International Journal of Theoretical Physics, Vol. 43, Nos. 7/8, August 2004 (© 2004)

Quantum Conservative Gates for Finite-Valued Logics

Gianpiero Cattaneo, Alberto Leporati, and Roberto Leporini

We introduce some conservative gates for finite-valued logics which are able to realize all the main connectives of the many-valued logics of Łukasiewicz, the MV-algebras of Chang and Brower–Zadeh algebras. After a brief exposition of the motivations for this work, the gates are defined and their properties are explored. Finally, a possible quantum realization of them is proposed, using three techniques: a "brute force" method—an extension of the Conditional Quantum Control argument, and a new technique which we call the Constants Method. For all these techniques, the unitary operator which describes the gate is a sum of local operators.

KEY WORDS: quantum gates; conservative logic; many-valued logics.

1. INTRODUCTION

Conservative logic is a mathematical model of computation introduced by Fredkin and Toffoli (1982), 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 which reflect physical principles such as the reversibility of micro-dynamical laws and the conservation of certain quantities, such as the energy of the physical system used to perform the computations. Conservative logic is based on the Fredkin gate, a universal three-inputs/three-outputs gate which is both conservative and reversible. As a matter of fact this gate was introduced by Petri some years before Fredkin in Petri (1967), and thus in the sequel we will call it the Petri–Fredkin gate.

On the other hand, many-valued logics and modal logics are extensions of the classical Boolean logic which have known a great diffusion due to their ability to manage incomplete and/or uncertain knowledge. Different approaches to many-valued and modal logics have been considered in literature (for instance, see Rescher (1969); Rosser-Turquette (1952)).

In Cattaneo *et al.* (2000a) conservative logic as been extended to include the main features of many-valued logics with a finite number of truth values. The first

[1] Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano-Bicocca, Via Bicocca degli Arcimboldi 8, 20126 Milano, Italy; e-mail: leporini@disco.unimib.it.

result was the definition of some generalizations of the Petri–Fredkin gate for finite-valued reversible and conservative logics (notably finite-valued Łukasiewicz and finite-valued Gödel logics) which have the properties required by the conservative and many-valued paradigms. In particular, we introduced three gates for three-valued logics and three possible extensions of such gates for d-valued logics; two of them allow one to obtain all the main connectives of the Łukasiewicz logics as well as the Gödel implication, while the other was specifically designed to realize the operators which are typically found in MV-algebras and some modal logics. In Cattaneo *et al.* (2000b) we presented the quantum realization of the above gates using three techniques: a "brute force" method, in which the gates are expressed as sum of local operators, each one corresponding to a single row of the truth table, an extension of the Conditional Quantum Control technique originally introduced by Barenco, Deutsch, Ekert and Jozsa in Barenco *et al.* (1995), and a new technique which we have called the Constants Method.

One of the results exposed in Cattaneo *et al.* (2000a) is the following: if the number of truth values is greater than or equal to the number of input/output lines of the considered conservative gates, then it is not possible to realize the FAN-OUT (i.e., classical cloning) operator with such gates. Since FAN-OUT is a fundamental operator in reversible computing, we had two possible choices: to weaken the notion of conservativeness, or to use gates with a number of input/output lines greater than the number of truth values considered. In Cattaneo *et al.* (2000a) we followed the first way and we introduced the notion of *weak conservativeness*, with an associated physical interpretation. In this paper we partially follow the second line, and we present a conservative (in the sense of Fredkin and Toffoli (1982)) four-inputs/four-outputs gate (named CON_3) for three-valued logics that realize all the desired logical connectives as well as the FAN-OUT operator.

Another result of Cattaneo *et al.* (2000a) 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. Here we show that it is possible to overcome this problem by considering four-inputs/four-outputs gates: we define the gate wCON_d, and we show that different configurations of such gate realize not only all the logical connectives considered in Cattaneo *et al.* (2000a) but also the Goguen implication, although in a smaller set of truth values. Of course, for what we have said above, the gate wCON_d cannot be conservative in the sense of Fredkin and Toffoli (1982); however, it is weakly conservative.

Finally, we show that if we do not require that our gates be based upon the Petri–Fredkin gate then we can realize all the desired connectives as well as the FAN-OUT operator with three-inputs/three-outputs self-reversible and weakly conservative gates. We define one of such gates for Boolean logic and then we generalize it to d-valued logics. The gate ALN_d thus obtained allows one to realize all the operators implemented by wCON_d but the Goguen implication and the MV-algebras truncated sum.

This paper is organized as follows. In Section 2 we recall from Cattaneo *et al.* (2000a) some notions of conservative and many-valued logics, to set up the environment and the notation for our results. In Section 3 we define our new gates, and we discuss their properties. In Section 4 we give some examples of possible realizations of our gates in the paradigm of quantum computing, using the three techniques exposed in Cattaneo *et al.* (2000a,b): the "brute force" method, an extension of the Conditional Quantum Dynamics argument originally introduced by Barenco, Deutsch, Ekert and Jozsa in Barenco *et al.* (1995), and a new technique which we have called the Constants Method. Finally, in Section 5 we give the conclusions.

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 Cattaneo *et al.* (2000a); here we just recall some notations needed in the following.

2.1. Some Connectives for Many-Valued Logics

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 $L_d = \{0, \frac{1}{d-1}, \frac{2}{d-1}, \ldots, \frac{d-2}{d-1}, 1\}$ of truth values. 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 falsity and truth, whereas the other values of L_d indicate different degrees of indefiniteness.

For d-valued Łukasiewicz logics the following two connectives are assumed as primitives (see Łukasiewicz (1920), and also Rescher (1969), Rosser and Turquette (1952)):

$$x \rightarrow_L y := \min\{1, 1 - x + y\} \text{ (Łukasiewicz implication)}$$

$$\neg x := 1 - x \text{ (diametrical negation)}$$

Note that, if the use of the constant value 0 is allowed, we can obtain the diametrical negation as $\neg x = x \rightarrow_L 0$. Using these primitives it is possible to define the following connectives, which are usually found in d-valued Łukasiewicz logics:

$$x \vee y := (x \rightarrow_L y) \rightarrow_L y \text{ (Łukasiewicz disjunction)}$$

$$x \wedge y := \neg(\neg x \vee \neg y) \text{ (Łukasiewicz conjunction)}$$

$$x \leftrightarrow_L y := (x \rightarrow_L y) \wedge (y \rightarrow_L x) \text{ (Łukasiewicz equivalence)}$$

From these definitions it is easy 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 involved.

Following Chang (1958, 1959), the Łukasiewicz approach to many-valued logics can be equivalently recovered on the basis of the pair of connectives $\{\oplus, \neg\}$, where

$$x \oplus y := \min\{1, x + y\}$$

Indeed, it can be trivially verified that $x \rightarrow_L y = \neg x \oplus y$ and $x \oplus y = \neg x \rightarrow_L y$. In this Chang 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:

$$x \odot y := \max\{0, x + y - 1\}$$

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

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

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

Besides the diametrical negation ($\neg$), two other negation connectives can be considered: the *intuitionistic negation* (also called *impossibility* ($\sim$)) and the *anti-intuitionistic negation* (also called *contingency* ($\flat$)), whose definitions are:

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

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

Among the different types of implication connectives one can find in literature, in this paper we consider (apart from Łukasiewicz implication) the Gödel implication $\rightarrow_G$ and the Goguen implication $\rightarrow_\Pi$ (residue of product conjunction) defined as follows:

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

$$x \rightarrow_\Pi y := \begin{cases} \dfrac{y}{x} & \text{if } x, y \in GL_p \quad \text{and} \quad y < x \\ 1 & \text{if} \quad x, y \in GL_p \quad \text{and} \quad x \leq y \end{cases} \quad \text{(Goguen implication)}$$

where $GL_p = \{0\} \cup \{\frac{1}{2^i} \mid i \in \mathbb{Z} \text{ and } 0 \leq i \leq p-2\}$. Notice that the Goguen implication requires truth values which are implemented as non-equispaced rational numbers. If we let $d = 2^{p-2} + 1$ then all the numbers of GL_p are also elements of the set L_d. This means that we can look at $\rightarrow_\Pi$ as a binary map which is not closed on L_d but is closed on its subset GL_p. In other words, we can use a specially designed d-valued gate to compute the Goguen implication for a p-valued logic. Of course this trick has nothing to do with the interpretation of the numbers of L_d and GL_p as truth values: they are different sets of truth values that happen to be implemented with the same numbers. For $d \geq 2$, in general it holds $GL_p \subset L_d$; we have $L_d = GL_p$ only for $d = p = 3$ and for $d = p = 2$, in which cases we can think that also the logical truth values coincide.

