# Fredkin gates for finite-valued reversible and conservative logics

INSTITUTE OF PHYSICS PUBLISHING JOURNAL OF PHYSICS A: MATHEMATICAL AND GENERAL

J. Phys. A: Math. Gen. **35** (2002) 9755–9785 PII: S0305-4470(02)37993-9

# Fredkin gates for finite-valued reversible and conservative logics

**G Cattaneo, A Leporati and R Leporini**

Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano – Bicocca, via Bicocca degli Arcimboldi 8, 20126 Milano, Italy

E-mail: cattang@disco.unimib.it, leporati@disco.unimib.it and leporini@disco.unimib.it

Received 10 June 2002, in final form 23 September 2002
Published 7 November 2002
Online at stacks.iop.org/JPhysA/35/9755

**Abstract**
The basic principles and results of *conservative logic* introduced by Fredkin and Toffoli in 1982, on the basis of a seminal paper of Landauer, are extended to *d*-valued logics, with a special attention to three-valued logics. Different approaches to *d*-valued logics are examined in order to determine some possible universal sets of logic primitives. In particular, we consider the typical connectives of Łukasiewicz and Gödel logics, as well as Chang's MV-algebras. As a result, some possible three-valued and *d*-valued universal gates are described which realize a functionally complete set of fundamental connectives. Two no-go theorems are also proved.

PACS numbers: 03.67.Lx, 02.10.−v

## 1. Introduction

The present paper is based on two different research areas that have been developed in the past years: *conservative logic* and *many-valued logics*. Conservative logic is a mathematical model of computation introduced by Fredkin and Toffoli in [FT82] on the basis of the seminal paper of Landauer [La61] (see also [Be73]) to improve the efficiency and performance of computing processes in terms of dissipated energy. The model is based on the *Fredkin gate*, a universal three-input/three-output Boolean gate which is both conservative and reversible. As a matter of fact this gate was introduced by Petri some years before Fredkin in [Pe67], and thus in the following we will call it the Petri–Fredkin gate. On the other hand, many-valued logics and modal logics, which have known a great diffusion due to their ability to manage incomplete and/or uncertain knowledge, are extensions of the classical Boolean logic. Different approaches to many-valued and modal logics have been considered in literature; for an overview see, for instance, [Re69, RT52]. These main subjects are briefly described in the next sections.

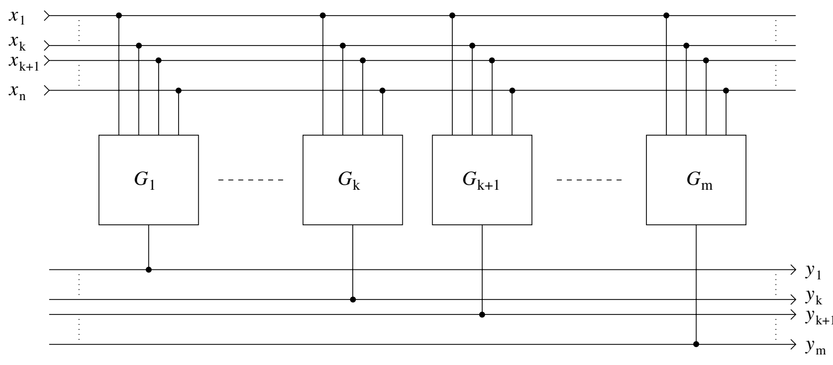

**Figure 1.** Standard parallel architecture of an *n*-input/*m*-output gate.

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. In particular, we consider the typical connectives of Łukasiewicz and Gödel logics, as well as Zawirski/Chang's MV-algebras. As a result, we describe some possible three-valued and finite-valued universal gates—that can be thought of as finite-valued extensions of the Petri–Fredkin gate—which realize a functionally complete set of fundamental connectives.

We also prove two no-go theorems: the first one states the impossibility of obtaining the FAN-OUT gate from a strictly conservative $d$-valued gate if $d$ is greater than the number $n$ of input/output lines. The second one says that it is impossible to realize a $d$-valued ($d \geqslant 3$) three-input/three-output reversible and conservative extension of the Petri–Fredkin gate able to realize simultaneously the Łukasiewicz connectives, the Gödel implication and the MV-connectives. Owing to these results, some weakening conditions are investigated and some $d$-valued universal gates that have the properties required by the conservative and many-valued paradigms are presented.

## 2. Reversibility, conservativeness and conditional control of Boolean gates

Computational models are usually based upon Boolean logic, and use some universal set of primitive connectives, such as {AND, NOT}.

From a general point of view, a (*classical deterministic*) *n-input/m-output gate* (where $n, m$ are positive integers) is a special-purpose *computer* schematized as a device able to compute (Boolean) *logical functions* $G : \{0, 1\}^n \to \{0, 1\}^m$. Any $\vec{x} = (x_1, x_2, \ldots, x_n) \in \{0, 1\}^n$ (resp., $\vec{y} = (y_1, y_2, \ldots, y_m) \in \{0, 1\}^m$) is called an *input* (resp., *output*) *vector*. For every $i \in \{1, 2, \ldots, n\}$ (resp., $j \in \{1, 2, \ldots, m\}$), called the *input* (resp., *output*) *bit* of position $i$ (resp., $j$), the Boolean value $x_i \in \{0, 1\}$ (resp., $y_j \in \{0, 1\}$) is said to be the *state* of bit $i$ (resp., $j$) of vector $\vec{x}$ (resp., $\vec{y}$). Finally, in the following we denote by $\lambda_f$ the generic vector belonging to the range of $G$.

The action of the multi-output map $G$ on an input vector $\vec{x}$ produces the output vector $G(\vec{x}) = (G_1(\vec{x}), G_2(\vec{x}), \ldots, G_m(\vec{x}))$ determined by the component *logical truth functions* (single-output maps) $G_j : \{0, 1\}^n \to \{0, 1\}$, for any $j = 1, 2, \ldots, m$, with a possible *parallel* implementation drawn in figure 1.

*Conservative logic* is a theoretical model of computation whose principal aim is to compute with zero *internal power dissipation*. This goal is reached by basing the model upon *reversible*

**Table 1.** The Landauer three-input/three-output gate.

| $x_1$ | $x_2$ | $x_3$ | $\rightarrow$ | $y_1$ | $y_2$ | $y_3$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 0 | 0 | 0 |
| 0 | 0 | 1 | | 1 | 1 | 0 |
| 0 | 1 | 0 | | 0 | 0 | 0 |
| 0 | 1 | 1 | | 1 | 1 | 0 |
| 1 | 0 | 0 | | 0 | 0 | 0 |
| 1 | 0 | 1 | | 1 | 1 | 0 |
| 1 | 1 | 0 | | 0 | 0 | 1 |
| 1 | 1 | 1 | | 1 | 1 | 1 |

and *conservative* primitives, which reflect physical principles such as the reversibility of microscopic dynamical laws and the conservation of certain *physical quantities*, such as the energy of the physical system used to perform the computations.

*Reversibility.* Most of the time, computational models lack *reversibility*; that is, one cannot in general deduce the input values of a gate from its output values. For example, knowing that the output of an AND gate is the logical value 0, one cannot deduce the input values that generated it. The original motivation for the study of reversibility in classical computing came from the observation that heat dissipation is one of the major obstacles for miniaturization of classical computers and the fact that the second law of thermodynamics implies that irreversible state changes during computation must dissipate heat. 'Thus, in the more abstract context of computing, the laws of "conservation of information" may play a role analogous to those of conservation of energy and momentum in physics.' [Tof80]

Lack of reversibility means that during the computation some information is lost. As shown by Landauer [La61] (see also [Be88] which can be found in [LR90]), a loss of information implies a loss of energy and therefore any computational model based on irreversible primitives is necessarily *informationally dissipative*. This is nowadays known (see [Be98]) as Landauer's principle

> To erase a bit of classical information within a computer, 1 bit of entropy must be expelled into the computer's environment, typically in the form of waste heat. Thus logical irreversibility is associated with physical irreversibility and requires a minimal heat generation, per machine cycle, typically of the order of $kT$ for each irreversible operation.

In practice the heat dissipation per bit processed by (irreversible) computers in use today is some orders of magnitude greater than the theoretical lower bound $kT \ln 2$ given by Landauer's principle. However, if computer hardware continues to shrink in size as so far, then the only feasible option to beat Landauer's lower bound seems to be reversible computation[1].

Let us make these considerations less informal by considering as a first example the logical function $L$ of table 1 computed by a three-input/three-output gate and discussed by Landauer in [La61]. Following Landauer,

> There are eight possible initial states, and in thermal equilibrium they will occur with equal probability. How much entropy reduction will occur in a machine cycle? States

[1] In modern computers heat dissipation is about $kT \times 10^8$ per logical operation. The heat must be removed by external means, for example, by constant cooling of all components by the thermal coupling of the circuits to a heat reservoir, i.e., air.

**Table 2.** Example of a two-input/four-output reversible gate.

| $x_1$ | $x_2$ | $\longmapsto$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ |
|---|---|---|---|---|---|---|
| 0 | 0 | | 0 | 0 | 0 | 0 |
| 0 | 1 | | 0 | 1 | 1 | 0 |
| 1 | 0 | | 0 | 1 | 1 | 1 |
| 1 | 1 | | 1 | 0 | 0 | 1 |

(111) and (001) occur with a probability 1/8 each; states (110) and (000) have a probability of occurrence of 3/8 each. The initial entropy was

$$S_i(3) = -k\sum_{\vec{x}} P(\vec{x})\log_e P(\vec{x}) = -k\sum \frac{1}{8}\log_e\frac{1}{8} = 3k\log_e 2.$$

The final entropy is

$$\begin{aligned} S_f(L) &= -k\sum_{\lambda_f} P_L(\lambda_f)\log_e P_L(\lambda_f) \\ &= -k\left(\frac{1}{8}\log_e\frac{1}{8} + \frac{1}{8}\log_e\frac{1}{8} + \frac{3}{8}\log_e\frac{3}{8} + \frac{3}{8}\log_e\frac{3}{8}\right). \end{aligned}$$

The difference $S_i(3) - S_f(L)$ is $0.82k$. The minimum dissipation, if the initial state has no useful information, is therefore $E_i(3) - E_f(L) = (S_i(3) - S_f(L))T = 0.82kT$.

More precisely, for any admissible output $\lambda_f = (y_1, y_2, y_3) \in \mathrm{Im}(L)$ we can introduce the set

$$M_L(\lambda_f) := L^{-1}(\lambda_f) = \{(x_1, x_2, x_3) \in \{0, 1\}^3 : L(x_1, x_2, x_3) = \lambda_f\}$$

whose cardinality $|M_L(\lambda_f)|$ expresses the *indistinguishability degree* of the output $\lambda_f$, i.e., the total number of possible inputs which cannot be distinguished by $L$ with respect to output $\lambda_f$. Then the above probabilities can be expressed as

$$P_L(\lambda_f) = \frac{|M_L(\lambda_f)|}{|\{0, 1\}^3|} = \frac{1}{8}|M_L(\lambda_f)|.$$

We now want to extend these considerations in order to compare the dissipation of *informational energy* in the case of devices whose number of output lines is not necessarily equal to the number of input lines. To this end, let us denote by $\mathcal{F}(\{0, 1\}, n, m) = (\{0, 1\}^m)^{\{0,1\}^n}$ the collection of all $n$-input/$m$-output Boolean gates $G : \{0, 1\}^n \to \{0, 1\}^m$ (from now on, with a little abuse of notation, we will refer interchangeably to gates and to the maps computed by them), and by $\mathcal{F}(\{0, 1\}, n, \mathbb{N}) = \bigcup_{m\in\mathbb{N}} \mathcal{F}(\{0, 1\}, n, m)$ the collection of all $n$-input/multi-output Boolean gates. For instance, $\mathcal{F}(\{0, 1\}, 2, \mathbb{N})$ contains both the gate $\mathrm{AND} : \{0, 1\}^2 \to \{0, 1\}$, associating with the Boolean pair $(x_1, x_2)$ the Boolean value $\mathrm{AND}(x_1, x_2) = x_1 \cdot x_2$, and the two-input/four-output gate defined in table 2.

In $\mathcal{F}(\{0, 1\}, n, \mathbb{N})$, owing to the assumption that in thermal equilibrium all possible input vectors $\vec{x}$ will occur with equal probability $P(\vec{x})$, the *input information entropy* is independent of the particular gate and equal to

$$\begin{aligned} S_i(n) :&= -k\sum_{\vec{x}} P(\vec{x})\log_e P(\vec{x}) \\ &= -k\sum_{\vec{x}} \frac{1}{2^n}\log_e\frac{1}{2^n} = nk\log_e 2. \end{aligned}$$

What depends on the gate $G \in \mathcal{F}(\{0,1\}, n, \mathbb{N})$ is the set of $\lambda_f$*-indistinguishable input vectors*, where $\lambda_f \in \mathrm{Im}(G)$ is any *admissible output vector* of $G$:

$$\begin{aligned} M_G(\lambda_f) :&= G^{-1}(\lambda_f) \\ &= \{(x_1, x_2, \ldots, x_n) \in \{0,1\}^n : G(x_1, x_2, \ldots, x_n) = \lambda_f\}. \end{aligned}$$

Let us notice that the collection $\{M_G(\lambda_f) : \lambda_f \in \mathrm{Im}(G)\}$ of all such subsets constitutes a partition of $\{0,1\}^n$. Borrowing some terminology from axiomatic quantum mechanics, elements $\lambda_f$ from $\mathrm{Im}(G)$ can be called *eigenvalues* (possible output values) of $G$, $\mathrm{Im}(G)$ is the *spectrum* of $G$, the set $M_G(\lambda_f)$ is the *eigenspace* (set of possible inputs) associated with the eigenvalue $\lambda_f$, and the characteristic function $\chi_{M_G(\lambda_f)}$ (=1 if $\vec{x} \in M_G(\lambda_f)$, and 0 otherwise) is the spectral projection associated with the eigenspace. The collection of all spectral projections of $G$, for $\lambda_f$ ranging on the spectrum of $G$, is a *spectral identity resolution* of $G$:

$$\mathrm{Id} = \sum_{\lambda_f \in \mathrm{Im}(G)} \chi_{M_G(\lambda_f)} \qquad G = \sum_{\lambda_f \in \mathrm{Im}(G)} \lambda_f \chi_{M_G(\lambda_f)}.$$

The *indistinguishability degree* of the admissible output vector $\lambda_f \in \mathrm{Im}(G)$ is defined as $|M_G(\lambda_f)|$, and the *probability of occurrence* of $\lambda_f$ as

$$P_G(\lambda_f) = \frac{1}{2^n}|M_G(\lambda_f)|$$

with corresponding *output information entropy*

$$S_f(G) := -k \sum_{\lambda_f \in \mathrm{Im}(G)} P_G(\lambda_f) \log_e P_G(\lambda_f) \tag{1a}$$

$$= -\frac{k}{2^n} \sum_{\lambda_f \in \mathrm{Im}(G)} |M_G(\lambda_f)| \cdot \log_e |M_G(\lambda_f)| + S_i(n). \tag{1b}$$

Hence, the *information energy dissipation* of $G$ is

$$\begin{aligned} \Delta E(G) &= (S_i(n) - S_f(G)) \cdot T \\ &= \frac{kT}{2^n} \sum_{\lambda_f \in \mathrm{Im}(G)} |M_G(\lambda_f)| \cdot \log_e |M_G(\lambda_f)|. \end{aligned}$$

In particular, the information energy loss by the AND gate is $\Delta E(\mathrm{AND}) = \frac{3kT}{4}\log_e 3 \approx 0.82kT$ whereas the gate of table 2 has no information energy dissipation, owing to its reversibility.

