What is a Physical Realization of a Computational System?

Marco Giunti
ALOPHIS -- Applied LOgic, Philosophy and HIstory of Science*
Università di Cagliari
giunti@unica.it

1. Introduction

The concept of a physical realization of a computational system is one of the key notions of both functionalism and the symbolic approach to cognitive science or AI (Pylyshyn 1980: 113, 1984; Chalmers 1996: 309). Notwithstanding a widespread consensus on the theoretical importance of this concept, it comes somewhat as a surprise that a precise analysis or shared definition does not exist yet, either in the philosophical camp, or in the cognitive science and AI community[1]. In this paper, I will present my attempt at such an analysis. Before doing this, however, I will say a few words on why I believe that a seemingly promising alternative strategy is misguided.

* Postal address: Dipartimento di Pedagogia Psicologia Filosofia, via Is Mirrionis 1, 09123 Cagliari, Italy.

[1] Endicott (2005) distinguishes at least three different technical meanings of the term ‘realization’ among philosophers. Putnam claims that “every ordinary open system is a realization of every abstract finite automaton” (1988: 121) and, in the same vein, Searle (1990) argues that any physical system can be seen to realize any computation. Both Putnam’s and Searle’s arguments are clearly flawed (Chalmers 1996), but they do show that a precise explication of the notion of realization of a computational system is badly needed.

Marco Giunti, “What is a Physical Realization of a Computational System?”
in Flavia Marcacci and Maria Grazia Rossi, *Reasoning, metaphor and science*, pp. 177-192.

University of Urbino Carlo Bo
http://isonomia.uniurb.it/epistemologica

According to this strategy, both a computational system and its multiple physical realizations are to be thought as dynamical systems[2], and the realization relation is then analyzed as a special kind of structure preserving mapping from the state space of the computational system into the state space of the physical one (Giunti 1997: ch. 1; for a similar approach, see Fano *et al.* 2014). As the same kind of structure preserving mapping (i.e., an emulation function)[3] may very well exist between a single computational system and many physical systems, it might seem that we have a clearcut explanation of the basic feature of the realization relation, namely, multiple (low level) realizability of the same (high level) system.

However, there are two main reasons why the realization relation cannot be analyzed in terms of some form of emulation of a computational system by a physical one. The first is that emulation is a structure preserving mapping between two *mathematical* systems (*i.e.*, *dynamical* systems), and thus this kind of approach merely shifts the problem, for the realizing system (the physical one) is not a real or concrete system, but another mathematical or abstract system. So the question now is: what is a realization of a physical system? This question is not unsolvable in principle, but the only solution I can think of is subscribing to a form of Platonism, according to which at least some mathematical systems (those at the lowest level) are real.

Second, the emulation based strategy provides us with an *in principle* solution, which makes us think of realization as a relation between two dynamical systems — a high level computational system on the one hand, and a low level physical one on the other. However, no one would maintain that we could in fact know *the details* of the low level physical system, for

[2] Giunti and Mazzola (2012: Definition 1) define a deterministic dynamical system on any time model $L = (T, +)$ that satisfies the minimal requirement of being a monoid, as follows. *DS is a dynamical system on L* := DS is a pair $(M, (g^t)_{t\in T})$ and L is a pair $(T, +)$ such that (i) $L = (T, +)$ is a monoid. Any $t \in T$ is called a *duration* of the system, T is called its *time set*, and L its *time model*; (ii) M is a non-empty set. Any $x \in M$ is called a *state* of the system, and M is called its *state space*; (iii) $(g^t)_{t\in T}$ is a family indexed by T of functions from M to M. For any $t \in T$, the function g^t is called the *state transition of duration t* (briefly, *t-transition*, or *t-advance*) of the system; (iv) for any v, $t \in T$, for any $x \in M$, (a) $g^0(x) = x$, where 0 is the identity element of L; (b) $g^{v+t}(x) = g^v(g^t(x))$.

[3] Let $DS_1 = (M, (g^t)_{t\in T})$ be a dynamical system on $L = (T, +)$, and $DS_2 = (N, (h^v)_{v\in V})$ be a dynamical system on $P = (V, \oplus)$. An emulation function u of DS_2 *in* DS_1 is defined as follows (Giunti 2010a: sec. 2, 2014: sec. 2, Definition 8); *u is an emulation of DS_2 in DS_1* := u is an injective function from N to M and, for any $v \in V$, for any $c \in N$, there is $t \in T$ such that $u(h^v(c)) = g^t(u(c))$. Furthermore, we say that DS_1 *emulates* DS_2 just in case there is an emulation u of DS_2 in DS_1.

those details would admittedly be too complex. I believe that this kind of in principle solution is not in agreement with the question we have in mind when we ask what it means for a computational system to be realized by a physical one. This question does not ask for abstract in principle answers, but for detailed and concrete ones. For, in many cases, we can very well tell that a real system is a physical realization of a computational one, even though we do not have any low level description of the real system, nor do we even mention such description, think of, or make any reference to it.

