^{n-1}}\right)|\underline{x}\rangle+..+\left(-1+\frac{2^n-2}{2^{n-1}}\right)|1..1\rangle\right]\otimes\frac{|0\rangle-|1\rangle}{2^{\frac{(n+1)}{2}}}$$

which can be written as a block vector:

$G_{h=1}\|0..01\rangle$	
$\|0..0\rangle$	$[-1+(2^n-2)/2^{n-1}]\ /2^{n/2}H\|1\rangle$
…	…
$\|\underline{x}\rangle$	$[+1+(2^n-2)/2^{n-1}]\ /2^{n/2}H\|1\rangle$
…	…
$\|1..1\rangle$	$[-1+(2^n-2)/2^{n-1}]\ /2^{n/2}H\|1\rangle$

Now, we imagine to apply operator $(D_n \otimes I) \times U_F$ to a vector in this form:

$\|j\rangle$	
$\|0..0\rangle$	$aH\|1\rangle$
…	…
$\|\underline{x}\rangle$	$bH\|1\rangle$
…	…
$\|1..1\rangle$	$aH\|1\rangle$

where a and b are real number such that $(2^n-1)a^2+\beta^2=1$. The result is:

$U_F \times \|j\rangle$	
$\|0..0\rangle$	$aH\|1\rangle$
…	…
$\|\underline{x}\rangle$	$bCH\|1\rangle$
…	…
$\|1..1\rangle$	$aH\|1\rangle$

$(D_n \ddot{A} I) \times U_F \times \|j\rangle$	
$\|0..0\rangle$	$(-a+\Sigma_{j\in\{0,1\}^n-\{\underline{x}\}} a/2^{n-1}-b/2^{n-1})H\|1\rangle$
…	…
$\|\underline{x}\rangle$	$(+b+\Sigma_{j\in\{0,1\}^n-\{\underline{x}\}} a/2^{n-1}-b/2^{n-1})H\|1\rangle$
…	…
$\|1..1\rangle$	$(-a+\Sigma_{j\in\{0,1\}^n-\{x\}} a/2^{n-1}-b/2^{n-1})H\|1\rangle$

$(D_n \ddot{A} I) \times U_F \times \|j\rangle$	
$\|0..0\rangle$	$\{-a+[(2^n-1)a-b]/2^{n-1}\}H\|1\rangle$
…	…
$\|\underline{x}\rangle$	$\{+b+[(2^n-1)a-b]/2^{n-1}\}H\|1\rangle$
…	…
$\|1..1\rangle$	$\{-a+[(2^n-1)a-b]/2^{n-1}\}H\|1\rangle$

This means that if we start from vector $G_{h=1}|0..01\rangle$, which is in the form considered, and we apply h times operator $(D_n \otimes I) \times U_F$, the coefficients at time t are such that:

$$a_t = 2\frac{(2^n-1)a_{t-1}-b_{t-1}}{2^n} - a_{t-1}$$

$$b_t = 2\frac{(2^n-1)a_{t-1}-b_{t-1}}{2^n} + b_{t-1}$$

So, b increases, a decreases. Consider the vector superposition in fig.4.a:

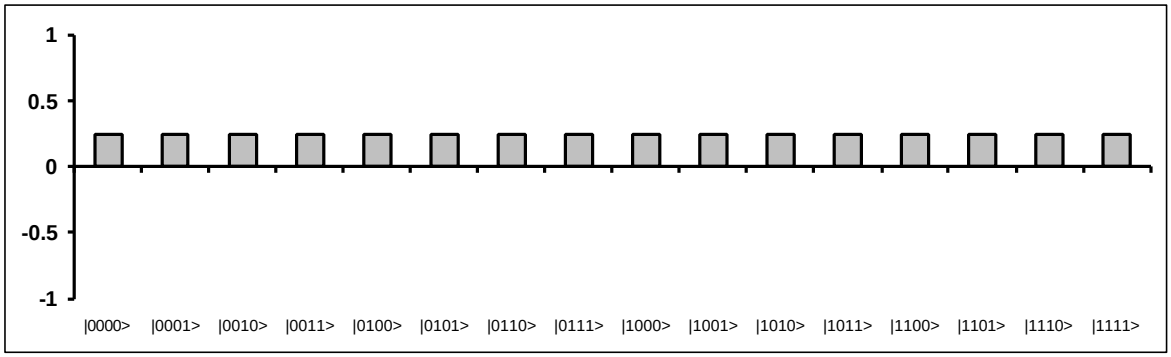


Figure 4.a: Vector Superposition

By applying ${}^4H$ the vector superposition becomes (fig.4.b):