From (1) it follows immediately that the output information entropy is bounded by

$$0 \leqslant S_f(G) \leqslant S_i(n).$$

A generic gate $G : \{0,1\}^n \to \{0,1\}^m$ is *reversible* (one-to-one mapping) if and only if $n = m$ and every element $\lambda_f$ of $\{0,1\}^n$ is an admissible output; in this case the corresponding $|M_G(\lambda_f)|$ is equal to 1 which leads to $S_i(n) - S_f(G) = 0$, and thus also $E_i(n) - E_f(G) = 0$. Precisely, the following proposition holds.

**Proposition 2.1.** *Let G be an n-input Boolean gate. Then the information energy dissipation is bounded by*

$$0 \leqslant \Delta E(G) \leqslant T S_i(n) = nkT \log_e 2.$$

*Moreover,*

1. $\mathrm{Im}(G)$ *is a singleton if and only if* $\Delta E(G) = T S_i(n)$*;*
2. *the gate is reversible (one-to-one) if and only if* $\Delta E(G) = 0$.

**Table 3.** Example of a two-input/two-output reversible gate, i.e., a permutation of the set $\{0, 1\}^2$.

| $x_1$ | $x_2$ | $\longmapsto$ | $y_1$ | $y_2$ |
|---|---|---|---|---|
| 0 | 0 | | 1 | 1 |
| 0 | 1 | | 1 | 0 |
| 1 | 0 | | 0 | 1 |
| 1 | 1 | | 0 | 0 |

**Table 4.** Example of a two-input/two-output conservative gate, i.e., in each row the output vector is a permutation of the input vector.

| $x_1$ | $x_2$ | $\longmapsto$ | $y_1$ | $y_2$ |
|---|---|---|---|---|
| 0 | 0 | | 0 | 0 |
| 0 | 1 | | 1 | 0 |
| 1 | 0 | | 1 | 0 |
| 1 | 1 | | 1 | 1 |

Let us stress that in the case of an $n$-input/$n$-output gate realizing the logical function $G : \{0, 1\}^n \to \{0, 1\}^n$ the reversibility condition corresponds to the fact that $G$ is a permutation of the set $\{0, 1\}^n$. For instance, a two-input/two-output reversible gate computes a permutation of the set $\{00, 01, 10, 11\}$. Table 3 shows an example of a gate of this kind.

*Conservativeness.* After [FT82], this condition is usually modelled by the property that each output $(y_1, y_2, \dots, y_n)$ of the gate is a permutation of the corresponding input $(x_1, x_2, \dots, x_n)$. We call this condition *strict conservativeness* of the gate. Trivially a gate of this kind must necessarily have the same number of input and output lines. Moreover, it is immediate to see that the number of 1s in the output $(y_1, y_2, \dots, y_n)$ is the same as in the input $(x_1, x_2, \dots, x_n)$. According to Toffoli [Tof80], in a conservative logic circuit the number of 1s in the input is the same as the number of 1s in different parts of the circuit. This quantity is additive, and can be shown to play a formal role analogous to that of energy in physical systems. In table 4 an example of a (strictly) conservative two-input/two-output gate is presented.

We observe that some conservative (but not reversible) circuits using complementary signal streams were discussed by von Neumann [vN56] as early as 1952. More recently, Kinoshita and associates [Ki76] worked out a classification of logic elements that ‘conserve’ 0s and 1s; their work, motivated by research in magnetic-bubble circuitry, mentions the possibility of more energy-efficient computation, but has apparently little concern for reversibility.

In general, in concrete devices the Boolean values 0 and 1 are realized by impulses of energy $\varepsilon_0$ and $\varepsilon_1$ respectively, with $0 < \varepsilon_0 < \varepsilon_1$ (see figure 2). Indeed, there should be a sufficiently large energy barrier between them so that no spontaneous transition, which would evidently be detrimental, can occur between the two states.

In the case of a generic (not necessarily conservative) gate which computes a logical function $G : \{0, 1\}^n \to \{0, 1\}^n$, a transition $\vec{x} = (x_1, x_2, \dots, x_n) \mapsto G(\vec{x}) = \vec{y} = (y_1, y_2, \dots, y_n)$ corresponding to a row of the tabular definition of the Boolean function produces a variation of the internal energy whose amount is

$$\Delta U(\vec{x}, \vec{y}) = \left(\varepsilon_{y_1} + \varepsilon_{y_2} + \cdots + \varepsilon_{y_n}\right) - \left(\varepsilon_{x_1} + \varepsilon_{x_2} + \cdots + \varepsilon_{x_n}\right).$$

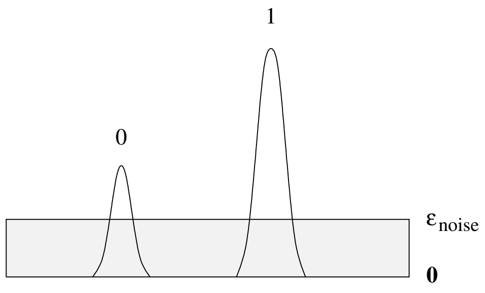

**Figure 2.** Realization of Boolean values 0 and 1 by impulses of energy $\varepsilon_0$ and $\varepsilon_1$, with $0 < \varepsilon_{\text{noise}} < \varepsilon_0 < \varepsilon_1$.

**Table 5.** The EXC two-input/two-output reversible and conservative gate.

| $x_1$ | $x_2$ | $\longmapsto$ | $y_1$ | $y_2$ |
|---|---|---|---|---|
| 0 | 0 | | 0 | 0 |
| 0 | 1 | | 1 | 0 |
| 1 | 0 | | 0 | 1 |
| 1 | 1 | | 1 | 1 |

Therefore, the total *internal energy dissipation* of $G$ is

$$\Delta U(G) = \sum_{\vec{x}\in\{0,1\}^n} \Delta U(\vec{x}, G(\vec{x})).$$

Conservativeness of $G$ trivially implies zero internal energy dissipation ($\Delta U(G) = 0$).

*Conclusions on reversibility and conservativeness.* Up to now the loss of energy due to irreversibility and non-conservativeness of logical primitives was irrelevant compared to the amount of energy dissipated by an electronic device implementing logical gates. But the problems rising from extreme miniaturization in electronics have led to the investigation of new ways of implementing circuits, borrowing the knowledge of quantum mechanics. These new research areas introduce the possibility of reversible and conservative computations based on reversible and conservative physical behaviour, encouraging the definition of new computational models.

Let us stress that conservativeness and reversibility are two independent notions: a gate can satisfy both properties, only one of them, or none. For example, a conservative non-reversible gate is given in table 4 (both inputs 01 and 10 are mapped into the same output 10) and a reversible non-conservative gate is shown in table 3 (transition $00 \mapsto 11$ is non-conservative). Another famous example of a reversible and non-conservative Boolean gate is the Toffoli gate, whose behaviour is defined by the following map CCNOT: $\{0, 1\}^3 \to \{0, 1\}^3$:

$$y_1 = x_1 \qquad y_2 = (x_1 \wedge x_3) \oplus x_2 \qquad y_3 = x_3. \tag{2}$$

Just as the Fredkin gate, the Toffoli gate was first presented by Petri in [Pe67], and thus it should be called the Petri–Toffoli gate. The same gate (with outputs $y_2$ and $y_3$ exchanged) has also been considered by Feynman in [Fe85], under the name of Controlled–Controlled-NOT (CCNOT).

A simple example of a reversible and conservative two-inputs/two-outputs gate is the realization of the *exchange* logical function EXC : $\{0, 1\}^2 \to \{0, 1\}^2$ whose tabular representation is given in table 5. In each row the output pair $(y_1, y_2)$ is a permutation of the

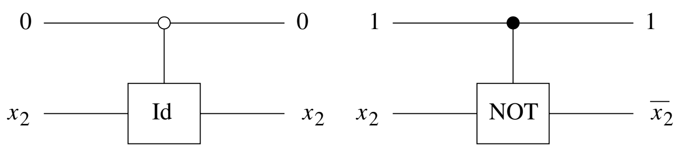

**Figure 3.** The conditional action of the Controlled-NOT gate.

**Table 6.** The Controlled-NOT reversible non-conservative gate.

| $x_1$ | $x_2$ | $\longmapsto$ | $y_1$ | $y_2$ |
|---|---|---|---|---|
| 0 | 0 | | 0 | 0 |
| 0 | 1 | | 0 | 1 |
| 1 | 0 | | 1 | 1 |
| 1 | 1 | | 1 | 0 |

corresponding input pair $(x_1, x_2)$, whereas the map EXC is a (global) permutation of the set $\{0, 1\}^2 = \{00, 01, 10, 11\}$.

*Conditional control gates.* Let us consider the Boolean two-input/two-output reversible non-conservative gate $G^{(CN)} : \{0, 1\}^2 \to \{0, 1\}^2$ whose component maps are the following:

$$G_1^{(CN)} : \{0, 1\}^2 \to \{0, 1\} \qquad G_1^{(CN)}(x_1, x_2) := x_1$$

$$G_2^{(CN)} : \{0, 1\}^2 \to \{0, 1\} \qquad G_2^{(CN)}(x_1, x_2) := x_1 \oplus x_2.$$

The corresponding truth table is given in table 6.

We can describe the behaviour 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 *target input* $x_2$, transforming it into the output $y_2$. To be precise, if the control input is 1 then the value of the target line is negated (i.e., the gate NOT acts on $x_2$ when $x_1 = 1$), otherwise it is left unchanged (i.e., the gate Id acts on $x_2$ when $x_1 = 0$). Formally, this is realized by a direct connection of the first input line with the first output line, whereas the action on the input of the second line is described by two maps $\delta_{x_1}^{(CN)} : \{0, 1\} \to \{0, 1\}$, where $\delta_{x_1}^{(CN)} = G_2^{(CN)}(x_1, \cdot)$ for $x_1 \in \{1, 0\}$. Precisely,

$$\delta_0^{(CN)} := G_2^{(CN)}(0, \cdot) = \text{Id} \qquad \text{and} \qquad \delta_1^{(CN)} := G_2^{(CN)}(1, \cdot) = \text{NOT}.$$

The input value of the control unit $x_1$ selects the map $\delta_{x_1}^{(CN)}$ (either the identity or the NOT map) which acts on the input value $x_2$ of the second line (figure 3). For this reason this gate is called the *Controlled*-NOT (usually abbreviated as CNOT) gate.

From the general viewpoint, the *conditional control* method applies to the cases in which the $n$-input/$n$-output gate under consideration can be divided into two parts: a *control unit* and a *target* (also *operating*) *unit* (see figure 4). The control unit has in general $k$ input and $k$ output lines, while the target unit has $(n - k)$ input and $(n - k)$ output lines. Thus any input vector $x_1, \ldots, x_k, x_{k+1}, \ldots, x_n$ can be split into two parts: the *control configuration* $x_1, \ldots, x_k$ and the *operating* (sometimes also called *target*) *input* $x_{k+1}, \ldots, x_n$. Any of the $2^k$ possible control configurations $x_1, \ldots, x_k$ is labelled by the integer number $a = \sum_{t=1}^{k} x_t 2^{t-1}$. Moreover, $2^k$ functions $\delta_0, \delta_1, \ldots, \delta_{2^k-1}$ of the kind $\{0, 1\}^{n-k} \to \{0, 1\}^{n-k}$ are stored in the memory of the control unit, the function $\delta_a$ being bijectively associated with the configuration labelled by the integer number $a \in \{0, \ldots, 2^k - 1\}$.

When a configuration $x_1, \ldots, x_k$ (which in the following will be denoted indifferently either by its vectorial notation $\vec{a}$ or by its decimal representation $a$, depending on the particular context) is fed as input to the control lines two things happen:

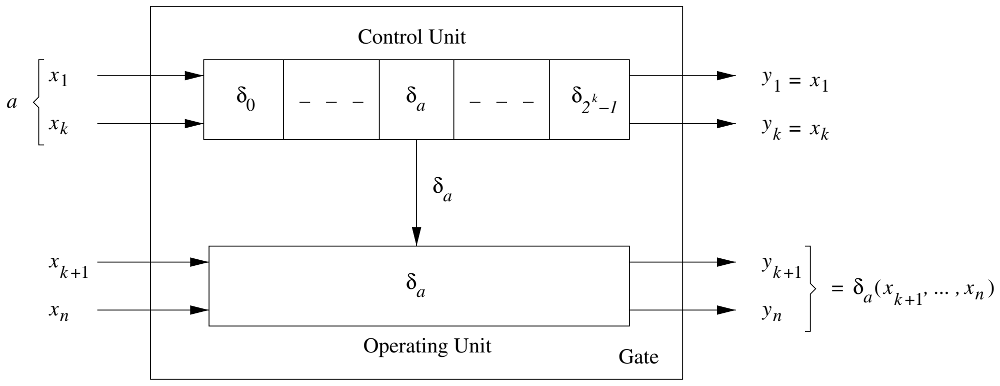

**Figure 4.** Ideal realization of a generic conditional control gate: the gate is divided into a *control unit* and an *operating unit*. The input values of the control unit are left unchanged and select a prescribed function to be applied to the input values of the operating unit.

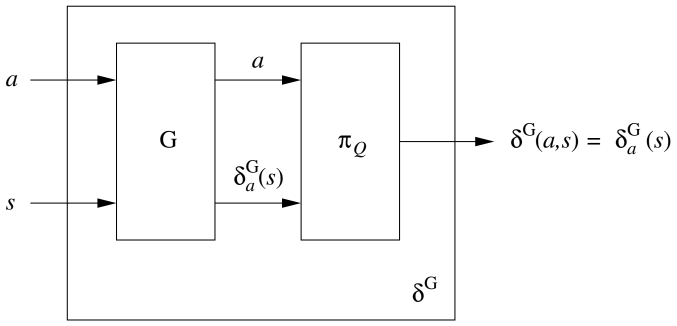

**Figure 5.** Finite automaton generated by a conditional control gate.

1. the control configuration $x_1, \dots, x_k$ is returned unchanged on the output lines of the control unit;
2. the function $\delta_a$ bijectively associated with the control configuration is selected and applied to the input vector $x_{k+1}, \dots, x_n$ of the operating unit, producing the output vector $\delta_a(x_{k+1}, \dots, x_n)$.