2. Computational systems as setup interpreted dynamical systems

My present strategy envisages that an adequate solution to the realization problem will emerge once the quite complex nature of computational systems is fully recognized and brought to light. According to this view, computational systems are more similar to *empirically correct dynamical models* than to dynamical systems tout court. Thus, the solution of the realization problem is to be sought among the *modeling* relations between *dynamical systems* and *phenomena*, and not among the *emulation* relations between purely mathematical dynamical systems.[4]

In my view a computational system $\boldsymbol{CS} = (DS, H, I_{DS,H})$ is a complex object which consists of three parts. First, a mathematical part $DS = (M, (g^t)_{t\in T})$, which is a *discrete*[5] *n-component*[6] *dynamical system.*[7] Second, a

[4] While emulation is the wrong kind of relation for an analysis of *realization*, emulation is indeed an adequate basis for a representational analysis of *reduction* of empirically interpreted dynamical systems (Giunti 2010a, 2014). The two issues are closely related but, in order to avoid conceptual confusion, they cannot be lumped together. The first one has to do with a relation between a computational system and its realizers, where each realizer is a *concrete* or *real* system. The second one deals with a relation between two mathematical dynamical systems, both of which are *models* of real phenomena.

[5] A dynamical system $DS = (M, (g^t)_{t\in T})$ is *discrete* := (i) the time model of DS is the additive monoid $(T, +)$, where the time set T is either the set of the non-negative integers $Z_{\geq 0}$ or the integers Z, and + is the usual operation of sum of two integers; (ii) the state space M is at most countably infinite (Giunti 2010a: sec. 2, 2014: sec. 2).

[6] For any i $(1 \leq i \leq n)$, let X_i be a non-empty set, and let $DS = (M, (g^t)_{t\in T})$ be a dynamical system whose state space $M \subseteq X_1 \times \ldots \times X_n$; for any i, the set $C_i = \{x_i :$ for some n-tuple $x \in M$, x_i is the i-th element of $x\}$ is called the *i-th component of M*, and DS is called an *n-component dynamical system* (Giunti 2014: sec. 4.1, 2016: Definition 2).

[7] The two requirements of (i) being a *discrete* dynamical system with (ii) a *finite* number of state space components make up a *necessary*, but not sufficient condition for the mathematical part DS of a computational system. The debate on which other requirements should be considered for an adequate definition of the purely mathematical part of a computational system is still open (Gandy 1980; Giunti 1997, 2006; Giunti & Giuntini

computational setup[8] $H = (F, B_F)$, which is made up of a *theoretical* part F, and a *real part* B_F. And, third, an interpretation $I_{DS,H}$, which links the dynamical system DS with the setup H.

2.1. The computational setup *H*

Let us now see in more detail what the computational setup $H = (F, B_F)$ looks like. First, its theoretical part F is a *functional description* which provides a sufficiently detailed specification of:

a) the internal constitution and organization, or functioning, of any real system of a certain *type* AS_F;
b) a *causal scheme* CS_F of the external interactions of any real system of type AS_F. In particular, the description of the causal scheme CS_F must include the specification of:
   (b.1) the *initial conditions* that an arbitrary temporal evolution of any real system of type AS_F must satisfy;
   (b.2) the *boundary conditions* during the whole subsequent evolution;
   (b.3) and, possibly, the *final conditions* under which the evolution terminates.

Second, the real part B_F of the computational setup H is the set of all real or concrete systems which satisfy the functional description F or, in other words, B_F is the set of all real systems of type AS_F whose temporal evolutions are all constrained by the causal scheme CS_F. B_F is called the realization domain[9] (or application domain) of H. Any real system $b_F \in B_F$ is called an *F-realizer*.

As $H = (F, B_F)$ is a *computational* setup, any temporal evolution of an arbitrary F-realizer $b_F \in B_F$ proceeds in discrete time steps, which correspond to basic operation cycles of b_F. A basic operation cycle can always be interpreted as the (serial or parallel) execution, by b_F, of a finite

2007; Sieg 1999, 2002a, 2002b; Shagrir 2002; Dershowitz & Gurevich 2008; Fano *et al.* 2014).

[8] A *computational setup* is in fact a special kind of *dynamical phenomenon*, see Giunti (2010a: sec. 4, 2010b: Appendice, 2014: sec. 4.1, 2016: sec. 3).

[9] Since the functional description F typically contains several idealizations, no real system *exactly* satisfies F, but it rather fits F up to a *certain degree*. Thus, from a formal point of view, the realization domain B_F of a computational setup $H = (F, B_F)$ would be more faithfully described as a *fuzzy* set.

number of instructions for symbol manipulation. Therefore, the temporal evolutions of an F-realizer are in fact its *computations*.

A clear example of a computational setup is provided by a typical informal presentation of a Turing machine. An arbitrary *Turing machine setup*, $H_{TM} = (F_{TM}, B_{F_{TM}})$, is detailed below.

