IS BEING COMPUTATIONAL AN INTRINSIC PROPERTY OF A DYNAMICAL SYSTEM?

MARCO GIUNTI
Università di Cagliari, Italy

Abstract: I consider whether or not a discrete dynamical system has two isomorphic representations, one recursive and the other non-recursive; if it does not, the system can be said to be an intrinsic computational system. I prove that intrinsic computational systems exist, as well as non-intrinsic ones, and I finally argue that some representation of a non-intrinsic computational system is not effective with respect to the state-space structure of the system.

Key words: dynamical systems theory, discrete system, computational system, computation, computability theory, recursive function, effective procedure, representability.

1. INTRODUCTION

By the term *computational system* I refer to any device of the kind studied by standard (or elementary) computation theory. Thus, for example, Turing machines, register machines, cellular automata, and finite state automata are four different types of computational systems. Discreteness and determinism are two properties shared by all such devices. Therefore, so called analog computers are not computational systems in this sense.

Computational systems can in fact be thought as dynamical systems of a special kind. From an intuitive point of view, the computational systems can be identified with those discrete, deterministic, dynamical systems that can be described or represented effectively. Elsewhere, I gave a formal explication of this intuitive concept, and I showed that Turing machines and all other systems that have been actually studied by standard computation theory (register machines, cellular automata, monogenic production systems, etc.) satisfy the formal definition (Giunti 1992, 1997).

According to this definition, being a computational system is a property that reduces to the existence of an *effective representation* of a discrete dynamical system. The condition of the existence of an effective representation can be made precise by requiring that the discrete system admit at least one recursive isomorphic system, *i.e.*, an isomorphic system (i) whose state space is a recursively enumerable[1] subset of the natural numbers and (ii) whose state transitions are recursive.

However, it is quite natural to ask whether or not the property of being computational is *intrinsic* to the dynamics of a discrete system. In fact, a discrete system might admit two isomorphic numeric representations, such that one is recursive and the other is not. In this case, the property of being computational could not be said to be intrinsic to the dynamics of the system, for it would depend on the numeric representation of the dynamics we choose.

In sec. 2 of this paper, I will lay down the formal apparatus necessary to discuss this problem and, in sec. 3, I will prove that some computational systems are intrinsic, but some are not. I will also raise the question whether intrinsic non-computational systems exist, *i.e.*, whether there are discrete systems whose numeric representations are all non-recursive.

Finally, in sec. 4, I will make a few remarks about a different way of looking at this kind of problem. One of the results of sec. 3 show that some discrete systems are non-intrinsic computational systems, for they admit two numeric representations such that one is recursive and the other is not. One may wonder whether this result might depend on the fact that at least one of the two *representations* is not intrinsic to the dynamics of the system.

By an *intrinsic representation* of a discrete[2] dynamical system DS I mean a pair (u, $DS_{\#}$) such that: (i) $DS_{\#}$ is a dynamical system whose state space is the set of the natural numbers Z^{+}; (ii) u is an isomorphism of $DS_{\#}$ in DS (and thus, u is a bijective enumeration from Z^{+} to the state space M of DS); (iii) the enumeration $u: Z^{+} \rightarrow M$ can be constructed effectively by means of a mechanical procedure that takes as given the *whole structure* of the state space M, and *nothing more*.

A precise definition of the intuitive idea of an intrinsic representation of a discrete dynamical system, however, requires some further general concepts of dynamical systems theory and graph theory, as well as the new notion of an *enumerating machine*, *i.e.*, a machine that effectively produces an enumeration of the state space by moving from state to state in accordance with the state transitions determined by the dynamics of the system. These developments go beyond the limits of the present work, and will be the subject of a forthcoming paper.

2. DYNAMICAL AND COMPUTATIONAL SYSTEMS

A dynamical system is a mathematical model that expresses the idea of an arbitrary deterministic system, either reversible or irreversible, with discrete or continuous time or state space (Arnold 1977; Szlensk 1984). Let Z be the integers, Z^+ the non-negative integers, R the reals and R^+ the non-negative reals; below is the exact definition of a dynamical system.