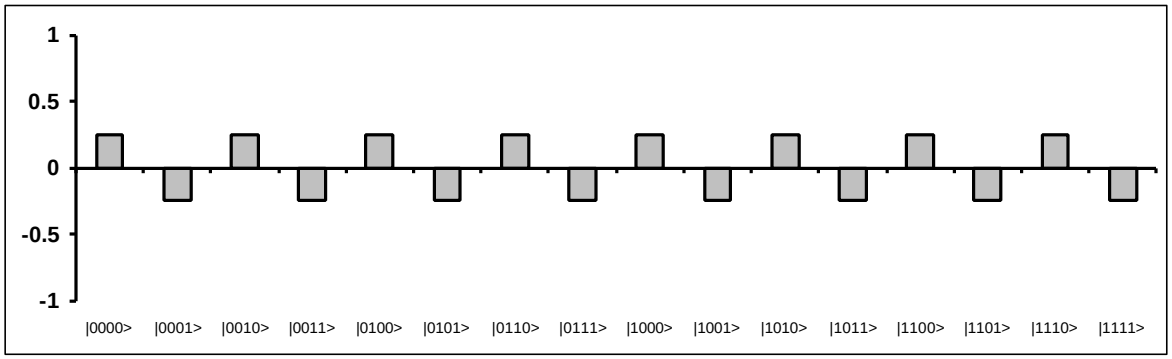


Figure 4.b: Superposition of fig.4.a after ${}^4H$ has been applied

Operator U_F (with $\underline{x}$=001) generates the following vector superposition (fig.4.c):

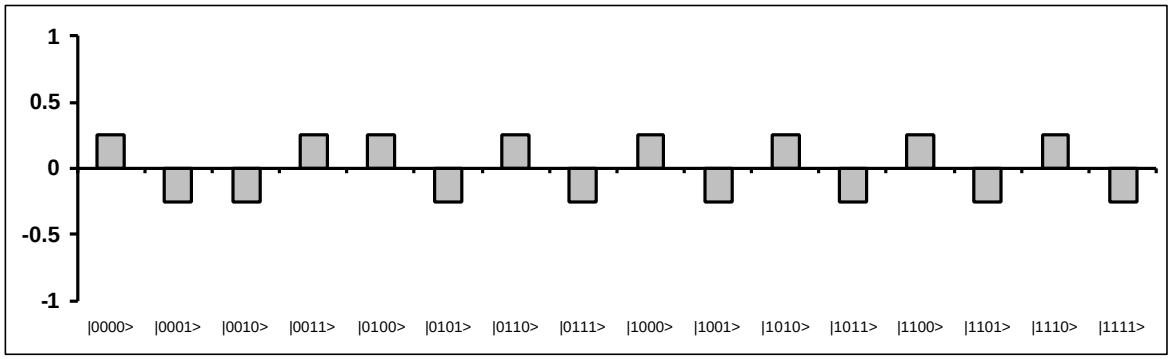


Figure 4.c: U_F (with $\underline{x}$=001) has been applied and entanglement created

Finally, after the action of $D_n \otimes I$ the superposition is:

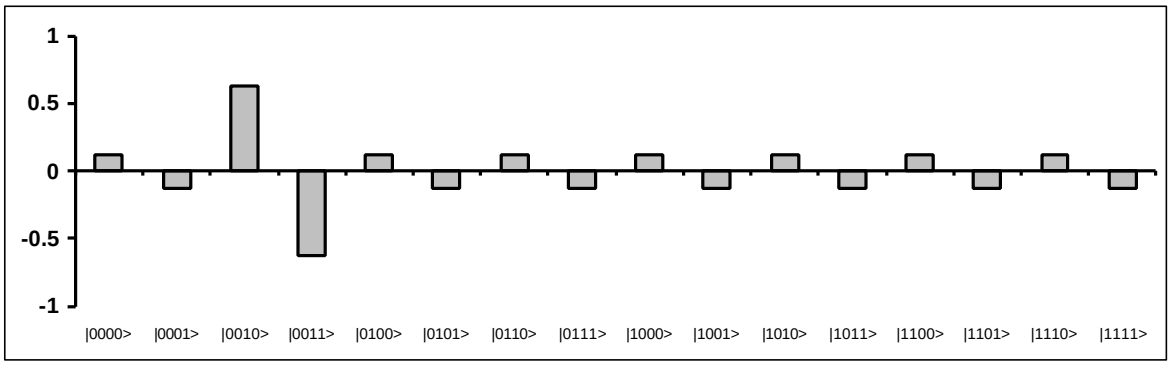


Figure 4.d: $D_n \otimes I$ has acted and interference took place.