F_{TM}: Functional description of a Turing machine (theoretical part of a Turing machine setup H_{TM})

(A) SPECIFICATION OF ANY REAL SYSTEM OF *TURING MACHINE TYPE* $AS_{F_{TM}}$

A real system of Turing machine type $AS_{F_{TM}}$ is made up of two main parts: an external part and an internal one. The *external part* consists of two devices. First, the *external memory*, which may be thought as a linear arrangement of a finite number of cells (*e.g.*, a tape divided into squares). Each cell always contains exactly one symbol taken from a finite alphabet $A = \{b, a_1, \ldots , a_m\}$ of m+1 ($m \geq 1$) different symbols. The number of cells is always finite, but new cells can be added as needed (see below), either to the left of the leftmost cell, or to the right of the rightmost one. However, when a cell is added, it always contains the special symbol b, called the *blank.*

Second, a *read/write/move head*, which is a device that, at each instant, is located on exactly one cell of the external memory, and is capable of doing three operations: (1) read the symbol contained in the cell where it is located (*scanned symbol*), and send it to the *control unit* (in the internal part) to which it is constantly connected; (2) replace the scanned symbol with a symbol received from the control unit; (3) move one cell to the right, one cell to the left, or stay put, according to the moving command +1, −1, or 0 that it receives from the control unit. If the head receives the command +1 (−1) when it is located on the rightmost (leftmost) cell, it first adds one blank cell to the right (left), and then moves to the newly added cell.

The *internal part* of the system also consists of two devices. First, an *internal memory*, which may be thought as a single cell that always contains exactly one symbol (*internal state*) taken from a finite alphabet $Q = \{q_1, \ldots , q_n\}$ of n ($n \geq 1$) different symbols. The internal memory is constantly connected to the control unit and, besides being just a container for an internal state, it is also capable of two operations: (1) read its internal state, and send it to the *control unit*; (2) replace its internal state with an internal state received from the control unit.

The second internal device is a *control unit*, which is a deterministic input/output mechanism. Its inputs are state-symbol pairs q_ia_j, while its outputs are symbol-movement-state triples a_kXq_l. (X stands for one of the three commands +1, −1, 0.) For any possible input pair $q_i a_j$, the control unit always responds with a fixed output triple $a_k X q_l$. The input-output behavior of the control unit is thus completely described by three finite tables, **T1**, **T2**, **T3**, in which each input pair $q_i a_j$ is listed together with, respectively, the corresponding output symbol a_k, the movement command X, or the new internal state q_l. Both the symbol a_k and the command X of an output triple are always sent to the head, while the new internal state q_l is sent to the internal memory.

An operation cycle of the whole system is as follows. First, the current internal state q and the symbol a in the cell where the head is located are simultaneously read and sent to the control unit, which responds with an output triple $a'Xq'$. Second, the new symbol a' and the moving command X are sent to the head, which first replaces a with a' and then moves according to X; at the same time, the new internal state q' is sent to the internal memory, where it replaces q. The operation cycle is thus complete, and the system is ready to start a new cycle.

(B) SPECIFICATION OF THE CAUSAL SCHEME $CS_{F_{TM}}$ OF THE EXTERNAL INTERACTIONS OF ANY REAL SYSTEM OF TURING MACHINE TYPE $AS_{F_{TM}}$

(1) *Initial conditions*. A sequence of operation cycles (*i.e., a temporal evolution, or a computation*) of a system of type $AS_{F_{TM}}$ starts as soon as the following settings are performed: (i) an initial internal state is fixed, (ii) a symbol in each cell of the external memory is fixed, and (iii) the cell where the head is initially located is fixed.

(2) *Boundary conditions*. During the whole subsequent computation there is no further interaction with the external environment.

(3) *Final conditions*. The computation terminates immediately after an operation cycle that satisfies *all* the following conditions: (i) $q' = q$, (ii) $a' = a$, (iii) $X = 0$.

$B_{F_{TM}}$: Realization domain (real part of a Turing machine setup H_{TM})

The realization domain $B_{F_{TM}}$ of a Turing machine setup H_{TM} is the (fuzzy)[10] set of all real or concrete systems that satisfy the given specification of the Turing machine type $AS_{F_{TM}}$, and whose temporal evolutions all satisfy the given specification of the Turing machine interaction scheme $CS_{F_{TM}}$. Any system $b_{F_{TM}} \in B_{F_{TM}}$ is called a *Turing machine realizer.*

[10] See footnote 9.

2.2. The discrete *n*-component dynamical system *DS*

So far, we have mainly focused on the setup part H of a computational system. But, as mentioned above, a computational system is a complex object that also includes a purely mathematical part DS, and an interpretation $I_{DS,H}$ of the mathematical part DS on the setup H. In particular, we have seen that the mathematical part DS is a *discrete n-component dynamical system*. Let us now see what this dynamical system looks like in the special case of a Turing machine. An arbitrary *Turing machine dynamical system*, DS_{TM}, is described below.

An arbitrary Turing machine dynamical system $\boldsymbol{DS_{TM}}$

From the purely mathematical point of view, an arbitrary Turing machine can be identified with the discrete, 3-component dynamical system $DS_{TM} = (Q{\times}P{\times}C, (g^t)_{t\in Z_{\geq 0}})$, which is the one completely specified by the difference equation stated below (Table 2.3). Note first that this dynamical system is *discrete*, for its time model is the additive monoid $(Z_{\geq 0}, +)$ of the non-negative integers, and its state space $Q{\times}P{\times}C$ is *countably* infinite (because of the finitary restriction on the functions in C, see next paragraph).