Observe that if the use of the constant value 0 is allowed then we can get the intuitionistic negation as $\sim x = x \rightarrow_G 0$. Analogously, for $x \in GL_p$ it holds $\sim x = x \rightarrow_\Pi 0$. Moreover, when $d = p = 2$ the three implications $\rightarrow_L$, $\rightarrow_G$ and $\rightarrow_\Pi$ collapse to the classical implication, whereas for $p = d = 3$ the non classical implications $\rightarrow_G$ and $\rightarrow_\Pi$ coincide.

2.2. Computational Paradigms of Reversible and Conservative Logic

According to Fredkin and Toffoli (1982), *conservativeness* is usually modeled by the property that the output values of the involved gates are always a permutation of the values given in the input. We call this condition *strict conservativeness*. Let us stress that this does not mean that input and output lines are permanently connected by a fixed pattern; roughly speaking, it is as if the line connections *depend* on the values given in the input.

On the other hand, *reversibility* is modeled as the fact that the gate computes an invertible mapping. Let us note that conservativeness and reversibility are two independent notions: a gate can satisfy both properties, only one of them, or none. For example, a conservative Boolean gate may map two different input configurations (e.g., (101) and (011)) to the same permuted output configuration (e.g., (110)); on the other hand, a famous example of reversible but non conservative Boolean gate is the Controlled–Controlled–NOT gate, also known as the Toffoli gate, whose behavior is defined by the following map CCNOT : $\{0, 1\}^3 \rightarrow \{0, 1\}^3$:

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= (x_1 \wedge x_3) \oplus x_2 \\ y_3 &= x_3 \end{aligned} \tag{2.1}$$

Just as the Fredkin gate, the Toffoli gate was first presented by Petri in Petri (1967), and thus in the sequel we will call it the Petri–Toffoli gate.

Since every reversible circuit computes a bijective mapping between input and output patterns, and every conservative circuit performs a permutation of each input pattern, it follows that they must necessarily have the same number of input and output lines. In this paper we are mainly interested into gates which are both conservative and reversible, such as the Petri–Fredkin gate that computes the following map FG : $\{0, 1\}^3 \to \{0, 1\}^3$:

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= (x_1 \wedge x_2) \vee (\neg x_1 \wedge x_3) \\ y_3 &= (\neg x_1 \wedge x_2) \vee (x_1 \wedge x_3) \end{aligned}$$

In particular, the Petri–Fredkin gate is *self-reversible*, i.e., FG is the inverse of itself with respect to map composition. This means that FG is a bijection on the set $\{0, 1\}^3$ that can be expressed as the composition of fixed points and disjoint cycles of length two. Self-reversibility is a particularly desirable feature in constructing a quantum circuit (Fredkin and Toffoli (1982), p. 247). Note that self-reversibility implies the reversibility property while the converse is not generally true.

The extension of conservativeness and reversibility to d-valued logics is immediate: a map $f : L_d^n \to L_d^m$ is *reversible* if and only if it is a bijection. This implies $n = m$, so that also reversible d-valued circuits must have the same number of input and output lines. On the other hand, a map $f : L_d^n \to L_d^m$ is *conservative* if and only if for each possible choice of $\underline{x} \in L_d^n$, the m-tuple $\underline{y} = f(\underline{x})$ is a permutation of the n-tuple $\underline{x}$. Once again, this implies that $n = m$. As in the Boolean case, conservativeness and reversibility are two independent notions.

A well-known technique to transform irreversible gates into reversible ones consists in adding some input and output lines in order to store the information that, given some output value, allow one to recover the input that generated it. Thus, for example, the AND gate is not reversible, but we can reversibly compute the AND connective through the Petri–Fredkin gate. This technique allows one to give universal sets of reversible gates for Boolean logic starting from universal sets of Boolean gates and transforming the irreversible gates into reversible ones. A similar idea can be used to make a gate conservative, as we show for the (universal and self-reversible) non conservative Petri–Toffoli gate, defined by Equations (2.1). Let us consider a five-inputs/five-outputs gate, described by the following equations:

$$\forall \underline{x} \in \{0, 1\}^5 \; T(\underline{x}) := \begin{cases} (0, 0, 1, 1, 1) & \text{if} \;\; \underline{x} = (1, 0, 1, 0, 1) \\ (1, 0, 1, 0, 1) & \text{if} \;\; \underline{x} = (0, 0, 1, 1, 1) \\ (1, 1, 1, 0, 1) & \text{if} \;\; \underline{x} = (1, 0, 1, 1, 1) \\ (1, 0, 1, 1, 1) & \text{if} \;\; \underline{x} = (1, 1, 1, 0, 1) \\ (x_1, x_2, x_3, x_4, x_5) & \text{otherwise} \end{cases}$$

As we can see, by fixing $x_1 = 1$ and $x_2 = 0$ in the input, and ignoring the corresponding output lines y_1 and y_2, this gate realizes the Petri–Toffoli gate on the last three input and output lines. The proposed gate is conservative, since for each input pattern $\underline{x}$ the corresponding output pattern $\underline{y} = T(\underline{x})$ is a permutation of $\underline{x}$.

From now on, a d-valued n-inputs/n-outputs gate will be called an (n, d)-gate for short.

A very important gate in reversible computing is the FAN-OUT : $L_d \to L_d^2$, defined by the law FAN–OUT$(x) = (x, x)$. In other words, the FAN-OUT operator simply clones the input value. When dealing with classical circuits, the FAN-OUT operator is implemented by sticking two output wires to an existing input wire; on the other hand, when we deal with quantum circuits we have to take into account the so-called *no cloning theorem* Woot and Zurek (1982), 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 (d-valued) truth values, and not each possible quantum state such as, for example, superpositions of vectors from the canonical orthonormal basis, where each vector represents a classical truth value. From a classical point of view, it is easy to see that the FAN-OUT operator cannot be realized by a two-inputs/two-outputs *conservative* gate, even when working with Boolean values. On the contrary, the Controlled–NOT gate, defined as

$$y_1 = x_1$$

$$y_2 = x_1 \oplus x_2$$

is a *reversible* two-inputs/two-outputs gate that realizes the FAN-OUT operator when the input x_2 is fixed to 0.

In this paper we are interested in the design of universal gates which are able to implement the FAN-OUT operator together with as many connectives from d-valued logics as possible. However, in Cattaneo *et al.* (2000a) we have shown that if $d \geq 3$ then there is no *strictly conservative* $(3, d)$-gate which is able to realize in its configurations the FAN-OUT operator. As a consequence, in Cattaneo *et al.* (2000a) we proposed the notion of *weak conservativeness*: that is, we required that the sum of the output values be always equal to the sum of the input values. We showed also that, from a physical point of view, this property is equivalent to the requirement that the energy needed to build the pattern of input values is equal to the energy that would be required to build the output pattern "from scratch," that is directly without using any gate. A trivial observation is that strict and weak conservativeness coincide in the Boolean case.