Here, the probability amplitudes of non-interesting vectors are not null, but they are very small.
Suppose to apply operator U_F again. The resulting superposition is reported in fig.4.e.

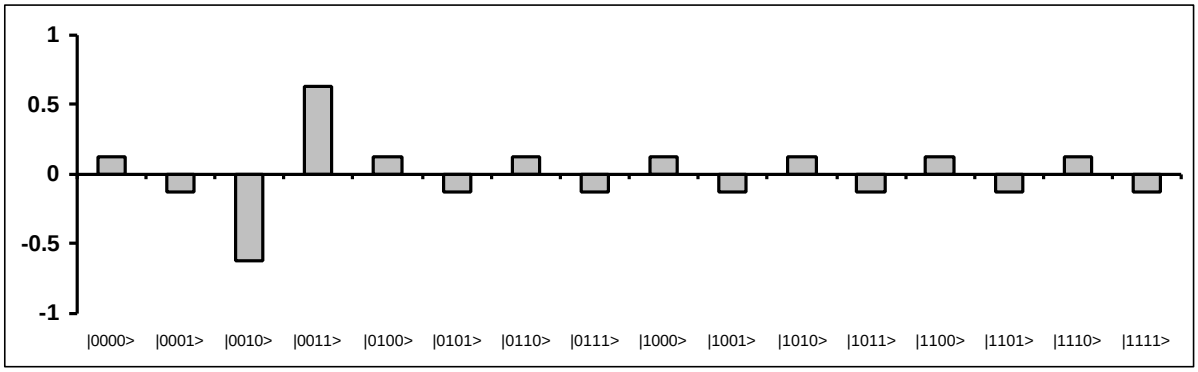


Figure 4.e: U_F is applied a second time

Then, by applying $D_n \otimes I$, we obtain the vector linear combination of fig.4.f.

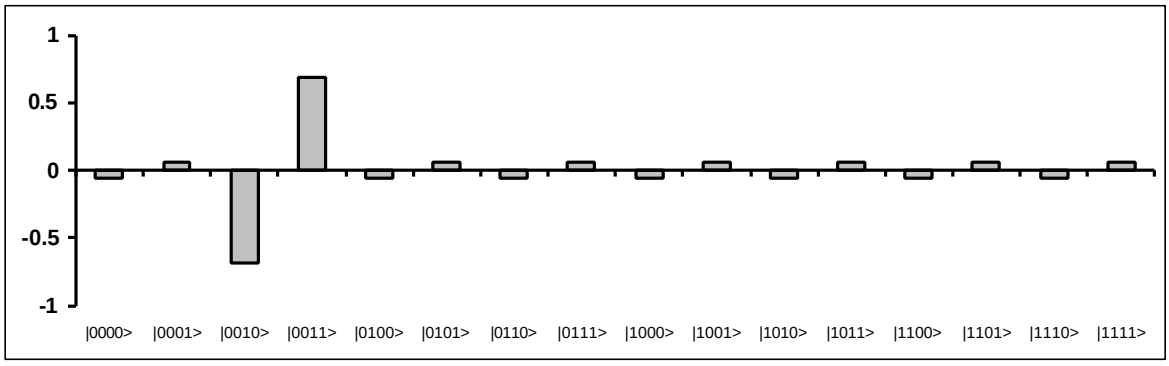


Figure 4.f: $D_n \otimes I$ is applied a second time

We can observe that the probability amplitude of the desired vectors has increased in modulus. This means a greater probability to measure vectors |0010> or |0011>.
If we do measurement after h repetitions of operator $D_{n'}U_F$, what is the probability $P(h)$ to measure vectors $|x\rangle\otimes|0\rangle$ or $|x\rangle\otimes|1\rangle$? We can show that the:

$$P'(h)=O(2^{-n/2})$$

The quantum block is repeated only 1 time with a sufficiently large $h=O(2^{n/2})$.
So, the final collected basis vector is unique.