Second, DS_{TM} is a 3-*component* dynamical system, for its state space has the three components Q, P, and C. $Q = \{q_1, \ldots, q_n\}$, where q_i is an arbitrary internal state of the Turing machine; $P = Z$ is the set of all integers, which is intended to represent all the possible positions of the head, and C is the set of all functions $c: P \to \{b, \mathrm{a}_1, \ldots, a_m\}$ that satisfy the finitary restriction: $c(x) \neq b$ for at most finitely many $x \in P$. Any such function is intended to represent a possible content of the Turing machine external memory. Let us now consider the five variables and the six functions defined in Tables 2.1 and 2.2 below.

Variable	*Domain*	*State variable*
a	$A := \{b, \mathrm{a}_1, \ldots, a_m\}$	*NO*
q	$Q := \{q_1, \ldots, q_n\}$	*YES*
p	$P := Z$, the integers	*YES*
c	$C :=$ the set of all functions $c: P \to A$ such that $c(p) \neq b$ for at most finitely many $p \in P$	*YES*
m	$M := \{-1, 0, +1\}$	*NO*

Table 2.1 The variables needed for describing an arbitrary Turing machine dynamical system $DS_{TM} = (Q{\times}C{\times}P, (g^t)_{t\in Z_{\geq 0}})$.

Then, an arbitrary *Turing machine dynamical system* is the 3-component discrete dynamical system $DS_{TM} := (Q\times P\times C, (g^t)_{t\in Z_{\geq 0}})$, whose time set is $Z_{\geq 0}$, whose state space is $Q\times P\times C$, and whose family of state transition functions $(g^t : Q\times P\times C \rightarrow Q\times P\times C)_{t\in Z_{\geq 0}}$ is defined by the 3-component difference equation shown in Table 2.3.[11]

Function	***Codomain***	***Definition***
$\mathbf{READ}(c, p)$	A	$\mathbf{READ}(c, p) := c(p)$
$\mathbf{WRITE}(a, c, p)$	C	$\mathbf{WRITE}(a, c, p) := c' \in C$ such that, for any $x \in P$, if $x = p$, $c'(x) = a$; else, $c'(x) = c(x)$
$\mathbf{MOVE}(p, m)$	P	$\mathbf{MOVE}(p, m) := p + m$
$\mathbf{A}(q, a)$	A	$\mathbf{A}(q, a) :=$ for any q and a, $\mathbf{A}(q, a)$ is listed in a given finite table T1
$\mathbf{M}(q, a)$	M	$\mathbf{M}(q, a) :=$ for any q and a, $\mathbf{M}(q, a)$ is listed in a given finite table T2
$\mathbf{Q}(q, a)$	Q	$\mathbf{Q}(q, a) :=$ for any q and a, $\mathbf{Q}(q, a)$ is listed in a given finite table T3

Table 2.2 The functions needed for describing an arbitrary Turing machine dynamical system $DS_{TM} = (Q\times C\times P, (g^t)_{t\in Z_{\geq 0}})$.

State variable	***Difference equation***
q	$q' = \mathbf{Q}(q, \mathbf{READ}(c, p))$
p	$p' = \mathbf{MOVE}(p, \mathbf{M}(q, \mathbf{READ}(c, p)))$
c	$c' = \mathbf{WRITE}(\mathbf{A}(q, \mathbf{READ}(c, p)), c, p)$

Table 2.3 The 3-component difference equation that univocally individuates an arbitrary Turing machine dynamical system $DS_{TM} = (Q\times C\times P, (g^t)_{t\in Z_{\geq 0}})$.

2.3. The interpretation $I_{DS, H}$

The final step of my analysis focuses on the nature and role of the third part of a computational system, that is, the interpretation $I_{DS,H}$ of the mathematical part DS on the computational setup H. Let $DS = (M, (g^t)_{t\in T})$

[11] More precisely, the family of state transition functions $(g^t: Q\times P\times C \rightarrow Q\times P\times C)_{t\in Z_{\geq 0}}$ is defined as follows: (i) g^0 is the identity function; (ii) g^1 is the function defined by the 3-component difference equation in Table 2.3; for any $t \in Z_{\geq 1}$, g^t is the t-th iteration of g^1.

be a discrete n-component dynamical system, and $H = (F, B_F)$ be a computational setup. An interpretation $I_{DS,H}$ of DS on H consists in stating that (i) each component C_i of the state space M is included in, or is equal to, the set $V(\boldsymbol{M}_i)$ of all the possible values of a magnitude $\boldsymbol{M}_i$ of setup H, and (ii) the time set T of DS is equal to the set $V(\boldsymbol{T})$ of all the possible values of the time magnitude $\boldsymbol{T}$ of setup H.[12] In other words, an interpretation $I_{DS,H}$ can always be identified with a particular set of n+1 statements. We thus define:

$I_{DS,H}$ is an interpretation of DS on H $:= I_{DS,H} = \{C_1 \subseteq V(\boldsymbol{M}_1), \ldots , C_n \subseteq V(\boldsymbol{M}_n), T = V(\boldsymbol{T})\}$, where C_i is the i-th component of M, $\boldsymbol{M}_i$ is a magnitude of the setup H, $V(\boldsymbol{M}_i)$ is the set of all possible values of $\boldsymbol{M}_i$, $\boldsymbol{T}$ is the time magnitude of H and, for any i and j, if $i \neq j$, then $\boldsymbol{M}_i \neq \boldsymbol{M}_j$.