We can look at a controlled gate as a finite automaton. The original space $\{0, 1\}^n$ on which a controlled gate $G$ acts can be split into the set $\mathcal{A} := \{0, 1\}^k$, called the *alphabet* of the gate, and the set $Q := \{0, 1\}^{n-k}$, called the *phase* space of the gate; elements of $\mathcal{A}$ are *symbols* of the alphabet and elements of $Q$ are *states* of the gate. Hence, the gate can be represented as a mapping $G : \mathcal{A} \times Q \to \mathcal{A} \times Q$, associating with any symbol–state pair $(\vec{a}, \vec{s})$ a new symbol–state pair $G(\vec{a}, \vec{s}) := \left(\vec{a}, \delta^G_a(\vec{s})\right)$. Therefore, if we put the gate in cascade with the trivial *decoder* (according to [Tof80]) $\pi_Q : \mathcal{A} \times Q \to Q$ associating with any pair $(a, s)$ the state $\pi_Q(a, s) := s$ one obtains a deterministic finite *automaton* $\mathbb{A}^G = \langle \mathcal{A}, Q, \delta^G \rangle$ with (finite) alphabet $\mathcal{A}$, set of states $Q$, and *next state* (also *transition*) *function* $\delta^G := (\pi_Q \circ G) : \mathcal{A} \times Q \to Q$ associating with any symbol–state pair $(\vec{a}, \vec{s})$ the 'next' state $\vec{s}\,' = \delta^G(\vec{a}, \vec{s}) := \pi_Q(G(\vec{a}, \vec{s})) = \delta^G_a(\vec{s})$ (see figure 5). We observe that this automaton can be equivalently described by the pair $\left\langle Q, \left\{\delta^G_0, \delta^G_1, \dots, \delta^G_{2^k-1}\right\}\right\rangle$ consisting of the phase space $Q = \{0, 1\}^{n-k}$ and the collection of $2^k$ transformations of the phase space $\delta^G_a : Q \to Q$, for $a$ running in $\{0, 1, \dots, 2^k - 1\}$.

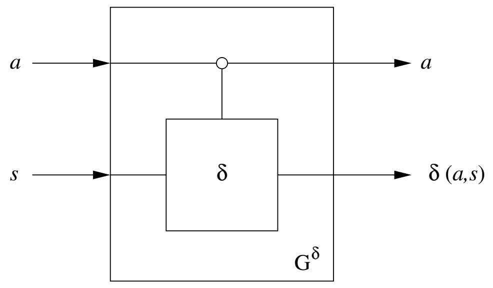

**Figure 6.** Conditional control gate generated by a finite automaton.

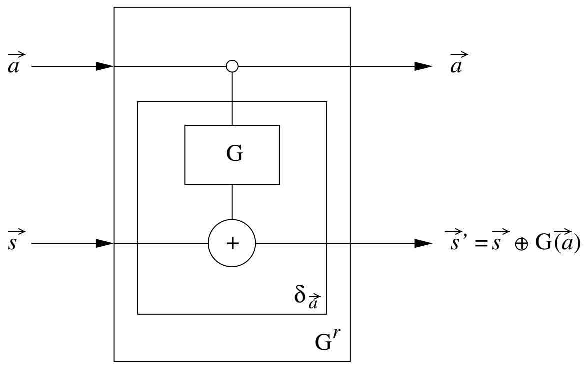

**Figure 7.** Reversible conditional control gate generated by a non-reversible gate $G$.

Vice versa, any (finite) automaton $\mathbb{A} = \langle \mathcal{A}, Q, \delta \rangle$ consisting of the (finite) alphabet $\mathcal{A}$, the (finite) phase space $Q$, and the next state function $\delta : \mathcal{A} \times Q \to Q$ can be equivalently described by the pair $\langle Q, \{\delta_0, \delta_1, \ldots, \delta_{|\mathcal{A}|-1}\} \rangle$ based on the phase space $Q$ and the (finite) collection of phase space transformations $\delta_a : Q \to Q$ (for $a \in \{0, 1, \ldots, |\mathcal{A}| - 1\}$) associating with any state $\vec{s}$ the next state $\vec{s}\,' = \delta_a(\vec{s}) := \delta(\vec{a}, \vec{s})$. This automaton generates a controlled gate $G^\delta : \mathcal{A} \times Q \to \mathcal{A} \times Q$ (see figure 6) associating with the symbol–state input pair $(\vec{a}, \vec{s})$ the symbol–state output pair $G^\delta(\vec{a}, \vec{s}) := (\vec{a}, \delta(\vec{a}, \vec{s}))$. Trivially, if $|A| = 2^k$ and $|Q| = 2^h$ then by a suitable binary representation of each symbol $\vec{a}$ and each state $\vec{s}$ this conditional control gate is realized by a mapping $G^\delta : \{0, 1\}^n \to \{0, 1\}^n$, with $n = h + k$.

In conclusion, the class of Boolean conditional control gates is categorically equivalent to the class of deterministic finite automata in which both the alphabet and the phase space have a power of 2 cardinality.

*From a generic gate to a (conservative) reversible conditional control gate, and back.* If a Boolean gate $G : \{0, 1\}^n \to \{0, 1\}^m$ is not reversible, then it is always possible to construct a corresponding reversible gate $G^r : \{0, 1\}^{m+n} \to \{0, 1\}^{m+n}$ associating with the input pair $(\vec{a}, \vec{s}) \in \{0, 1\}^n \times \{0, 1\}^m$ the output pair $(\vec{a}, \vec{s} \oplus G(\vec{a})) \in \{0, 1\}^n \times \{0, 1\}^m$. Moreover, this is a controlled gate, that is a finite automaton with respect to the alphabet $\mathcal{A} = \{0, 1\}^n$, the phase space $Q = \{0, 1\}^m$, and the set of next state functions $\delta_{\vec{a}}$ (for $\vec{a} \in \{0, 1\}^n$) associating with any state $\vec{s} \in \{0, 1\}^m$ the next state $\vec{s}\,' = \delta_{\vec{a}}(\vec{s}) := \vec{s} \oplus G(\vec{a}) \in \{0, 1\}^m$ (see figure 7).

Generally, $G^r$ is a non-conservative gate. If this is the case, we can extend it to a conservative gate $G^{rc}$ by adding some new input and output lines, maintaining the original reversibility. Let $\{\vec{x}\}_1 = \sum_{i=1}^{n+m} x_i$ be the number of ones contained in the input $\vec{x}$; analogously, let $\{G^r(\vec{x})\}_1 = \sum_{i=1}^{n+m} G_i^r(\vec{x})$ be the number of ones contained in the corresponding output $G^r(\vec{x})$. We denote by $E(\vec{x})$ the quantity $\{G^r(\vec{x})\}_1 - \{\vec{x}\}_1$. Clearly $E(\vec{x})$ is an integer number from the interval $[-(n+m), n+m]$. It is immediately seen that if $G^r$ were conservative then it would hold $E(\vec{x}) = 0$ for every $\vec{x} \in \{0,1\}^{n+m}$. On the other hand, since we have assumed that $G^r$ is a non-conservative gate, there exists an $\vec{x} \in \{0,1\}^{n+m}$ such that $E(\vec{x}) \neq 0$.

For the moment, let us suppose that $E(\vec{x}) > 0$. Then there exists an $\vec{x}' \in \{0,1\}^{n+m}$ such that $E(\vec{x}') < 0$. In fact we can express the quantity $\sum_{\vec{x}\in\{0,1\}^{n+m}} E(\vec{x})$ as follows:

$$\begin{aligned}\sum_{\vec{x}\in\{0,1\}^{n+m}} E(\vec{x}) &= \sum_{\vec{x}\in\{0,1\}^{n+m}} (\{G^r(\vec{x})\}_1 - \{\vec{x}\}_1) \\ &= \sum_{\vec{x}\in\{0,1\}^{n+m}} \{G^r(\vec{x})\}_1 - \sum_{\vec{x}\in\{0,1\}^{n+m}} \{x\}_1. \end{aligned} \tag{3}$$

Since $G^r$ is reversible, it is a permutation over the set $\{0,1\}^{n+m}$. This means that the two sums in (3) are calculated over the same elements, and thus

$$\sum_{\vec{x}\in\{0,1\}^{n+m}} E(\vec{x}) = 0.$$

As a consequence, if $E(\vec{x}) > 0$ there must exist an $\vec{x}' \in \{0,1\}^{n+m}$ such that $E(\vec{x}') < 0$. In a completely analogous way we can show that if $E(\vec{x}) < 0$ then there exists an $\vec{x}' \in \{0,1\}^{n+m}$ such that $E(\vec{x}') > 0$.

For the considerations above, if we define $\ell = -\min_{\vec{x}} E(\vec{x})$ and $h = \max_{\vec{x}} E(\vec{x})$, and the gate $G^r$ is non-conservative, then $\ell$ and $h$ are positive integers. For any $\vec{x} \in \{0,1\}^{n+m}$ such that $E(\vec{x}) < 0$, let $E_\ell(\vec{x})$ be the string $\underbrace{1,\ldots,1}_{-E(\vec{x})},0,\ldots,0$ of length $\ell$ (if $\ell = 0$ then $E_\ell(\vec{x})$ is the empty string); analogously, whenever $E(\vec{x}) > 0$ we define $E_h^c(\vec{x})$ as the string $\underbrace{0,\ldots,0}_{E(\vec{x})},1,\ldots,1$ of length $h$ (and also in this case, if $h = 0$ then $E_h^c(\vec{x})$ is the empty string).

To extend $G^r$ to a reversible and conservative gate $G^{rc}$ we can use $\ell$ ancillae lines (that we briefly indicate with $\vec{y}$) to provide $-E(\vec{x})$ ones whenever $E(\vec{x}) < 0$, and $h$ ancillae lines (that we indicate with $\vec{z}$) to remove $E(\vec{x})$ ones whenever $E(\vec{x}) > 0$. More precisely, we define $G^{rc} : \{0,1\}^{n+m+\ell+h} \to \{0,1\}^{n+m+\ell+h}$ as follows:

$\forall\vec{x} \in \{0,1\}^{n+m}, \forall\vec{y} \in \{0,1\}^{\ell}, \forall\vec{z} \in \{0,1\}^{h}$

$$G^{rc}(\vec{x},\vec{y},\vec{z}) := \begin{cases} (G^r(\vec{x}), E_\ell(\vec{x}), \vec{1}_h) & \text{if } E(\vec{x}) < 0, \vec{y} = \vec{0} \text{ and } \vec{z} = \vec{1} & \text{(i)} \\ (\vec{k}, \vec{0}_\ell, \vec{1}_h) & \text{if } G^r(\vec{k}) = \vec{x}, E(\vec{k}) < 0, \\ & \quad \vec{y} = E_\ell(\vec{k}) \text{ and } \vec{z} = \vec{1} & \text{(ii)} \\ (G^r(\vec{x}), \vec{0}_\ell, \vec{1}_h) & \text{if } E(\vec{x}) = 0, \vec{y} = \vec{0} \text{ and } \vec{z} = \vec{1} & \text{(iii)} \\ \big(G^r(\vec{x}), \vec{0}_\ell, E_h^c(\vec{x})\big) & \text{if } E(\vec{x}) > 0, \vec{y} = \vec{0} \text{ and } \vec{z} = \vec{1} & \text{(iv)} \\ (\vec{k}, \vec{0}_\ell, \vec{1}_h) & \text{if } G^r(\vec{k}) = \vec{x}, E(\vec{k}) > 0, \\ & \quad \vec{y} = \vec{0} \text{ and } \vec{z} = E_h^c(\vec{k}) & \text{(v)} \\ (\vec{x}, \vec{y}, \vec{z}) & \text{otherwise.} & \text{(vi)} \end{cases}$$

A direct inspection of $G^{rc}$ shows that the map $G^r$ is obtained in the first $n+m$ output lines when the ancillae lines $\vec{y}$ and $\vec{z}$ are fixed, respectively, with the input values $\vec{0}$ and $\vec{1}$. Notice

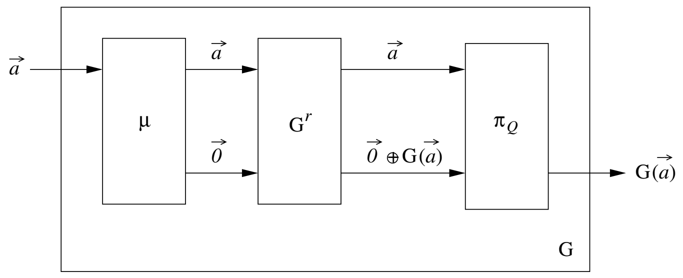

**Figure 8.** The original gate obtained from its (conservative and) reversible extension.

that rules (ii) and (v) are designed in order to provide the inverses of the tuples produced by rules (i) and (iv), respectively. On the other hand, the tuples produced by rule (iii) can be inverted by computing the inverse of the first $n+m$ components through the inverse of the map $G^r$. Finally, rule (vi) makes the gate behave as the identity when none of the previous rules are satisfied: as a consequence, the corresponding tuples can be trivially inverted. Summarizing, the inverse of $G^{rc}$ is obtained by substituting rule (iii) in the analytic expression of $G^{rc}$ with the following:

$$(\vec{k}, \vec{0}_\ell, \vec{1}_h) \qquad \text{if} \quad G^r(\vec{k}) = \vec{x},\, E(\vec{k}) = 0,\, \vec{y} = \vec{0} \text{ and } \vec{z} = \vec{1}.$$

Following Toffoli [Tof80], the original arbitrary Boolean gate $G$ can be recovered by means of the just constructed reversible and conservative gate $G^{rc}$ in the following way:

> In more general mathematical parlance, a *realization* of a function $G$ consists in a new function $G^{rc}$ together with two mappings $\mu$ and $\pi_Q$ (respectively, the *encoder* and the *decoder*) such that $G = \pi_Q \circ G^{rc} \circ \mu$. In this context, our plan is to obtain a realization $\pi_Q \circ G^{rc} \circ \mu$ of $G$ such that $G^{rc}$ is invertible [i.e., reversible] and conservative, and the mappings $\mu$ and $\pi_Q$ are essentially independent of $G$ and contain as little 'computing power' as possible.
> More precisely, though the form of $\mu$ and $\pi_Q$ must obviously reflect the number of input and output components of $G$, and thus the format of $G$'s truth table, we want them to be otherwise independent of the particular contents of such truth table as $G$ is made to range over the set of all combinatorial functions.

