In this paper we extend the basic principles and results of conservative logic to include the main features of many-valued logics with a finite number of truth values. Different approaches to many-valued logics are examined in order to determine some possible functionally complete sets of logic connectives. As a result, we describe some possible finite-valued universal gates which realize a functionally complete set of fundamental connectives. One of the purposes of this work is to show that the framework of reversible and conservative computation can be extended toward some non classical “reasoning environments”, originally proposed to deal with propositions which embed imprecise and/or uncertain information, that are usually based upon many-valued and modal logics. We also describe a possible quantum realization of the proposed gates, using creation and annihilation operators. In such realization the gates are expressed as formulas that are obtained using three techniques: a “brute force” technique, an extension of the Conditional Quantum Control method introduced by Barenco, Deutsch, Ekert and Jozsa in 1995, and a new technique that we call the Constants method. We show the Constants method allows one to reduce the number of local operators in the formulas which correspond to the proposed gates.

Keywords: quantum computing, conservative logic, many-valued logics.

Communicated by: to be filled by the Editorial

1 Introduction
Since the work of Fredkin and Toffoli on Conservative Logic [1] and the work of Bennett [2] on the minimum energy necessary to implement a computational process, a lot of papers appeared on these subjects investigating both theoretical and practical aspects. In particular, many possible physical realizations of the Petri–Fredkin gate have been proposed [3, 4, 5, 6]. More generally, many papers deal with the possibility to realize a reversible computer based on the laws of quantum mechanics: here we just recall the work of Benioff [7, 8, 9], Deutsch [10, 11], Feynman [12, 13] and Peres [14].

In [15] we extended conservative logic in order to include the main features of many–valued logics with a finite number \( d \) of truth values. The first result was the introduction of three \( d \)-valued universal gates satisfying the main properties required by both conservative and many–valued paradigms. In this paper we present the quantum realization of these gates using
three methods: the brute force method, an extension of the Conditional Quantum Control technique introduced in [16], and a new technique which we call the Constants method. Both techniques are used to write quantum operators which describe the behavior of classical gates. Such operators are sums of “local” operators, each of which is a tensor product of suitable compositions of the operators $a^\dagger$ and $a$, which are the finite dimensional versions of creation and annihilation operators usually found in quantum mechanics. An equivalent formulation is also given using spin–rising ($J_+$) and spin–lowering ($J_-$) operators. In particular we show that, for two of the three gates under consideration, the formulas obtained with our Constants method have $\Theta(d^2)$ terms instead of the $\Theta(d^3)$ terms obtained with the Conditional Quantum Control technique. The remaining gate cannot be expressed as a Conditional Control gate, and thus in this case a comparison cannot be performed.

The paper is organized as follows: in section 2 we recall some notation from [15]. In sections 3 and 4 we introduce the creation and annihilation operators for two–states systems, and we show how they can be used to write expressions which provide a quantum description of the behavior of $n$–input/$n$–output Boolean gates, starting from their truth table. In section 5 we consider the creation and annihilation operators for $d$–dimensional quantum systems, and we show how they can be used to describe the behavior of any gate for $d$–valued logics, starting from its truth table. In section 6 the quantum formulas corresponding to the gates $f_{1d}, f_{2d,\lambda}$ and $m_d$ introduced in [15] are built.

2 Preliminaries

We assume that the reader is already familiar with the notions concerning conservative logic and many–valued logics. An extended introduction to these subjects can be found in [15]. Here we just recall some notations needed in the following, and we introduce the connectives realized by our gates.

2.1 Finite–valued logical connectives

In this paper we will deal only with a finite number of truth values; more precisely we will consider, for every integer $d \geq 2$, the set $L_d = \{0, \frac{1}{d-1}, \frac{2}{d-1}, \ldots, \frac{d-2}{d-1}, 1\}$. As usually found in literature, we will use $L_d$ both as a set of truth values and as a numerical set equipped with the standard order relation on rational numbers. The values 0 and 1 denote respectively truth and falsity, while the values different from 0 and 1 indicate various degrees of indefiniteness.

For $d$–valued Lukasiewicz logic the following connective is assumed as primitive (see [17], and also [19, 20]):

$$x \rightarrow_L y := \begin{cases} 1 - x + y & \text{if } y < x \\ 1 & \text{otherwise} \end{cases}$$ (Lukasiewicz implication)

On the basis of this definition, it is possible to define some other connectives which are found in $d$–valued Lukasiewicz logic:

$$x \vee_L y := (x \rightarrow_L y) \rightarrow_L y$$ (Lukasiewicz disjunction)

$$x \land_L y := \neg(\neg x \vee \neg y)$$ (Lukasiewicz conjunction)

$$x \leftrightarrow_L y := (x \rightarrow_L y) \land (y \rightarrow_L x)$$ (Lukasiewicz equivalence)
Note that we can also obtain the so called *diametrical negation* as \( \neg x = x \rightarrow L 0 = 1 - x \). From these definitions it is not difficult to see that \( x \vee y = \max\{x, y\} \) and \( x \wedge y = \min\{x, y\} \) (with respect to the standard ordering of \( L_d \)). One important feature of all many–valued connectives now presented is that they coincide with the analogous Boolean connectives when only 0 and 1 are considered.

Following Zawirski [21] (and for an algebraic axiomatization see [22, 23]) the Lukasiewicz approach to many–valued logics can be equivalently recovered on the basis of the pair of connectives \( \{\oplus, \neg\} \), where

\[
x \oplus y := \begin{cases} x + y & \text{if } x + y < 1 \\ 1 & \text{otherwise} \end{cases}
\]

Indeed, it is immediately verified that \( x \rightarrow L y = \neg x \oplus y \) and \( x \oplus y = \neg x \rightarrow L y \). In this context one can consider another interesting derived connective defined by the rule \( x \odot y := \neg(\neg x \oplus \neg y) \) and explicitly written as:

\[
x \odot y := \begin{cases} x + y - 1 & \text{if } 1 < x + y \\ 0 & \text{otherwise} \end{cases}
\]

In this paper we consider also two modal connectives, *possibility* (\( \Diamond \)) and *necessity* (\( \Box \)), which are formally defined as follows:

\[
\Diamond x := \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{if } x \neq 0 \end{cases} \quad \text{(possibility)}
\]

\[
\Box x := \begin{cases} 0 & \text{if } x \neq 1 \\ 1 & \text{if } x = 1 \end{cases} \quad \text{(necessity)}
\]

Besides the diametrical negation (\( \neg \)) two other negation connectives can be considered: the *intuitionistic negation* (also *impossibility* \( \sim \)) and the *anti–intuitionistic negation* (also *contingency* \( \flat \)). They are derived as follows:

\[
\sim x := \neg \Diamond x = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x \neq 0 \end{cases} \quad \text{(intuitionistic negation)}
\]

\[
\flat x := \neg \Box x = \begin{cases} 1 & \text{if } x \neq 1 \\ 0 & \text{if } x = 1 \end{cases} \quad \text{(anti–intuitionistic negation)}
\]

Among various implication connectives one can find in literature, in this paper we consider the Gödel implication \( \rightarrow_G \), which is defined as follows:

\[
x \rightarrow_G y := \begin{cases} y & \text{if } y < x \\ 1 & \text{otherwise} \end{cases} \quad \text{(Gödel implication)}
\]

Note that the intuitionistic negation can be obtained from this implication connective as \( \sim x = x \rightarrow_G 0 \). Moreover, in the three–valued case the Gödel’s implication differs from the Lukasiewicz one for the input pair \( (\frac{1}{2}, 0) \): in fact, \( \frac{1}{2} \rightarrow_L 0 = \frac{1}{2} \) whereas \( \frac{1}{2} \rightarrow_G 0 = 0 \).
2.2 Computational paradigms of reversible and conservative logic

Conservative logic has been introduced by Fredkin and Toffoli in [1] as a mathematical model of computation that allows one to perform universal computations with zero internal power dissipation. This goal is reached by basing the model on reversible and conservative primitives. The latter reflect physical principles such as the reversibility of microscopic dynamical laws and the conservation of certain quantities, in particular the energy of the physical system used to perform the computations. More precisely, they consider the Petri–Fredkin gate (see section 4.4.1), a universal three–input/three–output Boolean gate introduced for the first time by Petri in [24]. This gate is also known as the controlled–swap gate since when the first (controlling) line is in state 0, the other two (controlled) lines exchange their values as a result of the action of the gate. Otherwise, if the controlling line has the value 1 then the two controlled lines keep their original values.

After [1] conservativeness is usually modelled by the property that the output values of the involved gate are always an input–dependent permutation of the incoming patterns. In the Boolean case, this is equivalent to require that the number of 1’s given in input is preserved into the output. This is what we have called strict conservativeness in [15].

On the other hand, reversibility is modelled as the fact that the gate computes an invertible function. Since every conservative circuit performs a permutation of each input pattern, and every reversible circuit computes a bijective mapping between input and output patterns, a gate which is simultaneously reversible and conservative must necessarily have the same number of input and output lines, that is, it must be an $n$–input/$n$–output gate. Here, we are mainly interested into gates which are both conservative and reversible, such as the Petri–Fredkin gate.

Obviously, there are reversible gates which are not conservative, such as the Controlled–Controlled–Not gate (see section 4.4.2) introduced by Petri [24] and successively considered by Feynman in [12], and the Peres gate (see section 4.4.3). Similarly, there are also conservative gates which are not reversible such as, for example, a two–input/two–output gate which maps both the input pairs $(0, 1)$ and $(1, 0)$ to $(1, 0)$. Let us recall that in [15] a procedure has been given to construct a conservative and reversible gate from a generic one, in such a way that the primitive behavior of the latter can be easily recovered.

A very important connective in reversible computing is the FanOut: $L_d \to L_d^2$, defined by the law $\text{FanOut}(x) = (x, x)$. In other words, the FanOut function simply clones the input value. When dealing with classical circuits, the FanOut function is implemented by sticking two output wires to an existing input wire. It is easy to see that the FanOut function cannot be realized by a two–input/two–output conservative gate, even when working with Boolean values. On the contrary, the Controlled–Not operator is a reversible two–input/two–output gate that realizes the FanOut function, as we shall discuss in section 4.2.

An important requirement for $d$–valued gates is functional completeness: that is, by fixing the values of a subset of their input lines we can obtain a set of connectives that, equipped with a suitable set of intermediate logical constants (i.e., from $L_d \setminus \{0, 1\}$) are able to realize any mapping from $L_d^n$ to $L_d$. The need for intermediate logical constants is given by the well known fact that both Łukasiewicz and Gödel logics without such constants are incomplete [20]: for example, they cannot express the function which is identically equal to $\frac{1}{d-1}$.

In [15] we have extended conservative logic to include the main features of $d$–valued logics,
and we have proposed some possible extensions of the Petri–Fredkin gate to deal with $d$ truth values. We were interested to design functionally complete gates extending the Petri–Fredkin gate and being able to implement the FanOut function together with as many connectives from $d$-valued logics as possible. However, we showed that it is impossible to find such $d$-valued three-input/three-output conservative gates for $d \geq 3$ (Proposition 6.1 of [15]). As a consequence, we proposed the notion of weak conservativeness, together with a possible physical interpretation: precisely, we required that the sum of output values be always equal to the sum of input values. This property can be interpreted as an energy conservation principle, in the sense that the energy needed to build the input values is equal to the energy that would be required to build the output values from scratch (that is, directly, instead of using the gate).

Needless to say, the functions presented in [15] are reversible mappings (and thus bijective) from $L_d^3$ to itself; better still, they are self-reversible (i.e., the composition of the functions with themselves gives the identity as a result). This means that the functions are permutations on the set $L_d^3$ that can be expressed as the product of disjoint transpositions.

Two further properties are considered in [15] in order to extend the Petri–Fredkin gate to deal with $d$ truth values: 0-regularity and 1-regularity. These properties are meaningful only for three-input/three-output gates. Precisely, let $G : L_d^3 \to L_d^3$ be the function computed by one of such gates. We say that the gate is 0-regular if and only if $G(0,x_2,x_3) = (0,x_3,x_2)$ for every $x_2,x_3$ in $L_d$. On the other hand, we say that the gate is 1-regular if and only if $G(1,x_2,x_3) = (1,x_2,x_3)$ for every $x_2,x_3$ in $L_d$. As it will be clear in the sequel, these two properties suggested us to adopt the Conditional Quantum Control of [16] as a technique to give a quantum description of our gates.

The next list summarizes the properties we tried to preserve when we looked for possible extensions of the Petri–Fredkin gate to $d$-valued logics:

F-1 ) it is a three-input/three-output gate, where each input and each output line may assume one of the values in $L_d$;
F-2 ) it is reversible;
F-2' ) it is self-reversible;
F-3 ) it is weakly conservative;
F-3' ) it is (strictly) conservative;
F-4 ) it is a universal gate, that is, from the patterns of the gate a functionally complete set of connectives is obtained, including FanOut;
F-5 ) it is 0-regular;
F-6 ) it is 1-regular;
F-7 ) $y_1 = x_1$, that is, the first output is always equal to the first input;
when fed with Boolean input triples it behaves as the Petri–Fredkin gate.

In [15] we have presented the gates \( F_1, F_2, \) and \( F_3 \), which can be thought of as three–valued extensions of the Petri–Fredkin gate. Moreover we have presented two functions, \( \mathcal{F}_d^1 \) and \( \mathcal{F}_d^2 \), which are designed to work with \( d \) truth values, and a family \( \mathcal{F}_d^{1,\lambda} \) of functions which are similar to \( \mathcal{F}_d^1 \) and which are able to realize the modal operator \( \Box \) by losing 0–regularity for some input/output pairs.