For example, let us consider a Turing machine dynamical system $DS_{TM} := (Q{\times}P{\times}C, (g^t)_{t\in Z_{\geq 0}})$ and a Turing machine setup $H_{TM} = (F_{TM}, B_{F_{TM}})$. Then, the *intended interpretation* of DS_{TM} on H_{TM} is described below.

The intended interpretation of DS_{TM} on H_{TM}

Let $\boldsymbol{Q}$ be the content of the internal memory of an arbitrary Turing machine realizer $b_{F_{TM}} \in B_{F_{TM}}$; then, $Q = V(\boldsymbol{Q})$.

Let $\boldsymbol{P}$ be the position of the head of $b_{F_{TM}}$. The head position is always individuated by the cell where it is located; as the cells are linearly arranged and their number is finite, if we choose one of them as the origin, we obtain a one-one correspondence between the cells and an initial segment of Z (in either the positive or the negative direction). Once such an integer coordinate system is fixed, an arbitrary possible value of $\boldsymbol{P}$ can be identified with the coordinate of the cell where the head is located; thus, $V(\boldsymbol{P}) = Z = P$.

Let $\boldsymbol{C}$ be the whole content of the external memory of $b_{F_{TM}}$, *i.e.*, the distribution of symbols over each of its cells. Then, with respect to the fixed integer coordinate system for the cells (see previous paragraph), an arbitrary

[12] In general, we take a magnitude of a computational setup $H = (F, B_F)$ to be a property $\boldsymbol{M}_j$ of every F-realizer $b_F \in B_F$ such that, at different instants, it can assume different values. The set of all possible values of magnitude $\boldsymbol{M}_j$ is indicated by $V(\boldsymbol{M}_j)$. We further assume that, among the magnitudes of any computational setup H, there always is its time magnitude, which we denote by $\boldsymbol{T}$. The set of all possible values (instants or durations) of the time magnitude of H is indicated by $V(\boldsymbol{T})$, Since the time of a computational setup evolves in discrete steps that correspond to its basic operation cycles, we take $V(\boldsymbol{T}) = Z_{\geq 0}$ (or $V(\boldsymbol{T}) = Z$, only if the setup is reversible).

possible value of magnitude $\boldsymbol{C}$ can be thought as any of the functions $c \in C$;[13] therefore, $C = V(\boldsymbol{C})$.

Finally, let $\boldsymbol{T}$ be the temporal ordering according to which an arbitrary temporal evolution of $b_{F_{TM}}$ occurs. Since any such evolution is in fact a sequence of successive operation cycles, each operation cycle corresponds to a non-negative integer; thus, $Z_{\geq 0} = V(\boldsymbol{T})$.

In short, *the intended interpretation of DS_{TM} on H_{TM}* consists of the following set of four identities.

$I_{DS_{TM}, H_{TM}} := \{Q = V(\boldsymbol{Q}), P = V(\boldsymbol{P}), C = V(\boldsymbol{C}), Z_{\geq 0} = V(\boldsymbol{T})\}$.

Turing machines as setup interpreted dynamical systems

Finally, as a typical example of a computational system, we can define a Turing machine as follows.

***TM** is a Turing machine* := $\boldsymbol{TM}$ is a triple $(DS_{TM}, H_{TM}, I_{DS_{TM}, H_{TM}})$, where $DS_{TM} = (Q \times C \times Z, (g^t)_{t \in Z_{\geq 0}})$ is a Turing machine dynamical system, $H_{TM} = (F_{TM}, B_{F_{TM}})$ is a Turing machine setup, $I_{DS_{TM}, H_{TM}}$ is the intended interpretation of DS_{TM} on H_{TM}, and the three tables T1, T2, T3 which, respectively, define the three functions **A**, **M**, **Q** of DS_{TM} are identical to the three tables **T1**, **T2**, **T3** that completely describe the input-output behavior of the control unit of an arbitrary real system of Turing machine type $AS_{F_{TM}}$, as specified by the functional description F_{TM} of setup H_{TM}.

3. Physical realizations of a computational system

Once an interpretation $I_{DS,H} = \{C_1 \subseteq V(\boldsymbol{M}_1), \ldots , C_n \subseteq V(\boldsymbol{M}_n), T = V(\boldsymbol{T})\}$ is given, we define the possible states of the setup $H = (F, B_F)$ as follows.

x is a possible state of H relative to $I_{DS,H}$:= $x \in V(\boldsymbol{M}_1) \times \ldots \times V(\boldsymbol{M}_n)$. $V(\boldsymbol{M}_1) \times \ldots \times V(\boldsymbol{M}_n)$ is called *the state space of H relative to $I_{DS,H}$*, and is indicated by $\boldsymbol{M}$.

The interpretation $I_{DS,H}$ also allows us to define the instantaneous state of an arbitrary F-realizer *of H*. Let $b_F \in B_F$ be an arbitrary F-realizer of setup H, and $j \in T$ an arbitrary instant. Then:

x is the state of b_F at instant j relative to $I_{DS,H}$:= $x = (x_1, \ldots , x_n)$, where x_i is the value at instant $j \in T$ of magnitude $\boldsymbol{M}_i$ of b_F (if, at instant j, such a value exists).