In the present case, the encoder is realized by the mapping $\mu : \{0,1\}^n \to \{0,1\}^{n+m+\ell+h}$ associating with the input $\vec{a} \in \{0,1\}^n$ the output 4-tuple $\mu(\vec{a}) := (\vec{a}, \vec{0}_m, \vec{0}_\ell, \vec{1}_h) \in \{0,1\}^n \times \{0,1\}^m \times \{0,1\}^\ell \times \{0,1\}^h$ (independent of the particular form of $G$). The decoder is realized by the projection mapping $\pi_Q : \mathcal{A} \times Q \times \{0,1\}^\ell \times \{0,1\}^h \to Q$. Trivially, for any $\vec{a} \in \{0,1\}^n$ one gets $(\pi_Q \circ G^{rc} \circ \mu)(\vec{a}) = (\pi_Q \circ G^{rc})(\vec{a}, \vec{0}_m, \vec{0}_\ell, \vec{1}_h) = \pi_Q(\vec{a}, G(\vec{a}), \vec{y}, \vec{z}) = G(\vec{a})$. This process is depicted in figure 8.

*The FAN-OUT gate as a cloning procedure induced by the Controlled-NOT gate.* A very important connective in reversible computing is FAN-OUT: $L \to L^2$, defined by the law FAN-OUT$(x) = (x, x)$. In other words, the FAN-OUT function simply clones the input value. When dealing with classical circuits, the FAN-OUT function is implemented by sticking two output wires to an existing input wire. The Controlled-NOT gate (see table 6) provides a possible realization of the FAN-OUT function by a two-input/two-output reversible gate. Indeed, if the operating line is fixed with the input value $x_2 = 0$, then the control input is cloned realizing in this way a classical FAN-OUT (see figure 9).

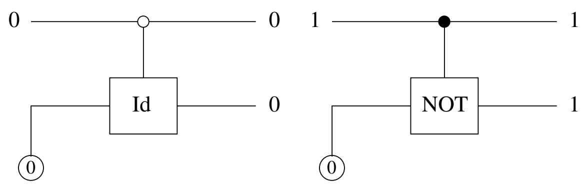

**Figure 9.** Realization of the FAN-OUT function with the Controlled-NOT gate.

**Table 7.** The Petri–Fredkin reversible and conservative gate.

| $x_1$ | $x_2$ | $x_3$ | $\longmapsto$ | $y_1$ | $y_2$ | $y_3$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 0 | 0 | 0 |
| 0 | 0 | 1 | | 0 | 1 | 0 |
| 0 | 1 | 0 | | 0 | 0 | 1 |
| 0 | 1 | 1 | | 0 | 1 | 1 |
| 1 | 0 | 0 | | 1 | 0 | 0 |
| 1 | 0 | 1 | | 1 | 0 | 1 |
| 1 | 1 | 0 | | 1 | 1 | 0 |
| 1 | 1 | 1 | | 1 | 1 | 1 |

## 3. The conservative and reversible Petri–Fredkin gate

**(F1)** One of the paradigmatic conservative and reversible primitives is the *Petri–Fredkin gate*, a three-input/three-output gate that computes the following function $\mathrm{FG} : \{0,1\}^3 \to \{0,1\}^3$:

$$y_1 = x_1 \qquad y_2 = (x_1 \wedge x_2) \vee (\neg x_1 \wedge x_3) \qquad y_3 = (\neg x_1 \wedge x_2) \vee (x_1 \wedge x_3).$$

Its truth table is presented in table 7.

**(F2)** The Petri–Fredkin gate is self-reversible, i.e., $\mathrm{FG} \circ \mathrm{FG} = \mathrm{Id}$. Hence, the inverse of function FG is FG itself. This is a particularly desirable feature for the construction of the quantum version of a reversible circuit ([FT82], p 247), since the part of the circuit which ‘undoes’ the computation (in order to disentangle input/output lines and the so-called ancillae lines) is thus completely symmetrical to the part which computes the output value.

Note that self-reversibility implies the reversibility property. The converse is not generally true: if $f : L^m \to L^m$ is reversible then it is a permutation over $L^m$ and, as is well known, in general the composition of a permutation with itself does not give the identity as a result. In particular, it can immediately be seen that only those permutations which are expressible as the composition of disjoint cycles of length 2 (and fixed points) are self-reversible.

**(F3)** Looking at table 7, we can immediately see that the Petri–Fredkin gate is conservative. This property allows for the realization of the Petri–Fredkin gate in the framework of ‘billiard ball’ computing, and led to the following observation concerning the physical meaning of conservativeness:

> In conservative logic, all signal processing is ultimately reduced to *conditional routing* of signals. Roughly speaking, signals are treated as unalterable objects that can be moved around in the course of a computation but never created or destroyed. ([FT82], p 227)

The billiard ball model, also developed by Fredkin and Toffoli in [FT82], is an excellent example of a toy scientific model of no immediate practical application but of large scientific impact. Balls of radius 1 travel on a unit grid in two directions. The direction of their

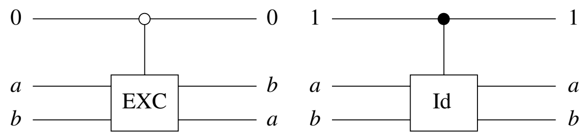

**Figure 10.** The Petri–Fredkin gate as a conditional switch.

movements can be changed either by an elastic collision, or by a reflection at a 'mirror'. Using this model it is possible to implement a *switch gate*; the Petri–Toffoli gate, whose behaviour is described in equation (2), can then be implemented with four of them.

**(F4)** If the first of the inputs is set to 0 then the Petri–Fredkin gate exchanges the second input with the third one, whereas if the first input is set to 1 it returns all the inputs unchanged, as shown in figure 10.

Therefore, the Petri–Fredkin gate is a conditional control gate with $x_1$ as control input, $\delta_0 = \text{EXC}$ and $\delta_1 = \text{Id}$.

**(F5)** From the Petri–Fredkin gate we can obtain some classical unitary and binary connectives by setting respectively two and one of the input lines to a constant value (that is, either 0 or 1). For example,

- by fixing $x_3 = 0$ in the input, the second output becomes $y_2 = (x_1 \wedge x_2) \vee (\neg x_1 \wedge 0) = x_1 \wedge x_2$, i.e., $y_2$ gives the logical AND between $x_1$ and $x_2$. In this case the outputs $y_1$ and $y_3$ are called *garbage*;
- by fixing $x_2 = 1$ and $x_3 = 0$ the input $x_1$ is negated in the output $y_3 = \neg x_1$, with garbage $y_1$ and $y_2$. In this way we realize the NOT connective.

**(F6)** Differently from the realization of the FAN-OUT gate by the reversible non-conservative Controlled-NOT gate, it is not difficult to see that it is impossible to realize the FAN-OUT function by a conservative two-input/two-output gate. Such a realization requires at least three input lines and three output lines, even when working with Boolean logic. The Petri–Fredkin gate supplies one of these possible realizations:

- if we fix $x_2 = 1$ and $x_3 = 0$ then the first input is cloned in the first and second outputs, i.e., we obtain the FAN-OUT function, with the output $y_3$ as garbage.

Summarizing, the Petri–Fredkin gate has the following properties:

(F1) it is a three-input/three-output gate, where each input/output assumes values in $\{0, 1\}$;
(F2) it is *reversible*, that is, a bijective mapping from $\{0, 1\}^3$ onto $\{0, 1\}^3$;
(F2′) it is *self-reversible*, that is, $\text{FG}^2 = \text{FG} \circ \text{FG} = \text{Id}$ (the identity function on $\{0, 1\}^3$);
(F3) it is *conservative*, in the sense that the number of 0's and 1's in the input is the same as in the output;
(F4) it is a *controlled gate*, that is, $x_1$ is a *control input* which is left unchanged but which determines a transformation of the *target input* $(x_2, x_3)$ into the output $(y_2, y_3)$ by the gate EXC if $x_1 = 0$ and by the identity gate if $x_1 = 1$;
(F5) it is a *universal primitive*, that is, from the configurations of the gate we can obtain the classical logical connectives AND, NOT (as well as logical disjunction and implication) which constitute a 'functionally complete' set of connectives for Boolean logic, that is a set of primitive truth functions with which all the possible truth functions (i.e., all functions $\{0, 1\}^n \to \{0, 1\}$ for $n$ ranging in $\mathbb{N}$) can be realized;
(F6) it realizes the FAN-OUT connective, which plays a central role in reversible computations since it clones a given input signal.

Our aim is to extend this computational Boolean framework based on the Petri–Fredkin gate to include the main features of many-valued logics, when a finite number of truth values are involved. In the next section we give a brief summary of the main aspects of this subject.

## 4. Many-valued logics

The simplest extension of classical two-valued logic consists in the introduction of a third 'intermediate', or 'neutral' or 'indeterminate' value. Łukasiewicz developed this idea in [Łu20]. In this paper he introduced a third truth value to take into account propositions which are neither true nor false, defining in this way a three-valued logic. This logic was then extended to deal with $d$ truth values as well as with an infinite number of truth values, in particular the $\aleph_0$ and $\aleph_1$ cardinalities.

Let us begin with a brief exposition of the main features of the many-valued logics of Łukasiewicz; the definition and the properties of the operators are the same for the finite- and the infinite-valued cases, unless otherwise stated.

Technically speaking, truth values of a logical system are defined just as syntactic labels, with no numerical meaning. In a subsequent step, it is possible to give an interpretation of the logical system in terms of an algebraic structure; only during such a process, the truth values are associated with elements of the structure, which can be more abstract mathematical objects than real or integer numbers. Indeed, all the notions here exposed can be restated in such a formal way; however, for our purposes it will be convenient to deal with the following sets of truth values, treated as numerical sets equipped with the standard total order relation induced by $\mathbb{R}$:

- $L_d = \left\{0, \frac{1}{d-1}, \frac{2}{d-1}, \ldots, \frac{d-2}{d-1}, 1\right\}$, with $d \geqslant 2$, for $d$-valued logics;
- $L_{\aleph_0} = [0, 1] \cap \mathbb{Q}$, that is the set of rational values in the interval [0, 1], for infinite-valued logics with $\aleph_0$ truth values;
- $L_{\aleph_1} = [0, 1]$, that is the set of real values in the interval [0, 1], for infinite-valued logics with $\aleph_1$ truth values.

The numbers of $L_\alpha, \alpha \in \{d, \aleph_0, \aleph_1\}$ are interpreted, after Łukasiewicz, as the possible truth values which logical sentences can be assigned to. As usually done in the literature, the values 1 and 0 denote respectively truth and falseness, whereas all the other values are used to indicate different degrees of indefiniteness. With the introduction of the new truth values, the propositional connectives of Boolean logic must be redefined. Accordingly, many-valued logics represent strong generalizations of bivalent (i.e., classical) logic.

### *4.1. Łukasiewicz approach*

The Łukasiewicz system on the totally ordered numerical set of truth values $L_\alpha$, with $\alpha \in \{d, \aleph_0, \aleph_1\}$, considers as primitive the *implication* ($\to_L$) connective, which is defined by the following equation:

$$\begin{aligned} x \to_L y &:= \min\{1, 1 - x + y\} \qquad \text{(Łukasiewicz implication)} \\ &= \begin{cases} 1 - x + y & \text{if} \quad y < x \\ 1 & \text{otherwise.} \end{cases} \end{aligned}$$

In the system $\langle L_\alpha, \to_L \rangle$ a *negation* ($\neg$) connective is derived according to the rule:

$$\begin{aligned} \neg x &: = x \to_L 0 \qquad \text{(diametrical negation)} \\ &= 1 - x. \end{aligned}$$

Using these two connectives, Łukasiewicz defines some other derived ones as

$$\begin{array}{ll} x \vee y := (x \rightarrow_L y) \rightarrow_L y & \text{(Łukasiewicz disjunction)} \\ x \wedge y := \neg(\neg x \vee \neg y) & \text{(Łukasiewicz conjunction)} \\ x \leftrightarrow_L y := (x \rightarrow_L y) \wedge (y \rightarrow_L x) & \text{(Łukasiewicz equivalence)} \end{array}$$

the former two being the algebraic realizations of the logical connectives OR and AND respectively.

From these definitions it is easy to see that the following equalities hold:

$$x \vee y = \max\{x, y\} \qquad \text{and} \qquad x \wedge y = \min\{x, y\}$$

where max and min are the lub and glb of the pair of numbers $x$, $y$ with respect to the standard total order of $L_\alpha$, which can also be expressed in the form

$$x \leqslant y \qquad \Longleftrightarrow \qquad x \rightarrow_L y = 1.$$

One important feature of all many-valued connectives now presented is that they are equal to the analogous Boolean connectives when only 0 or 1 are involved.

Following Chang [Ch58, Ch59], the Łukasiewicz approach to many-valued logics can be equivalently recovered on the basis of the pair of connectives $\{\oplus, \neg\}$, where $\oplus$ is the binary operation of *truncated sum*—introduced for the first time by Zawirski in [Za34]—defined as follows:

$$\begin{aligned} x \oplus y : &= \min\{1, x + y\} \qquad \text{(truncated sum)} \\ &= \begin{cases} x + y & \text{if} \quad x + y < 1 \\ 1 & \text{otherwise.} \end{cases} \end{aligned}$$

The Łukasiewicz system $\langle L_\alpha, \rightarrow_L\rangle$ and the Zawirski one $\langle L_\alpha, \oplus, \neg\rangle$ are mutually equivalent, owing to the 'translation' rules:

$$x \oplus y = \neg x \rightarrow_L y \qquad \text{and} \qquad x \rightarrow_L y = \neg x \oplus y. \tag{4}$$

Furthermore, the following binary operation can be defined in the Zawirski $\langle L_\alpha, \oplus, \neg\rangle$-system:

$$\begin{aligned} x \odot y : &= \neg(\neg x \oplus \neg y) = \max\{0, x + y - 1\} \\ &= \begin{cases} x + y - 1 & \text{if} \quad 1 < x + y \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

In some semantic interpretations, $\oplus$ and $\odot$ are considered as algebraic realizations of the logical connectives VEL and ET respectively, and they are also called the *MV-disjunction* and the *MV-conjunction*.

Let us stress that on the basis of either the Łukasiewicz system or the Zawirski one it is always possible to derive a structure $\langle L_\alpha, \wedge, \vee, \neg\rangle$ of distributive lattice with a non-standard negation. The lattice join and meet operations, algebraic realizations of the logical connectives OR and AND, can be defined in the two systems respectively as follows:

$$x \vee y = \max\{x, y\} = (x \odot \neg y) \oplus y = \neg(\neg x \oplus y) \oplus y \tag{5a}$$

$$x \wedge y = \min\{x, y\} = (x \oplus \neg y) \odot y = \neg[\neg(x \oplus \neg y) \oplus \neg y]. \tag{5b}$$

Note that the *excluded middle law* holds in the case of the VEL connective ($\forall x \in L_\alpha$: $x \oplus \neg x = 1$), whereas in general this law does not hold for the OR connective ($\forall x \in L_\alpha \backslash \{0, 1\}$: $x \vee \neg x \neq 1$). A similar result is verified with respect to the *non-contradiction law* ($\forall x \in L_\alpha$: $x \odot \neg x = 0$ and $\forall x \in L_\alpha \backslash \{0, 1\}$: $x \wedge \neg x \neq 0$). However, the desirable law $x \vee x \rightarrow_L x = 1$ holds relatively to the OR connective, but for every $x \neq 0, 1$ one has that $x \oplus x \rightarrow_L x \neq 1$. In the Zawirski context, the standard ordering on $L_\alpha$ now assumes the form

$$x \leqslant y \qquad \Longleftrightarrow \qquad \neg x \oplus y = 1.$$

Two modal connectives, *possibility* ($\Diamond$) and *necessity* ($\Box$), can be introduced on $L_\alpha$ according to the following definitions:

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

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

Note that the restriction of both connectives to the Boolean values coincides with the identity function (these modalities are meaningless in the Boolean environment).