To be precise, the first function \( \mathcal{F}_d^1 : L_d^3 \rightarrow L_d^3 \) we introduced is defined as follows:

\[
\forall \mathbf{x} = (x_1, x_2, x_3) \in L_d^3 \quad \mathcal{F}_d^1(\mathbf{x}) := \begin{cases} 
(x_1, x_3, x_2) & \text{if } x_1 = 0 \text{ and } x_2 \neq x_3 \\
(x_1, x_3, x_2) & \text{if } 0 < x_1 \leq x_3 < 1 \text{ and } x_2 = 1 \\
(x_1, x_3, x_2) & \text{if } 0 < x_1 \leq x_2 < 1 \text{ and } x_3 = 1 \\
(x_1, x_1, 1 - x_1 + x_3) & \text{if } x_3 < x_1 < 1 \text{ and } x_2 = 1 \\
(x_1, 1, x_3 + x_1 - 1) & \text{if } x_1 < 1, x_2 = x_1, x_3 + x_1 \geq 1 \text{ and } x_3 < 1 \\
(x_1, x_2 - x_1) & \text{if } 0 < x_1 < x_2 < 1 \text{ and } x_3 = 0 \\
(x_1, x_3 + x_1, 0) & \text{if } 0 < x_1, x_2 = x_1, x_3 + x_1 < 1 \text{ and } x_3 > 0 \\
(x_1, x_2, x_3) & \text{otherwise}
\end{cases}
\]

It is not difficult to see that \( \mathcal{F}_d^1 \) is also a universal function. In fact, as it is shown in Table 1, through suitable configurations of constants assigned to its arguments we obtain a set of connectives which suffice to generate, besides the \( \text{FanOut} \) function, all the operators of Lukasiewicz and Gödel \( d \)--valued logics.

Table 1. The operators obtained through function \( \mathcal{F}_d^1 \).

<table>
<thead>
<tr>
<th>Connective</th>
<th>Inputs</th>
<th>Constants</th>
<th>Outputs</th>
<th>Garbage</th>
</tr>
</thead>
<tbody>
<tr>
<td>FanOut</td>
<td>( x_1 )</td>
<td>( x_2 = 1, x_3 = 0 )</td>
<td>( y_1, y_2 )</td>
<td>( y_3 )</td>
</tr>
<tr>
<td>Pr(_1)</td>
<td>( x_2, x_3 )</td>
<td>( x_1 = 0 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>Pr(_2)</td>
<td>( x_2, x_3 )</td>
<td>( x_1 = 0 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \neg L )</td>
<td>( x_1, x_3 )</td>
<td>( x_2 = 1 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>( \neg G )</td>
<td>( x_1, x_2 )</td>
<td>( x_3 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \forall )</td>
<td>( x_1, x_3 )</td>
<td>( x_2 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \land )</td>
<td>( x_1, x_2 )</td>
<td>( x_3 = 0 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>Id</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = 0 )</td>
<td>( y_1 )</td>
<td>( y_2, y_3 )</td>
</tr>
<tr>
<td>( \neg )</td>
<td>( x_1 )</td>
<td>( x_2 = 1, x_3 = 0 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>( \sim )</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \Box )</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = 1 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
</tbody>
</table>

The family of functions \( \mathcal{F}_d^{1,\lambda} : L_d^3 \rightarrow L_d^3 \), parameterized with respect to \( \lambda \), is defined as
forall \( x = (x_1, x_2, x_3) \in L_3 \)

\[
\begin{align*}
    f_{d,\lambda}^2(x) := & \begin{cases} 
    (x_2, x_1, x_3) & \text{if } x_1 = 0, 0 < x_2 < 1 \text{ and } x_3 = \lambda \quad \text{i)} \\
    (x_2, x_1, x_3) & \text{if } 0 < x_1 < 1, x_2 = 0 \text{ and } x_3 = \lambda \quad \text{ii)} \\
    (x_1, x_3, x_2) & \text{if } x_1 = 0, x_2 \neq \lambda, x_3 \neq \lambda \\
    & \quad \text{ and } x_2 \neq x_3 \quad \text{iii)} \\
    (x_1, x_3, x_2) & \text{if } 0 \leq x_1 \leq x_3 < 1 \text{ and } x_2 = 1 \quad \text{iv)} \\
    (x_1, x_3, x_2) & \text{if } 0 \leq x_1 \leq x_2 < 1 \text{ and } x_3 = 1 \quad \text{v)} \\
    (x_1, 1, x_3 + x_1 - 1) & \text{if } x_1 < 1, x_2 = x_1, x_3 + x_1 \geq 1 \\
    & \quad \text{ and } x_3 < 1 \quad \text{vi)} \\
    (x_1, x_2 - x_1) & \text{if } 0 \leq x_1 < x_2 < 1 \text{ and } x_3 = 0 \quad \text{vii)} \\
    (x_1, x_3 + x_1, 0) & \text{if } 0 \leq x_1, x_2 = x_1, x_3 + x_1 < 1 \\
    & \quad \text{ and } x_3 > 0 \quad \text{viii)} \\
    (x_1, x_2, x_3) & \text{otherwise} \quad \text{x)}
\end{cases}
\]

For each fixed value of \( \lambda \) we get a function which realizes the connectives exposed in Table 2.

<table>
<thead>
<tr>
<th>Connective</th>
<th>Inputs</th>
<th>Constants</th>
<th>Outputs</th>
<th>Garbage</th>
</tr>
</thead>
<tbody>
<tr>
<td>FanOut</td>
<td>( x_1 )</td>
<td>( x_2 = 1, x_3 = 0 )</td>
<td>( y_1, y_2 )</td>
<td>( y_3 )</td>
</tr>
<tr>
<td>Pr1</td>
<td>( x_2, x_3 )</td>
<td>( x_1 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>Pr2</td>
<td>( x_2, x_3 )</td>
<td>( x_1 = 1 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>( \neg L )</td>
<td>( x_1, x_3 )</td>
<td>( x_2 = 1 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>( \neg G )</td>
<td>( x_1, x_2 )</td>
<td>( x_3 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \lor )</td>
<td>( x_1, x_3 )</td>
<td>( x_2 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \land )</td>
<td>( x_1, x_2 )</td>
<td>( x_3 = 0 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>Id</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = 0 )</td>
<td>( y_1 )</td>
<td>( y_2, y_3 )</td>
</tr>
<tr>
<td>( \neg )</td>
<td>( x_1 )</td>
<td>( x_2 = 1, x_3 = 0 )</td>
<td>( y_1 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>( \sim )</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = 1 )</td>
<td>( y_2 )</td>
<td>( y_1, y_3 )</td>
</tr>
<tr>
<td>( \diamond )</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = 1 )</td>
<td>( y_3 )</td>
<td>( y_1, y_2 )</td>
</tr>
<tr>
<td>( \lozenge )</td>
<td>( x_1 )</td>
<td>( x_2 = 0, x_3 = \lambda )</td>
<td>( y_1 )</td>
<td>( y_2, y_3 )</td>
</tr>
</tbody>
</table>

None of the gates just presented generates the MV connectives. This fact led us to define
the following function $m_d : L^3_d \to L^3_d$:
\[
\forall x = (x_1, x_2, x_3) \in L^3_d \quad m_d(x) :=
\begin{cases}
(x_1, x_3, x_2) & \text{if } x_1 = 0 \text{ and } x_2 \neq x_3 \\ 
(x_1, x_1 + x_3, 1 - x_1) & \text{if } x_1 > 0, x_2 = 1 \text{ and } x_1 + x_3 < 1 \\ 
(x_1, 1, x_2 - x_1) & \text{if } 0 < x_1 \leq x_2 < 1 \text{ and } x_3 = 1 - x_1 \\ 
1 - x_1 & \text{if } x_1 < 1, x_2 < 1, x_3 = 0 \text{ and } 
\end{cases}
\]

Table 3 reports the operators that can be obtained from function $m_d$ by fixing one or two input lines with constant values from $L_d$. As we can see, $m_d$ is a gate providing functional completeness for the finite–valued calculus of Lukasiewicz, regardless of the value assumed by $d$.

<table>
<thead>
<tr>
<th>Connectives</th>
<th>Inputs</th>
<th>Constants</th>
<th>Outputs</th>
<th>Garbage</th>
</tr>
</thead>
<tbody>
<tr>
<td>FanOut</td>
<td>$x_1$</td>
<td>$x_2 = 1, x_3 = 0$</td>
<td>$y_1, y_2$</td>
<td>$y_3$</td>
</tr>
<tr>
<td>$\text{Pr}_1$</td>
<td>$x_2, x_3$</td>
<td>$x_1 = 0$</td>
<td>$y_3$</td>
<td>$y_1, y_2$</td>
</tr>
<tr>
<td>$\text{Pr}_2$</td>
<td>$x_2, x_3$</td>
<td>$x_2 = 1$</td>
<td>$y_2$</td>
<td>$y_1, y_3$</td>
</tr>
<tr>
<td>$\ominus$</td>
<td>$x_1, x_3$</td>
<td>$x_1 = 0$</td>
<td>$y_2$</td>
<td>$y_1, y_3$</td>
</tr>
<tr>
<td>$\odot$</td>
<td>$x_1, x_2$</td>
<td>$x_3 = 0$</td>
<td>$y_2$</td>
<td>$y_1, y_3$</td>
</tr>
<tr>
<td>Id</td>
<td>$x_1$</td>
<td>$x_2 = 0, x_3 = 0$</td>
<td>$y_1$</td>
<td>$y_2, y_3$</td>
</tr>
<tr>
<td>$\neg$</td>
<td>$x_1$</td>
<td>$x_2 = 1, x_3 = 0$</td>
<td>$y_3$</td>
<td>$y_1, y_2$</td>
</tr>
<tr>
<td>$\sim$</td>
<td>$x_1$</td>
<td>$x_2 = 0, x_3 = 1$</td>
<td>$y_2$</td>
<td>$y_1, y_3$</td>
</tr>
<tr>
<td>$\otimes$</td>
<td>$x_1$</td>
<td>$x_2 = 0, x_3 = 1$</td>
<td>$y_3$</td>
<td>$y_1, y_2$</td>
</tr>
<tr>
<td>$\boxplus$</td>
<td>$x_3$</td>
<td>$x_1 + x_2 = 0, x_2 = 0$</td>
<td>$y_3$</td>
<td>$y_1, y_2$</td>
</tr>
</tbody>
</table>

One of the results exposed in [15] is that there is no weakly conservative and reversible three–inputs/three–outputs gate for $d$–valued logics that extends the Petri–Fredkin gate and realizes in its configurations all the desired connectives.

Since the functions $f^1_d, f^2_d$ and $m_d$ behave just like the three–valued gates $F_1$, $F_2$ and $F_3$ when $d = 3$, it will suffice to show how to describe the more general functions in a way which is suitable for a quantum realization. In order to reach our goal, we apply three techniques: the Brute Force method, an extension of the Conditional Quantum Control introduced in [16] and a new method that we call the Constans method. In both cases, the resulting expressions will be sums of local operators, each of which is the tensor product of suitable compositions of creation (or spin–rising) and annihilation (resp., spin–lowering) operators, introduced in section 3.5.
3 A Mathematical Model of Quantum Realization of Boolean Gates

In this section we describe a quantum model of computers based on Boolean logic, with particular regards to reversible and conservative gates. In the next section a possible extension of such formalism to $d$–valued logics is given, with some examples of the three–valued case.

From an abstract point of view a quantum computer can be considered as made up of interacting parts. The elementary units (memory cells) that compose these parts are two–level quantum systems called *qubits*. A qubit is typically implemented using the energies of a two–level atom, or the two spin states of a spin–1/2 atomic nucleus, or a polarization photon. In any digital computer, each bit must be stored in the state of some physical system. Contemporary computers use voltage levels to encode bits. The only requirement is that the physical system must possess at least two clearly distinguishable patterns, or *states*, that are sufficiently stable that they do not flip, spontaneously, from the state representing the bit 0 into the state representing the bit 1, or vice versa.

Fortunately, certain quantum systems possess properties that lend themselves to encoding bits as physical states. When we measure the “spin” of an electron, for example, we always find it to have one of two possible values. One value, called “spin up” or $|\uparrow\rangle$, means that the spin was found to be parallel to the axis along which the measurement was taken. The other possibility, “spin down” or $|\downarrow\rangle$, means that the spin was found to be anti–parallel to the axis along to which the measurement was taken.

This intrinsic “discreteness,” a manifestation of quantization, allows the spin of an electron to be considered a natural “bit.”

As a matter of fact, there is nothing special about spin systems. Any two–state quantum system, such as the direction of polarization of a photon or the discrete energy levels in an excited atom, would work equally well. Whatever the exact physical embodiment chosen, if a quantum system is used to represent a bit, then we call the resulting system a quantum bit, or just a “*qubit*” for short. [25] The mathematical description — independent of the practical realization — of a single qubit is based on the two–dimensional complex Hilbert space $\mathbb{C}^2$. The Boolean truth values 0 and 1 are represented in this framework by the unit vectors of the canonical orthonormal basis, called the *computational basis* of $\mathbb{C}^2$:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Inasmuch as we are talking about bits and qubits, we’d better find a way of distinguishing them. When we are talking about a *qubit* (a quantum bit) in a physical state that represents the bit 0, we will write the qubit state as $|0\rangle$. Likewise, a qubit in a physical state representing the bit value 1 will be written $|1\rangle$. [25] Qubits are thus the quantum analog of the classical notion of bit, but whereas bits can only take two different values, 0 and 1, qubits are not confined to their two basis states, $|0\rangle$ and $|1\rangle$, but can also exist in states which are coherent superpositions such as $|\psi\rangle = c_0 |0\rangle + c_1 |1\rangle$, where $c_0$ and $c_1$ are complex numbers satisfying the condition $|c_0|^2 + |c_1|^2 = 1$. A qubit in this state is not simply in state $|0\rangle$ or $|1\rangle$, nor is it in an intermediate state; rather the qubit is in both states *simultaneously* and a mere act of measure alter this state. Indeed, performing a measurement on a qubit in a state superposition of the kind described above will return the value 0 with probability $|c_0|^2$ and 1 with probability $|c_1|^2$; the state of the qubit after the measurement (post–measurement state)
will be $|0\rangle$ or $|1\rangle$ (depending on the outcome), and not $c_0 |0\rangle + c_1 |1\rangle$.

Let us stress that in axiomatic quantum mechanics a (pure) state is described by a one-dimensional subspace of the involved Hilbert space, whose vectors are representatives of this state. Thus, two unit vectors $|\psi\rangle$ and $|\varphi\rangle$ describe (belong to) the same state if and only if they differ of a phase factor, that is, if and only if there exists a real value $\theta \in [0, 2\pi)$ such that $|\psi\rangle = e^{i\theta} |\varphi\rangle$.