Using this new notion of conservativeness, in Cattaneo *et al.* (2000a) we have defined three (3, 3)-gates and three $(3, d)$-gates which are self-reversible and weakly conservative. A notable property of such gates is that they are *functionally complete*: from different configurations of the gates (that is, by fixing some of their input lines with constant values and considering some output lines as garbage) we can obtain a set of connectives that, equipped with some logical constants from L_d, are able to realize any mapping from L_d^n to L_d. The need for the logical constants is given by the well-known fact that Łukasiewicz and Gödel logics without constants are *incomplete* Rosser and Turquette (1952): for example, they cannot express the mapping which is identically equal to $\frac{1}{d-1}$ Cattaneo *et al.* (2000a). Notice that our notion of universality has nothing to do with the ones considered in quantum computing, where universality means that the gates can be used to realize any unitary operator under some suitable approximation.

Two further properties which we have taken into account in Cattaneo *et al.* (2000a) when designing our gates are 0-*regularity* and 1-*regularity*, both of them satisfied by the Petri–Fredkin gate. Formally, let $G : L_d^3 \to L_d^3$ be the map computed by a (3, d)-gate. 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 possible choice of 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 possible choice of x_2, x_3 in L_d. These two properties suggested us to adopt the Conditional Quantum Control of Barenco *et al.* (1995) as a technique to describe the quantum versions of our gates Cattaneo *et al.* (2000b), by looking at the first input line as a control line. In this paper we propose two extensions of these notions in order to use them with (n, d)-gates.

In the first extension, we say that an (n, d)-gate is 0-regular if and only if it is possible to fix $n - 2$ input lines to the Boolean value 0 so that the remaining two input values are exchanged. Analogously, we say that the gate is 1-regular if and only if it is possible to fix $n - 2$ input lines to 1 so that the remaining two input values are copied unaltered. A further requirement is that the $n - 2$ values fixed to 0 and to 1 (for 0 and 1-regularity, respectively) are returned unchanged onto the corresponding output lines. On the other hand, we do not require that the $n - 2$ lines fixed to obtain 0-regularity be the same fixed to obtain 1-regularity. Gate CON$_3$ defined in the next section satisfies these notions of 0 and 1-regularity.

A second possible extension is the following: an (n, d)-gate is 0-regular (respectively, 1-regular) if and only if, once fixed suitable $n - 3$ input lines with Boolean values, these are copied unchanged and there exists one of the remaining three lines that, when fixed to 0 (respectively, 1), produces in output the value 0 (respectively, 1) and causes the (non necessarily Boolean) values of the other two input lines to be exchanged (respectively, copied unaltered). A further requirement is that if the gate is both 0 and 1-regular, then the $n - 3$ fixed input lines, as well as the Boolean pattern used to fix them, must be the same; moreover, it is required that also the remaining input (control) line fixed either to 0 or to 1 (in order to exchange or to copy the values given in the last two input lines, respectively) must be the same. The gate wCON$_d$ defined in the next section satisfies these notions of 0 and 1-regularity.

The next list summarizes the fundamental properties we would like to preserve when looking for extensions of the Petri–Fredkin gate to d-valued logics:

(F-1) it is a (3, d)-gate;
(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 basic gate, that is, from the configurations of the gate all the logical connectives of Section 2.1. are obtained, included the FAN-OUT;
(F-5) it is 0-regular;
(F-6) it is 1-regular;
(F-7) at least one of the inputs is copied unaltered to the corresponding output line (useful for the quantum implementation with the Conditional Quantum Control technique);
(F-8) the restriction of the gate to Boolean input triples is just the Petri–Fredkin gate.

Just as we have done for 0-regularity and 1-regularity, we can extend property (F-8) by requiring that a given (n, d)-gate, when fixed $n - 3$ of its input lines with (non necessarily Boolean) constant values, behaves as the Petri–Fredkin gate when the remaining input lines are feeded with Boolean values.

3. CONSERVATIVE GATES

3.1. Finite-Valued Extensions of the Petri–Fredkin Gate

As stated in the introduction, in Cattaneo *et al.* (2000a) we proved the following proposition.

Proposition 3.1. *For $d \geq 3$, there is no (3,d)-gate satisfying properties (F-2), (F-3) and (F-8) which is able to realize the Łukasiewicz connectives ($\wedge, \vee, \rightarrow_L$), the Gödel implication ($\rightarrow_G$) and the MV-connectives ($\oplus, \odot$).*

As a consequence, if we want to build a single gate which satisfies properties (F-2)–(F-8) and seizes all the above connectives we have to look for an (n, d)-gate with $n \geq 4$.

The first gate we propose is the (4, 3)-gate CON_3, whose truth table is given in Table I. As we can see, CON_3 is self-reversible (and thus reversible) and strictly conservative. Table II shows the configurations of the gate that allow one to obtain all the required connectives of three-valued logics. Since the number of input/output lines is greater than the number of the involved truth values, we can realize the FAN-OUT without sacrificing strict conservativeness. Moreover, we observe that it holds $\mathrm{CON}_3(0, 0, x_3, x_4) = (0, 0, x_4, x_3)$ for every possible choice of x_3, x_4 in L_3, and $\mathrm{CON}_3(x_1, x_2, 1, 1) = (x_1, x_2, 1, 1)$ for every possible choice of x_1, x_2 in L_3; thus, the gate is both 0-regular and 1-regular with respect to the extended definition of such properties. As for property (F-7), it suffices to observe that the value on the first input line is always copied to the first output line, producing in this way a gate which is conditionally controlled by the first line. Finally, if we fix $x_1 = 1$ and we ignore the output line y_1 then the resulting (3, 3)-gate behaves as the Petri–Fredkin gate when feeded with Boolean input triples; this means that the gate satisfies also the extended definition of property (F-8).