5. DECODER

As in Deutsch-Jozsa's algorithm, when the output vector from the quantum gate has been measured, we must interpret it in order to find $\underline{x}$.
From the analyses we did above, this step is very simple. In fact, it is sufficient to choose a large h in order to get the searched vector $|\underline{x}\rangle|0\rangle$ or $|\underline{x}\rangle|1\rangle$ with probability near to 1. After getting it we encode back into their binary values the first n basis vector in the resulting tensor product, obtaining string $\underline{x}$ as final answer.

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Part 7: Comparing Quantum Algorithms

1. AIM

We want now to compare the algorithms analysed in the previous parts in order to define a general way to simulate quantum algorithms on classical computers.

2. PROBLEMS AND U_F STRUCTURES

We deal only with the four algorithms discussed in Parts 3, 4, 5 and 6, since Deutsch's algorithm is considered as a special case of Deutsch-Jozsa's algorithm. The four problems are resumed in Tab.1.

We reported the structure of U_F-operators for the general case $n>0$, in order to point out how the algorithms depend on the special structures of these matrices. We should note that every problem is characterised by a special "pattern" on the main diagonal of the block matrix describing U_F.

You see that Deutsch-Jozsa is a decision problem, whereas other problems are search problems. Nevertheless, all problems may be restated in a more general way. In fact, we always distinguish some disjoint classes for the input functions:

- constant and balanced functions in Deutsch-Jozsa's problem;
- functions characterised by a given s in Simon's problem;
- functions characterised by a given period r in Shor's problem;
- functions whose class-property is a binary string $\underline{x}$ in Grover's problem.

Every algorithm is thought in order to decide, given an input function, which class it belongs to. The fact to belong to a given class is in some sense encoded into matrix U_F. The algorithm's aim is to extract this information from U_F and encode it into some final vectors.

Let's observe that Deutsch-Jozsa and Grover's problems consider functions $f:\{0,1\}^n \rightarrow \{0,1\}$, whereas Simon and Shor's problems consider input functions $f:\{0,1\}^n \rightarrow \{0,1\}^n$. In general we talk of functions $f:\{0,1\}^n \rightarrow \{0,1\}^m$ where $m=1$ or $m=n$. In the first case we deal with truth functions, in the second with binary string functions.

<table>
<tr><th rowspan="2">NAME
AND
INPUT
FUNCTION</th><th rowspan="2">KINDS OF INPUT FUNCTIONS</th><th colspan="5">$n>0$</th><th rowspan="2">PROBLEM</th></tr>
<tr><td>U_F</td><td>$|0..0\rangle$</td><td>$|0..1\rangle$</td><td>…</td><td>$|1..1\rangle$</td></tr>
<tr><td>$|0..0\rangle$</td><td>$\mathbf{M_{0..0}}$</td><td>0</td><td>0</td><td>0</td><td rowspan="4">1. $M_i\in\{I, C\}$, $\forall i,j$: $M_i = M_j$
2. $M_i\in\{I, C\}$,
$|\{M_i : M_i = I\}| = |\{M_i : M_i = C\}|$</td></tr>
<tr><td>$|0..1\rangle$</td><td>0</td><td>$\mathbf{M_{0..1}}$</td><td>0</td><td>0</td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…</td><td>…</td></tr>
<tr><td>$|1..1\rangle$</td><td>0</td><td>0</td><td>0</td><td>$\mathbf{M_{1..1}}$</td></tr>
<tr><td>Deutsch-Jozsa

$f:\{0,1\}^n \rightarrow \{0,1\}$</td><td>1. A. $\forall x\in\{0,1\}^n$: $f(x)=0$
B. $\forall x\in\{0,1\}^n$: $f(x)=1$
2. $|\{x\in\{0,1\}^n : f(x)=0\}| = |\{x\in\{0,1\}^n : f(x)=1\}|$</td><td colspan="5">1. $M_i\in\{I, C\}$, $\forall i,j$: $M_i = M_j$
2. $M_i\in\{I, C\}$,
$|\{M_i : M_i = I\}| = |\{M_i : M_i = C\}|$</td><td>Decide if f is constant (1) or balanced (2)</td></tr>
<tr><td>Simon