A quantum register of size $n$ (also called an $n$-register) is mathematically described by the Hilbert space $\otimes^n C^2 = C^2 \otimes \ldots \otimes C^2$, representing a sequence of $n$ qubits labelled by the index $i \in \{1, \ldots, n\}$. An $n$-configuration is a vector $|x_1\rangle \otimes \ldots \otimes |x_n\rangle \in \otimes^n C^2$, where $x_i \in \{0, 1\}$ for every $i = 1, 2, \ldots, n$. This unit vector is usually written as $|x_1, \ldots, x_n\rangle$ and is considered as a quantum realization of the Boolean finite sequence $(x_1, \ldots, x_n) \in \{0, 1\}^n$.

Let us recall that the dimension of $\otimes^n C^2$ is $2^n$ and that $\{|x_1, \ldots, x_n\rangle : x_i \in \{0, 1\}\}$ is an orthonormal basis of this space called the $n$-register computational basis.

A classical computer is built from an electrical circuit containing wires and logic gates. The wires are used to carry information around the circuit, while logic gates perform manipulations of the information, converting it from one form to another. Analogously, a quantum computer is built from a quantum circuit containing elementary quantum gates to manipulate quantum information.

Unlike the situation of classical wired computers, where voltages of a wire go over voltages of another, in quantum realizations of classical Boolean gates (also called Boolean quantum gates in the sequel) something different happens. First of all, in this setting every gate must have the same number of input and output lines (that is, $n$-input/$n$-output gate). Each qubit of a given $n$-register is prepared in some particular quantum state ($|0\rangle$ or $|1\rangle$) in order to realize the required $n$-configuration $|x_1, \ldots, x_n\rangle$, quantum realization of an input Boolean sequence of length $n$. Then, a linear operator $G : \otimes^n C^2 \to \otimes^n C^2$ is applied to the $n$-register. The application of $G$ has the effect of transforming the $n$-configuration $|x_1, \ldots, x_n\rangle$ into a new $n$-configuration $G(|x_1, \ldots, x_n\rangle) = |y_1, \ldots, y_n\rangle$, which is the quantum realization of the output pattern of the gate. In other words, a quantum realization $G$ of a Boolean $n$-input/$n$-output gate is a linear operator transforming vectors of the $n$-register computational basis into vectors of the same basis. Let us stress that in particular such operator $G$ changes the state $|x_i\rangle$ (with $x_i \in \{0, 1\}$) of each qubit of the register into a new state $|y_i\rangle$ (with $y_i \in \{0, 1\}$) of the same qubit, and we interpret such modifications as the computation made by the corresponding gate.

The action of the operator $G$ on a non-factorized vector $\Phi = \sum c_{i_1 \ldots i_n} |x_{i_1}, \ldots, x_{i_n}\rangle$, expressed as a linear combination of the elements of the $n$-register basis, is obtained by linearity: $G(\Phi) = \sum c_{i_1 \ldots i_n} G(|x_{i_1}, \ldots, x_{i_n}\rangle)$. In particular, self-reversible and unitary gates are characterized by the operator conditions $G = G^{-1} = G^*$.  

3.1 The “matrix” representation of quantum mechanics on tensor products of Hilbert spaces

In this section, in order to prevent possible confusions, we briefly summarize the so-called matrix representation of standard Hilbert space quantum mechanics in the particular case of a tensor product of Hilbert spaces. For the sake of simplicity we consider the tensor product
For the sake of simplicity, in the sequel we set one–to–one correspondence:

\[
\text{Fourier expansion with respect to the basis } C^B \text{ is not difficult to build a unitary representation } \mathcal{U}_{CB} \text{ of the Hilbert space } C^2 \otimes C^2 \text{ onto the Hilbert space } C^4, \text{ which to every vector } |\Psi\rangle \text{ of } C^2 \otimes C^2, \text{ expressed according to its Fourier expansion with respect to the basis } CB \text{ as } |\Psi\rangle = \sum_{i=1}^4 \langle u_i | \Psi \rangle | u_i \rangle, \text{ associates the vector } \mathcal{U}_{CB}(|\Psi\rangle) = [(|u_1|\Psi), \langle u_2|\Psi\rangle, \langle u_3|\Psi\rangle, \langle u_4|\Psi\rangle]^T \text{ of } C^4 \text{ consisting of the ordered Fourier coefficients. For the sake of simplicity, in the sequel we set } |\Psi_{CB}\rangle := \mathcal{U}_{CB}|\Psi\rangle. \text{ In particular, } \mathcal{U}_{CB} \text{ identifies any factorized vector from } C^2 \otimes C^2 \text{ with an element of } C^4 \text{ according to the one–to–one correspondence:}
\[
\begin{bmatrix}
\varphi_1 \\
\varphi_2 \\
\varphi_3 \\
\varphi_4
\end{bmatrix} \otimes
\begin{bmatrix}
\psi_1 \\
\psi_2 \\
\psi_3 \\
\psi_4
\end{bmatrix} =
\begin{bmatrix}
\varphi_1 \psi_1 \\
\varphi_1 \psi_2 \\
\varphi_3 \psi_1 \\
\varphi_3 \psi_2
\end{bmatrix}
\]
(1)

In the case of the states of the computational orthonormal basis of } C^2 \otimes C^2 \text{ we have:

\[
\begin{array}{c|c}
C^2 \otimes C^2 & C^4 \\
\hline
|1, 1\rangle & [0, 0, 0, 1]^T \\
|1, 0\rangle & [0, 1, 0, 1]^T \\
|0, 1\rangle & [0, 1, 0, 1]^T \\
|0, 0\rangle & [1, 0, 0, 1]^T
\end{array}
\]

An identification holds also between operators. Let us denote by } \mathcal{L}(C^2 \otimes C^2) \text{ (resp., } \mathcal{L}(C^4) \text{) the (noncommutative) associative algebra with unity of all linear operators on } C^2 \otimes C^2 \text{ (resp., } C^4) \text{. Then the mapping } \mathcal{U}_{CB} : \mathcal{L}(C^2 \otimes C^2) \mapsto \mathcal{L}(C^4) \text{ associating to the linear operator } T : C^2 \otimes C^2 \mapsto C^2 \otimes C^2 \text{ the linear operator } T_{CB} = \mathcal{U}_{CB} \circ T \circ \mathcal{U}^{-1}_{CB} : C^4 \mapsto C^4 \text{ is an algebra isomorphism whose behavior is depicted by the following commutative diagram.}

\[
\begin{array}{ccc}
C^2 \otimes C^2 & \xrightarrow{T} & C^2 \otimes C^2 \\
\downarrow \mathcal{U}_{CB} & \equiv & \downarrow \mathcal{U}_{CB} \\
C^4 & \xrightarrow{T_{CB}} & C^4
\end{array}
\]

The operator } T_{CB} \text{ can be described using an order 4 complex matrix whose entry } T^i_j \text{ (corresponding to the row } i \in \{1, 2, 3, 4\} \text{ and the column } j \in \{1, 2, 3, 4\} \text{) is } T^i_j := \langle u_i | T u_j \rangle. \text{ In particular, the tensor product } T \otimes R \text{ of the two operators } T = \begin{bmatrix}
\varphi_1 & \varphi_2 \\
\varphi_3 & \varphi_4
\end{bmatrix} \text{ and } R = \begin{bmatrix}
\psi_1 & \psi_2 \\
\psi_3 & \psi_4
\end{bmatrix} \text{ defined on } C^2 \text{ is isomorphic to an operator on } C^4 \text{ according to the following identification:
\[
\begin{bmatrix}
\varphi_1 \\
\varphi_2 \\
\varphi_3 \\
\varphi_4
\end{bmatrix} \otimes
\begin{bmatrix}
\psi_1 \\
\psi_2 \\
\psi_3 \\
\psi_4
\end{bmatrix} =
\begin{bmatrix}
\varphi_1 \psi_1 & \varphi_1 \psi_2 & \varphi_2 \psi_1 & \varphi_2 \psi_2 \\
\varphi_1 \psi_3 & \varphi_1 \psi_4 & \varphi_2 \psi_3 & \varphi_2 \psi_4 \\
\varphi_3 \psi_1 & \varphi_3 \psi_2 & \varphi_4 \psi_1 & \varphi_4 \psi_2 \\
\varphi_3 \psi_3 & \varphi_3 \psi_4 & \varphi_4 \psi_3 & \varphi_4 \psi_4
\end{bmatrix}
\]
(2)
If $A$ is a self-adjoint operator (that is, a quantum observable) on $C^2 \otimes C^2$, written $A \in \mathcal{O}(C^2 \otimes C^2)$, then $A_{CB}$ is such that $A_{CB} = (A_{CB})^* = (A_{CB})^t$ and if $U$ is a unitary operator then $U_{CB}$ is such that $(U_{CB})^{-1} = (U_{CB})^* = (U_{CB})^t$.

As to the quantities which are experimentally tested in quantum mechanics, the mean value $\langle A \rangle_\Psi := \langle \Psi | A | \Psi \rangle$ of the observable $A$ when the system is prepared into the pure state represented by the unit vector $\Psi$ and the corresponding standard deviation $(\Delta A)_\Psi = \langle (A - \langle A \rangle_\Psi) | (A - \langle A \rangle_\Psi) \rangle_\Psi$, the two descriptions on $C^2 \otimes C^2$ and $C^4$ by state–observable pairs $(\Psi, A)$ and $(\Psi_{CB}, A_{CB})$ are physically indistinguishable since:

$$\langle A \rangle_\Psi = \langle A_{CB} \rangle_{\Psi_{CB}} \quad \text{and} \quad (\Delta A)_\Psi = (\Delta A_{CB})_{\Psi_{CB}}$$

Let us notice that linear operators which act on $n$–registers can be represented as order $2^n$ square matrices of complex entries.

### 3.2 Single qubit quantum gates

In this paper we are interested to give a quantum description of physical systems acting as Boolean gates. As state above, in a classical computer the processing of information is performed by logic gates. A logic gate maps the state of its input bits into another state according to a truth table. The only nontrivial single bit classical gate is the Not gate, a one-bit gate which negates the state of the input bit: 0 becomes 1 and vice versa. The corresponding quantum gate is implemented via a unitary operator that evolves the states of the computational basis of $C^2$ into the corresponding states of the same basis according to the same truth table. For instance, the quantum version of the Not is the unitary operator $U_{\text{Not}}$ such that:

$$\begin{align*}
U_{\text{Not}}|0\rangle &= |1\rangle \\
U_{\text{Not}}|1\rangle &= |0\rangle
\end{align*}$$

(3)

corresponding to the unitary matrix operator:

$$U_{\text{Not}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Of course there exist also genuine quantum gates, i.e., gates that have no classical counterpart and thus characterized by the fact that some input configurations (tensor product of vectors of the computational basis) are transformed into non–trivial superpositions of configurations. An example of an operation of this kind is the $\sqrt{\text{Not}}$ gate, acting on configurations of a single qubit, that can be thought of as the realization of a 1–register. Formally, the map $\sqrt{\text{Not}} : C^2 \rightarrow C^2$ is described by the order 2 unitary matrix:

$$\sqrt{\text{Not}} := \frac{1}{2} \begin{bmatrix} 1 + i & 1 - i \\ 1 - i & 1 + i \end{bmatrix}$$

With respect to this gate, the elements of the computational basis are transformed according to the rules:

$$\begin{align*}
\sqrt{\text{Not}}(|0\rangle) &= \frac{1 + i}{2} |0\rangle + \frac{1 - i}{2} |1\rangle \\
\sqrt{\text{Not}}(|1\rangle) &= \frac{1 + i}{2} |0\rangle + \frac{1 - i}{2} |1\rangle
\end{align*}$$

(4)
and so the quantum representative of classical truth values are not transformed into classical representative of classical truth values. This means, in particular, that there is no classical one–input/one–output truth table corresponding to the $\sqrt{\text{Not}}$ gate. Trivially, $\sqrt{\text{Not}} \circ \sqrt{\text{Not}} = \text{Not}.$

Another possible realization of this kind of gate is the map $\sqrt{\text{Not}}^{(i)} : C^2 \to C^2$ defined by the unitary matrix:

$$
\sqrt{\text{Not}}^{(i)} := \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}
$$

Also in this case the quantum representatives of classical truth values are not transformed into quantum representatives of classical truth values (compare with (3)).

\[
\begin{align*}
\sqrt{\text{Not}}^{(i)} |0\rangle &= \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \\
\sqrt{\text{Not}}^{(i)} |1\rangle &= \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle
\end{align*}
\] (5)

Trivially, $\sqrt{\text{Not}}^{(i)} \circ \sqrt{\text{Not}}^{(i)} = i \text{Not},$ which corresponds to the $\text{Not}$ gate up to the phase factor $i.$

Another genuine one qubit quantum gate is the Hadamard gate, also acting on quantum registers of size 1. Formally, the map $U_H : C^2 \to C^2$ is described by the following order 2 unitary matrix:

$$
U_H := \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
$$

and the corresponding action on qubits is given by:

\[
\begin{align*}
U_H |0\rangle &= \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \\
U_H |1\rangle &= \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle
\end{align*}
\] (6)

We need to say also a word of caution about the FanOut function. As we have seen, when dealing with classical (non–reversible) circuits the FanOut function is implemented by sticking two output wires to an existing input wire, whereas if we want to perform computations in a reversible way we have to design an appropriate gate to perform the FanOut. When we deal with quantum circuits, we have to take also into account the so called no cloning theorem ([26, 27]), which states that there is no unitary operator that duplicates any possible quantum state. However, this is not a problem for our purposes, since our aim is to give a quantum simulation of classical gates. This means that we need to duplicate only those states which correspond to classical (Boolean, or $d$–valued) truth values, and not each possible quantum state such as, for example, superpositions of vectors from the canonical orthonormal basis.

3.3 The two–levels single system Hamiltonian

In describing a computer it is important, from the point of view of quantum mechanics, to give the Hamiltonian operator for the physical system that constitutes the computing machinery. As it is well known, the Hamiltonian operator describes the energy of the quantum system and allows one to derive its time evolution.