Table I. Truth Table of the (4,3)-Gate CON_3

x_1	x_2	x_3	x_4	$\mapsto$	y_1	y_2	y_3	y_4	x_1	x_2	x_3	x_4	$\mapsto$	y_1	y_2	y_3	y_4	x_1	x_2	x_3	x_4	$\mapsto$	y_1	y_2	y_3	y_4
0	0	0	0		0	0	0	0	$\frac{1}{2}$	0	0	0		$\frac{1}{2}$	0	0	0	1	0	0	0		1	0	0	0
0	0	0	$\frac{1}{2}$		0	0	$\frac{1}{2}$	0	$\frac{1}{2}$	0	0	$\frac{1}{2}$		$\frac{1}{2}$	0	0	$\frac{1}{2}$	1	0	0	$\frac{1}{2}$		1	0	0	$\frac{1}{2}$
0	0	0	1		0	0	1	0	$\frac{1}{2}$	0	0	1		$\frac{1}{2}$	1	0	0	1	0	0	1		1	0	0	1
0	0	$\frac{1}{2}$	0		0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	0		$\frac{1}{2}$	0	$\frac{1}{2}$	0	1	0	$\frac{1}{2}$	0		1	0	$\frac{1}{2}$	0
0	0	$\frac{1}{2}$	$\frac{1}{2}$		0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$		$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$\frac{1}{2}$		1	0	$\frac{1}{2}$	$\frac{1}{2}$
0	0	$\frac{1}{2}$	1		0	0	1	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	1		$\frac{1}{2}$	0	$\frac{1}{2}$	1	1	0	$\frac{1}{2}$	1		1	$\frac{1}{2}$	0	1
0	0	1	0		0	0	0	1	$\frac{1}{2}$	0	1	0		$\frac{1}{2}$	0	1	0	1	0	1	0		1	0	1	0
0	0	1	$\frac{1}{2}$		0	0	$\frac{1}{2}$	1	$\frac{1}{2}$	0	1	$\frac{1}{2}$		$\frac{1}{2}$	0	1	$\frac{1}{2}$	1	0	1	$\frac{1}{2}$		1	0	1	$\frac{1}{2}$
0	0	1	1		0	0	1	1	$\frac{1}{2}$	0	1	1		$\frac{1}{2}$	0	1	1	1	0	1	1		1	0	1	1
0	$\frac{1}{2}$	0	0		0	$\frac{1}{2}$	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0		$\frac{1}{2}$	$\frac{1}{2}$	0	0	1	$\frac{1}{2}$	0	0		1	$\frac{1}{2}$	0	0
0	$\frac{1}{2}$	0	$\frac{1}{2}$		0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$		$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	1	$\frac{1}{2}$	0	$\frac{1}{2}$		1	$\frac{1}{2}$	0	$\frac{1}{2}$
0	$\frac{1}{2}$	0	1		0	1	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	1		$\frac{1}{2}$	1	0	$\frac{1}{2}$	1	$\frac{1}{2}$	0	1		1	0	$\frac{1}{2}$	1
0	$\frac{1}{2}$	$\frac{1}{2}$	0		0	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0		$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0	1	$\frac{1}{2}$	$\frac{1}{2}$	0		1	$\frac{1}{2}$	$\frac{1}{2}$	0
0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$		0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$		$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$		1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
0	$\frac{1}{2}$	$\frac{1}{2}$	1		0	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1		$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$	1		1	$\frac{1}{2}$	$\frac{1}{2}$	1
0	$\frac{1}{2}$	1	0		0	1	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1	0		$\frac{1}{2}$	1	$\frac{1}{2}$	0	1	$\frac{1}{2}$	1	0		1	$\frac{1}{2}$	1	0
0	$\frac{1}{2}$	1	$\frac{1}{2}$		0	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$		$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	1	$\frac{1}{2}$	1	$\frac{1}{2}$		1	$\frac{1}{2}$	1	$\frac{1}{2}$
0	$\frac{1}{2}$	1	1		0	$\frac{1}{2}$	1	1	$\frac{1}{2}$	$\frac{1}{2}$	1	1		$\frac{1}{2}$	$\frac{1}{2}$	1	1	1	$\frac{1}{2}$	1	1		1	$\frac{1}{2}$	1	1
0	1	0	0		0	1	0	0	$\frac{1}{2}$	1	0	0		$\frac{1}{2}$	0	0	1	1	1	0	0		1	1	0	0
0	1	0	$\frac{1}{2}$		0	$\frac{1}{2}$	1	0	$\frac{1}{2}$	1	0	$\frac{1}{2}$		$\frac{1}{2}$	$\frac{1}{2}$	0	1	1	1	0	$\frac{1}{2}$		1	1	$\frac{1}{2}$	0
0	1	0	1		0	1	0	1	$\frac{1}{2}$	1	0	1		$\frac{1}{2}$	1	1	0	1	1	0	1		1	1	1	0
0	1	$\frac{1}{2}$	0		0	$\frac{1}{2}$	0	1	$\frac{1}{2}$	1	$\frac{1}{2}$	0		$\frac{1}{2}$	$\frac{1}{2}$	1	0	1	1	$\frac{1}{2}$	0		1	1	0	$\frac{1}{2}$
0	1	$\frac{1}{2}$	$\frac{1}{2}$		0	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$		$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1	1	1	$\frac{1}{2}$	$\frac{1}{2}$		1	1	$\frac{1}{2}$	$\frac{1}{2}$
0	1	$\frac{1}{2}$	1		0	1	$\frac{1}{2}$	1	$\frac{1}{2}$	1	$\frac{1}{2}$	1		$\frac{1}{2}$	1	1	$\frac{1}{2}$	1	1	$\frac{1}{2}$	1		1	1	$\frac{1}{2}$	1
0	1	1	0		0	1	1	0	$\frac{1}{2}$	1	1	0		$\frac{1}{2}$	1	0	1	1	1	1	0		1	1	0	1
0	1	1	$\frac{1}{2}$		0	1	1	$\frac{1}{2}$	$\frac{1}{2}$	1	1	$\frac{1}{2}$		$\frac{1}{2}$	1	$\frac{1}{2}$	1	1	1	1	$\frac{1}{2}$		1	1	1	$\frac{1}{2}$
0	1	1	1		0	1	1	1	$\frac{1}{2}$	1	1	1		$\frac{1}{2}$	1	1	1	1	1	1	1		1	1	1	1

Table II. The Operators Realized by the Gate CON_3

Connective	Inputs	Constants	Outputs	Garbage
FAN-OUT	x_1	$x_2 = \frac{1}{2}, x_3 = 0, x_4 = 1$	y_1, y_4	y_2, y_3
Pr_1	x_3, x_4	$x_1 = 0, x_2 = 0$	y_4	y_1, y_2, y_3
Pr_2	x_3, x_4	$x_1 = 0, x_2 = 0$	y_3	y_1, y_2, y_4
$\rightarrow_L$	x_2, x_4	$x_1 = \frac{1}{2}, x_3 = 1$	y_3	y_1, y_2, y_4
$\rightarrow_G$	x_3, x_2	$x_1 = \frac{1}{2}, x_4 = 1$	y_2	y_1, y_3, y_4
$\rightarrow_\Pi$	x_3, x_2	$x_1 = \frac{1}{2}, x_4 = 1$	y_2	y_1, y_3, y_4
$\vee$	x_3, x_4	$x_1 = \frac{1}{2}, x_2 = 1$	y_2	y_1, y_3, y_4
$\wedge$	x_1, x_4	$x_2 = \frac{1}{2}, x_3 = 0$	y_4	y_1, y_2, y_3
$\oplus$	x_1, x_4	$x_2 = \frac{1}{2}, x_3 = 1$	y_3	y_1, y_2, y_4
$\odot$	x_2, x_4	$x_1 = 1, x_3 = 0$	y_3	y_1, y_2, y_4
$\mathbb{I}$	x_4	$x_1 = 0, x_2 = 0, x_3 = 0$	y_3	y_1, y_2, y_4
$\neg$	x_2	$x_1 = \frac{1}{2}, x_3 = 1, x_4 = 0$	y_3	y_1, y_2, y_4
$\sim$	x_3	$x_1 = \frac{1}{2}, x_2 = 0, x_4 = 1$	y_2	y_1, y_3, y_4
$\flat$	x_4	$x_1 = \frac{1}{2}, x_2 = 1, x_3 = 0$	y_4	y_1, y_2, y_3
$\Diamond$	x_2	$x_1 = 0, x_3 = 0, x_4 = 1$	y_2	y_1, y_3, y_4
$\Box$	x_2	$x_1 = 0, x_3 = 0, x_4 = 1$	y_4	y_1, y_2, y_3