$f:\{0,1\}^n \rightarrow \{0,1\}^n$</td><td>$\exists s\in\{0,1\}^n - \{0,..,0\}: \forall x,y\in\{0,1\}^n$:
$f(x)=f(y) \Leftrightarrow (x=y \vee x=y\oplus s)$</td><td colspan="5">$M_i = P_1\otimes..\otimes P_n$, $P_k\in\{\mathrm{I}, \mathrm{C}\}$, $k=1,..,n$
and $M_i = M_j \Leftrightarrow (j=i \vee j=i\oplus s)$</td><td>Find s</td></tr>
<tr><td>Shor

$f:\{0,1\}^n \rightarrow \{0,1\}^n$</td><td>$\exists r\in\{0,1\}^n - \{0,..,0\}: \forall x,y\in\{0,1\}^n$:
$f(x)=f(y) \Leftrightarrow (x=y \vee x=(y+r)\mathrm{mod}2^n)$</td><td colspan="5">$M_i = P_1\otimes..\otimes P_n$, $P_k\in\{\mathrm{I}, \mathrm{C}\}$, $k=1,..,n$
and $M_i = M_j \Leftrightarrow (j=i \vee j=(i+r)\mathrm{mod}2^n)$</td><td>Find r</td></tr>
<tr><td>Grover

$f:\{0,1\}^n \rightarrow \{0,1\}$</td><td>$\exists x\in\{0,1\}^n$:
$(f(x)=1 \wedge \forall y\in\{0,1\}^n : y\neq x \Rightarrow f(x)=0)$</td><td colspan="5">$M_i\in\{I, C\}$,
$\exists i$: $(M_i = C \wedge \forall j \neq i \Rightarrow M_i = I)$</td><td>Find x</td></tr>
</table>

Table 1: Quantum Problems and U_F structure

3. QUANTUM GATES

The true heart of the quantum block is the quantum gate. Its function is to extract information from U_F and encode it into some basis vectors in order to answer the initial question of the problem. We used quantum circuits in order to provide a high-level description of a quantum gate. Using this description technique, we can now give a unified general representation of the quantum gates we employed in our analyses. Every gate is here intended as a special case of the quantum circuit reported in the general prototype of quantum block pictured in fig.1.