In the case of a two levels quantum system described by the Hilbert space $C^2,$ the Hamiltonian can be expressed by the diagonal matrix:

$$
H = \begin{bmatrix} \varepsilon_0 & 0 \\ 0 & \varepsilon_1 \end{bmatrix} = \varepsilon_0 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \varepsilon_1 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}
$$ (7)
where the two real eigenvalues $\varepsilon_0 < \varepsilon_1$ represent the ground and the excited energy levels respectively. The vector describing the eigenstate of ground (resp., excited) energy $\varepsilon_0$ (resp., $\varepsilon_1$) is $|H = \varepsilon_0\rangle = |0\rangle$ (resp., $|H = \varepsilon_1\rangle = |1\rangle$). In the spectral resolution of the Hamiltonian $H$ (see the second identity of (7)), the orthogonal projections $P_{H}(\varepsilon_0)$ and $P_{H}(\varepsilon_1)$ describing the sharp events “a measure of energy yields the value $\varepsilon_0$” and “a measure of energy yields the value $\varepsilon_1$” are respectively expressed by the matrices:

$$P_{H}(\varepsilon_0) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad P_{H}(\varepsilon_1) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

### 3.4 The spin–$1/2$ Pauli basis of order 2 complex matrices

The collection of all order 2 complex matrices is a 4–dimensional complex linear space, a basis of which is composed by Pauli’s matrices $\sigma_k$, with $k \in \{x, y, z\}$, plus the order 2 identity matrix $I_2$:

$$\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Once stressed that the above matrices of the Pauli basis are self–adjoint, one has that $\{\sigma_x, \sigma_y, \sigma_z, I_2\}$ is also a basis of the 4–dimensional real linear space of all self–adjoint matrices on the space $\mathbb{C}^2$.

Let us recall that the $z$ component of a spin–$1/2$ angular momentum is described by the matrix

$$J_z^{(1/2)} = \frac{\hbar}{2} \sigma_z = \frac{\hbar}{2} \begin{bmatrix} 0 & 0 \\ 0 & -\frac{\hbar}{2} \end{bmatrix} = \frac{\hbar}{2} \begin{bmatrix} 1 & 0 \\ 0 & \frac{\hbar}{2} \end{bmatrix} = \frac{\hbar}{2} \sigma_z (8)$$

whose spectral resolution in the second identity allows one to represent the events “a measure of the spin $z$ component yields the value $\frac{\hbar}{2}$” and “a measure of the spin $z$ component yields the value $-\frac{\hbar}{2}$” by the orthogonal projections:

$$P_{J_z^{(1/2)}} \left( \frac{\hbar}{2} \right) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad P_{J_z^{(1/2)}} \left( -\frac{\hbar}{2} \right) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

The spin–$1/2$ eigenvectors corresponding to the eigenvalues $-\frac{\hbar}{2}$ and $+\frac{\hbar}{2}$ are $|J_z^{(1/2)} = -\frac{\hbar}{2}\rangle = |1\rangle$ and $|J_z^{(1/2)} = +\frac{\hbar}{2}\rangle = |0\rangle$, respectively.

In the particular case where $\varepsilon_0 := -\frac{\hbar \omega_0}{2} = -\frac{\hbar v_0}{2}$ and $\varepsilon_1 := \frac{\hbar \omega_0}{2} = \frac{\hbar v_0}{2}$, corresponding to an energy quantum jump $\Delta \varepsilon = \hbar \omega_0 = \hbar v_0$ between the eigenstates $|0\rangle$ and $|1\rangle$, the above Hamiltonian (7) assumes the form:

$$H^{(1/2)} = -\omega_0 J_z^{(1/2)} = -\frac{\hbar v_0}{2} \sigma_z$$

This is the Hamiltonian of a spin $1/2$ particle in a uniform magnetic field $\vec{B}_0$, chosen the $Oz$ axis along $\vec{B}_0$ and settled $\omega_0 = \gamma B_0$.

### 3.5 The fermion-$1/2$ canonical basis of order 2 complex matrices

Another basis of the order 2 complex matrices linear space is the canonical one:
\[ N' = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad a = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad a^\dagger = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad N = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \]

Notice that in this basis only \( N \) and \( N' \) are self-adjoint (and thus they represent some observables of the system), whereas \( a \) and \( a^\dagger \) are one the adjoint of the other. Any linear operator on \( C^2 \) can be expressed as a linear combination of the operators \( a, a^\dagger, N \) and \( N' \).

However, exploiting the whole algebraic structure of the associative algebra of linear operators (in particular, the composition operator) we can generate any linear operator on \( C^2 \) using only the pair of operators \( a \) and \( a^\dagger \), since:

\[ N = a^\dagger a \quad N' = aa^\dagger \]

In particular, being \( N |0\rangle = 0 |0\rangle \) e \( N |1\rangle = 1 |1\rangle \), this self-adjoint operator can be interpreted as the observable number of particles of a systems consisting of at most 1 particle. Precisely, the ket \( |0\rangle \) (resp., \( |1\rangle \)) describes the eigenstate of zero (resp., one) particle in the system. The spectral resolution of \( N \) is:

\[ N = 0 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + 1 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \]

Hence, the events “the total number of particles is 0” and “the total number of particles is 1” are realized by the orthogonal projections:

\[ P_N(0) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad P_N(1) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \]

In the state \( |0\rangle \) (resp., \( |1\rangle \)) the event \( P_N(0) \) (resp., \( P_N(1) \)) is certain, i.e., it occurs with probability one. All this in agreement with the above interpretation: \( |N = 0\rangle = |0\rangle \) and \( |N = 1\rangle = |1\rangle \) are the eigenstates of the system composed by zero and the eigenstate of the system composed by one particle, respectively.

The operator \( a^\dagger \) transforms the vacuum state \( |N = 0\rangle \) into the one-particle state \( |N = 1\rangle \). If the state of the system was originally \( |N = 1\rangle \) then this operator produces the null vector \( 0 \); an alternative point of view is that the operator, applied to the state \( |N = 1\rangle \), does not change this state and produces the number 0 as an “output value”: \( a^\dagger |1\rangle = 0 |1\rangle = 0 \). Hence, \( a^\dagger \) can be interpreted as a creation operator.

Analogously, \( a \) is an operator which transforms the one-particle state \( |N = 1\rangle \) into the vacuum state \( |N = 0\rangle \). If the input state of the system was originally \( |N = 0\rangle \) then this operator produces the null vector \( 0 \); also in this case, an alternative point of view is that this operator, applied to the state \( |N = 0\rangle \), does not change this state and produces the number 0 as an “output value”: \( a |0\rangle = 0 |0\rangle = 0 \). Hence, \( a \) can be interpreted as an annihilation operator.

Also in the present finite dimensional case of \( C^2 \) one has that the canonical anticommutation rule for fermions (semi-integer spin particles, in our case 1/2) \( \{a, a^\dagger\} = a^\dagger a + aa^\dagger = N + N' = I_2 \) are satisfied; moreover \( [a, a^\dagger] = \sigma_z \). Summarizing:

\[ \{a, a^\dagger\} = I_2 \quad [a, a^\dagger] = \sigma_z \]
They also satisfy:
\[ a^2 = (a^\dagger)^2 = 0 \]  
\[ (10) \]

On the basis of these operators, once introduced \( \Delta \epsilon := \epsilon_1 - \epsilon_2 \), i.e., the energy quantum jump between the two energy levels, the Hamiltonian (7) can be rewritten as

\[ H = \epsilon_0 I_2 + \Delta \epsilon N = \epsilon_0 I_2 + \Delta \epsilon a^\dagger a \]

In particular, one can consider the two levels Hamiltonian on \( C^2 \) with energy quantum jump \( \Delta \epsilon = \bar{\hbar} \omega_0 \) and fundamental energy level \( \epsilon_0 = \frac{1}{2} \bar{\hbar} \omega_0 \):

\[ \hat{H} = \bar{\hbar} \omega_0 \left( \frac{1}{2} I_2 + \hat{a}^\dagger \hat{a} \right) = \left( \begin{array}{cc} \frac{1}{2} \hbar \omega_0 & 0 \\ 0 & \frac{3}{2} \hbar \omega_0 \end{array} \right) \]  
\[ (11) \]

In the first of the above identities, the Hamiltonian resembles the one of the single–particle infinite dimensional quantum harmonic oscillator, but applied to the Hilbert space \( C^2 \). Whereas in the second identity of (11) one realizes that the presently adopted hamiltonian can be considered as the infinite dimensional harmonic oscillator truncated to the second eigenvalue.

One can also try to mimic the infinite dimensional approach to creation and annihilation operators introducing the following two (self–adjoint) operators as the finite dimensional quantum version on \( C^2 \) of the position and momentum observables (which depend upon the real parameter \( \eta \)):

\[ \hat{x} := \sqrt{\frac{\hbar}{2 \eta}} (\hat{a} + \hat{a}^\dagger) = \sqrt{\frac{\hbar}{2 \eta}} \sigma_x \]
\[ \hat{p} := -i \sqrt{\frac{\hbar \eta}{2}} (\hat{a} - \hat{a}^\dagger) = \sqrt{\frac{\hbar \eta}{2}} \sigma_y \]

However these operators satisfy the commutation relation \([\hat{x}, \hat{p}] = \frac{\hbar}{2} [\sigma_x, \sigma_y] = i \hbar \sigma_z \) (and not the canonical commutation relation \([\hat{x}, \hat{p}] = i \hbar I \)). Using these operators, one has the following Hamiltonian with the single degenerate energy eigenvalue \( \frac{\bar{\hbar} \omega_0}{2} \) (putting \( \eta = m \bar{\hbar} \omega_0 \)):

\[ \hat{H}_d = \frac{1}{2m} [\hat{p}^2 + \eta^2 \hat{x}^2] = \frac{\hbar \omega_0}{2} I_2 \]

Therefore, differently from the general infinite dimensional case we have that \( \hat{H}_d \neq \hat{H} \).

Lastly, the Hamiltonian (8) can be written as:

\[ H^{(1/2)} = \frac{\hbar \omega_0}{2} [a^\dagger, a] \]

Let us remark that in this spin context, the non–Hermitian operators \( a \) and \( a^\dagger \) can be expressed with respect to the Pauli basis as

\[ a = \frac{1}{2} (\sigma_x + i \sigma_y) = \frac{\hbar}{2} J_+ \quad \quad a^\dagger = \frac{1}{2} (\sigma_x - i \sigma_y) = \frac{\hbar}{2} J_- \]

where \( J_+ = \frac{\hbar}{2} (\sigma_x + i \sigma_y) \) and \( J_- = \frac{\hbar}{2} (\sigma_x - i \sigma_y) \) are the spin 1/2 “rising” and “lowering” operators, respectively, since \( J_+ \mid -\frac{1}{2} \hbar \rangle = \mid +\frac{1}{2} \hbar \rangle \) (so rising the spin of the particle) and \( J_+ \mid \frac{1}{2} \hbar \rangle = 0 \) (similarly for \( J_- \)). Thus, spin rising is just the number annihilation, and vice versa.
3.6 Creation and annihilation for quantum registers

In section 3.5 creation and annihilation operators on a single qubit register $C^2$ have been introduced. Since the logical gates we shall consider in the sequel involve quantum registers of length 2 or 3, that is Hilbert spaces $C^2 \otimes C^2$ or $\otimes^3(C^2)$, we want to extend the application of the above operators to this context of tensor product of the Hilbert space $C^2$. We will discuss the simple case of length-2 registers, since the extension to the general case is straightforward.

We will use a slight modification in the notation, precisely we denote by $(C^2)_1 \otimes (C^2)_2$ the tensor product of the two identical Hilbert spaces $C^2$ where the lower script denotes the qubit one refers to. Each of the Hilbert space $(C^2)_i$ ($i = 1, 2$) can be equipped with its own pair of creation–annihilation operators $a_i$ and $a_i^\dagger$, through which all other linear operators on the same space can be expressed making use of the algebraic operations of sum $+$, product with respect to complex numbers $\cdot$, and composition $\circ$.

We can now extend these operators to the tensor product in the following way:

$$\hat{a}_1 := a_1 \otimes I, \quad \hat{a}_1^\dagger := a_1^\dagger \otimes I,$$

$$\hat{a}_2 := I \otimes a_2, \quad \hat{a}_2^\dagger := I \otimes a_2^\dagger$$

and also, the number of particles for each qubit:

$$\hat{N}_1 = \hat{a}_1^\dagger \circ \hat{a}_1 = (a_1^\dagger \circ a_1) \otimes I = N_1 \otimes I$$

$$\hat{N}_2 = \hat{a}_2^\dagger \circ \hat{a}_2 = I \otimes (a_2^\dagger \circ a_2) = I \otimes N_2$$

It is easy to verify that the usual anticommutation relations for fermions (extension to the present case of (9)) are satisfied:

$$\{ \hat{a}_1, \hat{a}_1^\dagger \} = \hat{I}, \quad [\hat{a}_1, \hat{a}_1^\dagger] = \hat{\sigma}_{1,z} \quad (12a)$$

$$\{ \hat{a}_2, \hat{a}_2^\dagger \} = \hat{I}, \quad [\hat{a}_2, \hat{a}_2^\dagger] = \hat{\sigma}_{2,z} \quad (12b)$$

where $\hat{I} := I \otimes I$, $\hat{\sigma}_{1,z} = \sigma_z \otimes I$, and $\hat{\sigma}_{2,z} = I \otimes \sigma_z$.

On the contrary, the commutations relations hold for each couple of operators on decoupled qubits. In particular:

$$[\hat{a}_1, \hat{a}_2^\dagger] = [\hat{a}_1^\dagger, \hat{a}_2] = 0$$

$$[\hat{N}_1, \hat{N}_2] = 0$$

The number of particles operators $N_i$ on $(C)_i$ have the eigenstate $|0\rangle$ and $|1\rangle$ (i.e., $N_i$ $|n_i\rangle = n_i$ $|n_i\rangle$, with $n_i = 0, 1$). Therefore, $\{|n_1, n_2\} = |n_1\rangle \otimes |n_2\rangle : n_1, n_2 \in \{0, 1\}$ is an orthonormal basis of $(C^2)_1 \otimes (C^2)_2$.