[1] *DS is a dynamical system* iff there is M, T, $(g^t)_{t\in T}$ such that $DS = (M, (g^t)_{t\in T})$ and

1. M is a non-empty set; M represents all the possible states of the system, and it is called the *state space*;
2. T is either Z, Z^+, R, or R^+; T represents the time of the system, and it is called the *time set*[3];
3. $(g^t)_{t\in T}$ is a family of functions from M to M; each function g^t is called a *state transition* or a *t-advance* of the system;
4. for any $t, v \in T$, for any $x \in M$,
   a. $g^0(x) = x$;
   b. $g^{t+v}(x) = g^v(g^t(x))$.

[2] A *discrete*[4] *dynamical system* is a dynamical system whose state space is finite or denumerable, and whose time set is either Z or Z^+; [3] a *continuous dynamical system* is a dynamical system that is not discrete; [4] a *cascade* is a dynamical system with discrete time, *i.e.*, whose time set is either Z or Z^+. Thus, all discrete dynamical systems are cascades, but the reverse is not true.

[5] A dynamical system is *reversible* iff its time set is either Z or R; [6] it is *irreversible* iff its time set is either Z^+ or R^+. If a dynamical system DS is reversible, then any state transition is bijective, and the set of all state transitions $\{g^t\}_{t\in T}$ is a commutative group with respect to the composition of functions; the unit is g^0 and, for any $t \in T$, the algebraic inverse of g^t is g^{-t} = the inverse function $(g^t)^{-1}$. If DS is irreversible, $\{g^t\}_{t\in T}$ is a commutative monoid with respect to the composition operation, with unit g^0.

Any t-advance ($t > 0$) of an irreversible cascade can always be thought as being generated by iterating t times a given function $g: M \to M$ (thus, $g^1 = g$). As for a reversible cascade, the generating function $g: M \to M$ must be bijective; the positive t-advances are then obtained as before, while the negative ones ($t < 0$) are generated by iterating $|t|$ times the inverse function g^{-1}.

[7] $DS = (M, (g^t)_{t\in T})$ is a *possible dynamical system* iff DS satisfies the first three conditions of def. 1. We can now define the concept of an isomorphism between two possible dynamical systems as follows. [8] *u is an isomorphism of DS_1 in DS_2* iff $DS_1 = (M, (g^t)_{t\in T})$ and $DS_2 = (N, (h^v)_{v\in V})$ are possible dynamical systems, $T = V$, $u: M \to N$ is a bijection and, for any $t \in T$, for any $x \in M$, $u(g^t(x)) = h^t(u(x))$.

[9] DS_1 *is isomorphic* to DS_2 iff there is u such that u is an isomorphism of DS_1 in DS_2. It is easy to verify that the isomorphism relation is an equivalence relation on any given set of possible dynamical systems. (The concept of *set of all possible dynamical systems* is inconsistent, and we must then take as the basis of the theory of dynamical systems a specific, sufficiently large, set of possible dynamical systems.)

It is also not difficult to prove that the relation of isomorphism is a congruence with respect to the property of being a dynamical system, that is to say: if DS_1 is isomorphic to DS_2 and DS_1 is a dynamical system, then DS_2 is a dynamical system. This allows us to speak of abstract dynamical systems in exactly the same sense we talk of abstract groups, fields, lattices, order structures, etc. We can thus define: [10] an *abstract dynamical system* is any equivalence class of isomorphic dynamical systems.

[11] A *representation* of a dynamical system DS is a pair $(u, DS_\#)$ such that u is an isomorphism of $DS_\#$ in DS. [12] A *numeric representation* of a dynamical system DS is a representation $(u, DS_\#)$ of DS such that the state space of $DS_\#$ is a subset of Z^+. By def. 12 and 1, it immediately follows that any discrete system has a numeric representation. [13] A *recursive representation* of a discrete dynamical system DS is a numeric representation $(u, DS_\#)$ of DS such that (i) the state space of $DS_\#$ is a recursively enumerable subset of Z^+; (ii) any state transition of $DS_\#$ is a recursive function.