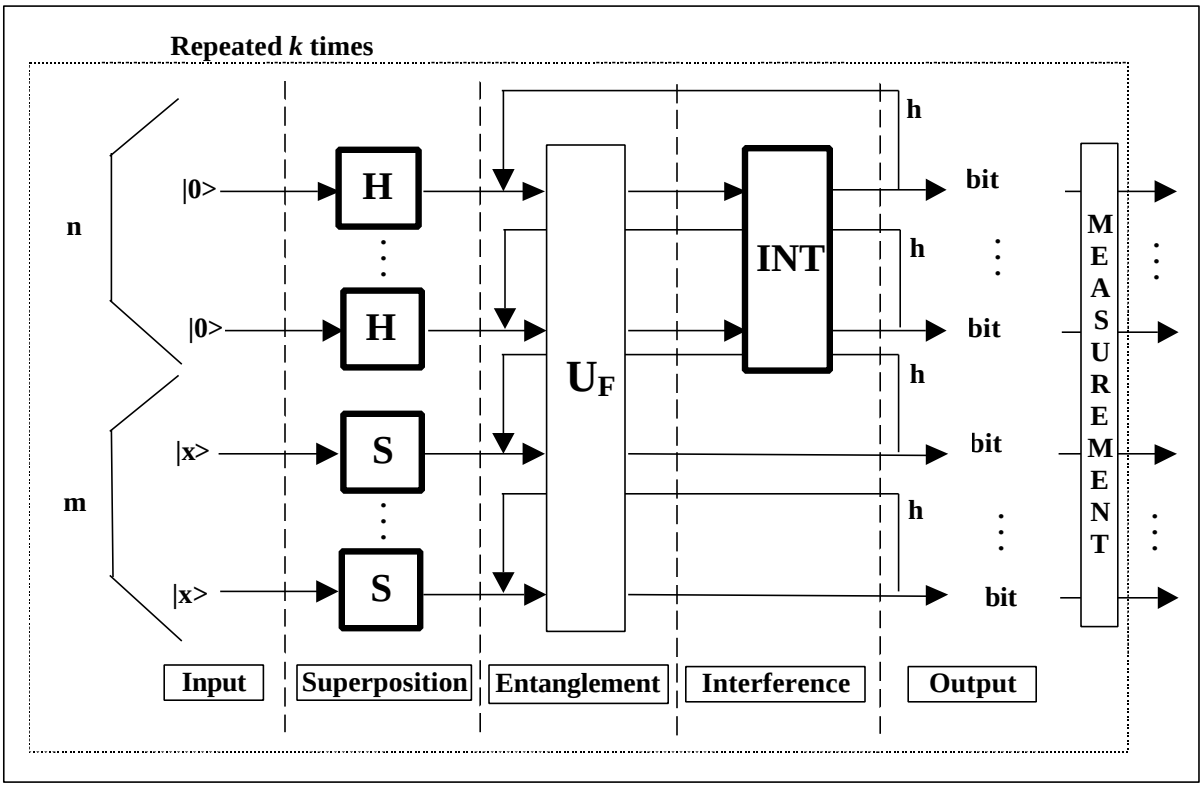


Figure 1: General Prototype of Quantum Block

We should finally observe that the difference between the four analysed quantum algorithms is essentially in the choice of the interference operator *Int* and of the superposition operator *S*. The input vector is a sort of message that traverses a quantum channel made of three main sub-channels: superposition, entanglement and interference. The entanglement channel is the true input of the algorithm gate. It belongs to a given family depending on the problem to solve and on its input. The superposition and especially the interference channels are chosen in such a way that some measurements effectuated at the end of the channel reveal what kind of entanglement has taken place at the middle of the channel.

Our next objective is to determine a sort of mathematical law that connects the choice of operator *Int* to the form of operator U_F and so to the initial problem. We hope in this way to be able to define a general method for automatic quantum programming.

In table 2 we report the values of the different parameters in the prototype of fig.1 for every one of the algorithms discussed in this paper:

NAME	ALGORITHM	GATE SYMBOLIC FORM: $\left[\underbrace{\left(Int\otimes^{m} I\right)}_{Interferen\,ce}\cdot \underbrace{U_F}_{Enta\,nglement}\right]^{h+1}\cdot\left(\underbrace{^{n}H\otimes^{m} S}_{Superposit\,ion}\right)$
Deutsch-Jozsa	• $m=1$ • $S=H\,(x=1)$ • $Int=^{n}H$ • $k=1$ • $h=0$	$\left(^{n}H\otimes I\right)\cdot U_F\cdot\left(^{n+1}H\right)$
Simon	• $m=n$ • $S=I\,(x=0)$ • $Int=^{n}H$ • $k=O(n)$ • $h=0$	$\left(^{n}H\otimes^{n} I\right)\cdot U_F\cdot\left(^{n}H\otimes^{n} I\right)$
Shor	• $m=n$ • $S=I\,(x=0)$ • $Int=QFT_n$ • $k=O(Poly(n))$ • $h=0$	$\left(QFT_n\otimes^{n} I\right)\cdot U_F\cdot\left(^{n}H\otimes^{n} I\right)$
Grover	• $m=1$ • $S=H\,(x=1)$ • $Int=D_n$ • $k=1$ • $h=O\left(2^{n/2}\right)$	$\left(D_n\otimes I\right)\cdot U_F\cdot\left(^{n+1}H\right)$

Table 2: Parameters for Quantum Algorithms

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

Conclusions

1. The methodology of quantum algorithm design based on R&D of quantum gates is represented.
2. The benchmarks of quantum gates for Deutsch's, Deutsch-Jozsa's, Simon's, Shor's, and Grover`s algorithms in general form (Part 7–Tab.2) are described.
3. The roles of quantum operators as superposition, entanglement and interference on coherent states are discussed.

SIMULATION OF QUANTUM ALGORITHMS ON CLASSICAL COMPUTERS

References

A. General Concepts of Quantum Computation

[1] R.P. Feynman, *Feynman Lectures on Computation*", Addison-Wesley Publ. Company, Inc., N.Y., 1996

[2] Y.I. Manin, *The Computable and Incomputable* (in Russian), Sovetskoe Radio Publ., Moscow, 1980

[3] J. Preskill, *Quantum Information and Computation*, http://www.theory. .caltech.edu/people/preskill/ph224, Caltech, 1997

[4] C.P. Williams, S.H. Clearwater, *Explorations in Quantum Computing*, Springer-Verlag, N.Y., 1998

[5] T. Nishino, *Introduction in Quantum Computing* (in Japanese), Japan, 1997

[6] A. Barenco, *Quantum Computation*, A thesis for the degree of Doctor of Philosophy at the University of Oxford, 1996