The second extension of the Petri–Fredkin gate we propose is the (4, d)-gate wCON_d, defined as follows:
$\forall \underline{x} \in L_d^4$

$$\mathrm{wCON}_d(\underline{x}) := \begin{cases} (0, x_2, x_2 + x_3, 1 - x_2) & \text{if } x_1 = 0, x_2 > 0, x_4 = 1 \text{ and } x_2 + x_3 < 1 \\ (0, x_2, x_3 - x_2, 1) & \text{if } x_1 = 0, 0 < x_2 \leq x_3 < 1 \text{ and } x_4 = 1 - x_2 \\ (0, x_2, 1, x_3) & \text{if } x_1 = 0, x_4 = 1, x_3 < 1 \text{ and } x_2 + x_3 \geq 1 \\ (0, x_2, x_4, 1) & \text{if } x_1 = 0, x_3 = 1, x_4 < 1 \text{ and } x_2 + x_4 \geq 1 \\ (0, x_2, x_2 + x_4 - 1, 1 - x_2) & \text{if } x_1 = 0, x_3 = 0, x_4 < 1 \text{ and } x_2 + x_4 > 1 \\ (0, x_2, 0, x_3 + x_4) & \text{if } x_1 = 0, 0 < x_3 < x_2 \text{ and } x_4 = 1 - x_2 \\ \left(\frac{1}{2}, x_2, \frac{x_4}{x_2}, 1 + x_4 - \frac{x_4}{x_2}\right) & \text{if } x_1 = \frac{1}{2}, x_2, x_4 \in GL_p, x_4 < x_2 \text{ and } x_3 = 1 \\ \left(\frac{1}{2}, x_2, 1, x_2x_3\right) & \text{if } x_1 = \frac{1}{2}, x_2, x_2x_3 \in GL_p, x_2 > 0, x_3 < 1 \\ & \quad \text{and } x_3 + x_4 - 1 = x_2x_3 \\ (1, 0, x_4, x_3) & \text{if } x_1 = 1, x_2 = 0 \text{ and } x_3 \neq x_4 \\ (1, x_2, x_4, 1) & \text{if } x_1 = 1, 0 < x_2 \leq x_4 < 1 \text{ and } x_3 = 1 \\ (1, x_2, 1, x_3) & \text{if } x_1 = 1, 0 < x_2 \leq x_3 < 1 \text{ and } x_4 = 1 \\ (1, x_2, x_2, 1 - x_2 + x_4) & \text{if } x_1 = 1, x_4 < x_2 < 1 \text{ and } x_3 = 1 \\ (1, x_2, 1, x_4 + x_2 - 1) & \text{if } x_1 = 1, x_2 < 1, x_3 = x_2, x_4 + x_2 \geq 1 \\ & \quad \text{and } x_4 < 1 \\ (1, x_2, x_2, x_3 - x_2) & \text{if } x_1 = 1, 0 < x_2 < x_3 < 1 \text{ and } x_4 = 0 \\ (1, x_2, x_4 + x_2, 0) & \text{if } x_1 = 1, 0 < x_2, x_3 = x_2, x_4 + x_2 < 1 \\ & \quad \text{and } x_4 > 0 \\ (x_1, x_2, x_3, x_4) & \text{otherwise} \end{cases}$$

Table III. The Operators Realized by the Gate wCON$_d$

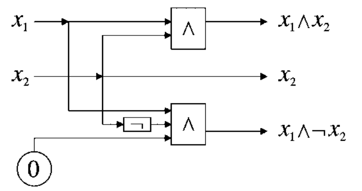
Connective	Inputs	Constants	Outputs	Garbage
FAN-OUT	x_2	$x_1 = 1, x_3 = 1, x_4 = 0$	y_2, y_3	y_1, y_4
Pr$_1$	x_3, x_4	$x_1 = 1, x_2 = 1$	y_3	y_1, y_2, y_4
Pr$_2$	x_3, x_4	$x_1 = 1, x_2 = 1$	y_4	y_1, y_2, y_3
$\rightarrow_L$	x_2, x_4	$x_1 = 1, x_3 = 1$	y_4	y_1, y_2, y_3
$\rightarrow_G$	x_2, x_3	$x_1 = 1, x_4 = 1$	y_3	y_1, y_2, y_4
$\rightarrow_\Pi$	x_2, x_4	$x_1 = \frac{1}{2}, x_3 = 1$	y_3	y_1, y_2, y_4
$\vee$	x_2, x_4	$x_1 = 1, x_3 = 1$	y_3	y_1, y_2, y_4
$\wedge$	x_2, x_3	$x_1 = 1, x_4 = 0$	y_3	y_1, y_2, y_4
$\oplus$	x_2, x_3	$x_1 = 0, x_4 = 1$	y_3	y_1, y_2, y_4
$\odot$	x_2, x_4	$x_1 = 0, x_3 = 0$	y_3	y_1, y_2, y_4
$\mathbb{I}$	x_4	$x_1 = 1, x_2 = 1, x_3 = 0$	y_4	y_1, y_2, y_3
$\neg$	x_2	$x_1 = 1, x_3 = 1, x_4 = 0$	y_4	y_1, y_2, y_3
$\sim$	x_2	$x_1 = 1, x_3 = 0, x_4 = 1$	y_3	y_1, y_2, y_4
$\flat$	x_2	$x_1 = 0, x_3 = 1, x_4 = 0$	y_3	y_1, y_2, y_4
$\Diamond$	x_2	$x_1 = 1, x_3 = 0, x_4 = 1$	y_4	y_1, y_2, y_3
$\Box$	x_2	$x_1 = 0, x_3 = 1, x_4 = 0$	y_4	y_1, y_2, y_3

Table III reports the connectives realized by wCON$_d$. As stated above, since wCON$_d$ realizes the FAN-OUT operator, for $d \geq 4$ it cannot be strictly conservative; however, it is easily verified that it is weakly conservative and self-reversible. The gate wCON$_d$ satisfies the property (F-4), as we can see from the set of the connectives which are realized. As for properties (F-5)–(F-6) we observe that wCON$_d(1, 0, x_3, x_4) = (1, 0, x_4, x_3)$ and wCON$_d(1, 1, x_3, x_4) = (1, 1, x_3, x_4)$ for every possible choice of x_3, x_4 in L_d. Moreover, the first two inputs are always copied to the first two outputs, thus satisfying the property (F-7). Finally, if we fix $x_1 = 1$ and we ignore the first output line then the resulting $(3, d)$-gate behaves as the Petri–Fredkin gate when fed with Boolean input triples.

3.2. An Alternative to the Petri–Fredkin Gate

As we have seen, Proposition 3.1 states that we cannot realize both the connectives of Łukasiewicz many-valued logics and of Chang's MV-algebras with the same $(3, d)$-gate, if we require that such gate is a d-valued extension of the Petri–Fredkin gate. However, if we disregard the properties (F-4)–(F-8) we can easily build a $(3, 2)$-gate whose d-valued extension allows one to realize all the desired connectives. The truth table of this $(3, 2)$-gate is the following:

x_1	x_2	x_3	$\mapsto$	y_1	y_2	y_3	x_1	x_2	x_3	$\mapsto$	y_1	y_2	y_3
0	0	0		0	0	0	1	0	0		0	0	1
0	0	1		1	0	0	1	0	1		0	1	1
0	1	0		0	1	0	1	1	0		1	1	0
0	1	1		1	0	1	1	1	1		1	1	1



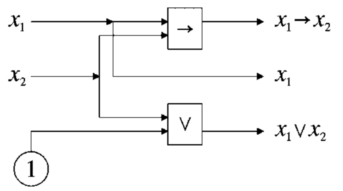


Fig. 1. A self-reversible and conservative $(3, 2)$-gate. Note that in the gate on the right $x_1 \wedge \neg x_2 = \neg(x_1 \rightarrow x_2)$.

The behavior of the gate in the two cases $x_3 = 0$ and $x_3 = 1$ is depicted in Figure 1. Since this gate is both strictly conservative and self-reversible, it is a valid *alternative* to the Petri–Fredkin gate upon which to base conservative logic.