Note that, since any discrete system is a cascade, condition (ii) of def. 13 reduces to the following condition. Let $g_\#$ be the generating function of the positive state transitions of $DS_\#$. Then, if DS is irreversible, (ii) is equivalent to requiring that $g_\#$ be recursive; if DS is reversible, (ii) is equivalent to requiring that both $g_\#$ and its inverse $g_\#^{-1}$ be recursive. However, by condition (i), the domain of $g_\#$ is a recursively enumerable subset of Z^+; this, together with the recursivity of $g_\#$, entails that $g_\#^{-1}$ is recursive too. Therefore, condition (ii) is equivalent to requiring that $g_\#$ be recursive.

[14] A *canonic numeric representation* of a dynamical system DS is a numeric representation $(u, DS_\#)$ of DS such that either the state space $Z_\#$ of $DS_\#$ is an initial segment of Z^+, or $Z_\# = Z^+$. By def. 14 and 1, it is obvious that any discrete dynamical system has a canonic numeric representation. [15] A *recursive canonic representation* of a discrete dynamical system DS

is a canonic numeric representation (u, $DS_{\#}$) of *DS*, which is also a recursive representation of *DS*.

Obviously, the recursive canonic representations are a proper subset of the recursive representations. Nevertheless, if a discrete dynamical system has a recursive representation, it has a recursive canonic representation as well. This is established by the following theorem.

Theorem 1 [*recursive canonic representability*]
For any discrete dynamical system *DS*, if *DS* has a recursive representation, then *DS* has a recursive canonic representation.

Proof
Let $DS = (M, (g^t)_{t\in T})$ be a discrete dynamical system, and let (u, $DS_{\#}$) be a recursive representation of *DS*, where $DS_{\#} = (N, (g_{\#}{}^t)_{t\in T})$. The proof is trivial if *M* is finite. Let us then assume that *M* is denumerable. By def. 13, *N* is recursively enumerable. By means of this fact, we construct a canonic numeric representation of *DS*, and we then show that it is recursive.

Since *N* is recursively enumerable, there is $e: Z^+ \rightarrow N$ such that *e* is surjective and recursive. Let us define $c: Z^+ \rightarrow N$ as follows: for any $m \in Z^+$, $c(m) = e$(the least $n \geq m$ such that, for any $k < m$, $c(k) \neq e(n)$). By its definition, *c* is a bijection from Z^+ to *N* and, since *e* is recursive, *c* is recursive as well. Since the domain of *c* is the whole Z^+, its inverse c^{-1} is recursive too. Let us then define $u': Z^+ \rightarrow M$ as follows: for any $m \in Z^+$, $u'(m) = u(c(m))$. By its definition, u' is a bijection from Z^+ to *M*. Let $g_{\#}': Z^+ \rightarrow N$ be defined as follows: for any $m \in Z^+$, $g_{\#}'(m) = u'^{-1}(g(u'(m)))$, where $g = g^1$ is the generating function of *DS*. Let $DS_{\#}' = (Z^+, (g_{\#}'^t)_{t\in T})$ be the discrete dynamical system generated by $g_{\#}'$. Then, by construction, (u', $DS_{\#}'$) is a canonic numeric representation of *DS*.

In addition, by the definitions of u' and $g_{\#}'$, and since *u* is an isomorphism of $DS_{\#}$ in *DS*, it follows that, for any $m \in Z^+$, $g_{\#}'(m) = c^{-1}(g_{\#}(c(m)))$. Thus, being a composition of recursive functions, $g_{\#}'$ is recursive. Therefore, (u', $DS_{\#}'$) is a recursive representation of *DS*. Since it is also a canonic numeric representation of *DS*, by def. 15, the thesis follows.
Q.E.D.

Finally, we can state the precise definition of the concept of a computational system. [16] *DS* is a *computational system* iff *DS* is a discrete dynamical system, and there is a recursive representation of *DS*.