Besides the diametrical negation ($\neg$) two other negation connectives can be defined as many-valued extensions of the standard Boolean negation: the *intuitionistic negation* (also *impossibility* $\sim$) and the *anti-intuitionistic negation* (also *contingency* $\flat$) defined as

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

and

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

In agreement with the intuitionistic propositional logic of Brouwer and Heyting, the intuitionistic negation (impossibility) fails the excluded middle law ($\forall x \in L_\alpha \backslash \{0, 1\}$: $x \vee \sim x = x \oplus \sim x = x \neq 1$), but does not fail the non-contradiction law ($\forall x \in L_\alpha$: $x \wedge \sim x = x \odot \sim x = 0$). Note that the restriction of the three negations to the Boolean values collapses in a unique (standard) negation ($\forall x \in \{0, 1\} : \neg x = \sim x = \flat x = 1 - x$).

The intuitionistic negation is a primitive one, together with the diametrical negation, in *BZ-lattice structures*, of which the system $\langle L_\alpha, \wedge, \vee, \neg, \sim \rangle$ is a standard model. For further information on BZ structures see [CN89].

In conclusion, in the algebraic approach to many-valued logics we have considered as primitives two mutually interdefinable (according to (4)) systems, the Łukasiewicz one $\langle L_\alpha, \rightarrow_L \rangle$ and the Zawirski one $\langle L_\alpha, \oplus, \neg \rangle$. A new system of distributive lattice with diametrical negation $\langle L_\alpha, \wedge, \vee, \neg \rangle$ can always be induced. Moreover, the set of unitary connectives $\{\Box, \Diamond, \sim, \flat\}$ (two modalities and two negations) are mutually interdefinable making use of the diametrical negation ($\neg$) according to the following diagram:

$$\begin{tikzcd} \Diamond \arrow[r, leftrightarrow, "\neg(\cdot)\neg"] \arrow[d, leftrightarrow, "\neg(\cdot)"'] & \Box \arrow[d, leftrightarrow, "\neg(\cdot)"] \\ \sim \arrow[r, leftrightarrow, "\neg(\cdot)\neg"'] & \flat \end{tikzcd}$$

Let us briefly see what happens in the finite-valued case. In the three-valued logic $L_3$ one has

$$\begin{aligned} \Diamond x &= \neg x \rightarrow_L x = x \oplus x \\ \Box x &= \neg(x \rightarrow_L \neg x) = \neg(\neg x \oplus \neg x) = x \odot x. \end{aligned}$$

Let us stress that in this case, $L_3$, the above definition of 'it is possible that $x$' coincides with 'if not $x$ then $x$' ($\Diamond x = \neg x \rightarrow_L x$); in [Łu30] Łukasiewicz mentioned that Tarski, a student of his, in 1921 proposed this as the definition of possibility. Therefore in this particular case we can derive the modal connectives from the system $\langle L_3, \rightarrow_L \rangle$ (equivalently, $\langle L_3, \oplus, \neg \rangle$)

which is thus sufficient to generate all connectives introduced above. In particular, the two negations impossibility and contingency have the form $\sim x = \neg(\neg x \rightarrow_L x) = \neg(x \oplus x)$ and $\flat x = x \rightarrow_L \neg x = \neg x \oplus \neg x$, respectively.

The link between possibility and truncated sum connectives is extended to the more general finite $d$-valued case by the following identity, which is true for every $x \in L_d$:

$$\Diamond x := \underbrace{x \oplus x \oplus \cdots \oplus x}_{(d-1)\text{-times}}.$$

Thus owing to this result and the relation

$$\Box x = \neg\Diamond\neg x = \underbrace{x \odot x \odot \cdots \odot x}_{(d-1)\text{-times}}$$

also in any finite-valued case the modal connectives of possibility and necessity can both be derived inside the system $\langle L_d, \oplus, \neg\rangle$.

We observe that, for infinite-valued logics, it is not possible to derive from $\rightarrow_L$ and $\neg$ the modal operators $\Box$ and $\Diamond$, and the intuitionistic and anti-intuitionistic negations $\sim$ and $\flat$ as we have just done for the finite-valued case. In fact, in [Mc51] the following theorem has been proved:

**Theorem 4.1.** *Let $L \in \{L_{\aleph_0}, L_{\aleph_1}\}$. A function $f : L^m \rightarrow L$ is expressible as a formula containing only the operators $\rightarrow_L$ and $\neg$ if and only if it is continuous.*

### *4.2. Gödel approach*

The extension of classical connectives to many-valued logics is not unique. For example, different types of implications have been defined in literature; one of these, which is often used, is the implication $\rightarrow_G$ defined by Gödel:

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

Note that if the use of the constant value 0 is allowed then we can obtain the intuitionistic negation as $\sim x = x \rightarrow_G 0$. Moreover, in the three-valued case Gödel's implication differs from $\rightarrow_L$ only for the input pair $\left(\frac{1}{2}, 0\right)$: in fact, $\frac{1}{2} \rightarrow_L 0 = \frac{1}{2}$ whereas $\frac{1}{2} \rightarrow_G 0 = 0$.

## 5. Functional completeness of finite-valued calculus

We face now the problem of determining whether any conceivable function $f : L_d^n \rightarrow L_d$, for $n$ ranging in $\mathbb{N}$, is expressible using only the operators $\neg$ and $\rightarrow_L$, i.e., the *functional completeness problem* on $L_d$ of the pair of connectives $\{\neg, \rightarrow_L\}$.

The following result, originally due to Jerzy Słupecki (see, for example, [RT52]), gives a negative answer.

**Theorem 5.1.** *The d-valued (with $d \geqslant 3$) propositional calculus of Łukasiewicz based on operators $\neg$ and $\rightarrow_L$ is not functionally complete. That is, there exist functions $f : L_d^n \rightarrow L_d$ which are not expressible as a composition of the logical functions $\neg$ and $\rightarrow_L$ (from which we stress that it is possible to derive the logical functions $\vee, \wedge, \oplus, \odot, \Diamond, \Box, \flat, \sim, \leftrightarrow_L$).*

**Proof.** The result follows directly from the fact that every function built-up using only $\neg$ and $\rightarrow_L$ gives a result in $\{0, 1\}$ when its arguments are assigned with values in this set. As a consequence we cannot represent, for example, the constant function which is identically equal to $\frac{1}{d-1}$. $\Box$

To make the $d$-valued sentential calculus functionally complete Słupecki introduced a new unary connective, called *tertium*, which is defined by the constant function $\mathcal{T} : L_d \to L_d$:

$$\forall x \in L_d \qquad \mathcal{T}(x) := \frac{1}{d-1}.$$

In fact, the following theorem holds (the interested reader can find the proof of this theorem in [RT52]).

**Theorem 5.2.** *The d-valued (with $d \geqslant 2$) propositional calculus of Łukasiewicz is functionally complete with respect to the set of primitive truth functions* $\{\neg, \to_L, \mathcal{T}\}$.

On the other hand, also $\langle L_d, \to_L \rangle$ is functionally complete (recall that $\neg x = x \to_L 0$) according to the following definition:

**Definition 5.1.** *A collection of primitive truth functions and a set of constants from $L_d$ are* universal *or (according to [RT52])* functionally complete *if all possible truth functions $L_d^n \to L_d$, with $n \in \mathbb{N}$, are expressible using these primitive functions and assigned constants.*

Hence it is functionally equivalent to assume the tertium function or the inclusion of constants in the original set of primitives $\{\neg, \to_L\}$.

## 6. Finite-valued conservative logics

In this section we extend conservative logic to include the main features of $d$-valued logics, with particular attention to three-valued logics. Since conservative logic is based on the Petri–Fredkin gate, we will extend it in order to deal with $d$ possible truth values on its input and output lines.

First of all we restrict our attention to gates having the same number of input and output lines. For brevity, we call the $(n, d)$-gate an $n$-input/$n$-output gate whose input and output lines may assume values from $L_d$. Thus, an $(n, d)$-gate computes a function $f : L_d^n \to L_d^n$, where $L_d^n = \underbrace{L_d \times \cdots \times L_d}_{n \text{ times}}$.

*Reversibility.* The extension of the *reversibility* property is simple: an $(n, d)$-gate is reversible if and only if the function computed by the gate is one-to-one (or, in other words, a permutation of the set $L_d^n$). A similar argument holds for *self-reversibility*: an $(n, d)$-gate is self-reversible if and only if the corresponding function applied twice is the identity function.

*Conservativeness.* The case of *conservativeness* is more complicated. A gate is *strictly conservative* if and only if each output vector is obtained by a permutation of the components of the input vector. This definition reflects perfectly the observation made by Fredkin and Toffoli in [FT82], cited above. As in the Boolean case, strict conservativeness and reversibility are two independent notions.

Notice that the permutation of the input values is not fixed, but varies depending on the pattern of values presented to the input lines; an example can be seen in figure 10, where two possible permutations are chosen according to the value fed to the first input of the Petri–Fredkin gate.

Clearly the two-valued Petri–Fredkin gate is strictly conservative, and in our first attempt to make an extension of this gate to the finite-valued case we tried to preserve this property. Unfortunately, if the number $n$ of input/output lines of a strictly conservative gate for a $d$-valued logic is not greater than $d$, then it is impossible to realize in its configurations the FAN-OUT function, as stated in the following proposition:

**Proposition 6.1.** *If $n$ and $d$ are two integer numbers such that $0 < n \leqslant d$ then there is no function $f : L_d^n \to L_d^n$ which corresponds to a strictly conservative gate realizing in its configurations the FAN-OUT function.*

**Proof.** If $n = 1$ then the gate has one output, and thus it cannot realize the FAN-OUT function. So, assume that $1 < n \leqslant d$, and that there exists a strictly conservative gate realizing FAN-OUT and corresponding to a function $f : L_d^n \to L_d^n$. In the gate configuration realizing the FAN-OUT function, one input line is fed with a variable value and $n - 1$ input lines are fed with constant values. Since $n - 1 \leqslant d - 1$, there exists at least one truth value $\ell \in L_d$ which does not appear in the fixed constant input values. When the variable value of the input is set to $\ell$, both the following properties should hold:

- the output vector should be a permutation of the input vector (since the gate is strictly conservative), and
- $\ell$ should appear twice in the output values (as the gate realizes the FAN-OUT function),

which is clearly impossible. □

If the condition $n \leqslant d$ in proposition 6.1 is relaxed, then it is not difficult to see that FAN-OUT can be realized through gates which are both reversible and strictly conservative: see, for example, the Petri–Fredkin gate, where $n = 3$ and $d = 2$.

*Weak conservativeness.* An alternative approach is to weaken the conservativeness property in order to obtain some reasonable gate that computes the FAN-OUT function. Thus we say that a gate is *weakly conservative* if and only if the sum of output values is always equal to the sum of input values. Clearly if a gate is strictly conservative then it is also weakly conservative, while the converse is not generally true. Moreover, it can immediately be seen that strict and weak conservativeness coincide in the Boolean case.

For example, if the input of a gate is $(\lambda, 0, 1)$ and the corresponding output is $(0, 1, \lambda)$ then the gate is both strictly conservative and weakly conservative for this input/output pair, regardless of the numerical value associated with $\lambda \in L_d$. On the other hand, if the corresponding output is $(\lambda, \lambda, \lambda)$ then the gate is weakly conservative if and only if we associate with $\lambda$ the numerical value $\frac{1}{2}$, while it is not strictly conservative, whatever the numerical value associated with $\lambda$. Indeed it is easy to see that, for a given input vector, the set of admissible output vectors prescribed by the weak conservativeness property varies depending upon the numerical values associated with the truth values.

Assuming $L_d$ as the set of truth values, we propose a possible physical interpretation of the weak conservativeness property. To produce a given input vector for a gate we need some amount of energy. A 'conservative' gate has to build the output vector in such a way that this energy is preserved; in other words, the output produced must have the property that, if built from scratch, it requires the same amount of energy as was required to build the input. The simplest way to satisfy this property is to produce a permutation of the input values, as strictly conservative gates do.

Now, let us suppose encoding the $d$ truth values on a physical system which has energy levels that are equally spaced and ordered according to the numerical value associated with the truth values. Thus, to switch from a given truth value, say $\frac{k}{d-1}$, to the next, that is $\frac{k+1}{d-1}$, we need to provide a fixed amount $\Delta E$ of energy. Analogously, when passing from a given truth value to the previous, the same amount $\Delta E$ of energy is released.

For a gate to be conservative, it must build the output pattern without requiring energy from an external source nor dissipating energy towards the environment; this means that it can switch a line from a truth value $\frac{k_1}{d-1}$ to a higher value $\frac{k_2}{d-1}$ if and only if the energy needed (which is equal to $(k_2 - k_1) \cdot \Delta E$) becomes available by lowering by the same amount the truth value stored in some other line. This is clearly equivalent to requiring that the sum of the values on the output lines be equal to the sum of the values on the input lines.

*0- and 1-regularity.* We now define two other properties of the Petri–Fredkin gate. They are not fundamental properties but characterize, for $d$-valued logics, three-input/three-output gates that have a behaviour which is similar to the two-valued Petri–Fredkin gate. We recall that the Petri–Fredkin gate exchanges the second input with the third one when the first input is set to 0, and it gives as outputs the unchanged inputs when the first input is set to 1. According to this point of view, let $G : L_d^3 \to L_d^3$ be the function 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 $L_d$. Analogously, 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 $L_d$.

*Functional completeness.* The last fundamental property satisfied by the Petri–Fredkin gate is *universality* (in the sense of *functional completeness*). Indeed, according to the definition given above, with the $d$-valued extensions of the Petri–Fredkin gate we will introduce in the next sections it is possible to realize some universal sets of connectives for $d$-valued logics.

*Conclusions.* In the next sections we look for universal gates for $d$-valued logics which preserve as many of the following properties as possible:

(F1) it is a $(3, d)$-gate, that is, a three-input/three-output gate where each input and each output line may assume one of the values in $L_d = \left\{0, \frac{1}{d-1}, \frac{2}{d-1}, \ldots, \frac{d-2}{d-1}, 1\right\}$;
(F2) it is reversible;
(F2′) it is self-reversible;
(F3) it is weakly conservative;
(F3′) it is strictly conservative;
(F4) it is a universal gate, that is, from the configurations of the gate a universal (functionally complete) set of connectives is obtained, including FAN-OUT;
(F5) it is 0-regular;
(F6) it is 1-regular;
(F7) $y_1 = x_1$, that is, the first output is always equal to the first input (conditional control condition);
(F8) when fed with Boolean input triples, it behaves as the classical Petri–Fredkin gate.