Looking for a d-valued extension of this gate, we have chosen to minimize the number of triples for which the gate behaves differently from the identity gate, with the constraint that it realizes all the desired connectives from d-valued logics. As it will be explained in the following, this is a fundamental step of our Constants Method, that in some cases allows one to give particularly short formulas for the unitary operator corresponding to a quantum realization of the gate. This process led us to the following analytic expression of $\mathrm{ALN}_d : L_d^3 \rightarrow L_d^3$:

$$\forall \underline{x} \in L_d^3$$

$$\mathrm{ALN}_d(\underline{x}) := \begin{cases} (1 - x_1 + x_2, x_1, x_1) & \text{if } x_2 \leq x_1, x_2 < 1 \text{ and } x_3 = 1 \\ (x_2, x_1 + x_2 - 1, 1) & \text{if } 1 \leq x_1 + x_2 < 2 \text{ and } x_2 = x_3 \\ (1, x_1, x_2) & \text{if } x_1 < x_2 < 1 \text{ and } x_3 = 1 \\ (x_2, x_3, 1) & \text{if } x_1 = 1 \text{ and } x_2 < x_3 < 1 \\ (x_3 - x_2, x_2, x_2) & \text{if } x_1 = 0 \text{ and } x_2 < x_3 < 1 \\ (0, x_2, x_1 + x_2) & \text{if } x_1 > 0, x_1 + x_2 < 1 \text{ and } x_2 = x_3 \\ (1 - x_1, x_1 + x_3 - 1, x_1) & \text{if } x_1 < 1, x_1 + x_3 > 1, x_2 = 0 \\ & \quad \text{and } x_3 < 1 \\ (x_3, 0, x_1 + x_2) & \text{if } x_1 = 1 - x_3 \text{ and } 0 < x_2 < x_3 < 1 \\ (x_1, x_2, x_3) & \text{otherwise} \end{cases}$$

As one can easily verify, the gate is self-reversible and weakly conservative. Moreover it is universal, as it can realize all the operators listed in Table IV. Observe that the MV-algebras truncated sum ($\oplus$) and the Goguen implication ($\rightarrow_{\Pi}$, with $p > 3$) are not realized.

4. QUANTUM REALIZATION OF THE GATES

In this section we describe a mathematical formalism which can be used to realize a quantum version of the gates proposed in the previous section.

Table IV. The Operators Realized by the Gate ALN$_d$

Connective	Inputs	Constants	Outputs	Garbage
FAN-OUT	x_1	$x_2 = 0, x_3 = 1$	y_2, y_3	y_1
Pr$_1$	x_1, x_2	$x_3 = 1$	y_2	y_1, y_3
Pr$_2$	x_1, x_2	$x_3 = 0$	y_2	y_1, y_3
$\rightarrow_L$	x_1, x_2	$x_3 = 1$	y_1	y_2, y_3
$\rightarrow_G$	x_2, x_3	$x_1 = 1$	y_3	y_1, y_2
$\vee$	x_1, x_2	$x_3 = 1$	y_3	y_1, y_2
$\wedge$	x_2, x_3	$x_1 = 0$	y_3	y_1, y_2
$\odot$	x_1, x_3	$x_2 = 0$	y_2	y_1, y_3
$\mathbb{I}$	x_2	$x_1 = 0, x_3 = 0$	y_2	y_1, y_3
$\neg$	x_1	$x_2 = 0, x_3 = 1$	y_1	y_2, y_3
$\sim$	x_2	$x_1 = 1, x_3 = 0$	y_3	y_1, y_2
$\flat$	x_3	$x_1 = 0, x_2 = 1$	y_2	y_1, y_3
$\Diamond$	x_2	$x_1 = 1, x_3 = 0$	y_1	y_2, y_3
$\Box$	x_3	$x_1 = 0, x_2 = 1$	y_1	y_2, y_3

A quantum gate acts on memory cells that are d-level quantum systems called *qudits* (see Cattaneo *et al.* (2000b) and Gottesman (1999)). A qudit is typically implemented using the energy levels of an atom, a nuclear spin or a polarization photon. The mathematical description—independent of the practical realization—of a single qudit is based on the d-dimensional complex Hilbert space $\mathbb{C}^d$. The truth values of L_d are represented in this framework by the unit vectors of the canonical orthonormal basis, called the *computational basis* of $\mathbb{C}^d$:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}, \left|\frac{1}{d-1}\right\rangle = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}, \dots, \left|\frac{d-2}{d-1}\right\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

A *quantum register* of size n is mathematically described by the Hilbert space $\otimes^n \mathbb{C}^d = \underbrace{\mathbb{C}^d \otimes \cdots \otimes \mathbb{C}^d}_{n\text{-times}}$ representing a set of n qudits labeled by the index $i = 1, \dots, n$. An *n-configuration* (quantum realization of a classical sequence of length n) is a vector $|x_1\rangle \otimes \cdots \otimes |x_n\rangle$, simply written as $|x_1, \dots, x_n\rangle$, where each x_i runs on the set L_d of classical truth values. Let us recall that the dimension of $\otimes^n \mathbb{C}^d$ is d^n and the collection of all n-configurations $\{|x_1, \dots, x_n\rangle : x_i \in L_d\}$ is an orthonormal basis of this space, called the *n-register computational basis*.

Unlike the situation of the classical wired computer where voltages of a wire go over voltages of another, in quantum realizations of classical gates something different happens. In this setting every gate of this kind must have the same number of input and output lines (that is, it must be an (n, d)-gate) and so its quantum realization is a linear operator $G : \otimes^n \mathbb{C}^d \mapsto \otimes^n \mathbb{C}^d$ which transforms vectors $|x_1, \ldots, x_n\rangle$ of the n-register computational basis into vectors $G(|x_1, \ldots, x_n\rangle) = |x'_1, \ldots, x'_n\rangle$ of the same basis. Let us stress that G modifies the state $|x_i\rangle$ of each qudit of the register into a new state $|x'_i\rangle$ of the same qudit, and that we interpret such modifications as the computation made by the corresponding gate. The action of G on a non-factorized vector $\Phi = \sum \alpha_{i_1 \ldots i_n} |x_{i_1}, \ldots, x_{i_n}\rangle$, expressed as a linear combination of the vectors of the n-register basis, is obtained by linearity: $G(\Phi) = \sum \alpha_{i_1 \ldots i_n} G(|x_{i_1}, \ldots, x_{i_n}\rangle)$.

Here we are interested into the quantum realization of classical gates; of course there exist also *genuine quantum gates*, characterized by the fact that some input configurations are transformed into non-trivial superpositions of the vectors of the computational basis. Two examples of such gates are the $\sqrt{\text{NOT}}$ gate, acting on configurations of a single qubit, and the *Hadamard* gate, acting on quantum registers of size 2. In this paper we do not consider this kind of gates.

The collection of all linear operators on $\mathbb{C}^d$ 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 $E_{x,x'}|y\rangle = \mathbf{0}$ 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 $\mathbb{C}^d$ into 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 into the null vector all the $|y\rangle$ with $y \in L_d$ and $y \neq x$. For $i, j \in \{0, 1, \ldots, d-1\}$, the operator $E_{\frac{i}{d-1}, \frac{j}{d-1}}$ can be represented as an order d square matrix having 1 in position $(j+1, i+1)$ and 0 in every other position:

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

A quantum realization of classical (n, d)-gates can be obtained as sums of tensor products of the operators $E_{x,x'}$. For example, let us consider the following "brute force" approach. Let $x_1 x_2 \cdots x_n \mapsto y_1 y_2 \cdots y_n$ be a generic row of the truth table of an (n, d)-gate. For what we have said 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 configuration $x_1 x_2 \cdots x_n$ into the output configuration $y_1 y_2 \cdots y_n$, and collapses all the other input configurations of the n-register 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 (n, d)-gate.