Moreover, we can introduce the self-adjoint operator $\hat{N} = \hat{N}_1 + \hat{N}_2$ which can be interpreted as the observable “number of particles of a system consisting of at most 2 particles”. We obtain:

$$\hat{N} |n_1, n_2\rangle = (n_1 + n_2) |n_1, n_2\rangle$$

$|0, 0\rangle$ is the eigenstate of the system composed by zero particles, $\{|0, 1\rangle, |1, 0\rangle\}$ are the eigenstates of the system composed by one particle (double degeneracies of the eigenvalue 1) and
$|1, 1\rangle$ is the eigenstate of the system composed by two particles. For the creation operators, we have:

\[
\hat{a}_1^\dagger |0, n_2\rangle = |1, n_2\rangle \quad \hat{a}_1^\dagger |1, n_2\rangle = 0
\]

\[
\hat{a}_2 |n_1, 0\rangle = |n_1, 1\rangle \quad \hat{a}_2 |n_1, 1\rangle = 0
\]

Analogously, for the annihilation operators, we have:

\[
\hat{a}_1 |0, n_2\rangle = 0 \quad \hat{a}_1 |1, n_2\rangle = 0
\]

\[
\hat{a}_2 |n_1, 0\rangle = 0 \quad \hat{a}_2 |n_1, 1\rangle = 0
\]

Notice that we used notation $\hat{a}_1, \hat{a}_2, \hat{a}_3, \ldots$ instead of the one $\hat{a}, \hat{b}, \hat{c}, \ldots$, which is commonly used in the description of quantum logical gates on registers of length more than one (see, for instance the controlled-not gate by Feynman in [13]).

Moreover, in this simplified notation one also omits the composition operator $\circ$. For example, the operator $\hat{a}_1^\dagger \circ \hat{a}_1 \circ (\hat{a}_2^\dagger + \hat{a}_2) + \hat{a}_1 \circ \hat{a}_1^\dagger$ on the Hilbert space $C^2 \otimes C^2$, written according to the complete notation, is simplified as $a^\dagger a (b^\dagger + b) + a a^\dagger$.

### 4 Quantum Realization of Boolean Gates

Now that we have a mathematical model to interpret boolean truth values as vectors of the qubit Hilbert space $C^2$, and the quantum version of Boolean $n$–input/$n$–output gates as linear operators $G : \otimes^n C^2 \rightarrow \otimes^n C^2$, let us address the following problem.

Given the truth table of a Boolean $n$–input/$n$–output gate $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$, express the linear operator $G_f : \otimes^n C^2 \rightarrow \otimes^n C^2$ that furnishes the quantum realization of the gate as a formula containing only the linear operators $I_2, a^\dagger, a$ and the algebraic operations $+, -, \circ, \otimes$ among them.

As an example of solution to an instance of the above problem, the operator (3) corresponding to logical negation (NOT) can be expressed as:

\[
\text{NOT} := a + a^\dagger = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]

It is immediate to see that this unitary operator is self–reversible.

As another example, the reversible non–conservative 2–input/2–output gate which negates the first input and leaves unchanged the second input can be expressed through the following unitary operator:

\[
\text{NOT} \otimes I_2 = (a + a^\dagger) \otimes I_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}
\]

As these two examples show, if the Boolean gate $f$ is reversible then the corresponding operator $G_f$ is unitary, as commonly required in quantum computing. However, we are interested to give a quantum description of all possible Boolean $n$–input/$n$–output gates and thus, as we will see in the sequel, generally the operators we will obtain are not necessarily unitary.
In the next three sections we expose three methods that can be used to implement any \(n\)-input/\(n\)-output Boolean gate, the so called “brute force” method and our proposed Constants method, as well as an extension of the Conditional Quantum Control method — originally proposed by Barenco, Deutsch, Ekert and Jozsa in [16] — that allows one to easily express any Conditional Control gate.

### 4.1 The “brute force” quantum realization of boolean gates

Observe that the construction of the operator shown in equation (13) is particularly simple since the gate acts onto the states of the two inputs separately; in other words, the action of the operator on each input is independent of the state of the other input. The case where the action of the gate on one input depends on the state of one or more of the other inputs is of course more complicated. However, we can still write the global operator as a sum of so called local operators, by a “brute force” procedure, where each local operator corresponds to a single row of the truth table which describes the gate. Precisely, in order to translate the generic boolean truth table row:

\[
x_1, x_2, \ldots, x_n \mapsto y_1, y_2, \ldots, y_n
\]

we build the “local” operator:

\[
E_{x_1, y_1} \otimes E_{x_2, y_2} \otimes \cdots \otimes E_{x_n, y_n}
\]

where \(E_{x,y} := |y\rangle \langle x|\), with \(x, y \in \{0,1\}\), is the operator that transforms the single qubit vector \(|x\rangle\) into the vector \(|y\rangle\), and returns the null vector if it is applied to the other vector of the computational basis \(\{|0\rangle, |1\rangle\}\).

For instance, the above operator in equation (13) can be written as:

\[
\text{Not} \otimes I_2 = E_{0,1} \otimes E_{0,0} + E_{0,1} \otimes E_{1,1} + E_{1,0} \otimes E_{0,0} + E_{1,0} \otimes E_{1,1}
\]

Since the quantum realization of the general truth table rule is composed by operators of the form \(E_{x,y}\), from the fact that

\[
E_{1,1} = a^\dagger a \quad \quad \quad \quad E_{0,1} = a^\dagger \\
E_{1,0} = a \quad \quad \quad \quad E_{0,0} = aa^\dagger
\]

we can conclude that every local operator is a tensor product of suitable compositions of creation and annihilation operators.

Note that this brute force procedure in some sense can be applied also to genuine quantum gates. For instance the \(\sqrt{\text{Not}}\) gate, once considered the superpositions \(\psi_0 = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle\) and \(\psi_1 = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle\), can be written as \(\sqrt{\text{Not}} = E_{0,\psi_0} + E_{1,\psi_1}\), where for \(x \in \{0,1\}\) we define \(E_{x,\psi_x} := |\psi_x\rangle \langle x|\). More generally, the generic truth table row of a genuine quantum gate (whatever be the logical meaning of this expression) \(x_1, x_2, \ldots, x_n \mapsto \psi_{x_1}, \psi_{x_2}, \ldots, \psi_{x_n}\) is realized by the operator \(E_{x_1,\psi_{x_1}} \otimes E_{x_2,\psi_{x_2}} \otimes \cdots \otimes E_{x_n,\psi_{x_n}}\).

### 4.2 The generalized “conditional quantum control” method

Let us apply a method derived from Conditional Quantum Control [16] on a simple truth table of a non–reversible non–conservative 2–input/2–output Boolean gate:
We can describe the behavior of this gate by considering $x_1$ as a control input which is left unchanged but which determines the action of a prescribed operation on the operating (sometimes, target) input $x_2$, that transforms it into the output $y_2$.

The quantum realization of such “controlled behavior” can be obtained by making use of the operators $N = E_{1,1} = \ket{1}\bra{1}$ and $N' = E_{0,0} = \ket{0}\bra{0}$, in this context considered as simplified form of $a^\dagger a$ and $aa^\dagger$ respectively. Indeed, if we want to realize a gate performing the condition: “if the control qubit is $\ket{1}$ then the operator $O_1$ is applied to the operating qubit (and the control qubit is left unchanged)”, then we can build the operator $N \otimes O_1$, where $N$ checks for the condition “the control qubit is $\ket{1}$” and $O_1$ is the operator which acts on the operating qubit $\ket{x_2}$. Note that if the control qubit is $\ket{0}$ then the operator $N \otimes O_1$ produces the null vector of $C^2 \otimes C^2$. Similarly, $N' \otimes O_0$ realizes the condition “if the control qubit is $\ket{0}$ then the operator $O_0$ is applied to the operating qubit $\ket{x_2}$ (and the control qubit is left unchanged)”.

In particular, in the case of the Boolean gate described by the truth table above we can distinguish the following two cases:

- if $x_1 = 1$ then $y_2 = x_2$. This condition is immediately realized by the operator $N \otimes I_2$;
- if $x_1 = 0$ then $y_2 = 1$, whatever be the operating input $x_2$. In this case we need an operator that transforms the state $\ket{0}$ of the second qubit into the $\ket{1}$ and that leaves the second qubit state $\ket{1}$ unchanged. Such an operator can be obtained as the sum of the following two operators: $a^\dagger$, which transforms the state $\ket{0}$ into $\ket{1}$ collapsing $\ket{1}$ to the null vector, and $N$, which leaves $\ket{1}$ unchanged collapsing $\ket{0}$ to the null vector. Thus, the required operator is $N' \otimes (a^\dagger + N)$, where $N'$ selects the qubit $\ket{0}$ on the control line leaving this state unchanged, and $a^\dagger + N$ performs the prescribed transformations on the input of the operating qubit.

From the sum of the operators defined in the two points above we get the non–unitary operator which realizes the given truth table:

$$G = N \otimes I_2 + N' \otimes (a^\dagger + N) \equiv \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Notice that the method we have used here to design the gate is a generalization of the Conditional Quantum Control method of [16]. In fact, recall that in Conditional Control gates $2^k$ functions $\delta_0, \ldots, \delta_{2^k-1}$ are stored in the memory of the control unit, the function $\delta_a$ being bijectively associated to the control input configuration labelled by the integer number $a \in \{0, \ldots, 2^k-1\}$ (see Figure 1). In [16] these functions are described through unitary operators $U_0, U_1, \ldots, U_{2^k-1}$, defined on the Hilbert space $\otimes^{n-k} C^2$; here we drop the requirement that
such operators, as well as the global operator defined on $\otimes^n \mathbb{C}^2$, be unitary. Moreover, in the sequel we will apply this method to realize $d$-valued gates.

Thus, when a configuration $|x_1, \ldots, x_k\rangle$ is fed to the control lines of a Conditional Control gate two things happen:

- the control configuration $|x_1, \ldots, x_k\rangle$ is returned unchanged into the output lines of the control unit, and

- the (non necessarily unitary) operator $U_a$ bijectively associated to the control configuration is selected and applied to the input configuration $|x_{k+1}, \ldots, x_n\rangle$ of the target unit, producing the output configuration $U_a|x_{k+1}, \ldots, x_n\rangle$.

The global operator on $[\otimes^k \mathbb{C}^2] \otimes [\otimes^{n-k} \mathbb{C}^2]$ which describes the behavior of the gate can thus be written as:

$$P_0 \otimes U_0 + P_1 \otimes U_1 + \ldots + P_{2^k-1} \otimes U_{2^k-1}$$

(14)

where $P_a = E_{a,a} = |a\rangle \langle a|$ is the orthogonal projection of the Hilbert space $\otimes^k \mathbb{C}^2$ which selects the $a$-th control configuration and collapses to the null vector all the other control configurations, and $U_a$ is the corresponding operator on $\otimes^{n-k} \mathbb{C}^2$ which has to be applied to the target configuration. Making use of Dirac notation, expression (14) can be equivalently written as (see [16]):

$$|0\rangle \langle 0| \otimes U_0 + |1\rangle \langle 1| \otimes U_1 + \ldots + |2^k-1\rangle \langle 2^k-1| \otimes U_{2^k-1}$$

(15)

Summarizing, “in order to implement a [quantum controlled gate] it is sufficient, from the experimental point of view, to induce a conditional control of physical bits, i.e., to perform [suitable] transformations on one physical subsystem conditioned upon the quantum state of another subsystems [according to (15)], where the projectors refer to quantum states of the control subsystem and the operations $U_a$ are performed on the target subsystem.” [28].

Let us note that if in (14) all operators $U_a$ are unitary, then the resulting operator is unitary too and this correspond to the original Conditional Quantum Control method.

As an example, the quantum realization of the Controlled–Not, C\text{Not} : $C^2 \otimes C^2 \rightarrow C^2 \otimes C^2$, gate with the Conditional Quantum Control method is the following unitary operator.
on two qubits:

$$\text{CNOT} = |0\rangle \langle 0| \otimes I_2 + |1\rangle \langle 1| \otimes \text{Not}$$

It is immediately seen that the action of this unitary quantum realization of the Controlled–Not gate on configurations of length 2, $|x_1, x_2\rangle$ for $x_1, x_2 \in \{0, 1\}$, is the following:

$$\text{CNOT}(|x_1, x_2\rangle) = |x_1, x_1 \oplus x_2\rangle$$

Let us note that fixing to 0 the second qubit one obtains the “clonation” of the vectors of the computational basis:

$$\text{CNOT}(|x_1, 0\rangle) = |x_1, x_1\rangle \quad \text{for any } x_1 \in \{0, 1\}$$

but there are well known no–go theorems about the extension of this result to all vectors of the Hilbert space $C^2$ [26, 27].

The major drawback of the method derived from Conditional Quantum Control is that it does not allow one to describe every conceivable operator, since there are gates that cannot be divided into a control unit and an operating unit.

### 4.3 The constants method

In this section we introduce a new method to describe 2–input/2–output Boolean gates in the quantum context, particularly useful when the behaviors of the gate under consideration and of the identity gate differ only for few input/output pairs. In such a case it is convenient to build the global operator as the identity plus an operator which takes into account the cases where the gate behaves differently from the identity gate. In other words, we choose constants and input lines to be fixed with such constants in order to minimize the length of expression (16).

As an example of application of our method, let us consider a two–input/two–output non–reversible and non–conservative Boolean gate whose behavior is identical to the identity for all but the input/output transition $(1, 1) \mapsto (1, 0)$. The corresponding operator can be defined as:

$$I + O$$

\[(16)\]

The method works as follows. Let us suppose that we want to design a 2–input/2–output Boolean gate which realizes some set $C$ of connectives. These connectives are realized by fixing some input lines of the gate with constant values from $\{0, 1\}$; if we appropriately choose these input lines and constant values, we can minimize the number of input/output pairs for which the gate behaves differently from the identity gate. In other words, we choose constants and input lines to be fixed with such constants in order to minimize the length of expression (16).

As an example of application of our method, let us consider a two–input/two–output non–reversible and non–conservative Boolean gate whose behavior is identical to the identity for all but the input/output transition $(1, 1) \mapsto (1, 0)$. The corresponding operator can be defined as:

$$I_1 + N \otimes (E_{1, 0} - E_{1, 1})$$

In this expression, the first term coincides with the identity gate and the second term applies $(E_{1, 0} - E_{1, 1})$ on $|x_2\rangle$ when $|x_1\rangle = |1\rangle$; the operator $E_{1, 0}$ transforms $|x_2\rangle = |1\rangle$ into $|y_2\rangle = |0\rangle$, while $E_{1, 1}$ is subtracted to cancel the effect of the identity operator.

The quantum realizations of $f_{d, 1}, f_{d, \lambda}^1$ and $m_{d, 1}$ have been obtained applying the Constants method. As we will see, for function $f_{d, 1}$ the Constants method allows to get $\Theta(d^2)$ terms in the corresponding expression versus the $\Theta(d^3)$ terms obtained with Conditional Quantum Control. A similar result has been obtained with function $m_{d, 1}$; whereas for functions $f_{d, \lambda}^2$ the Conditional Quantum Control technique cannot be applied.
4.4 Implementation of some universal gates

In this section we give a quantum description of some celebrated universal gates, each time using the most convenient method between the ones we have exposed. The gates here considered are the Petri–Fredkin gate [1], the Feynman Controlled–Controlled–Not [13] and the Peres gate [14].