[7] A. Ekert, R. Jozsa, *Quantum Computation and Shor's Factoring Problems*, Rev. Modern Physics, vol.68, 1996, p.733

[8] T.P. Spiller, *Quantum Information Processing: Cryptography, Computation and Teleportation*, Proc. IEEE, vol.84, 1996

[9] A. Steane, *Quantum Computing*, Rep. Prog. Phys., vol.61, 1998. P.117

[10] A. Ekert, C. Macchiavello, *An Overview of Quantum Computing*, Unconventional Models of Computation, Eds: C.S. Claude, J. Casti and M.J. Dinneen, Springer-Verlag Singapore Pte.Ltd, 1998, p.19

[11] D. Aharonov, *Quantum Computation*, 1998, http://xxx.lanl.gov/ps/quant-ph/9812039

[12] A.Y. Kitaev, *Quantum Computations: Algorithms and Error Correction*, Russian Math. Surveys, vol.52, n.6, 1997, p.1191

[13] C.M. Bennet, P. Shor, *Quantum Information Theory*, IEEE Trans. Information Theory, vol.44, n.6, 1998, p.2724

[14] P. Olivari, *Problemi e metodi del calcolo quantistico* (in Italian), Tesi di Laurea: relatore Prof. D. De Falco, correlatore Prof. D. Mundici, Univeristà degli Studi di Milano, Dipartimento di Scienze dell'Informazione,1996

[15] E. Rieffel, W. Polak, *An Introduction to Quantum Computing for Non-Physicists*, ACM Computing Surveys, 1998, http://xxx.lanl.gov/ps/quant-ph/9809016

[16] J. Gruska, *Quantum Computing*, McGraw-Hill Publ. Company, 1999

[17] V. Vedral, M.B. Plenio, *Basics of Quantum Computation*, http://xxx.lanl.gov/ /ps/quant-ph/9802065

[18] D. Deutsch, *The Fabric of Reality*, Viking-Penguin Publ., London, 1997

[19] I.V. Volovich, *Mathematical Models of Quantum Computers*, Lectures Delivered at the Moscow State University, 1998

[20] A.Y. Kitaev, *Classical and Quantum Computations*, Lectures Notes Independent University, Moscow, 1998

[21] C. Moore, M. Nillson, *Some Notes on Parallel Quantum Computation*, LANL, http://xxx.lanl.gov/ps/quant-ph/9804034

[22] C. Moore, M. Nillson, *Parallel Quantum Computation and Quantum Codes*, LANL, http://xxx.lanl.gov/ps/quant-ph/9808027

[23] B. Tsirelson, *Quantum Information Processing*, Lecture Notes, Tel-Aviv University, 1997

[24] A. Berthiaume, *Quantum Computation*, in Complexity Theory Retrospective II, Ed. by L. Hemaspaandra and A. Saleman, Springer-Verlag, 1997

[25] M. Brooks (Ed.), *Quantum Computing and Communications*, Springer-Verlag, N.Y., 1999

[26] C.H. Bennet, P. Gacs, M. Li, P.M.B. Vitanyi, W.H. Zurek, *Information distance*, IEEE Transactions on Information Theory, vol.44, n.4, July 1998

[27] D. Gottesman, *Theory of Fault-tolerant Quantum Computation*, Phys. Rev. A, vol.57, n.1, January 1998

[28] A. Fijany, C.P. Williams, *Quantum Wavelet Transforms: Fast Algorithms and Complete Circuits*, Presented at 1st NASA Int. Conf. On Quantum Computing and Communication, palm Spring, CA, Feb.17-21, 1998, LANL http://xxx.lanl.gov/ps/quant-ph/9809004

B. Quantum Algorithms

[29] R.Jozsa, *Quantum Algorithms and the Fourier Transform*, SIAM Journal of Computing 26 (october 1997), http://xxx.lanl.gov/ps/quant-ph/9707033

[30] D. Collins, K.W. Kim, W.C. Holton, *Deutsch-Jozsa Algorithm as a Test of Quantum Computation*, Phys. Rev. A, 1998, http://xxx.lanl.gov/ps/quant-ph/9807012

[31] D.R. Simon, *On the Power of Quantum Computation*, Proc. 35th Annual IEEE Symposium on the Foundations of Computer Science (IEEE Computer Society Press, Los Alamitos, CA, 19994), p.116, http://feynman.stanford.edu/qcomp/ /simon/simon-94.ps.gz