Unfortunately, the formal expressions obtained with the above brute force method are very long, independently of the (n, d)-gate considered. If the gate has a particular structure, we can do better. For example, in Barenco *et al.* (1995) $(n, 2)$-gates are considered whose first input line is a *control* line that determines the action of the gate on the remaining input lines. This approach has been extended in Cattaneo *et al.* (2000b) to the case where an (n, d)-gate can be divided into a k-inputs/k-outputs *control unit* and an $(n-k)$-inputs/$(n-k)$-outputs *target* (also *operating*) *unit*. Thus any input configuration $|x_1, \ldots, x_k, x_{k+1}, \ldots, x_n\rangle$ can be splitted 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 $\otimes^{n-k}\mathbb{C}^2$, 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 (n, d)-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 $\otimes^k\mathbb{C}^2$ 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.

It is clear that the method derived from Conditional Quantum Control does not describe every possible operator, since there are gates which cannot be divided as a control unit and an operating unit. As a consequence, in Cattaneo *et al.* (2000b) we proposed a general method that in some cases allows one to shorten considerably the length of the expression of the unitary operator that constitute a quantum implementation the gate. Basically speaking, our Constants Method works as follows. Let us suppose that we want to give an (n, d)-gate which realizes some 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, then 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 $\mathbb{I} + T$, where T is a linear operator which takes into account the cases where the gate behaves differently from the identity gate.

Whatever is the method used to express the linear operator associated to the (n, d)-gate, the resulting expression is a sum of tensor products of some operators

$E_{x,y}$. Each of these operators can be expressed, using the whole algebraic structure of the associative algebra of operators, as a suitable composition of creation and annihilation operators or, alternatively, as a suitable composition of spin-creation and spin-annihilation operators. We recall that the action of the *creation* operator $a^\dagger$ and of the *annihilation* operator a on the vectors of the canonical orthonormal basis of $\mathbb{C}^d$ is, respectively:

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

and

$$a \left| \frac{k}{d-1} \right\rangle = \sqrt{k} \left| \frac{k-1}{d-1} \right\rangle \quad \text{for} \quad k = 1, 2, \ldots, d-1$$
$$a|0\rangle = \mathbf{0}$$

The action of the *spin-creation* operator J_+ and of the *spin-annihilation* operator J_- on the same vectors is:

$$J_- \left| \frac{k}{d-1} \right\rangle = \sqrt{(k+1)(d-(k+1))} \left| \frac{k+1}{d-1} \right\rangle \quad \text{for} \quad k = 0, 1, \ldots, d-2$$
$$J_-|1\rangle = \mathbf{0}$$

and

$$J_+ \left| \frac{k}{d-1} \right\rangle = \sqrt{k(d-k)} \left| \frac{k-1}{d-1} \right\rangle \quad \text{for} \quad k = 1, 2, \ldots, d-1$$
$$J_+|0\rangle = \mathbf{0}$$

Thus, let $A_{u,v}^{p,q,r}$ denote the expression

$$\underbrace{v \cdots v}_{r} \underbrace{v^* \cdots v^*}_{q} \underbrace{v \cdots v}_{p} u \tag{4.1}$$

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 $E_{\frac{i}{d-1}, \frac{j}{d-1}}$ in terms of creation and annihilation operators as follows:

$$E_{\frac{i}{d-1},\frac{j}{d-1}} = \begin{cases} \dfrac{\sqrt{j!}}{(d-1)!} A^{d-2,d-1-j,0}_{a^\dagger,a^\dagger} & \text{if} \quad i=0 \\ \dfrac{\sqrt{j!}}{(d-1)!} A^{d-1,d-1-j,0}_{a,a^\dagger} & \text{if} \quad i=1 \text{ and } j \geq 1 \\ \dfrac{\sqrt{i!}}{(d-1)!\sqrt{j!}} A^{d-2-i,d-1,j}_{a^\dagger,a^\dagger} & \text{if} \quad (i=1,\, j=0 \text{ and } d \geq 3) \\ & \quad \text{or}\,(1 < i < d-2 \text{ and } j \leq i) \\ \dfrac{\sqrt{j!}}{(d-1)!\sqrt{i!}} A^{i-1,d-1,d-1-j}_{a,a} & \text{if}(i=d-2,\, j=d-1 \text{ and } d \geq 3) \\ & \quad \text{or } (1 < i < d-2 \text{ and } j > i) \\ \dfrac{1}{\sqrt{(d-1)!j!(d-1)}} A^{d-1,j,0}_{a^\dagger,a} & \text{if} \quad i=d-2 \text{ and } j \leq d-2 \\ \dfrac{1}{\sqrt{(d-1)!j!}} A^{d-2,j,0}_{a,a} & \text{if} \quad i=d-1 \end{cases}$$

Alternatively, we can express the operators $E_{\frac{i}{d-1},\frac{j}{d-1}}$ in terms of spin-creation and spin-annihilation operators. Let us consider the formal expression (4.1) applied to $u, v \in \{J_+, J_-\}$; moreover, let

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

Then it holds,

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

5. CONCLUSIONS

We have defined some conservative gates for finite-valued logics, which are able to realize all the main connectives of the many-valued logics of Łukasiewicz, the MV-algebras of Chang and Brower–Zadeh algebras. In particular, the gate CON_3 was a strictly conservative gate for three-valued logics, while the gate wCON_d was a weakly conservative gate for d-valued logics. Moreover, we have proposed a strictly conservative (3, 2)-gate which is (in our opinion) a valid alternative to the Petri–Fredkin gate as a foundation for conservative logic. A d-valued extension of such gate, ALN_d, allows one to seize all the desired connectives, with some minor exceptions.

A mathematical framework that allows one to give quantum realizations of the gates has been introduced, and the possibility to realize our gates with three techniques (the so-called "brute force" method, the Conditional Quantum Control and our Constants Method) has been exposed.

One of the purposes of our work was 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 uncertain information, that are usually based on many-valued and modal logics.

APPENDIX

In this section we give an example of the application of the technique derived from Conditional Quantum Control, and of our Constants Method, to express the unitary operators that constitute a quantum realization of some (n, d)-gates. The formal expression of the unitary operator associated to the gate wCON_d has been obtained through the application of the Conditional Quantum Control technique, while the expression for the unitary operator that implements ALN_d results from the application of our Constants Method. Observe how the latter formula is significantly shorter than the former; this is due to the fact that the gates wCON_d and ALN_d have been conceived with the Constants Method in mind.

Realization of wCON_d with the Conditional Quantum Control Technique

$$P_0 \otimes \left(P_0 \otimes \mathbb{I}_d \otimes \mathbb{I}_d + P_{\frac{1}{d-1}} \otimes \left(\sum_{j=0}^{d-3} E_{\frac{j}{d-1},\frac{j+1}{d-1}} \otimes E_{1,\frac{d-2}{d-1}} + P_0 \otimes \sum_{j=0}^{d-2} P_{\frac{j}{d-1}} \right.\right.$$

$$\left. + \sum_{j=1}^{d-2} \left(E_{\frac{j}{d-1},\frac{j-1}{d-1}} \otimes E_{\frac{d-2}{d-1},1} + P_{\frac{j}{d-1}} \otimes \sum_{k=0}^{d-3} P_{\frac{k}{d-1}} \right) + E_{\frac{d-2}{d-1},1} \otimes E_{1,\frac{d-2}{d-1}} \right.$$

$$+ P_1 \otimes \sum_{j=0}^{d-3} P_{\frac{j}{d-1}} + E_{1,\frac{d-2}{d-1}} \otimes E_{\frac{d-2}{d-1},1} + P_1 \otimes P_1 \Bigg)$$

$$+ \sum_{i=2}^{d-2} P_{\frac{i}{d-1}} \otimes \Bigg(\sum_{j=d-i}^{d-2} E_{0,\frac{i+j}{d-1}-1} \otimes E_{\frac{j}{d-1},1-\frac{i}{d-1}} + P_0 \otimes \sum_{j=0}^{d-1-i} P_{\frac{j}{d-1}}$$