[13] Recall that, for any $c \in C$, c: $Z \rightarrow A = \{b, a_1, \ldots, a_m\}$.

Obviously, if x is the state of b_F at instant j, then $x \in \boldsymbol{M}$. Note, however, that, depending on the instant j, the value of magnitude $\boldsymbol{M}_i$ of b_F may not exist.[14] If this happens, the state of b_F at instant j relative to $I_{DS,H}$ is not defined.

Now, relative to the interpretation $I_{DS,H}$, we may define the set C_F of all those possible states of H (if any) that *actually* are initial states of H.

$C_F :=$ {x : for some $b_F \in B_F$, for some temporal evolution e of b_F, for some $j \in T$, j is the initial instant of e and x is the state of b_F at j relative to $I_{DS,H}$}. C_F is called *the set of all initial states of H, relative to interpretation* $I_{DS,H}$.

Intuitively, the set C_F may be thought as the set of all those states in $\boldsymbol{M}$ that are consistent with the initial conditions specified by the causal scheme CS_F and are in fact initial states of some realizer $b_F \in B_F$. Also note that, depending on the interpretation $I_{DS,H}$, C_F may be empty, or C_F may not be a subset of the state space M of DS.[15] The definition of an *admissible interpretation* (see below) will exclude these somewhat pathological interpretations.

Let $C_F \neq \emptyset$. Let us now define, with respect to interpretation $I_{DS,H}$, the set of all initial instants of the evolutions of a given F-realizer $b_F \in B_F$, whose initial state $x \in C_F$ be fixed. We call this set $J_{b_F,x}$.

$J_{b_F,x} := \{j_{b_F,x} : j_{b_F,x}$ is the initial instant of some evolution of b_F, and x is the state of b_F at instant $j_{b_F,x}\}$. $J_{b_F,x}$ is called *the set of the initial instants of* b_F *whose initial state is x, relative to* $I_{DS,H}$.

Note that, for some $b_F \in B_F$ and $x \in C_F$, $J_{b_F,x}$ may be empty.[16] However, by the definition of C_F, for any $x \in C_F$, there is $b_F \in B_F$ such that $J_{b_F,x} \neq \emptyset$.

As the setup H is usually taken to be deterministic, the existence and identity of the instantaneous state, at any fixed stage of an evolution of any realizer b_F, is not intended to depend on either the initial instant, or the

[14] If, for some reason, b_F no longer exists at instant $j \in T$, then *a fortiori* the value at j of magnitude $\boldsymbol{M}_i$ of b_F does not exist either. Furthermore, we are not making any assumption about the continuous existence of the values of a magnitude during any interval of time. Thus, it is in principle possible that the value of magnitude $\boldsymbol{M}_i$ of b_F exists at some instant j of b_F's existence, but does not exist at some other instant k of its existence.

[15] In fact, by the definition of instantaneous state, C_F is empty if, for any $b_F \in B_F$ and any state evolution e of b_F, some magnitude $\boldsymbol{M}_i$ does not have a value at the initial instant of e. Also recall that, according to interpretation $I_{DS,H}$, each component C_i of the state space M is in general a subset of $V(\boldsymbol{M}_i)$. Thus, if for some $x \in C_F$, its i-th component $x_i \in V(\boldsymbol{M}_i)$ is not a member of C_i, then $C_F \not\subseteq M$.

[16] In fact, $J_{b_F,x}$ is empty if x is not the state of b_F at the initial instant of any of its evolutions.

identity of b_F, but only on the initial state. Thus, any admissible interpretation $I_{DS,H}$ should at least ensure that the condition below holds.

Condition D (Determinism). For any b_F, $d_F \in B_F$, for any $x \in C_F$, for any $j_{b_F,x} \in J_{b_F,x}$, for any $k_{d_F,x} \in J_{d_F,x}$, for any $t \in T$, if $t + j_{b_F,x}$ is an instant of the evolution of b_F that starts at $j_{b_F,x}$ and the state of b_F at instant $t + j_{b_F,x}$ exists, then $t + k_{d_F,x}$ is an instant of the evolution of c_F that starts at $k_{d_F,x}$, the state of c_F at instant $t + k_{d_F,x}$ exists as well, and the state of b_F at instant $t + j_{b_F,x}$ = the state of d_F at instant $t + k_{d_F,x}$.

Let $C_F \neq \emptyset$. For any initial state $x \in C_F$, let us consider the set of all F-realizers whose initial state is x. This set, denoted by B_{F_x}, is in other words the collection of all F-realizers b_F whose set $J_{b_F,x}$ is not empty. Note that also this definition, as the previous ones, depends on the interpretation $I_{DS,H}$.

B_{F_x} := $\{b_F \in B_F$ such that $J_{b_F,x} \neq \emptyset\}$. B_{F_x} is called *the set of all F-realizers whose initial state is x, relative to interpretation* $I_{DS,H}$.

We noticed above that, for any $x \in C_F$, there is $b_F \in B_F$ such that $J_{b_F,x} \neq \emptyset$. Therefore, by the definition just stated, for any $x \in C_F$, $B_{F_x} \neq \emptyset$.