[32] P.W. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, Proc. 34th Annual Symposium on the Foundations of Computer Science, edited by S. Goldwasser (IEEE Computer Society Press, Los Alamitos, CA, 1994), p.124, http://xxx.lanl.gov/ps/quant-ph/9508027

[33] L.K. Grover, *A Fast Quantum Mechanical Algorithm for Database Search*, Proceedings of the 28th ACM Symposium on Theory of Computing, Pennsylvania, May 22-24, 1996, p.212, http://xxx.lanl.gov/ps/quant-ph/9605043

[34] L.K. Grover, *A Framework for Fast Quantum Mechanical Algorithms*, Proceedings of the 30th ACM Symposium on Theory of Computing 53, 1998, http://xxx.lanl.gov/ /ps/quant-ph/9711043

[35] Yu. I. Manin, *Classical Computing, Quantum Computing and Shor's Factoring Algorithm*, expository talk written for the Bourbaki Seminar, June, 1999, http://xxx.lanl.gov/ps/quant-ph/9903008

[36] T. Hogg, C. Mochon, *Tools for Quantum Algorithms*, 1998, http://xxx.lanl.gov/ps/quant-ph/ 9811073

[37] R. Cleve, A. Ekert, C. Macchiavello, M. Mosca, *Quantum Algorithms Rivisited*,

Proc. Roy. Soc. Lond. A, 1997, http://xxx.lanl.gov/ps/quant-ph/9708016

[38] R. Cleve, A. Ekert, L. Henderson, C. Macchiavello, M. Mosca, *On Quantum Algorithms*, LANL, http://xxx.lanl.gov/ps/quant-ph/9903061

[39] A. Ekert, *Quantum Algorithms: Entanglement Enhanced Information Processing*, Phil. Trans. Roy. Soc. Lond., 1998, Proceedings of Royal Society Discussion Meeting "Quantum Computation: Theory and Experiment" held in November 1997, http://xxx.lanl.gov/ps/quant-ph/9803072

[40] R. Jozsa, *Quantum Effects in Algorithms*, Proc. 1st International Conference on Quantum Computation and Quantum Communication, Palm Springs, February 1998 (appearing in a special issue of the journal Chaos, Solitons and Fractals, 1998), http://xxx.lanl.gov/ps/quant-ph/9805086

[41] T.P. Spiller, *Quantum Information processing: Cryptography, Computation and Teleportation*, Proc. IEEE, vol. 84, 1996, p.1719

[42] E. Bernstein, U. Vazirani, *Quantum Complexity Theory*, SIAM Computing, vol.26, n.5, 1997, p.1411

[43] L.M. Adleman, J. Demarrais, M.-D.A. Huang, *Quantum Computability*, Proc. R. Soc. Lond., vol. A 454, 1998, p.339

[44] V. Vedral, A. Barenco, A. Ekert, *Quantum networks for Elementary Arithmetic Operations*, Rep. Prog. Phys., vol. A 54, n.1, 1996, p.147

[45] A. Steane, *Quantum Computing*, Rep. Prog. Phys., vol. 61, 1998, p.117

[46] S. Ulyanov, I. Kurawaki, F. Arai, T. Fukuda, K. Yamafuji, G.G. Rizzotto, *Physical Limits and Information Bounds of Micro Control*. *Part* 2., Proc. of Intern. Symp. On Micromechanics and Human Science (MHS'98), Nagoya, 1998, p.149

In order to provide material about Quantum Computing, we collected a lot of papers in the volumes:

[1] Quantum Computing Bibliography, Part I, *Mathematical and Physical Background of Quantum Computing*, vol.0 (Tutorials)

[2] Quantum Computing Bibliography, Part I, *Mathematical and Physical Background of Quantum Computing*, vol.1 (Author index A-B)

[3] Quantum Computing Bibliography, Part I, *Mathematical and Physical Background of Quantum Computing*, vol.2 (Author index B-E)

[4] Quantum Computing Bibliography, Part I, *Mathematical and Physical Background of Quantum Computing*, vol.3 (Author index F-P)

[5] Quantum Computing Bibliography, Part I, *Mathematical and Physical Background of Quantum Computing*, vol.4 (Author index P-W)

These volumes can be found in Polo Didattico e di Ricerca di Crema – Università degli Studi di Milano.