Properties (F5)–(F8) are not essential from the point of view of conservative logic, but nonetheless are desirable in a $d$-valued extension of the Petri–Fredkin gate.

## 7. Three-valued universal gates

In order to devise a universal gate for a three-valued logic, the first idea that comes to mind is to take the equations which define the input/output behaviour of the Petri–Fredkin gate and to interpret $\neg$, $\vee$ and $\wedge$, respectively, as the Łukasiewicz negation, disjunction and conjunction. However this approach does not work, as the mapping from $L_3^3$ to $L_3^3$ thus obtained is not even a bijection. As a consequence, we have to look for gates which are universal and preserve as many properties from (F1) to (F8) as possible.

The following table presents all the binary three-valued connectives that we are interested in realizing with our three-valued universal gates: the Łukasiewicz implication $\to_L$, the Gödel implication $\to_G$, the Łukasiewicz disjunction $\vee$, the Łukasiewicz conjunction $\wedge$, the MV-disjunction $\oplus$ and the MV-conjunction $\odot$:

| $x$ | $y$ | $\to_L$ | $\to_G$ | $\wedge$ | $\vee$ | $\oplus$ | $\odot$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | $\frac{1}{2}$ | 1 | 1 | 0 | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| $\frac{1}{2}$ | 0 | $\frac{1}{2}$ | 0 | 0 | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 |
| $\frac{1}{2}$ | $\frac{1}{2}$ | 1 | 1 | $\frac{1}{2}$ | $\frac{1}{2}$ | 1 | 0 |
| $\frac{1}{2}$ | 1 | 1 | 1 | $\frac{1}{2}$ | 1 | 1 | $\frac{1}{2}$ |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{2}$ | 1 | 1 | $\frac{1}{2}$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

The unitary connectives here considered are, besides the trivial identity connective Id, the negation connectives $\neg$, $\sim$, $\flat$ and the modal connectives $\Diamond$ and $\Box$ depicted in the following table:

| $x$ | $\neg$ | $\sim$ | $\flat$ | $\Diamond$ | $\Box$ |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 0 |
| $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |

It is important to stress that besides unitary and binary connectives we must consider the FAN-OUT function which plays a fundamental role for reversible computations. Due to proposition 6.1, the presence of this function forbids the strict conservativeness of a universal (3, 3)-gate.

The first three-valued gate that we introduce ($F_1$) allows one to obtain from its configurations all the main connectives of the Łukasiewicz logic $\langle L_3, \to_L \rangle$, as well as the Gödel implication. The truth table of the gate is given in table 8; as can be seen, the gate is self-reversible (and thus reversible), 0-regular and 1-regular. Moreover, it satisfies properties (F7) and (F8).

Table 9 shows all the relevant connectives which can be obtained from the gate by fixing one or two of its input lines with constant values from $L_3$; $\mathrm{Pr}_1$ and $\mathrm{Pr}_2$ are the projector

**Table 8.** Truth table of gate $F_1$.

| $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ | $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ | $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ |
|---|---|---|---|---|---|---|---|---|
| 0 0 0 | | 0 0 0 | $\frac{1}{2}$ 0 0 | | $\frac{1}{2}$ 0 0 | 1 0 0 | | 1 0 0 |
| 0 0 $\frac{1}{2}$ | | 0 $\frac{1}{2}$ 0 | $\frac{1}{2}$ 0 $\frac{1}{2}$ | | $\frac{1}{2}$ 0 $\frac{1}{2}$ | 1 0 $\frac{1}{2}$ | | 1 0 $\frac{1}{2}$ |
| 0 0 1 | | 0 1 0 | $\frac{1}{2}$ 0 1 | | $\frac{1}{2}$ 0 1 | 1 0 1 | | 1 0 1 |
| 0 $\frac{1}{2}$ 0 | | 0 0 $\frac{1}{2}$ | $\frac{1}{2}$ $\frac{1}{2}$ 0 | | $\frac{1}{2}$ $\frac{1}{2}$ 0 | 1 $\frac{1}{2}$ 0 | | 1 $\frac{1}{2}$ 0 |
| 0 $\frac{1}{2}$ $\frac{1}{2}$ | | 0 $\frac{1}{2}$ $\frac{1}{2}$ | $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ | ** | $\frac{1}{2}$ 1 0 | 1 $\frac{1}{2}$ $\frac{1}{2}$ | | 1 $\frac{1}{2}$ $\frac{1}{2}$ |
| 0 $\frac{1}{2}$ 1 | | 0 1 $\frac{1}{2}$ | $\frac{1}{2}$ $\frac{1}{2}$ 1 | | $\frac{1}{2}$ 1 $\frac{1}{2}$ | 1 $\frac{1}{2}$ 1 | | 1 $\frac{1}{2}$ 1 |
| 0 1 0 | | 0 0 1 | $\frac{1}{2}$ 1 0 | ** | $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ | 1 1 0 | | 1 1 0 |
| 1 1 $\frac{1}{2}$ | | 0 $\frac{1}{2}$ 1 | $\frac{1}{2}$ 1 $\frac{1}{2}$ | | $\frac{1}{2}$ $\frac{1}{2}$ 1 | 1 1 $\frac{1}{2}$ | | 1 1 $\frac{1}{2}$ |
| 0 1 1 | | 0 1 1 | $\frac{1}{2}$ 1 1 | | $\frac{1}{2}$ 1 1 | 1 1 1 | | 1 1 1 |

**Table 9.** The operators obtained through gate $F_1$.

| Connective | Inputs | Constants | Outputs | Garbage |
|---|---|---|---|---|
| FAN-OUT | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_1, y_2$ | $y_3$ |
| $\mathrm{Pr}_1$ | $x_2, x_3$ | $x_1 = 0$ | $y_3$ | $y_1, y_2$ |
| $\mathrm{Pr}_2$ | $x_2, x_3$ | $x_1 = 0$ | $y_2$ | $y_1, y_3$ |
| $\rightarrow_L$ | $x_1, x_3$ | $x_2 = 1$ | $y_3$ | $y_1, y_2$ |
| $\rightarrow_G$ | $x_1, x_2$ | $x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\vee$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ |
| $\wedge$ | $x_1, x_2$ | $x_3 = 0$ | $y_2$ | $y_1, y_3$ |
| Id | $x_1$ | $x_2 = 0, x_3 = 0$ | $y_1$ | $y_2, y_3$ |
| $\neg$ | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_3$ | $y_1, y_2$ |
| $\sim$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\diamond$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_3$ | $y_1, y_2$ |

connectives defined as $\mathrm{Pr}_1(x_1, x_2) = x_1$ and $\mathrm{Pr}_2(x_1, x_2) = x_2$, respectively. We observe that this gate realizes two negations (the diametrical and the intuitionistic one) and both the Łukasiewicz and Gödel implications introduced in section 4; as a consequence, the universality property (F4) is satisfied for both kinds of three-valued logic. On the other hand, the necessity modal connective, the anti-intuitionistic negation and both the binary MV-connectives are not realized.

Due to proposition 6.1, gate $F_1$ cannot be strictly conservative, as it realizes the FAN-OUT function. More precisely, strict conservativeness is lost in the two table rows marked with $(**)$. However, for these rows the gate is weakly conservative, and therefore the entire gate is weakly conservative.

The next two gates we introduce are part of the results of an exhaustive search—performed with a program written on purpose—over all three-valued gates having the following properties:

(F1) it is a (3, 3)-gate;
(F2′) it is self-reversible;
(F3) it is weakly conservative;
(F8) when fed with Boolean input triples it behaves as the Petri–Fredkin gate.

The first of the two gates we have obtained ($F_2$) is substantially equivalent to $F_1$; its truth table is given in table 10. As we can see, this gate differs from $F_1$ only for the input triples $\left(0, \frac{1}{2}, \frac{1}{2}\right)$ and $\left(\frac{1}{2}, 0, \frac{1}{2}\right)$. It is only 1-regular and it does not satisfy property (F7).

Table 11 shows all the relevant connectives which can be obtained from the gate by fixing one or two of its input lines with constant values from $L_3$. Let us observe that the set of connectives is the same as $F_1$'s with the exception of the modal connective $\square$, which is not present in the first gate. Thus, the deficiencies with respect to gate $F_1$ concerning the properties enjoyed by the gate are balanced with a richer set of realized connectives. As does happen with gate $F_1$, the set of connectives realized by gate $F_2$ satisfies condition (F4) of universality.

The last gate ($F_3$) we introduce allows one to realize the MV-connectives of the three-valued case; its truth table is given in table 12. Besides properties (F1), (F2′), (F3) and (F8), used by our program as criteria for the exhaustive search, this gate satisfies property (F7) of conditional control; moreover, it is 0-regular and 1-regular.

Table 13 shows all the relevant connectives which can be obtained from the gate. Given the correspondences between the operators of the three-valued Zawirski system $\langle L_3, \oplus, \neg \rangle$

**Table 10.** Truth table of gate $F_2$.

| $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ | $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ | $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ |
|---|---|---|---|---|---|---|---|---|
| $0\ 0\ 0$ | | $0\ 0\ 0$ | $\frac{1}{2}\ 0\ 0$ | | $\frac{1}{2}\ 0\ 0$ | $1\ 0\ 0$ | | $1\ 1\ 0$ |
| $0\ 0\ \frac{1}{2}$ | | $0\ \frac{1}{2}\ 0$ | $\frac{1}{2}\ 0\ \frac{1}{2}$ | | $0\ \frac{1}{2}\ \frac{1}{2}$ | $1\ 0\ \frac{1}{2}$ | | $1\ 0\ \frac{1}{2}$ |
| $0\ 0\ 1$ | | $0\ 1\ 0$ | $\frac{1}{2}\ 0\ 1$ | | $\frac{1}{2}\ 0\ 1$ | $1\ 0\ 1$ | | $1\ 0\ 1$ |
| $0\ \frac{1}{2}\ 0$ | | $0\ 0\ \frac{1}{2}$ | $\frac{1}{2}\ \frac{1}{2}\ 0$ | | $\frac{1}{2}\ \frac{1}{2}\ 0$ | $1\ \frac{1}{2}\ 0$ | | $1\ \frac{1}{2}\ 0$ |
| $0\ \frac{1}{2}\ \frac{1}{2}$ | | $\frac{1}{2}\ 0\ \frac{1}{2}$ | $\frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}$ | | $\frac{1}{2}\ 1\ 0$ | $1\ \frac{1}{2}\ \frac{1}{2}$ | | $1\ \frac{1}{2}\ \frac{1}{2}$ |
| $0\ \frac{1}{2}\ 1$ | | $0\ 1\ \frac{1}{2}$ | $\frac{1}{2}\ \frac{1}{2}\ 1$ | | $\frac{1}{2}\ 1\ \frac{1}{2}$ | $1\ \frac{1}{2}\ 1$ | | $1\ \frac{1}{2}\ 1$ |
| $0\ 1\ 0$ | | $0\ 0\ 1$ | $\frac{1}{2}\ 1\ 0$ | | $\frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}$ | $1\ 1\ 0$ | | $1\ 1\ 0$ |
| $0\ 1\ \frac{1}{2}$ | | $0\ \frac{1}{2}\ 1$ | $\frac{1}{2}\ 1\ \frac{1}{2}$ | | $\frac{1}{2}\ \frac{1}{2}\ 1$ | $1\ 1\ \frac{1}{2}$ | | $1\ 1\ \frac{1}{2}$ |
| $0\ 1\ 1$ | | $0\ 1\ 1$ | $\frac{1}{2}\ 1\ 1$ | | $\frac{1}{2}\ 1\ 1$ | $1\ 1\ 1$ | | $1\ 1\ 1$ |

**Table 11.** The operators obtained through gate $F_2$.

| Connective | Inputs | Constants | Outputs | Garbage |
|---|---|---|---|---|
| FAN-OUT | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_1, y_2$ | $y_3$ |
| $\mathrm{Pr}_1$ | $x_2, x_3$ | $x_1 = 1$ | $y_2$ | $y_1, y_3$ |
| $\mathrm{Pr}_2$ | $x_2, x_3$ | $x_1 = 1$ | $y_3$ | $y_1, y_2$ |
| $\to_L$ | $x_1, x_3$ | $x_2 = 1$ | $y_3$ | $y_1, y_2$ |
| $\to_G$ | $x_1, x_2$ | $x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\vee$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ |
| $\wedge$ | $x_1, x_2$ | $x_3 = 0$ | $y_2$ | $y_1, y_3$ |
| Id | $x_1$ | $x_2 = 0, x_3 = 0$ | $y_1$ | $y_2, y_3$ |
| $\neg$ | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_3$ | $y_1, y_2$ |
| $\sim$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\Diamond$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_3$ | $y_1, y_2$ |
| $\Box$ | $x_1$ | $x_2 = 0, x_3 = \frac{1}{2}$ | $y_1$ | $y_2, y_3$ |

**Table 12.** Truth table of gate $F_3$.

| $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ | $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ | $x_1x_2x_3$ | $\mapsto$ | $y_1y_2y_3$ |
|---|---|---|---|---|---|---|---|---|
| $0\ 0\ 0$ | | $0\ 0\ 0$ | $\frac{1}{2}\ 0\ 0$ | | $\frac{1}{2}\ 0\ 0$ | $1\ 0\ 0$ | | $1\ 0\ 0$ |
| $0\ 0\ \frac{1}{2}$ | | $0\ \frac{1}{2}\ 0$ | $\frac{1}{2}\ 0\ \frac{1}{2}$ | | $\frac{1}{2}\ \frac{1}{2}\ 0$ | $1\ 0\ \frac{1}{2}$ | | $1\ 0\ \frac{1}{2}$ |
| $0\ 0\ 1$ | | $0\ 1\ 0$ | $\frac{1}{2}\ 0\ 1$ | | $\frac{1}{2}\ 0\ 1$ | $1\ 0\ 1$ | | $1\ 0\ 1$ |
| $0\ \frac{1}{2}\ 0$ | | $0\ 0\ \frac{1}{2}$ | $\frac{1}{2}\ \frac{1}{2}\ 0$ | | $\frac{1}{2}\ 0\ \frac{1}{2}$ | $1\ \frac{1}{2}\ 0$ | | $1\ \frac{1}{2}\ 0$ |
| $0\ \frac{1}{2}\ \frac{1}{2}$ | | $0\ \frac{1}{2}\ \frac{1}{2}$ | $\frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}$ | | $\frac{1}{2}\ 1\ 0$ | $1\ \frac{1}{2}\ \frac{1}{2}$ | | $1\ \frac{1}{2}\ \frac{1}{2}$ |
| $0\ \frac{1}{2}\ 1$ | | $0\ 1\ \frac{1}{2}$ | $\frac{1}{2}\ \frac{1}{2}\ 1$ | | $\frac{1}{2}\ \frac{1}{2}\ 1$ | $1\ \frac{1}{2}\ 1$ | | $1\ \frac{1}{2}\ 1$ |
| $0\ 1\ 0$ | | $0\ 0\ 1$ | $\frac{1}{2}\ 1\ 0$ | | $\frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}$ | $1\ 1\ 0$ | | $1\ 1\ 0$ |
| $0\ 1\ \frac{1}{2}$ | | $0\ \frac{1}{2}\ 1$ | $\frac{1}{2}\ 1\ \frac{1}{2}$ | | $\frac{1}{2}\ 1\ \frac{1}{2}$ | $1\ 1\ \frac{1}{2}$ | | $1\ 1\ \frac{1}{2}$ |
| $0\ 1\ 1$ | | $0\ 1\ 1$ | $\frac{1}{2}\ 1\ 1$ | | $\frac{1}{2}\ 1\ 1$ | $1\ 1\ 1$ | | $1\ 1\ 1$ |

and those of the Łukasiewicz one $\langle L_3, \to_L\rangle$ expressed by equation (4), we have that the set of connectives realized by gate $F_3$ satisfies condition (F4) of universality.