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

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

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

$$+ P_1 \otimes \sum_{j=0}^{d-2-i} P_{\frac{j}{d-1}} + P_1 \otimes P_1 \Bigg)$$

$$+ P_1 \otimes \Bigg(P_0 \otimes P_0 + \sum_{j=0}^{d-2} \left(E_{\frac{j}{d-1},1} \otimes E_{1,\frac{j}{d-1}} + E_{1,\frac{j}{d-1}} \otimes E_{\frac{j}{d-1},1}\right)$$

$$+ \sum_{j=1}^{d-2} \left(E_{0,\frac{j}{d-1}} \otimes E_{\frac{j}{d-1},0} + E_{\frac{j}{d-1},0} \otimes E_{0,\frac{j}{d-1}}\right)$$

$$+ \sum_{j=1}^{d-2} P_{\frac{j}{d-1}} \otimes \sum_{k=1}^{d-2} P_{\frac{k}{d-1}} + P_1 \otimes P_1 \Bigg)\Bigg)$$

$$+ P_{\frac{1}{2}} \otimes \Bigg(P_0 \otimes \mathbb{I}_d \otimes \mathbb{I}_d$$

$$+ \sum_{i=1}^{d-1} P_{\frac{i}{d-1}} \otimes \Bigg(\sum_{j=0}^{i-1} E_{1,\frac{j}{i}} \otimes E_{\frac{j}{d-1},1-\frac{j}{d-1}-\frac{j}{i}} + P_1 \otimes \sum_{j=i}^{d-1} P_{\frac{j}{d-1}}$$

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

$$+ P_{\frac{j}{d-1}} \otimes \sum_{\substack{k=0\\ k\neq d-1-j\left(1-\frac{i}{d-1}\right)}}^{d-1} P_{1-\frac{k}{d-1}\left(1-\frac{k}{d-1}\right)} \Bigg)\Bigg)\Bigg)$$

$$+ \sum_{\substack{i=1 \\ i \neq \frac{i}{d-1}}}^{d-2} P_{\frac{i}{d-1}} \otimes \mathbb{I}_d \otimes \mathbb{I}_d \otimes \mathbb{I}_d$$

$$+ P_1 \otimes \Bigg(P_0 \otimes \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} E_{\frac{i}{d-1},\frac{j}{d-1}} \otimes E_{\frac{j}{d-1},\frac{i}{d-1}}$$

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

$$+ \sum_{j=0}^{i-1} \Big(E_{1,\frac{i}{d-1}} \otimes E_{\frac{j}{d-1},1+\frac{j-i}{d-1}} + P_{\frac{j}{d-1}} \otimes \mathbb{I}_d \Big)$$

$$+ \sum_{j=i+1}^{d-2} \Big(E_{\frac{j}{d-1},\frac{i}{d-1}} \otimes E_{0,\frac{j-i}{d-1}} + P_{\frac{j}{d-1}} \otimes \sum_{k=1}^{d-2} P_{\frac{k}{d-1}} \Big)$$

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

$$+ P_{\frac{i}{d-1}} \otimes P_0 + P_1 \otimes P_1 \Bigg)$$

$$+ P_{\frac{d-2}{d-1}} \otimes \Bigg(E_{1,\frac{d-2}{d-1}} \otimes E_{\frac{d-2}{d-1},1} + E_{\frac{d-2}{d-1},1} \otimes E_{1,\frac{d-2}{d-1}}$$

$$+ \sum_{j=0}^{d-3} \Big(E_{1,\frac{d-2}{d-1}} \otimes E_{\frac{j}{d-1},\frac{j+1}{d-1}} + P_{\frac{j}{d-1}} \otimes \mathbb{I}_d \Big)$$

$$+ \sum_{j=1}^{d-2} E_{\frac{d-2}{d-1},1} \otimes E_{\frac{j}{d-1},\frac{j-1}{d-1}} + P_{\frac{d-2}{d-1}} \otimes P_0 + P_1 \otimes P_1 \Bigg)$$

$$+ P_1 \otimes \mathbb{I}_d \otimes \mathbb{I}_d \Bigg)$$

Realization of ALN$_d$ with the Constants Method

$$\sum_{i=0}^{d-2} \sum_{j=i}^{d-1} \Big(E_{\frac{j}{d-1},1-\frac{i+j}{d-1}} \otimes E_{\frac{i}{d-1},\frac{j}{d-1}} \otimes E_{1,\frac{j}{d-1}} - P_{\frac{j}{d-1}} \otimes P_{\frac{i}{d-1}} \otimes P_1 \Big)$$

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

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

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

$$+\sum_{i=1}^{d-2}\sum_{j=0}^{i-1}\left(E_{0,\frac{i-j}{d-1}}\otimes E_{\frac{j}{d-1},\frac{j}{d-1}}\otimes E_{\frac{i}{d-1},\frac{j}{d-1}}-P_0\otimes P_{\frac{j}{d-1}}\otimes P_{\frac{i}{d-1}}\right)$$

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

$$+\sum_{i=2}^{d-2}\sum_{j=d-i}^{d-2}\left(E_{\frac{i}{d-1},1-\frac{i}{d-1}}\otimes E_{0,\frac{i+j}{d-1}-1}\otimes E_{\frac{j}{d-1},\frac{i}{d-1}}-P_{\frac{i}{d-1}}\otimes P_0\otimes P_{\frac{j}{d-1}}\right)$$

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

$$+\mathbb{I}_d\otimes\mathbb{I}_d\otimes\mathbb{I}_d$$

ACKNOWLEDGMENT

This work has been supported by the MURST Project "Unconventional Computational Models: Syntactic and Combinatorial Methods."

REFERENCES

Barenco, A., Deutsch, D., Ekert, A., and Jozsa, R. (1995). Conditional quantum control and logic gates. *Physical Review Letters* **74**, 4083–4086.

Cattaneo, G., Leporati, A., and Leporini, R. (2002). Fredkin gates for finite-valued reversible and conservative logics. *Journal of Physics A: Mathematical and General* **35**, 9755–9785.

Cattaneo, G., Leporati, A., and Leporini, R. (2000). Quantum realization of fredkin gates for finite-valued reversible and conservative logics, Manuscript.

Chang, C. C. (1958). Algebraic analysis of many valued logics. *Transactions of American Mathematical Society* **88**, 467–490.

Chang, C. C. (1959). A new proof of the completeness of Łukasiewicz axioms. *Transactions of American Mathematical Society* **93**, 74–80.

Fredkin, E. and Toffoli, T. (1982). Conservative logic. *International Journal of Theoretical Physics* **21**, 219–253.

Gottesman, D. (1999). Fault-tolerant quantum computation with higher-dimensional systems. *Chaos, Solitons, and Fractals* **10**, 1749–1758.

Lukasiewicz, J. (1920). O logice trójwartośćiowjei. *Ruch Filozoficzny* **5**, 170. (English Version: On three-valued logic. In *Selected Works*, L. Borkowski, ed., North-Holland, Amsterdam, p. 87).

Lukasiewicz, J. (1970). In *Selected Works*, L. Borkowski, ed., North-Holland, Amsterdam.

Petri, C. A. (1967). Gründsatzliches zur Beschreibung diskreter Prozesse. In *Proceedings of the 3rd Colloquium über Automatentheorie (Hannover, 1965)*, Birkhäuser Verlag, Basel, pp. 121–140. (English Version: Fundamentals of the representation of discrete processes, ISF Report 82.04 (1982), translated by H. J. Genrich and P. S. Thiagarajan).

Rescher, N. (1969). *Many-Valued Logics*, McGraw-Hill.

Rosser, J. B. and Turquette, A. R. (1952). *Many-Valued Logics*, North-Holland, Amsterdam.

Wootters, W. K. and Zurek, W. H. (1982). A single quantum cannot be cloned. *Nature* **299**, 802–803.