4.4.1 The self–reversible conservative Petri–Fredkin gate

We recall that the Petri–Fredkin gate is described by the following Boolean equations:

\[
\begin{align*}
    y_1 &= x_1 \\
    y_2 &= (x_1 \land x_2) \lor (\neg x_1 \land x_3) \\
    y_3 &= (\neg x_1 \land x_2) \lor (x_1 \land x_3)
\end{align*}
\]

The set of Boolean primitives generated by this gate is:

\[
\{ \lor, \land, \rightarrow, \text{FanOut}, \Pr_1, \Pr_2, \neg, \neg \rightarrow \}
\]

where \( \Pr_1 \) and \( \Pr_2 \) are the projectors over the first and the second component of a Boolean pair.

It is immediately seen that the input line \( x_1 \) can be considered as a control line: when \( |x_1⟩ = |1⟩ \) the gate behaves as the identity gate, and when \( |x_1⟩ = |0⟩ \) the behavior of the gate can be described by the sum of four “local” operators, each one corresponding to a pattern of Boolean values fed to the other two input lines. Hence, the global operator which describes the Petri–Fredkin gate is (recall that \( N = a^† a \) and \( N' = a^† \)):

\[
M_{PF} = N' \otimes (N' \otimes N + a^† \otimes a + a \otimes a^† + N \otimes N)
\]

\[
\equiv \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} = \begin{bmatrix}
N' & a^† & a & N \\
a & N' & \text{O}_4 \\
\text{O}_4 & \text{I}_4
\end{bmatrix}
\]

In the conventional notation:

\[
M_{PF} = a a^† (b b^† c c^† + b^† c + b c^† + b^† b c^† c) + a^† a
\]

4.4.2 The self–reversible non–conservative Feynman Controlled–Controlled–Not

The Feynman Controlled–Controlled–Not (CCNOT) gate is described by the following Boolean equations:

\[
\begin{align*}
    y_1 &= x_1 \\
    y_2 &= x_2 \\
    y_3 &= [\neg (x_1 \land x_2) \land x_3] \lor [(x_1 \land x_2) \land \neg x_3] = (x_1 \land x_2) \oplus x_3
\end{align*}
\]
where $\oplus$ means the exclusive Or between the arguments. This gate is self-reversible and non-conservative (for example, the triple $(1, 1, 0)$ is mapped to $(1, 1, 1)$) and generates the following set of Boolean primitives:

$$\{\oplus, \land, \text{FanOut}, \text{Pr}_1, \text{Pr}_2, \neg, \neg\land\}$$

We can easily build the corresponding operator by looking at $x_1$ and $x_2$ as control lines and by noting that the third input line is negated only when both $x_1$ and $x_2$ are in state $|1\rangle$, otherwise it is left unchanged. Thus we get the following operator:

$$M_{\text{CCNot}} = I_2 \otimes I_2 \otimes I_2 + N \otimes N \otimes (M_\neg - I_2)$$

As we can see from the matrix above, $M_{\text{CCNot}}$ differs from the identity gate only for two of the eight possible input/output pairs; it is thus convenient to express it as follows:

$$M_{\text{CCNot}} = I + a^\dagger a b^\dagger b(c^\dagger + c - I_2)$$

4.4.3 The reversible non-conservative Peres gate

The Peres gate is described by the following equations:

$$y_1 = (\neg x_2 \land x_3) \lor (x_2 \land \neg x_3) = x_2 \oplus x_3$$

$$y_2 = x_2$$

$$y_3 = (\neg x_1 \land x_2 \land x_3) \lor (x_1 \land \neg(x_2 \land x_3)) = x_1 \oplus (x_2 \land x_3)$$

This gate is reversible and non-conservative (for example, the triple $(1, 1, 1)$ is mapped to $(0, 1, 0)$) and generates the following set of Boolean primitives:

$$\{\lor, \land, \text{FanOut}, \text{Pr}_1, \text{Pr}_2, \neg, \neg\land, \text{FanOut} \neg, \neg\text{Pr}_2\}$$

Although $x_2$ could be considered as a control line, we get no particular advantage by adopting this point of view. Moreover, the behavior of the gate differs from that of the identity gate for five of the eight possible input/output pairs (the unchanged patterns are $(0, 0, 0), (0, 1, 1)$ and $(1, 0, 1)$), and thus expressing the gate through its differences from the
identity gate is of no particular advantage as well. As a consequence, we just look at each row of the gate’s truth table and we build the corresponding local operator. The global operator which describes the Peres gate is thus obtained as the sum of the following eight local operators:

\[
M_{\text{Pe}} = N' \otimes N' \otimes N' + a^\dagger \otimes N' \otimes a \\
+ a^\dagger \otimes N \otimes N' + N' \otimes N \otimes N + a \otimes N' \otimes a^\dagger \\
+ N \otimes N' \otimes N + N \otimes N \otimes a^\dagger + a \otimes N \otimes a
\]

\[
= \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}
= \begin{bmatrix}
N' & O_2 & a^\dagger & O_2 \\
O_2 & N & O_2 & a \\
a & O_2 & N & O_2 \\
O_2 & N' & O_2 & a^\dagger
\end{bmatrix}
\]

In the conventional notation:

\[
M_{\text{Pe}} = a^\dagger b b^\dagger c c^\dagger + a^\dagger b b^\dagger b c c^\dagger + a^\dagger b^\dagger b c c^\dagger \\
+ a b b^\dagger c^\dagger + a^\dagger a b b^\dagger c^\dagger c + a^\dagger a b^\dagger b c^\dagger + a b^\dagger b c
\]

5 Quantum Realization of \(d\)-valued Gates

In this section we extend the mathematical model introduced in section 3 to \(d\)-valued logics. In this setting, the \(d\)-valued versions of qubits are usually called qudits [29]. As it happens with qubits, we can think of qudits as implemented by the energy levels of a quantum system, or as the number of particles in the system, or as the value of the \(z\) component of the spin of a particle. From a strictly mathematical point of view, the \(d\) truth values (with \(d \geq 2\)) of \(L_d = \{0, \frac{1}{d-1}, \frac{2}{d-1}, \ldots, \frac{d-2}{d-1}, 1\}\) can be realized by the unit vectors of the canonical orthonormal basis of the Hilbert space \(C^d\) according to:

\[
|0\rangle = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \frac{1}{d-1} |0\rangle = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \quad \frac{d-2}{d-1} |0\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}
\]

The collection of all linear operators on \(C^d\) is a \(d^2\)-dimensional linear space whose canonical basis is:

\[
\{E_{x,x'} = |x'\rangle \langle x| : x, x' \in L_d\}
\]

Since \(E_{x,x'}|x\rangle = |x'\rangle\) and \(E_{x,x'}|y\rangle = 0\) for every \(y \in L_d\) such that \(y \neq x\), this operator transforms the unit vector \(|x\rangle\) into the unit vector \(|x'\rangle\), collapsing all the other vectors of the canonical orthonormal basis of \(C^d\) to the null vector. The case \(x = x'\) corresponds to the orthogonal projection \(P_x = E_{x,x} = |x\rangle \langle x|\) which leaves unchanged \(|x\rangle\) collapsing to the null vector all \(|y\rangle\)'s with \(y \in L_d\) and \(y \neq x\). For \(i, j \in \{0, 1, \ldots, d-1\}\), the operator \(E_{x+i/\sqrt{d}, x+j/\sqrt{d}}\) can
be represented as an order $d$ square matrix having 1 in position $(j + 1, i + 1)$ and 0 in every other position:

$$E_{\frac{j}{d-1}, \frac{i}{d-1}} = (\delta_{r,j+1}\delta_{i+1,s})_{r,s=1,2,...,d}$$

In the following three sections we show how the vectors of the orthonormal canonical basis of $C^d$ can be interpreted as the eigenvectors of the energy states of a truncated quantum harmonic oscillator. Moreover, we extend creation and annihilation operators in order to work with qudits. We also introduce spin–rising and spin–lowering operators, that allow one to vary the value of the $z$ component of the spin of a particle. Finally, we show how creation (or spin–rising) and annihilation (resp., spin–lowering) can be used to compute the operators $E_{x,y}$, on their turn used to implement $d$–valued $n$–input/$n$–output gates.

### 5.1 The $d$–levels single system Hamiltonian

The quantum realization of $d$–valued one–input/one–output gates can be done by considering single quantum systems whose Hamiltonian on $C^d$ is:

$$H = \begin{bmatrix} \varepsilon_0 & 0 & \cdots & 0 \\ 0 & \varepsilon_0 + \Delta \varepsilon & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon_0 + (d-1)\Delta \varepsilon \end{bmatrix}$$  \hspace{1cm} (17)$$

The energy eigenvalues $\varepsilon_k = \varepsilon_0 + k\Delta \varepsilon$ of $H$, starting from the ground energy state $\varepsilon_0$ and equispaced by the quantum of energy $\Delta \varepsilon$, are the ones of the infinite dimensional quantum harmonic oscillator truncated at the $d-1$ excited level (see Figure 2).

The unit vector $|H = \varepsilon_k\rangle = \left| \frac{k}{\sqrt{d-1}} \right\rangle$, for $k \in \{0,1,\ldots,d-1\}$, is the eigenvector of the state of energy $\varepsilon_0 + k\Delta \varepsilon$. The spectral resolution of the above truncated harmonic oscillator Hamiltonian (17) is:

$$H = \sum_{k=0}^{d-1} (\varepsilon_0 + k\Delta \varepsilon)P_{\varepsilon_k}$$

where each orthogonal projection $P_{\varepsilon_k} = P_{\frac{k}{\sqrt{d-1}}}$ is the quantum realization of the sharp event “a measure of the system energy yields the value $\varepsilon_0 + k\Delta \varepsilon$".
5.2 Creation and annihilation on qudits

We extend now the creation and annihilation operators considered in section 3.5 in order to deal with the $d$–dimensional Hilbert spaces $C^d$. A natural way to perform such extension is to define the creation operator in such a way that, when applied to a state different from $|1\rangle$ it returns the “next” state in the ordering from $|0\rangle$ to $|1\rangle$, while if applied to the state $|1\rangle$ it returns the null vector or — as we have seen above for two–states systems — it returns the state $|1\rangle$ with eigenvalue 0. The annihilation operator is defined in an analogous way: when applied to a state different from $|0\rangle$ it returns the “previous” state in the ordering from $|0\rangle$ to $|1\rangle$ while, if applied to the state $|0\rangle$, it returns the null vector or — in the alternative point of view — it returns the state $|0\rangle$ with eigenvalue 0.

Formally, creation and annihilation operators on the Hilbert space $C^d$ are respectively defined as:

\[
\hat{a}^\dagger = \begin{bmatrix}
0 & 0 & \cdots & 0 & 0 \\
1 & 0 & \cdots & 0 & 0 \\
0 & \sqrt{2} & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & \sqrt{d-1} & 0
\end{bmatrix}
\quad \hat{a} = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & \sqrt{2} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & \sqrt{d-1}
\end{bmatrix}
\]

The operators $\hat{a}^\dagger$ and $\hat{a}$ are non–Hermitian, adjoints of each other. From these formulas it follows that the action of $\hat{a}^\dagger$ on the vectors of the canonical orthonormal basis of $C^d$ is the following:

\[
\hat{a}^\dagger \left| \frac{k}{d-1} \right\rangle = \sqrt{k+1} \left| \frac{k+1}{d-1} \right\rangle \quad \text{for } k \in \{0,1,\ldots,d-2\}
\]
\[
\hat{a}^\dagger |1\rangle = 0
\]

whereas the action of $\hat{a}$ is:

\[
\hat{a} \left| \frac{k}{d-1} \right\rangle = \sqrt{k} \left| \frac{k-1}{d-1} \right\rangle \quad \text{for } k \in \{1,2,\ldots,d-1\}
\]
\[
\hat{a} |0\rangle = 0
\]

Using $\hat{a}^\dagger$ and $\hat{a}$, we can introduce the following operators representing the $d$–dimensional extension of the corresponding two–dimensional case (see section 3.5):

\[
N = \hat{a}^\dagger \hat{a} = \begin{bmatrix}
0 & 0 & \cdots & 0 & 0 \\
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & d-2 & 0 \\
0 & 0 & \cdots & 0 & d-1
\end{bmatrix}
\quad \hat{a} \hat{a}^\dagger = \begin{bmatrix}
1 & 0 & \cdots & 0 & 0 \\
0 & 2 & 0 & \cdots & 0 \\
0 & 0 & 3 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & d-2 & 0 \\
0 & 0 & \cdots & 0 & d-1
\end{bmatrix}
\]

Hence, these operators satisfy the following commutation and anticommutation relations,
the maximum number of particles that can be simultaneously in the system is
apply the creation operator
it switch to the “previous” state
of energy
null vector as a result.
annihilation operator to an empty particle system does not affect the system and returns the
on the system, and returns as a result the null vector. Analogously, the application of the
application of the creation operator to a full
k
to the
physical systems consisting of a maximum number of
particles state
expression:
of linear operators to express any such operator (i.e., any order
whole algebraic structure (in particular, the composition operation) of the associative algebra
ε
possibility of expressing the Hamiltonian (17) as follows:

Thus, if one excludes the case \( d = 2 \) where it is satisfied the fermion anticommutation rule, nei-
ther the bosons commutation rule \([a, a^\dagger] = I\) nor the fermion anticommutation rule \([a, a^\dagger] = I\)
of the infinite dimensional case hold.
The eigenvalues of the self–adjoint operator \( N \) are \( 0, 1, 2, \ldots, d – 1 \), and the eigenvector

The notation adopted in [29], where the qudit base states are denoted by \(|0\rangle, |1\rangle, \ldots, |d – 1\rangle\), and
it is assumed that a qudit can be in a superposition of the
d base states:

with \( c_i \in \mathbb{C} \) for \( i \in \{0, 1, \ldots, d – 1\} \), and \(|c_0|^2 + |c_1|^2 + \ldots + |c_{d–1}|^2 = 1\).