It is worth noting that, as a consequence of proposition 6.1, none of the gates presented in this section is strictly conservative.

We conclude this section with the following proposition.

**Table 13.** The operators obtained through gate $F_3$.

| Connective | Inputs | Constants | Outputs | Garbage |
|---|---|---|---|---|
| FAN-OUT | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_1, y_2$ | $y_3$ |
| $\mathrm{Pr}_1$ | $x_2, x_3$ | $x_1 = 0$ | $y_3$ | $y_1, y_2$ |
| $\mathrm{Pr}_2$ | $x_2, x_3$ | $x_1 = 0$ | $y_2$ | $y_1, y_3$ |
| $\oplus$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ |
| $\odot$ | $x_1, x_2$ | $x_3 = 0$ | $y_2$ | $y_1, y_3$ |
| Id | $x_1$ | $x_2 = 0, x_3 = 0$ | $y_1$ | $y_2, y_3$ |
| $\neg$ | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_3$ | $y_1, y_2$ |
| $\sim$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\diamond$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_3$ | $y_1, y_2$ |
| $\square$ | $x_3$ | $x_1 = \frac{1}{2}, x_2 = 0$ | $y_3$ | $y_1, y_2$ |

**Table 14.** Configurations of a $(3, d)$-gate that realize the $\oplus$ connective. Here $\lambda$ is an element of $L_d\backslash\{0, 1\}$.

| Connective | Inputs | Constant | Output | Garbage | Proof |
|---|---|---|---|---|---|
| $\oplus$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ | (1) |
| $\oplus$ | $x_1, x_2$ | $x_3 = 1$ | $y_3$ | $y_1, y_2$ | (2) |
| $\oplus$ | $x_2, x_3$ | $x_1 = \lambda$ | $y_1$ | $y_2, y_3$ | (3) |
| $\oplus$ | $x_2, x_3$ | $x_1 = \lambda$ | $y_2$ | $y_1, y_3$ | (4) |
| $\oplus$ | $x_2, x_3$ | $x_1 = \lambda$ | $y_3$ | $y_1, y_2$ | (5) |
| $\oplus$ | $x_1, x_3$ | $x_2 = \lambda$ | $y_1$ | $y_2, y_3$ | (6) |
| $\oplus$ | $x_1, x_3$ | $x_2 = \lambda$ | $y_2$ | $y_1, y_3$ | (7) |
| $\oplus$ | $x_1, x_3$ | $x_2 = \lambda$ | $y_3$ | $y_1, y_2$ | (8) |
| $\oplus$ | $x_1, x_2$ | $x_3 = \lambda$ | $y_1$ | $y_2, y_3$ | (9) |
| $\oplus$ | $x_1, x_2$ | $x_3 = \lambda$ | $y_2$ | $y_1, y_3$ | (10) |
| $\oplus$ | $x_1, x_2$ | $x_3 = \lambda$ | $y_3$ | $y_1, y_2$ | (11) |

**Proposition 7.1.** *For $d \geqslant 3$, there is no (3, d)-gate satisfying properties (F2), (F3) and (F8) which is able to realize the Łukasiewicz connectives ($\wedge, \vee, \rightarrow_L$), the Gödel implication ($\rightarrow_G$) and the MV-connectives ($\oplus, \odot$).*

**Proof.** The only configurations that allow one to realize the classical implication with a Boolean Fredkin gate are $x_2 = 1$ and $x_3 = 1$. Thus, if we impose property (F8) on our $(3, d)$-gate we get the following two possibilities to implement $\rightarrow_L$ and $\rightarrow_G$:

| Connective | Inputs | Constant | Output | Garbage |
|---|---|---|---|---|
| $\rightarrow_L$ | $x_1, x_3$ | $x_2 = 1$ | $y_3$ | $y_1, y_2$ |
| $\rightarrow_G$ | $x_1, x_2$ | $x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\rightarrow_G$ | $x_1, x_3$ | $x_2 = 1$ | $y_3$ | $y_1, y_2$ |
| $\rightarrow_L$ | $x_1, x_2$ | $x_3 = 1$ | $y_2$ | $y_1, y_3$ |

However, in both cases there is no configuration that allows one to realize $\oplus$. In table 14 we explore all possible configurations and, for each case, we give a short proof of the incompatibility. In what follows, $\gamma$ is an unspecified element of $L_d$, whereas $\lambda$ is a fixed element of $L_d\backslash\{0, 1\}$.

(1) For $\lambda \leqslant \frac{1}{2}$, all the triples $\lambda 1(1-\lambda)$ are mapped to 011, thus violating (F2).
(3) Triples $\lambda 11$ are mapped to 111, thus violating (F3).
(4) For triples $\lambda 01$, it should be $0 \oplus 1 = 1$ on $y_2$, and also $\lambda \rightarrow_L 0 = 1-\lambda$ or $\lambda \rightarrow_G 0 = 0$ on the same output.
(6) Triples $0\lambda 1$ are mapped to $11\gamma$, thus violating (F3).
(7) For triples $1\lambda 1$, it should be $1 \oplus 1 = 1$ on $y_2$, and also $1 \rightarrow_L \lambda = 1-\lambda$ or $1 \rightarrow_G \lambda = \lambda$ on the same output.
(9) Triples $01\lambda$ are mapped to $1\gamma 1$, thus violating (F3).
(10) For triples $11\lambda$, it should be $1 \oplus 1 = 1$ on $y_2$, and also $1 \rightarrow_L \lambda = 1-\lambda$ or $1 \rightarrow_G \lambda = \lambda$ on the same output.

The cases (2), (5), (8) and (11) can be obtained, respectively, from (1), (4), (7) and (10) by exchanging the second and third input/output lines of the gate. □

If a strictly conservative gate that realizes all the three-valued connectives mentioned above is needed then, due to propositions 6.1 and 7.1, it is necessary to look for $(n, 3)$-gates with $n \geqslant 4$. In [CLL02], a (4, 3)-gate which has all the required properties is presented.

## 8. Finite-valued universal gates

After the discovery of the generalizations of the Petri–Fredkin gate to three-valued logics exposed in the previous section, we obviously tried to generalize further to $d$-valued logics.

The approach followed in the previous section, that is making an exhaustive search in the space of truth tables of all $(3, d)$-gates, is clearly not feasible to find a solution which is valid for every value of $d$. As a consequence, we looked for some analytic expressions which define the new reversible and conservative gates independently of the cardinality of the set of truth values.

### *8.1. A gate for Łukasiewicz and Gödel d-valued logics*

The first function $\underline{f}_d^1 : L_d^3 \rightarrow L_d^3$ we introduce is defined as follows:

$\forall \underline{x} = (x_1, x_2, x_3) \in L_d^3$

$$\underline{f}_d^1(\underline{x}) := \begin{cases} (x_1, x_3, x_2) & \text{if } \; x_1 = 0 \text{ and } x_2 \neq x_3 & \text{(i)} \\ (x_1, x_3, x_2) & \text{if } \; 0 < x_1 \leqslant x_3 < 1 \text{ and } x_2 = 1 & \text{(ii)} \\ (x_1, x_3, x_2) & \text{if } \; 0 < x_1 \leqslant x_2 < 1 \text{ and } x_3 = 1 & \text{(iii)} \\ (x_1, x_1, 1 - x_1 + x_3) & \text{if } \; x_3 < x_1 < 1 \text{ and } x_2 = 1 & \text{(iv)} \\ (x_1, 1, x_3 + x_1 - 1) & \text{if } \; x_1 < 1, x_2 = x_1, x_3 + x_1 \geqslant 1 & \\ & \quad \text{and } x_3 < 1 & \text{(v)} \\ (x_1, x_1, x_2 - x_1) & \text{if } \; 0 < x_1 < x_2 < 1 \text{ and } x_3 = 0 & \text{(vi)} \\ (x_1, x_3 + x_1, 0) & \text{if } \; 0 < x_1, x_2 = x_1, x_3 + x_1 < 1 & \\ & \quad \text{and } x_3 > 0 & \text{(vii)} \\ (x_1, x_2, x_3) & \text{otherwise.} & \text{(viii)} \end{cases}$$

A direct inspection of the definition allows one to conclude that function $\underline{f}_d^1$ is well defined; that is, each triple $(x_1, x_2, x_3)$ of $L_d^3$ is associated by $\underline{f}_d^1$ with a single triple $(y_1, y_2, y_3)$ of $L_d^3$.

Let us see some properties of $\underline{f}_d^1$.

**Proposition 8.1.** *$\underline{f}_d^1$ is self-reversible.*

**Proof.** We have to prove that $\forall \underline{x} \in L_d^3$, $\underline{f}_d^1(\underline{f}_d^1(\underline{x})) = \underline{x}$. We can proceed by dividing the elements of the domain as in rules (i), (ii), ..., (viii).

Let $a$ and $b$ be two arbitrary elements of $L_d$. Considering the above rules it holds:

(i) $\underline{f}_d^1(\underline{f}_d^1(0, a, b)) = \underline{f}_d^1(0, b, a) = (0, a, b)$.

(ii) Let $\underline{x} = (a, 1, b)$ with $0 < a \leqslant b < 1$. Since $\underline{y} = \underline{f}_d^1(a, 1, b) = (a, b, 1)$ satisfies the conditions in (iii), we have $\underline{f}_d^1(a, b, 1) = (a, 1, b) = \underline{x}$.

(iii) Let $\underline{x} = (a, b, 1)$ with $0 < a \leqslant b < 1$. It holds $\underline{y} = \underline{f}_d^1(a, b, 1) = (a, 1, b)$, that satisfies the conditions in (ii), and thus $\underline{f}_d^1(a, 1, b) = (a, b, 1) = \underline{x}$.

(iv) Let $\underline{x} = (a, 1, b)$ with $b < a < 1$. It holds $\underline{y} = \underline{f}_d^1(a, 1, b) = (a, a, 1 - a + b)$. Since $0 \leqslant b < a < 1, 1 - a + b + a \geqslant 1$ and $1 - a + b < 1$, we have that $\underline{y}$ satisfies the conditions in (v), and thus $\underline{f}_d^1(a, a, 1-a+b) = (a, 1, 1-a+b+a-1) = (a, 1, b) = \underline{x}$.

(v) Let $\underline{x} = (a, a, b)$ with $a < 1, b + a \geqslant 1$ and $b < 1$. It holds $\underline{y} = \underline{f}_d^1(a, a, b) = (a, 1, b + a - 1)$. Since $b + a \geqslant 1, b < 1$ and $0 \leqslant b + a - 1 < a < 1$, we have that $\underline{y}$ satisfies the conditions in (iv), and thus $\underline{f}_d^1(a, 1, b + a - 1) = (a, a, 1 - a + b + a - 1) = (a, a, b) = \underline{x}$.

(vi) Let $\underline{x} = (a, b, 0)$ with $0 < a < b < 1$. It holds $\underline{y} = \underline{f}_d^1(a, b, 0) = (a, a, b - a)$. Since $b - a + a < 1$ and $b - a > 0$, we have that $\underline{y}$ satisfies the conditions in (vii), and consequently $\underline{f}_d^1(a, a, b - a) = (a, b, 0) = \underline{x}$.

(vii) Let $\underline{x} = (a, a, b)$ with $0 < a, b+a < 1$ and $b > 0$. Since $\underline{y} = \underline{f}_d^1(a, a, b) = (a, b+a, 0)$ satisfies the conditions in (vi), it holds $\underline{f}_d^1(a, b + a, 0) = (a, a, b) = \underline{x}$.

(viii) Trivial. □

The proof of the previous proposition shows the method used to build the function $\underline{f}_d^1$. Rules (i) and (viii) allow the function to behave as the Petri–Fredkin gate when its arguments are restricted to {0, 1}. Rules (ii) and (iv) have been introduced in order to allow the gate to generate the Łukasiewicz implication on the third output line and the Łukasiewicz disjunction on the second output line when the second input line is set to 1. Rules (iii) and (v) are the inverses of rules (ii) and (iv); this is done in order to guarantee the self-reversibility of the gate. Rules (vi) and (viii) realize the Łukasiewicz conjunction on the second output line when the third input line is set to 0, whereas (vii) and (viii) are their inverses. Rule (viii) uses the simplest self-reversible function (the identity function) to deal with the cases not considered by the other rules.

Properties (F5), (F6), (F7) and (F8) are trivially satisfied by $\underline{f}_d^1$. Moreover, each rule was written in order to verify the property of weak conservativeness. In fact, the following proposition holds, whose proof is straightforward, and thus omitted.

**Proposition 8.2.** *$\underline{f}_d^1$ is weakly conservative.*

It is not difficult to see that $\underline{f}_d^1$ is also a universal function. In fact, as shown in table 15, through suitable configurations of constants assigned to its arguments we obtain a set of connectives which suffice to generate, besides the FAN-OUT function, all the operators of Łukasiewicz and Gödel $d$-valued logics.

It is important to emphasize that, as said before, the properties of the gate do not depend on the number of truth values involved. Moreover, when $d = 3$ the function $\underline{f}_d^1$ behaves just like the gate $F_1$ presented in the previous section.