Suppose $C_F \neq \emptyset$. Then, for any $x \in C_F$, for any $b_F \in B_{F_x}$, for any $j_{b_F,x} \in J_{b_F,x}$, we define the following set of durations:

$q_{b_F,j_{b_F,x}}(x) := \{t$: $t \in T$, $t + j_{b_F,x}$ is an instant of the evolution of b_F that starts at $j_{b_F,x}$, and there is $y \in \boldsymbol{M}$ such that y is the state of b_F at $t + j_{b_F,x}$, relative to $I_{DS,H}\}$.

Note that this definition, like the previous ones, is *relative* to the interpretation $I_{DS,H}$. Furthermore, $q_{b_F,j_{b_F,x}}(x) \neq \emptyset$, for $0 \in q_{b_F,j_{b_F,x}}(x)$.

Also note that, whenever Condition D above holds, $q_{b_F,j_{b_F,x}}(x)$ depends on x, but does not depend on either b_F or $j_{b_F,x}$; therefore, if Condition D holds, we simply write "$q_F(x)$" instead of "$q_{b_F,j_{b_F,x}}(x)$".

By Condition D and the previous definition, for any $x \in C_F$, $q_F(x)$ is the set of all durations t that transform the initial state x of an arbitrary F-realizer $b_F \in B_{F_x}$ into some other state of b_F. More briefly, we call $q_F(x)$ *the set of all durations that transform the initial state x of H into some other state*.

As we are not interested in any interpretation $I_{DS,H}$ such that (a) $C_F = \emptyset$, or (b) $C_F \not\subseteq M$, or (c) Condition D does not hold[17], we define:

[17] If either (a), (b), or (c), the interpretation $I_{DS,H}$, is obviously incorrect, for: if (a) holds, no evolution of any F-realizer b_F can be represented by means of the state transition family $(g^t)_{t\in T}$ of $DS = (M, (g^t)_{t\in T})$; if (b) holds, some evolution of some F-realizer b_F cannot be represented by $(g^t)_{t\in T}$; if (c) holds, some evolution of some F-realizer b_F cannot be correctly represented by $(g^t)_{t\in T}$.

$I_{DS,H}$ *is an admissible interpretation of DS on H* := (i) $C_F \neq \emptyset$ and (ii) $C_F \subseteq M$ and (iii) Condition D holds.

We can now precisely state the conditions for an interpretation $I_{DS,H}$ to be correct. The intuitive idea is this. As soon as an interpretation $I_{DS,H}$ is fixed, the dynamical system $DS = (M, (g^t)_{t \in T})$ provides us with a representation of the real systems (F-realizers) in the realization domain of computational setup H. Such a representation is in fact provided by the state transition family $(g^t)_{t \in T}$ of dynamical system DS. The interpretation $I_{DS,H}$ will thus turn out to be correct if the representation, provided by $(g^t)_{t \in T}$, of *all* temporal evolutions of *all* F-realizers of H is correct. This intuitive idea is formally expressed by the definition below.

$I_{DS,H}$ *is a correct interpretation of DS on H* := (i) $I_{DS,H}$ is an admissible interpretation of DS on H and (ii) for any $x \in C_F$, for any $t \in q_F(x)$, for any $b_F \in B_{F_x}$, for any $j_{b_F,x} \in J_{b_F,x}$, $g^t(x)$ = the state of b_F at instant $t + j_{b_F,x}$ relative to $I_{DS,H}$.

Let $\boldsymbol{CS} = (DS, H, I_{DS,H})$ be a computational system, and B_F be the realization domain of H. We can thus finally define:

b is a physical realization of $\boldsymbol{CS}$:= $b \in B_F$ and $I_{DS,H}$ is a correct interpretation of DS on H.

It is now easy to prove that, in the special but important case of an arbitrary Turing machine $\boldsymbol{TM}$, the intended interpretation $I_{DS_{TM},H_{TM}}$ of the Turing machine dynamical system DS_{TM} on the Turing machine setup H_{TM} *is indeed* a correct interpretation of DS_{TM} on H_{TM}, so that any Turing machine realizer $b_{F_{TM}} \in B_{F_{TM}}$ turns out to be a physical realization of $\boldsymbol{TM}$, and conversely.

Theorem
If $\boldsymbol{TM} = (DS_{TM}, H_{TM}, I_{DS_{TM},H_{TM}})$ is a Turing machine, then $I_{DS_{TM},H_{TM}}$ is a correct interpretation of DS_{TM} on H_{TM}.
Proof
The thesis is a straightforward consequence of the definitions of Turing machine and correct interpretation of DS on H, in conjunction with the theoretical part F_{TM} of the Turing machine setup H_{TM}, the specification of DS_{TM}, and the intended interpretation $I_{DS_{TM},H_{TM}}$. *Q.E.D.*

Corollary
For any Turing machine $\boldsymbol{TM}$, b is a physical realization of $\boldsymbol{TM}$ iff $b \in B_{F_{TM}}$.