One possible physical interpretation of operators \( a^\dagger \) and \( a \), by operator \( N \), follows from the
possibility of expressing the Hamiltonian (17) as follows:

In this case \( a^\dagger \) (resp., \( a \)) realizes the transition from the eigenstate of energy \( \epsilon_k = \epsilon_0 + k \Delta \epsilon \)
to the “next” (resp., “previous”) eigenstate of energy \( \epsilon_{k+1} = \epsilon_0 + (k + 1) \Delta \epsilon \) (resp., \( \epsilon_{k–1} = \epsilon_0 + (k – 1) \Delta \epsilon \)) for any \( 0 \leq k < d – 1 \) (resp., \( 0 < k \leq d – 1 \)), while it collapses the last excited
(resp., ground) state of energy \( \epsilon_0 + (d – 1) \Delta \epsilon \) (resp., \( \epsilon_0 \)) to the null vector.

Another physical interpretation of operators \( a^\dagger \) and \( a \), by operator \( N \), follows from the
possibility of expressing the Hamiltonian (17) as follows:

Generalizing the two–dimensional case, also in the \( d \)–dimensional setting we can use the
whole algebraic structure (in particular, the composition operation) of the associative algebra
of linear operators to express any such operator (i.e., any order \( d \) complex matrix) as a linear
combination of suitable compositions of creations and annihilations. Indeed, let \( \mathcal{A}^{p,q,r}_{u,v} \) denote
the expression:

\[ v \ldots v^* v^* \ldots v v^* v \ldots v u \]
where \( u, v \in \{a^\dagger, a\} \), \( v^* \) is the adjoint of \( v \), and \( p, q, r \) are non-negative integer values. For \( i, j \in \{0, 1, \ldots, d - 1\} \), we can express the operator \( E_{\frac{1}{d^2} \cdot \frac{1}{d^2}} \) in terms of creation and annihilation as follows:

\[
E_{\frac{1}{d^2} \cdot \frac{1}{d^2}} = \begin{cases}
\sqrt{\frac{d-2}{d}} A_{a,a}^{d-2,d-1,j,0} & \text{if } i = 0 \\
\sqrt{\frac{d-1}{d}} A_{a,a}^{d-1,d-1,j,0} & \text{if } i = 1 \text{ and } j \geq 1 \\
\frac{\sqrt{d-1}}{\sqrt{(d-1)!}} A_{a,a}^{d-2-i,d-1,j} & \text{if } (i = 1, j = 0 \text{ and } d \geq 3) \text{ or } (1 < i < d - 2 \text{ and } j \leq i) \\
\frac{1}{\sqrt{(d-1)!j!(d-1-j)!}} A_{a,a}^{d-1,d-1,d-1-j} & \text{if } (i = d - 2, j = d - 1 \text{ and } d \geq 3) \text{ or } (1 < i < d - 2 \text{ and } j > i) \\
\frac{1}{\sqrt{(d-1)!j!(d-1-j)!}} A_{a,a}^{d-2,j,0} & \text{if } i = d \text{ and } j \leq d - 2 \\
\frac{1}{\sqrt{(d-1)!j!(d-1-j)!}} A_{a,a}^{d-2,j,0} & \text{if } i = d - 1
\end{cases}
\]

5.3 The angular momentum interpretation of qudits

As it is well known, for a fixed integer \( d \geq 2 \) the angular momentum based on the Hilbert space \( C^d \) consists of the triple of self-adjoint operators \( J = (J_z, J_y, J_x) \). Moreover, for \( j = \frac{d-1}{2} \), the real value \( j(j+1) \) is an eigenvalue of the operator \( J^2 = J_x^2 + J_y^2 + J_z^2 \). The matrix representation of the \( z \) component of this angular momentum with respect to the orthonormal basis of its eigenvectors is:

\[
J_z = \begin{bmatrix}
d-rac{1}{2} & 0 & \cdots & 0 & 0 \\
0 & d-rac{3}{2} & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 3-rac{d}{2} & 0 \\
0 & 0 & \cdots & 0 & \frac{1}{2} - \frac{d}{2}
\end{bmatrix}
\]

Thus, the \( z \) component of the angular momentum can assume \( d \) possible eigenvalues:

\[
m = \frac{d - (2k + 1)}{2} \quad \text{for } k \in \{0, 1, \ldots, d - 1\}
\]

with corresponding eigenvectors:

\[
\left| J_z = \frac{d - (2k + 1)}{2} \right> = \left| \frac{k}{d - 1} \right>
\]

Let us consider the two operators \( J_+ \) and \( J_- \) on the Hilbert space \( C^d \) which are obtained from the general angular momentum operators as:

\[
J_+ = J_x + iJ_y \quad J_- = J_x - iJ_y
\]

The operators \( J_+ \) and \( J_- \) are non-Hermitian, adjoints of each other, and satisfy the canonical commutation relation \([J_+, J_-] = 2J_z\). In matrix form they can be expressed as follows:

\[
J_+ = \begin{bmatrix}
0 & \sqrt{d-1} & 0 & \cdots & 0 & 0 \\
0 & 0 & \sqrt{2(d-2)} & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & \sqrt{2(d-2)} & 0 \\
0 & 0 & 0 & \cdots & 0 & \sqrt{d-1}
\end{bmatrix}
\]
30 Quantum Conservative Many-valued Computing

and

\[
J_- = \begin{bmatrix}
\sqrt{d-1} & 0 & \cdots & 0 & 0 & 0 \\
0 & \sqrt{2(d-2)} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & \cdots & \sqrt{2(d-2)} & 0 & 0 \\
0 & 0 & \cdots & 0 & \sqrt{d-1} & 0 \\
\end{bmatrix}
\]

That is, for \( r, s \in \{1, 2, \ldots, d\} \), the entry in position \((r, s)\) of matrices \(J_+\) and \(J_-\) is, respectively:

\[
(J_+ )_{r,s} = \sqrt{r(r-\delta_{r,s})} \delta_{r,s-1}
\]

\[
(J_- )_{r,s} = \sqrt{s(s-\delta_{r,s+1})} \delta_{r,s+1}
\]

As it is well known, the action of operators \(J_+\) and \(J_-\) on the vectors of the orthonormal basis of \(C^d\) formed by the eigenvectors of \(J_z\) is the following:

\[
J_+ |J_z = m\rangle = \sqrt{j(j+1) - m(m+1)} |J_z = m+1\rangle \quad \text{for } m = -j, \ldots, j
\]

and

\[
J_- |J_z = m\rangle = \sqrt{j(j+1) - m(m-1)} |J_z = m-1\rangle \quad \text{for } m = -j, \ldots, j
\]

Thus, we can interpret these operators as follows: the application of \(J_+\) has the effect of changing the \(z\) component of the angular momentum to the next value. If applied to a system which has already a maximum value of \(J_z\), \(J_+\) leaves the system unchanged and returns as a result the null vector. Analogously, the application of \(J_-\) has the effect of switching the system to the previous value of the \(z\) component of the angular momentum. If applied to a system which has already a minimum value of \(J_z\), \(J_-\) does not affect the system and returns as a result the null vector. In analogy to the creation and annihilation operators introduced in the previous section, we call \(J_+\) the \(spin\text{-}rising\) operator and \(J_-\) the \(spin\text{-}lowering\) operator on \(C^d\).

The actions of \(J_+\) and \(J_-\) on the vectors of the qudit orthonormal basis are the following:

\[
J_+ \begin{bmatrix}
k \\
\frac{d-1}{d}
\end{bmatrix} = \sqrt{k(d-k)} \begin{bmatrix}
k-1 \\
\frac{d-1}{d}
\end{bmatrix} \quad \text{for } k \in \{1, 2, \ldots, d-1\}
\]

\[
J_+ |0\rangle = 0
\]

and

\[
J_- \begin{bmatrix}
k \\
\frac{d-1}{d}
\end{bmatrix} = \sqrt{(k+1)(d-(k+1))} \begin{bmatrix}
k+1 \\
\frac{d-1}{d}
\end{bmatrix} \quad \text{for } k \in \{0, 1, \ldots, d-2\}
\]

\[
J_- |1\rangle = 0
\]
In particular, let us note that $J_+$ switches a qudit to the previous truth value in $L_d$, whereas $J_-$ switches it to the next truth value. Thus, we have that $J_+$ behaves as a spin–rising and, simultaneously, as a truth value annihilation operator, whereas $J_-$ behaves as a spin–lowering and as a truth value creation operator.

Let us note also that when dealing with two truth values (that is, when $d = 2$) it holds:

$$a^\dagger = J_- = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \quad \text{and} \quad a = J_+ = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

Therefore it holds also $N = J_- J_+$ and $N' = J_+ J_-$, whereas in general, for $d > 2$, such equalities do not hold.

We conclude this section by presenting the expressions that allow one to obtain the operators $E_{x,x'}$ in terms of spin–rising and spin–lowering. Let us consider the formal expression (18) applied to $u, v \in \{J_+, J_-\}$; moreover, let:

$$c_{r,s} = \prod_{k=r}^{s} \sqrt{k(d-k)} \prod_{k=1}^{d-1} k(d-k)$$

where $s, r$ are two non negative integers. For $i, j \in \{0, 1, \ldots, d-1\}$, it holds:

$$E_{x,x'} = \begin{cases} 
& c_{i,j} A_{J_- J_+}^{d-2,i-1,j,0} 
\quad \text{if } i = 0 \\
& c_{i,j} A_{J_+ J_-}^{d-1,i-1,j,0} 
\quad \text{if } i = 1 \text{ and } j \geq 1 \\
& c_{i+1,j} A_{J_+ J_-}^{d-2-i,d-1,j} 
\quad \text{if } (i = 1, j = 0 \text{ and } d \geq 3) \text{ or } (1 < i < d-2 \text{ and } j \leq i) \\
& c_{i+1,j} A_{J_- J_+}^{d-1,i,d-1,j-1} 
\quad \text{if } (i = d-2, j = d-1 \text{ and } d \geq 3) \text{ or } (1 < i < d-2 \text{ and } j > i) \\
& c_{2,d-1-j} A_{J_- J_+}^{d-1,j,0} 
\quad \text{if } i = d-2 \text{ and } j \leq d-2 \\
& c_{1,d-1-j} A_{J_+ J_-}^{d-2,j,0} 
\quad \text{if } i = d-1 
\end{cases}$$

### 5.3.1 The three–valued case

Due to the historical importance of three–valued logics, let us examine the three–valued setting more accurately. The truth values $0, \frac{1}{2}, 1$ of $L_3$ are quantum represented on the Hilbert space $C^3$ by the unit vectors of the canonical orthonormal basis:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad |\frac{1}{2}\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

The creation and annihilation operators are:

$$a^\dagger = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{bmatrix} \quad a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & \sqrt{2} \\ 0 & 0 & 0 \end{bmatrix}$$

with corresponding operators

$$N = a^\dagger a = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} \quad aa^\dagger = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$
from which it follows the commutation and anticommutation results:

\[
[a, a^\dagger] = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & -2
\end{bmatrix}
\]  \quad \{a, a^\dagger\} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 3 & 0 \\
0 & 0 & 2
\end{bmatrix}

With respect to \(a^\dagger\) and \(a\), the operators \(E_{x,x'}\) forming the canonical basis of the operator algebra on \(C^3\) can be expressed as:

\[
E_{0,0} = \frac{1}{2} a^2 (a^\dagger)^2 \quad E_{\frac{1}{2},0} = \frac{1}{2} a^2 a^\dagger \quad E_{1,0} = \frac{1}{\sqrt{2}} a^2
\]
\[
E_{0,\frac{1}{2}} = \frac{1}{2} a (a^\dagger)^2 \quad E_{\frac{1}{2},\frac{1}{2}} = \frac{1}{2} a^\dagger a \quad E_{1,\frac{1}{2}} = \frac{1}{\sqrt{2}} a^\dagger a^2
\]
\[
E_{0,1} = \frac{1}{\sqrt{2}} (a^\dagger)^2 \quad E_{\frac{1}{2},1} = \frac{1}{\sqrt{2}} (a^\dagger)^2 a \quad E_{1,1} = \frac{1}{2} (a^\dagger)^2 a^2
\]

Notice that the formulas used to express the operators \(E_{x,x'}\) as compositions of \(a\) and \(a^\dagger\) are not unique; for instance, \(E_{\frac{1}{2},\frac{1}{2}}\) can also be conveniently expressed as \(E_{\frac{1}{2},\frac{1}{2}} = \frac{1}{2} a^\dagger a^2 a^\dagger\).

Now, let us consider a system with three possible spin–values \(-1, 0\) and \(+1\), with corresponding eigenvectors \(|J_z = 1\rangle = |0\rangle\), \(|J_z = 0\rangle = |\frac{1}{2}\rangle\) and \(|J_z = -1\rangle = |1\rangle\).

The spin–rising and spin–lowering operators are:

\[
J_- = \begin{bmatrix}
0 & 0 & 0 \\
\sqrt{2} & 0 & 0 \\
0 & \sqrt{2} & 0
\end{bmatrix} \quad J_+ = \begin{bmatrix}
0 & \sqrt{2} & 0 \\
0 & 0 & \sqrt{2} \\
0 & 0 & 0
\end{bmatrix}
\]

Moreover it holds:

\[
J_- J_+ = \begin{bmatrix}
0 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 2
\end{bmatrix} \quad J_+ J_- = \begin{bmatrix}
2 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

The effect of operator \(J_+\) is depicted on the left side of Figure 3 for a spin–1 system on the Hilbert space \(C^3\). On the right side of the same figure the lowering action of the same operator on a three–levels system is given for comparison with the previous behavior. A similar figure with respect to \(J_-\) can be drawn showing its spin–1 lowering action with respect to the truth value creation behavior.

With respect to \(J_+\) and \(J_-\), the operators \(E_{x,x'}\) forming the canonical basis of the operator algebra on \(C^3\) can be expressed as:

\[
E_{0,0} = \frac{1}{4} (J_+)^2 (J_-)^2 \quad E_{\frac{1}{2},0} = \frac{1}{2\sqrt{2}} (J_+)^2 J_- \quad E_{1,0} = \frac{1}{2} (J_+)^2
\]
\[
E_{0,\frac{1}{2}} = \frac{1}{2\sqrt{2}} J_+ (J_-)^2 \quad E_{\frac{1}{2},\frac{1}{2}} = \frac{1}{4} J_+ (J_-)^2 J_+ \quad E_{1,\frac{1}{2}} = \frac{1}{2\sqrt{2}} J_- (J_+)^2
\]
\[
E_{0,1} = \frac{1}{2} (J_-)^2 \quad E_{\frac{1}{2},1} = \frac{1}{2\sqrt{2}} (J_-)^2 J_+ \quad E_{1,1} = \frac{1}{4} (J_-)^2 (J_+)^2
\]

Also in this case, the expressions of operators \(E_{x,x'}\) as compositions of operators \(J_+\) and \(J_-\) are not unique; for instance, we can also write \(E_{\frac{1}{2},\frac{1}{2}} = \frac{1}{4} J_- (J_+)^2 J_-\).
Fig. 3. The effect of the spin–rising operator on a spin–1 system and the corresponding lowering on three–level eigenstates.