**Table 15.** The operators obtained through function $\underline{f}_d^1$.

| Connective | Inputs | Constants | Outputs | Garbage |
|---|---|---|---|---|
| FAN-OUT | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_1, y_2$ | $y_3$ |
| $\mathrm{Pr}_1$ | $x_2, x_3$ | $x_1 = 0$ | $y_3$ | $y_1, y_2$ |
| $\mathrm{Pr}_2$ | $x_2, x_3$ | $x_1 = 0$ | $y_2$ | $y_1, y_3$ |
| $\to_L$ | $x_1, x_3$ | $x_2 = 1$ | $y_3$ | $y_1, y_2$ |
| $\to_G$ | $x_1, x_2$ | $x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\vee$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ |
| $\wedge$ | $x_1, x_2$ | $x_3 = 0$ | $y_2$ | $y_1, y_3$ |
| Id | $x_1$ | $x_2 = 0, x_3 = 0$ | $y_1$ | $y_2, y_3$ |
| $\neg$ | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_3$ | $y_1, y_2$ |
| $\sim$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\Diamond$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_3$ | $y_1, y_2$ |

### 8.2. A family of functions which realize necessity

Since $\underline{f}_d^1$ does not allow one to realize the modal operator $\Box$, we propose the following family of functions. Let $\lambda$ be any value from the set $L_d \backslash \{0, 1\}$; the family of functions $\underline{f}_{d,\lambda}^2 : L_d^3 \to L_d^3$, parametrized with respect to $\lambda$, is defined as follows:

$\forall \underline{x} = (x_1, x_2, x_3) \in L_d^3$

$$\underline{f}_{d,\lambda}^2(\underline{x}) := \begin{cases} (x_2, x_1, x_3) & \text{if } x_1 = 0, 0 < x_2 < 1 \text{ and } x_3 = \lambda & \text{(i)} \\ (x_2, x_1, x_3) & \text{if } 0 < x_1 < 1, x_2 = 0 \text{ and } x_3 = \lambda & \text{(ii)} \\ (x_1, x_3, x_2) & \text{if } x_1 = 0, x_2 \neq \lambda, x_3 \neq \lambda \text{ and } x_2 \neq x_3 & \text{(iii)} \\ (x_1, x_3, x_2) & \text{if } 0 \leqslant x_1 \leqslant x_3 < 1 \text{ and } x_2 = 1 & \text{(iv)} \\ (x_1, x_3, x_2) & \text{if } 0 \leqslant x_1 \leqslant x_2 < 1 \text{ and } x_3 = 1 & \text{(v)} \\ (x_1, x_1, 1 - x_1 + x_3) & \text{if } x_3 < x_1 < 1 \text{ and } x_2 = 1 & \text{(vi)} \\ (x_1, 1, x_3 + x_1 - 1) & \text{if } x_1 < 1, x_2 = x_1, x_3 + x_1 \geqslant 1 & \\ & \quad \text{and } x_3 < 1 & \text{(vii)} \\ (x_1, x_1, x_2 - x_1) & \text{if } 0 \leqslant x_1 < x_2 < 1 \text{ and } x_3 = 0 & \text{(viii)} \\ (x_1, x_3 + x_1, 0) & \text{if } 0 \leqslant x_1, x_2 = x_1, x_3 + x_1 < 1 & \\ & \quad \text{and } x_3 > 0 & \text{(ix)} \\ (x_1, x_2, x_3) & \text{otherwise.} & \text{(x)} \end{cases}$$

For each fixed value of $\lambda$ we get a function which realizes the connectives exposed in table 16. As can be seen, the price we pay to realize the modal connective $\Box$ together with all the connectives of $\underline{f}_d^1$ is that the functions $\underline{f}_{d,\lambda}^2$ lose 0-regularity in $2d - 5$ input/output pairs and property (F7) in $2d - 4$ input/output pairs. Properties (F6) and (F8) are trivially satisfied by functions $\underline{f}_{d,\lambda}^2$.

The following proposition holds. The proof of self-reversibility is similar to the one of proposition 8.1, and thus omitted; as for weak conservativeness, it suffices to inspect the definition of $\underline{f}_{d,\lambda}^2$.

**Proposition 8.3.** *For each fixed value of* $\lambda$ *in* $L_d \backslash \{0, 1\}$, *the function* $\underline{f}_{d,\lambda}^2$ *is self-reversible and weakly conservative.*

We observe that, for a fixed $\lambda$, the constants involved in the configurations depicted in table 16 are independent of $d$. Notice also that, when $d = 3$, the function $\underline{f}_{d,\frac{1}{2}}^2$ behaves just like the gate $F_2$ presented in the previous section.

**Table 16.** The operators obtained through functions $\underline{f}^{2}_{d,\lambda}$.

| Connective | Inputs | Constants | Outputs | Garbage |
|---|---|---|---|---|
| FAN-OUT | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_1, y_2$ | $y_3$ |
| $\mathrm{Pr}_1$ | $x_2, x_3$ | $x_1 = 1$ | $y_2$ | $y_1, y_3$ |
| $\mathrm{Pr}_2$ | $x_2, x_3$ | $x_1 = 1$ | $y_3$ | $y_1, y_2$ |
| $\rightarrow_L$ | $x_1, x_3$ | $x_2 = 1$ | $y_3$ | $y_1, y_2$ |
| $\rightarrow_G$ | $x_1, x_2$ | $x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\vee$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ |
| $\wedge$ | $x_1, x_2$ | $x_3 = 0$ | $y_2$ | $y_1, y_3$ |
| Id | $x_1$ | $x_2 = 0, x_3 = 0$ | $y_1$ | $y_2, y_3$ |
| $\neg$ | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_3$ | $y_1, y_2$ |
| $\sim$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\diamond$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_3$ | $y_1, y_2$ |
| $\square$ | $x_1$ | $x_2 = 0, x_3 = \lambda$ | $y_1$ | $y_2, y_3$ |

*8.3. A gate for MV-connectives*

None of the gates just presented generates the MV-connectives shown in section 4. This fact led us to define the following function $\underline{m}_d\colon L_d^3 \to L_d^3$:

$$\forall \underline{x} = (x_1, x_2, x_3) \in L_d^3$$

$$\underline{m}_d(\underline{x}) := \begin{cases} (x_1, x_3, x_2) & \text{if } x_1 = 0 \text{ and } x_2 \neq x_3 & \text{(i)} \\ (x_1, x_1 + x_3, 1 - x_1) & \text{if } x_1 > 0, x_2 = 1 \text{ and } x_1 + x_3 < 1 & \text{(ii)} \\ (x_1, 1, x_2 - x_1) & \text{if } 0 < x_1 \leqslant x_2 < 1 \text{ and } x_3 = 1 - x_1 & \text{(iii)} \\ (x_1, x_1 + x_2 - 1, \\ 1 - x_1) & \text{if } x_1 < 1, x_2 < 1, x_3 = 0 \text{ and} \\ & \quad x_1 + x_2 > 1 & \text{(iv)} \\ (x_1, x_2 + x_3, 0) & \text{if } 0 < x_2 < x_1 < 1 \text{ and } x_3 = 1 - x_1 & \text{(v)} \\ (x_1, x_3, x_2) & \text{if } 0 < x_1, x_2 > 0, x_3 = 0 \text{ and} \\ & \quad x_1 + x_2 \leqslant 1 & \text{(vi)} \\ (x_1, x_3, x_2) & \text{if } 0 < x_1, x_2 = 0, x_3 > 0 \text{ and} \\ & \quad x_1 + x_3 \leqslant 1 & \text{(vii)} \\ (x_1, x_2, x_3) & \text{otherwise.} & \text{(viii)} \end{cases}$$

In order to find this gate we used the technique previously shown: first we looked at the gate $F_3$ exposed in the previous section in order to know which configurations give rise to the operators $\oplus$ and $\odot$; successively, we wrote their inverses. Thus it is no wonder that, for $d = 3$, the function $\underline{m}_d$ behaves like the gate $F_3$ presented in the previous section.

As we have done with the previous functions, we can state the following proposition.

**Proposition 8.4.** *$\underline{m}_d$ is self-reversible and weakly conservative.*

Moreover, properties (F5), (F6), (F7) and (F8) are trivially satisfied by $\underline{m}_d$.

Table 17 reports the operators that can be obtained from function $\underline{m}_d$ by fixing one or two input lines with constant values from $L_d$. As we can see, $\underline{m}_d$ is a gate providing functional completeness for the finite-valued calculus of Łukasiewicz, regardless of the value assumed by $d$.

**Table 17.** The operators obtained through function $\underline{m}_d$.

| Connectives | Inputs | Constants | Outputs | Garbage |
|---|---|---|---|---|
| FAN-OUT | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_1, y_2$ | $y_3$ |
| $\mathrm{Pr}_1$ | $x_2, x_3$ | $x_1 = 0$ | $y_3$ | $y_1, y_2$ |
| $\mathrm{Pr}_2$ | $x_2, x_3$ | $x_1 = 0$ | $y_2$ | $y_1, y_3$ |
| $\oplus$ | $x_1, x_3$ | $x_2 = 1$ | $y_2$ | $y_1, y_3$ |
| $\odot$ | $x_1, x_2$ | $x_3 = 0$ | $y_2$ | $y_1, y_3$ |
| Id | $x_1$ | $x_2 = 0, x_3 = 0$ | $y_1$ | $y_2, y_3$ |
| $\neg$ | $x_1$ | $x_2 = 1, x_3 = 0$ | $y_3$ | $y_1, y_2$ |
| $\sim$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_2$ | $y_1, y_3$ |
| $\Diamond$ | $x_1$ | $x_2 = 0, x_3 = 1$ | $y_3$ | $y_1, y_2$ |
| $\Box$ | $x_3$ | $x_1 = \frac{1}{d-1}, x_2 = 0$ | $y_3$ | $y_1, y_2$ |

## 9. Conclusions and directions for future work

We presented some generalizations of the Petri–Fredkin gate for $d$-valued reversible and conservative logics, notably $d$-valued Łukasiewicz and $d$-valued Gödel logics. In particular, we introduced three gates for three-valued logics and three possible extensions of such gates for $d$-valued logics; one of the extensions was specifically designed to realize the MV-connectives. Moreover we showed how to realize, with such gates, the operators that characterize some modal logics.

One of the purposes of our work was to show that the framework of reversible and conservative computation can be extended towards some non-classical 'reasoning environments', originally proposed to deal with propositions which embed imprecise and uncertain information, that are usually based upon many-valued and modal logics.

It remains an open question as to how it is possible to extend further the framework towards infinite-valued logics, such as fuzzy logics, both with $\aleph_0$ and $\aleph_1$ truth values. We feel that in such settings many new and interesting questions arise; here we propose just a few of them. For example: since reversible circuits no longer need to have the same number of input and output lines, and moreover we can encode on a single input (or output) as much information as we want, what are the computational properties of such circuits? What are the differences with respect to reversible and conservative circuits for $d$-valued logics? How can we characterize the set of functions computed by such circuits?

Moreover, it is not difficult to extend proposition 6.1 to deal with an infinite number of truth values. A direct consequence is that there are no possible extensions of the Petri–Fredkin gate to infinite-valued logics which compute the FAN-OUT function and at the same time are strictly conservative. How does this change the notion of conservativeness? In this paper we proposed the alternative notion of weak conservativeness, together with a possible physical interpretation; however, when dealing with an infinite number of energy levels there are two possibilities: either the energy levels extend over an unlimited range, or the levels become increasingly close to each other. In the former case there is the problem of assigning the truth value 1 to some energy level. In the latter case, an infinite precision on the amount of energy can be required to switch from one level to another; in this situation, when the energy gap between the levels becomes smaller than the underlying thermal noise the computing physical system goes out of control. The above observations naturally lead to the following question: are the circuits for infinite-valued logics physically realizable? On the other hand, do we really need them?

## Acknowledgment

This work has been supported by MIUR\COFIN project ‘Formal Languages and Automata: Theory and Application’.

## References

[Be73] Bennett C H 1973 Logical reversibility of computation *IBM J. Res. Dev.* **17** 525–32

[Be88] Bennett C H 1988 Notes on the history of reversible computation *IBM J. Res. Dev.* **32** 16–23

[Be98] Bennett C H 1998 Quantum information *Technical Report, Tutorial, MFCS ’98*

[CLL02] Cattaneo G, Leporati A and Leporini R Quantum conservative gates for finite-valued logics *Int. J. Theor. Phys.* at press

[CN89] Cattaneo G and Nisticò G 1989 Brouwer–Zadeh posets and three-valued Łukasiewicz posets *Fuzzy Sets and Systems* **33** 165–90

[Ch58] Chang C C 1958 Algebraic analysis of many valued logics *Trans. Am. Math. Soc.* **88** 467–90

[Ch59] Chang C C 1959 A new proof of the completeness of Łukasiewicz axioms *Trans. Am. Math. Soc.* **93** 74–80

[Ch88] Chellas B F 1988 *Modal Logic, An Introduction* (Cambridge, MA: Cambridge University Press)

[Fe85] Feynman R P 1985 Quantum mechanical computers *Opt. News* **11** 11–20

[FT82] Fredkin E and Toffoli T 1982 Conservative logic *Int. J. Theor. Phys.* **21** 219–53

[Ki76] Kinoshita K *et al* 1976 On magnetic bubble circuits *IEEE Trans. Comput.* C **25** 247–53

[La61] Landauer R 1961 Irreversibility and heat generation in the computing process *IBM J. Res. Dev.* **3** 183–91

[LR90] Leff H S and Rex A F (ed) 1990 *Maxwell’s Demon: Entropy, Information, Computing* (Princeton, NJ: Princeton University Press)

[Lu20] Łukasiewicz J 1920 O logice *Ruch Filozoficzny* **5** 170–1 (Engl. transl. 1970 On three-valued logic *Selected Works* ed L Borkowski (Amsterdam: North-Holland) p 87)

[Lu30] Łukasiewicz J 1930 Philosophische Bemerkungen zu mehrwertigen Systemen des Aussagenkalküs *Comptes Rendus des Séances de la Société des Sciences et des Lettres de Varsovie* **23** 51–77 (Engl. transl. 1970 Philosophical remarks on many-valued systems of propositional logic *Selected Works* ed L Borkowski (Amsterdam: North-Holland) p 153)

[Lu70] Łukasiewicz J 1970 *Selected Works* ed L Borkowski (Amsterdam: North-Holland)

[Mc51] McNaughton R 1951 A theorem about infinite-valued sentential logic *J. Symb. Logic* **16** 1–13

[Pe67] Petri C A 1967 Gründsatzliches zur Beschreibung diskreter Prozesse *Proc. 3rd Coll. über Automatentheorie (Hannover, 1965)* (Basel: Birkhäuser) pp 121–40 (Engl. transl. 1982 Fundamentals of the representation of discrete processes *ISF Report 82.04*; transl. by H J Genrich and P S Thiagarajan)

[Re69] Rescher N 1969 *Many-Valued Logics* (New York: McGraw-Hill)

[RT52] Rosser J B and Turquette A R 1952 *Many-Valued Logics* (Amsterdam: North-Holland)

[Tof80] Toffoli T 1980 Reversible computing *Automata, Languages and Programming* ed J W de Bakker and J van Leeuwen (Berlin: Springer) p 632 Also available as *Technical Memo* MIT/LCS/TM-151 MIT Laboratory for Computer Science, February 1980

[vN56] von Neumann J 1956 Probabilistic logics and the synthesis of reliable organisms from unreliable components *Automata Studies* ed C E Shannon and J McCarthy (Princeton, NJ: Princeton University Press) pp 43–98

[Za34] Zawirski Z 1934 Relation of many-valued logic to probability calculus (in Polish, original title: Stosunek logiki wielowartościowej do rachunku prawdopodobieństwa) *Poznańskie Towarzystwo Przyjaciól Nauk.*