Before concluding this section, let me remark that, according to theorem 1, computational systems admit a *uniform* recursive representation. In fact, all computational systems whose state space has the same cardinality have a recursive canonic representation with *identical* state space (either the same initial segment of Z^+ for all finite systems with the same number of states, or Z^+ itself for all systems with a denumerable number of states). Thus, the recursive canonic representations of any two such systems only differ for the *state space structure*, that is, for the family of recursive state transitions that determine the specific dynamics of the different systems.

3. INTRINSIC VS. NON-INTRINSIC COMPUTATIONAL SYSTEMS

We are now in a position to distinguish two types of computational systems, according to whether the property of being computational is intrinsic or not. [17] *DS* is an *intrinsic computational system* iff *DS* is a discrete dynamical system, and any numeric representation of *DS* is a recursive representation of *DS*. [18] *DS* is a *non-intrinsic computational system* iff *DS* is a computational system, and there is a numeric representation of *DS* that is not a recursive representation of *DS*. [19] *DS* is an *intrinsic non-computational system* iff *DS* is a discrete dynamical system, and any numeric representation of *DS* is not a recursive representation of *DS*.

Note that computational systems and intrinsic non-computational systems constitute a partition of the set of all discrete dynamical systems. However, while the set of all computational systems is certainly not empty, it is an open question whether intrinsic non-computational systems exist.[5]

Analogously, intrinsic computational systems and non-intrinsic computational systems form a partition of the set of all computational systems. However, all *denumerable*[6] computational systems are non-intrinsic, and this entails that the set of all intrinsic computational systems is identical to the set of all finite discrete dynamical systems. This is the content of the theorem below.

Theorem 2 [*finiteness of any intrinsic computational system*]

1. If *DS* is a denumerable computational system, then *DS* is a non-intrinsic computational system;
2. *DS* is an intrinsic computational system iff *DS* is a finite discrete dynamical system.

Proof of 1

If *DS* is a denumerable computational system, it is always possible to find a non-recursive numeric representation (u, $DS_\#$) of *DS*. Take the state space N of $DS_\#$ to be an arbitrary non-recursively enumerable subset of Z^+. If M is the state space of *DS* and g is the generating function of *DS*, choose any bijection $u: N \to M$, and generate the state transitions of $DS_\#$

by $g_{\#}: N \rightarrow N$ such that, for any $n \in N$, $g_{\#}(n) = u^{-1}(g(u(n)))$. By construction, ($u$, $DS_{\#}$) is a non-recursive numeric representation of *DS*.

Proof of 2
The left/right implication follows from thesis 1. The converse is obvious. Q.E.D.

According to theorem 2, all denumerable computational systems are non-intrinsic (thesis 1). However, if we look at the proof of thesis 1, we realize that it depends crucially on condition (i) of def. 13, *i.e.*, on the requirement that the state space of any recursive representation be a recursively enumerable subset of Z^+. One may then think that the definition of an intrinsic computational system (def. 17) is too strong, for it fails for all denumerable systems *just because*, for any such system, there is a numeric representation whose state space in not recursively enumerable. But, as far as we know, this numeric representation might very well satisfy condition (ii) of def. 13. It is then natural to ask whether, by appropriately limiting the scope of the relevant numeric representations, we might get a refined concept of intrinsic computational system, for which the somewhat trivial proof of thesis 1 does not go through.

In effect, it is possible to get such a refined concept by just considering the *canonic* numeric representations, and not *all* the numeric representations like def. 17 does. The new definitions (where "*c*" is a reminder that these concepts are limited to canonic numeric representations) are as follows. [20] *DS* is a *c-intrinsic computational system* iff *DS* is a discrete dynamical system, and any canonic numeric representation of *DS* is a recursive representation of *DS*. [21] *DS* is a *c-non-intrinsic computational system* iff *DS* is a computational system, and there is a canonic numeric representation of *DS* that is not a recursive representation of *DS*. [22] *DS* is a *c-intrinsic non-computational system* iff *DS* is a discrete dynamical system, and any canonic numeric representation of *DS* is not a recursive representation of *DS*.

I will now show that the set of the *c*-intrinsic computational systems, as well as the one of the *c*-non-intrinsic computational systems, is not empty, and that both sets admit members whose state space is denumerable.