Proof
By the previous Theorem and the definition of physical realization of a computational system. *Q.E.D.*

4. Concluding remarks

It is my contention that the previous Theorem is not peculiar to Turing machines, but that an analogous theorem holds for each specific type of computational system (for instance, finite automata, register machines, cellular automata, and so on). If this is true, *all* computational systems are then characterized by a form of *a-priori* (or purely theoretical) *interpretation* of the mathematical part on the setup part, for the correctness of the interpretation can be *proved*. The *a-priori* character of the interpretation of a computational system distinguishes this kind of system from ordinary dynamical models of phenomena (Giunti 2010a: sec. 4, 2010b: Appendice, 2014: sec. 4.1, 2016: secs. 3 and 4), as found in empirical science. In fact, for this second kind of dynamical system, the correctness of the interpretation of the mathematical part on the intended *phenomenon* (which is the analog of the *setup* of a computational system) cannot be established *a-priori*—cannot be *proved*—but only *a-posteriori* or empirically.

References

Chalmers, D., 1996, "Does a rock implement every finite-state automaton?", in *Synthese*, 108, 3, pp. 309-333.

Dershowitz, N., Gurevich, Y., 2008, "A natural axiomatization of computability and proof of Church's thesis", in *The Bulletin of Symbolic Logic*, 14, 3, pp. 299-350.

Endicott, R., 2005, "Multiple Realizability", in D. Borchert (ed.), *The Encyclopedia of Philosophy*, vol. 6, 2nd Edition. US: Thomson Gale, Macmillan Reference, pp. 427-432.

Fano, V., Graziani, P., Macrelli, R., Tarozzi, G., 2014, "Are Gandy machines really local?", in V.C. Müller (ed.), *Computing and philosophy*, Synthese Library, vol. 375, New York: Springer, pp. 27-44.

Gandy, R., 1980, "Church's thesis and principles for mechanism", in J. Barwise, H.J. Keisler, & K. Kunen (eds.), *The Kleene Symposium*, Amsterdam: North Holland Publishing Company, pp. 123-148.

Giunti, M., 1997, *Computation, dynamics, and cognition*, New York, Oxford University Press.

Giunti, M., 2006, "Is being computational an intrinsic property of a dynamical system?", in G. Minati, E. Pessa, & M. Abram (eds.), *Systemics of emergence: Research and development*, New York: Springer, pp. 683-694.

Giunti, M., 2010a, "Reduction in dynamical systems", in M. D'Agostino, G. Giorello, F. Laudisa, T. Pievani, C. Sinigaglia (eds.), *SILFS New essays in logic and philosophy of science*, London: College Publications, pp. 677-694.

Giunti, M., 2010b, "Panorama e prospettive dell'approccio dinamico in scienza cognitiva", in *L&PS – Logic and Philosophy of Science*, VIII, 1, pp. 101-118. Retrieved from http://www2.units.it/episteme. English translation of the Appendice (Appendix) retrieved from http://dx.doi.org/10.13140/RG.2.1.3621.8089

Giunti, M., 2014, "A representational approach to reduction in dynamical systems", in *Erkenntnis*, 79, 4, pp. 943-968.

Giunti, M., 2016, "A real world semantics for deterministic dynamical systems with finitely many components", in L. Felline, A. Ledda, F. Paoli, & E. Rossanese (eds.), *New Directions in Logic and the Philosophy of Science*, London: College Publications, pp. 97-110. Online: https://dx.doi.org/10.13140/RG.2.1.3838.1923/1

Giunti, M., Giuntini, R., 2007, "Macchine, calcolo e pensiero", in S. Mancini (ed.), *Sguardi sulla scienza dal giardino dei pensieri*, Milano: Mimesis, pp. 39-67.

Giunti M., Mazzola C., 2012, "Dynamical systems on monoids: Toward a general theory of deterministic systems and motion", in G. Minati, M. Abram, E. Pessa (eds.), *Methods, models, simulations and approaches towards a general theory of change*, Singapore: World Scientific, pp. 173-185.

Searle, J., 1990, “Is the brain a digital computer?”, in *Proceedings and Adresses of The American Philosophical Association*, 64, 3, pp. 21-37.

Putnam, H., 1988, *Representation and reality*, Cambridge, MA, The MIT Press.

Pylyshyn, Z.W., 1980, “Computation and cognition: Issues in the foundations of cognitive science”, in *The Behavioral and Brain Sciences*, 3, pp. 111-169.

Pylyshyn, Z.W., 1984, *Computation and Cognition*, Cambridge, MA, The MIT Press.

Shagrir, O., 2002, “Effective computation by humans and machine”, in *Minds and Machines*, 12, pp. 221-240.

Sieg, W., 1999, “An abstract model for parallel computation: Gandy’s thesis”, in *The Monist*, 82, pp. 150-164.

Sieg, W., 2002a, “Calculations by man and machine: Mathematical presentation”, in P. Gärdenfors, J. Woleński, & K. Kijania-Placek (eds.), *In the scope of logic, methodology and philosophy of science*, vol. I. Dordrecht: Kluwer Academic Publishers, pp. 247-262.

Sieg, W., 2002b, “Calculations by man and machine: Conceptual analysis”, in W. Sieg, R. Sommer, & C. Talcott (eds.), *Reflections on the foundations of mathematics: Essays in honor of Solomon Feferman*, Association for Symbolic Logic, pp. 390-409.