5.4 Brute force, conditional quantum control, and constants method for \(d\)–valued gates

As it happens with Boolean gates, also classical \(d\)–valued \(n\)–input/\(n\)–output gates can be described in the quantum context as sums of “local” operators, each one being an appropriate tensor product of the operators \(E_{x,x'}\). In order to write such descriptions, we can use the \(d\)–valued versions of the “brute force”, extended Conditional Quantum Control, and Constants methods we have previously exposed.

For example, let us consider the “brute force” approach. Let \(x_1, x_2, \ldots, x_n \mapsto y_1, y_2, \ldots, y_n\) be a generic row of the truth table of a \(d\)–valued \(n\)–input/\(n\)–output gate. For what we have told above, the operator \(E_{x_1,y_1} \otimes E_{x_2,y_2} \otimes \cdots \otimes E_{x_n,y_n}\) transforms the input pattern \((x_1, x_2, \ldots, x_n)\) into the output pattern \((y_1, y_2, \ldots, y_n)\), whereas it collapses all the other input patterns of the \(n\)–register computational basis to the null vector. It is not difficult to see that if \(U_0, \ldots, U_{d^n-1}\) are the local operators associated to the \(d^n\) rows of the truth table, then the operator \(U = \sum_{i=0}^{d^n-1} U_i\) is a quantum realization of the \(d\)–valued \(n\)–input/\(n\)–output gate.

Also the extended Conditional Quantum Control method can be immediately applied to the \(d\)–valued case. If the \(d\)–valued \(n\)–input/\(n\)–output gate under consideration can be divided as a \(k\)–input/\(k\)–output control unit and an \((n – k)\)–input/\((n – k)\)–output target (also operating) unit, then any input configuration \(|x_1, x_2, \ldots, x_k, x_{k+1}, \ldots, x_n\rangle\) can be split into a control configuration \(|x_1, \ldots, x_k\rangle\) and a target configuration \(|x_{k+1}, \ldots, x_n\rangle\). The control configuration is returned unchanged on the \(k\) output lines of the control unit; as a side effect, it selects one of the \(d^k\) (non necessarily unitary) operators \(U_0, U_1, \ldots, U_{d^k-1}\), defined on the Hilbert space \(\otimes^{n-k} \mathbb{C}^d\), stored into the gate. The selected operator is applied to the target configuration in order to produce the output values of the target unit. The global operator that describes the behavior of the \(d\)–valued \(n\)–input/\(n\)–output gate has now the form:

\[
P_0 \otimes U_0 + P_1 \otimes U_1 + \cdots + P_{2^k-1} \otimes U_{2^k-1} = \sum_{X=0}^{2^k-1} P_X \otimes U_X
\]

where \(P_X = E_{X,x}\) is the orthogonal projection of the Hilbert space \(\otimes^k \mathbb{C}^d\) which selects the \(X\)–th control configuration, and collapses to the null vector all the other configurations. If many of the operators \(U_i\) are identical, this expression is much shorter than the one obtained with the brute force method. On the other hand, it is clear that the method derived from
Conditional Quantum Control cannot be used to describe every conceivable \( d \)-valued \( n \)-input/\( n \)-output gate, since there are gates which cannot be divided into a control unit and an operating unit.

Finally, the Constants method is practically unchanged. Let us suppose that we want to design a \( d \)-valued \( n \)-input/\( n \)-output gate which realizes a given set \( \mathcal{C} \) of connectives. These connectives are realized by fixing some input lines of the gate with constant values from \( L_d \); if we appropriately choose these input lines and constant values, we can minimize the number of input/output pairs for which the gate behaves differently from the identity gate. This means that we get a short expression by writing the operator associated to the gate in the form \( I + O \), where \( O \) is a linear operator which takes into account the cases where the gate behaves differently from the identity gate.

6 Quantum Implementation of \( f^1_d, f^2_d, \lambda \) and \( m_d \)

In this section we show how the functions \( f^1_d, f^2_d, \lambda \) and \( m_d \) can be expressed as appropriate linear combinations of tensor products of the operators \( E_{x,x'} \). We recall that the functions \( f^1_d, f^2_d, \lambda \) and \( m_d \) behave just like the three-valued gates \( F_1, F_2 \) and \( F_3 \) when \( d = 3 \); as a consequence, it suffices to give a quantum description of the more general functions. We use primarily our Constants method, since the functions have been designed in order to implement the required connectives and simultaneously to minimize the number of input/output pairs for which they do not behave as the identity function.

6.1 Realization of \( f^1_d \) with the constants method

Let us start with function \( f^1_d \); a few calculations will convince the reader that it can be expressed as:

\[
P_0 \otimes \sum_{i=0}^{d-1} \sum_{j \neq i}^{d-1} \left( E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{j}{d}, \frac{i}{d}} - P_{\frac{i}{d}} \otimes P_{\frac{j}{d}} \right) + \sum_{i=1}^{d-3} P_{\frac{i}{d}} \otimes \left( \sum_{j=1}^{d-2} \left( E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{j}{d}, \frac{i}{d}+1} - P_{\frac{i}{d}} \otimes P_{\frac{j}{d}} \right) + E_{\frac{i}{d}, \frac{j}{d}+1} \otimes E_{\frac{j}{d}, \frac{i}{d}} - P_{\frac{j}{d}} \otimes P_{\frac{i}{d}} \right) + \sum_{j=0}^{i-1} \left( E_{\frac{i}{d}, \frac{j}{d}+1} \otimes E_{\frac{j}{d}+1, \frac{i}{d}} - P_{\frac{j}{d}+1} \otimes P_{\frac{i}{d}} \right) + \sum_{j=d-1}^{d-2} \left( E_{\frac{i}{d}, \frac{j}{d}-1} \otimes E_{\frac{j}{d}-1, \frac{i}{d}} - P_{\frac{j}{d}-1} \otimes P_{\frac{i}{d}} \right) + \sum_{j=d}^{d-2} \left( E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{j}{d}, \frac{i}{d}-1} - P_{\frac{j}{d}} \otimes P_{\frac{i}{d}-1} \right) \right) \right)
\]
6.2 Realization of $f^1_d$ with the conditional quantum control method

If we look at the definition of function $f^1_d$, we can see that $x_1$ can be considered as a control line whose value determines the action to be performed by the operating unit, which has $x_2, x_3$ as input lines and $y_2, y_3$ as output lines. This means that we can express the function $f^1_d$ in the following form, using the method derived from Conditional Quantum Control:

$$f^1_d = \sum_{\lambda \in L_d} (|\lambda\rangle \langle \lambda| \otimes U_{\lambda}) = \sum_{\lambda \in L_d} (P_{\lambda} \otimes U_{\lambda})$$

This expression can be obtained from (20) by removing the term $I_2 \otimes I_2 \otimes I_2$ and the negative terms, and finally adding the terms corresponding to the input patterns which are mapped by the operating unit into themselves:

$$P_0 \otimes \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{i+1}{d}, \frac{j+1}{d}}$$

$$+ \sum_{i=1}^{d-3} P_{\frac{i}{d}} \otimes \left( \sum_{j=1}^{d-2} \left( E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{i+1}{d}, \frac{j+1}{d}} + E_{\frac{i+1}{d}, \frac{j+1}{d}} \otimes E_{\frac{i}{d}, \frac{j}{d}} \right) \right)$$

$$+ \sum_{j=0}^{d-1} E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{i-j}{d}, \frac{j-1}{d}} + \sum_{j=d-1}^{d-2} E_{\frac{i-j}{d}, \frac{j-i}{d}} \otimes E_{\frac{i}{d}, \frac{j}{d}}$$

$$+ \sum_{j=d+1}^{d-1} E_{\frac{i}{d}, \frac{j}{d}} \otimes E_{\frac{i+j}{d}, \frac{j+1}{d}}$$

$$+ \sum_{j=0}^{d-2} P_{\frac{i}{d}} \otimes I_2 + P_{\frac{i}{d-1}} \otimes P_0$$

$$+ \sum_{j=i+1}^{d-2} \sum_{k=1}^{d-2} P_{\frac{i}{d}} \otimes P_{\frac{k}{d}} + P_1 \otimes P_1$$
This expression contains $\frac{1}{2}(d^3 + 2d^2 - 5d + 4)$ terms versus the $8d^2 - 24d + 21$ terms of expression (20); thus, the expression obtained with the Constants method is in this case much shorter than the expression obtained with the method deriving from Conditional Quantum Control. This is due to the fact that functions $f^1_d$, $f^2_d$, and $m_d$, as told above, have been designed in order to realize as many connectives of $d$-valued logics as possible and, at the same time, to minimize the number of input/output pairs for which the behavior of the gates differs from the identity gate. Indeed, $d^3 - 4d^2 + 12d - 10$ out of $d^3$ input patterns are mapped by $f^1_d$ to themselves.

### 6.3 Realization of $f^2_d, \lambda$ with the constants method

As we have already told, the Constants method allows one to express every conceivable $d$-valued $n$-input/$n$-output gate, whereas the method derived from Conditional Quantum Control can only be applied to those gates which can be divided into a control unit and an operating unit. It is immediate to see that the functions $f^2_d, \lambda$ (where $\lambda \in L_d \setminus \{0, 1\}$) cannot be divided in this way; as a consequence, we can provide only the expression obtained with the Constants method:

$$
\left( \sum_{i=1}^{d-2} \left( E_{0, 0, i} \otimes E_{0, i, 0} - P_0 \otimes P_{i, 0} + E_{0, i, 0} \otimes E_{0, 0, i} - P_{i, 0} \otimes P_0 \right) \right) \otimes P_\lambda \\
+ P_0 \otimes \left( E_{0, 0} \otimes E_{0, 0} - P_0 \otimes P_0 + E_{0, 0} \otimes E_{0, 0} - P_0 \otimes P_0 \right) \\
+ E_{0, 0} \otimes E_{0, 0} - P_0 \otimes P_0 + E_{0, 0} \otimes E_{0, 0} - P_0 \otimes P_0 \\
+ P_0 \otimes \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \left( E_{0, i, j} \otimes E_{j, i} - P_{i, j} \otimes P_{j, i} \right) \\
+ \sum_{i=1}^{d-3} P_{i} \otimes \sum_{j=1}^{d-2} \left( E_{1, i, j} \otimes E_{0, i, j} - P_1 \otimes P_{i, j} \right) \\
+ E_{0, i, j} \otimes E_{1, i, j} - P_{0, i, j} \otimes P_{1, i, j} \\
+ \sum_{j=0}^{d-1} \left( E_{0, i, j} \otimes E_{0, i, j} - P_0 \otimes P_0 \right)
$$
6.4 Realization of $m_d$ with the constants method

As a further application of our Constants method, we show the expression obtained for function $m_d$:

\[
\begin{align*}
&+ \sum_{j=d-1}^{d-2} \left( E_{\frac{j}{d-1},1} \otimes E_{\frac{j}{d-1},1} - P_{\frac{j}{d-1}} \otimes P_{\frac{j}{d-1}} \right) \\
&+ \sum_{j=1}^{d-2} \left( E_{\frac{j}{d-1},\frac{j}{d-1}} \otimes E_{0,\frac{j}{d-1}} - P_{\frac{j}{d-1}} \otimes P_{0} \right) \\
&+ \sum_{j=1}^{d-2} \left( E_{\frac{j}{d-1},\frac{j}{d-1}} \otimes E_{\frac{j}{d-1},0} - P_{\frac{j}{d-1}} \otimes P_{\frac{j}{d-1}} \right) \\
&+ P_{\frac{d-1}{d-1}} \otimes \left( E_{1,\frac{d-1}{d-1}} \otimes E_{0,\frac{d-1}{d-1}} - P_{1} \otimes P_{\frac{d-1}{d-1}} + E_{\frac{d-1}{d-1},1} \otimes E_{1,\frac{d-1}{d-1}} \right. \\
&\left. - P_{\frac{d-1}{d-1}} \otimes P_{1} + \sum_{j=0}^{d-3} \left( E_{1,\frac{j+1}{d-1}} \otimes E_{1,\frac{j+1}{d-1}} - P_{1} \otimes P_{\frac{j}{d-1}} \right) \\
&\left. + \sum_{j=1}^{d-2} \left( E_{\frac{j}{d-1},1} \otimes E_{\frac{j}{d-1},1} - P_{\frac{j}{d-1}} \otimes P_{\frac{j}{d-1}} \right) \right)
\end{align*}
\]

\[+ I_2 \otimes I_2 \otimes I_2 \]
which is equal to:

\[
P_0 \otimes \sum_{i=0}^{d-1} \sum_{\begin{subarray}{c}j=0 \\ j \neq i \end{subarray}}^{d-1} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{i=2}^{d-2} P_{\frac{i}{d-1}} \otimes \left( \sum_{j=0}^{d-2-i} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{j=1}^{d-2} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{j=1}^{d-2} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{j=1}^{d-2} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ P_{\frac{i}{d-1}} \otimes \left( \sum_{j=0}^{d-3} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{j=1}^{d-2} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{j=1}^{d-2} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ \sum_{j=1}^{d-2} \left( E_{\frac{i}{d-1}, \frac{j}{d-1}} \otimes E_{\frac{i}{d-1}, \frac{j}{d-1}} - P_1 \otimes P_1 \right)
\]

\[
+ I_2 \otimes I_2 \otimes I_2
\]

As we have done for \( f_1 \), it is possible to obtain for \( m_{ij} \) also the expression resulting from the application of the Conditional Quantum Control method (not shown here), by considering \( x_1 \) as a control line. A moment’s thought will convince the reader that also in this case we obtain a sensibly longer formula.

Acknowledgements

This work has been supported by MIUR\COFIN project “Formal Languages and Automata: Theory and Application”
References