The proof concerning the *c*-intrinsic computational systems takes into account the discrete dynamical system $DS_1 = (Z^+, (i^n)_{n \in Z^+})$, generated by the identity function $i: Z^+ \rightarrow Z^+$. It is then almost immediate to show that DS_1 is a *c*-intrinsic computational system.

As for *c*-non-intrinsic computational systems, I will show that the discrete dynamical system $DS_2 = (Z^+, (s^n)_{n \in Z^+})$, generated by the successor function $s: Z^+ \rightarrow Z^+$, is a *c*-non-intrinsic computational system.

Theorem 3 [*existence of both c-intrinsic and c-non-intrinsic computational systems*]

1. $DS_1 = (Z^+, (i^n)_{n \in Z^+})$ is a *c*-intrinsic computational system;
2. $DS_2 = (Z^+, (s^n)_{n \in Z^+})$ is a *c*-non-intrinsic computational system.

Proof of 1

Obviously, DS_1 is a computational system, for (i, DS_1), where i is the identity function on Z^+, is a recursive representation of DS_1.

Note that an arbitrary canonic numeric representation of DS_1 is of the form $(p, DS_\#)$, where $p: Z^+ \to Z^+$ is an arbitrary bijection and $DS_\# = DS_1$. Thus, by def. 13, $(p, DS_\#)$ is a recursive representation of DS_1. Therefore, by def. 20, DS_1 is a *c*-intrinsic computational system.

Proof of 2

Obviously, DS_2 is a computational system, for (i, DS_2), where i is the identity function on Z^+, is a recursive representation of DS_2.

For any bijection $p: Z^+ \to Z^+$, let $s_p: Z^+ \to Z^+$ such that, for any $m \in Z^+$, $s_p(m) = p(s(p^{-1}(m)))$. Let $DS_p = (Z^+, (s_p{}^n)_{n \in Z^+})$ be the discrete dynamical system generated by s_p. Then, by construction, (p^{-1}, DS_p) is a canonic numeric representation of DS_2.

Note that, for any p, s_p can be thought as a "new successor function" on Z^+, corresponding to the order induced by p on Z^+. The first element of this order, so to speak the "new zero element", is $p(0)$, the "new 1" is $p(1)$, and so forth, so that, for any $m \in Z^+$, $p(m) = s_p{}^m(p(0))$. It is then easy to verify that, for any two different bijections p and q, $s_p \neq s_q$.[7]

Consequently, there are as many functions s_p as there are bijections $p: Z^+ \to Z^+$. But the number of such bijections is non-denumerable. Hence, there is p^* such that s_{p^*} is not recursive. It thus follows that the canonic numeric representation (p^{*-1}, DS_{p^*}) is not recursive. Therefore, by def. 21, DS_2 is a *c*-non-intrinsic computational system.

Q.E.D.

4. TOWARD A THEORY OF INTRINSIC REPRESENTABILITY

That the system generated by the identity function be a *c*-intrinsic computational system was to be expected. On the contrary, the proof that the computational system DS_2 generated by the successor function is *c*-non-intrinsic is surely surprising. There is a feeling of oddity in realizing that a dynamical system like DS_{p^*}, which has exactly the same structure as the sequence of the natural numbers, is generated by a non-recursive *pseudo-successor* function s_{p^*}, and that (p^{*-1}, DS_{p^*}) thus constitutes a *bona fide* non-recursive canonic representation of DS_2, which, in contrast, is generated by the *authentic* successor function that is obviously recursive.[8]

One may wonder that, after all, (p^{*-1}, DS_{p^*}) is not a *bona fide* representation of DS_2. That this way of looking at the problem might be promising is confirmed by the following observation. While it is obvious that, if we are given the whole structure of DS_2 (*i.e.*, the successor function $s: Z^+ \to Z^+$), we can mechanically produce the identity function i (by simply starting from state 0 and counting 0, then moving to state $s(0) = 1$ and counting 1, and so forth), it seems that, by just moving back and forth along the structure of DS_2 and counting whenever we reach a new state, in no way can we produce such a complex permutation of Z^+ like the bijection p^{*-1} (see fig. 1 below).

Also observe that the situation is exactly symmetrical if, instead, we imagine that we are given the whole structure of DS_{p^*} (*i.e.*, the pseudo-successor function $s_{p^*}: Z^+ \to Z^+$). In this case, we can easily produce p^* (by starting from state pseudo-0 = $p^*(0)$ and counting 0, then moving to state $s_{p^*}(0)$ = pseudo-1 and counting 1, and so forth), but it seems that, by just moving back and forth along the structure of DS_{p^*} and counting whenever we reach a new state, in no way can we produce such a simple enumeration of Z^+ like the identity function.

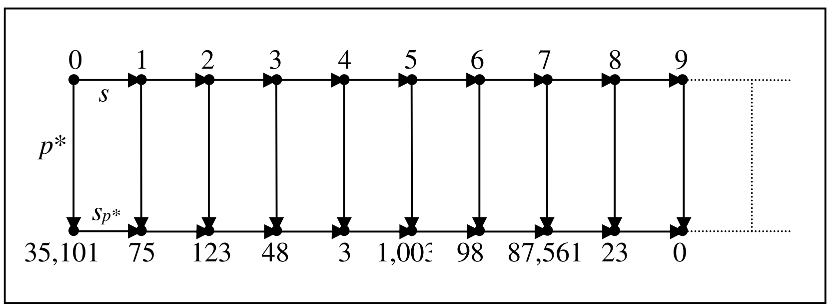

FIGURE 1 A hypothetical initial segment of p^*

Thus, summing up the two previous observations, we can describe the situation as follows: (i, DS_2), but not (p^{*-1}, DS_{p^*}), is a *bona fide* representation of DS_2; conversely, (p^*, DS_2), but not (i, DS_{p^*}), is a *bona fide* representation of DS_{p^*}; where, by a *bona fide representation* of a discrete dynamical system $DS = (M, (g^t)_{t\in T})$, I mean a canonic numeric representation $(u, DS_\#)$ of DS, such that the bijection $u: Z^+ \to M$ can be constructed effectively by means of a mechanical procedure that takes as given the whole structure of the state space M, and nothing more. In other words, a *bona fide* representation of *DS* is a canonic numeric representation (u, $DS_{\#}$) of DS, whose enumeration[9] u: $Z^+ \rightarrow M$ is effective with respect to the structure of the state space M. Let us stipulate that the term [23] *intrinsic representation* of *DS* is a synonym for *bona fide* representation of *DS*.

Note that, as it stands, def. 23 is not formally adequate, for I have not precisely defined the idea of an enumeration u: $Z^+ \rightarrow M$ that is effective with respect to the structure of the state space M. However, a precise definition of the intuitive idea of an intrinsic representation of a discrete dynamical system requires some further general concepts of dynamical systems theory and graph theory, as well as the new notion of an *enumerating machine*, *i.e.*, a machine that effectively produces an enumeration of the state space by moving from state to state in accordance with the state transitions determined by the dynamics of the system. These developments go beyond the scope of the present work, and will be the subject of a forthcoming paper.

---

1 In previous works (Giunti 1992, 1995, 1996, 1997, 1998), I required that the state space of the representing system be recursive, and not just recursively enumerable. From an intuitive point of view, the recursivity of the state space may seem too strong, for the important issue is that there exist an effective procedure for generating all the numbers that represent states of the system, and not that can we decide whether an arbitrary number stands for some state. In effect, however, it does not matter which of the two requirements we choose, for the two corresponding definitions of computational system are equivalent. This is an immediate consequence of the theorem of canonic recursive representability (th. 1), and of the fact that the state space of any canonic recursive representation is a recursive subset of the natural numbers Z^+ (because, by def. 15 and 14, such a state space is either finite or identical to Z^+).

2 At the moment, I only consider *denumerable* discrete systems, *i.e.*, discrete dynamical systems with a denumerable number of states. However, the complete definition of an intrinsic representation must also apply to the somehow trivial case of *finite* discrete systems.

3 It is important to keep in mind that T is not a *bare* set, but rather, a set on which we implicitly assume the whole usual structure of, respectively, the (non-negative) integers or the (non-negative) reals. More precisely, we could say that T is the domain of a model $(T, (\sigma_i)_{i \in I})$ of, respectively, the theory of the (non-negative) integers or the theory of the (non-negative) reals.

---

4 The term "discrete dynamical system" is often used as a synonym for "cascade", *i.e.*, a dynamical system with discrete time; see for example Kulenovic and Merino (2002), Martelli (1999), and Sandefour (1990). My use of the term "discrete dynamical system" is in accordance with Turing (1950).

5 In Giunti (1998, note 7) I claimed that the existence of intrinsic non-computational systems can be proved. I am no longer so confident that such a proof can be given.

6 By a *denumerable* (or *finite*) *dynamical system* I mean a system whose state space is denumerable (or finite).

7 The actual proof that if $p \neq q$, then $s_p \neq s_q$ is by *reductio* and by cases. In the first place, under the assumption $p \neq q$, we assume for *reductio* $s_p = s_q$. We then consider the two cases $p(0) \neq q(0)$ and $p(0) = q(0)$. In either case, keeping in mind the observation in the text, a contradiction readily follows.

8 But we may get even more surprised when we realize that this proof also entails the following: $(Z^+, s_{p^*}, p^*(0))$ is a model of Peano's axioms such that its successor function s_{p^*} is not recursive! This means that the property of being recursive is not an arithmetical property of the successor function, where by an *arithmetical property* I mean any property of a numeric entity which is invariant for all (isomorphic) models of Peano's axioms. To put it in a different way: the recursivity/non-recursivity of the successor function seems to depend on the model of arithmetic we choose. (And, quite obviously, this consideration can then be extendend to any other numeric entity to which the property of being recursive/non-recursive applies.)

9 If *DS* is a *finite* discrete system, then the enumeration u is not a bijection from the whole Z^+ to M, but from a finite initial segment of Z^+ to M.

REFERENCES

Arnold, Vladimir I. 1977. *Ordinary differential equations*. Cambridge: The MIT Press.

Kulenovic, Mustafa R. S., and Orlando Merino. 2002. *Discrete dynamical systems and difference equations with Mathematica*. Boca Raton: Chapman & Hall/CRC.

Giunti, Marco. 1992. *Computers, dynamical systems, phenomena and the mind*. Ph.D. dissertation. Bloomington, IN: Department of History and Philosophy of Science, Indiana University. (Microfilm published in 1993. Ann Arbor MI: University Microfilms Inc.)

Giunti, Marco. 1995. Dynamical models of cognition. In Port, Robert F. and Timothy van Gelder, eds. *Mind as motion: Explorations in the dynamics of cognition*, pp. 549-571. Cambridge: The MIT Press.

Giunti, Marco. 1996. Beyond computationalism. In Cottrel, Garrison W., ed. *Proceedings of the 18th annual conference of the Cognitive Science Society*, pp. 71-75. Mahwah, NJ: L. Erlbaum Associates.

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

Giunti, Marco. 1998. Is computationalism the hard core of cognitive science? In Abrusci, Vito M., Carlo Cellucci, Roberto Cordeschi and Vincenzo Fano, eds. *Prospettive della logica e della filosofia della scienza: Atti del convegno triennale della Società Italiana di Logica e Filosofia delle Scienze, Roma, 3-5 gennaio 1996*, pp. 255-267. Pisa: Edizioni ETS.

Martelli, Mario. 1999. *Introduction to discrete dynamical systems and chaos*. New York: Wiley.

Sandefur, James T. 1990. *Discrete dynamical systems: Theory and applications.* New York: Oxford University Press.

Szlensk, Wieslaw. 1984. *An introduction to the theory of smooth dynamical systems*. Chichister, England:John Wiley and Sons.

Turing, Alan M. 1950. Computing machinery and intelligence. *Mind* LIX, 236:433-460.