Natural Computation: the Cellular Automata Case

Martin Schüle [1]

Abstract. The question what makes natural systems "computational" is addressed by studying elementary cellular automata (ECA), a simple connectionist model of "natural computation". It is shown that ECA which show complicated dynamical and computational behaviour have certain dynamical system characteristics. In particular, it is argued that ECA capable of computational universality are sensitive but not chaotic and thus, in this sense, at the "edge of chaos". Based on the ECA case, the general features natural systems should display in order to be called "computational" are then identified: It is argued that "computation" in nature should show simulation, hierarchy and universality features, where the latter can be characterized to some degree by dynamical system properties.

1 INTRODUCTION

In what sense does nature "compute"? Conventional computers are physical systems that basically implement a universal Turing machine, the standard model of computation. Accordingly, these physical systems, special kinds of natural systems, may be called "computational". When presented with natural systems that have not been explicitly designed for computational purposes it becomes harder to judge whether there are "computational" in some sense. Especially in neuroscience, and more generally in biology, systems are often described in computational terms. However, as soon as we leave the firm basis of standard computation theory, we are left in the dark whether we are dealing here only with some vague, descriptive ideas or with "natural computation" in some more precise sense. Clarifying this issue leads to major conceptual and philosophical problems.

Part of the problem is that "natural computation" seems to be observer-relative. Without an observer, there is no computational task, hence no computation. However, the "observer" should not have the power to make any physical or biological process "computational" simply by observing it. What we are looking for is a notion of "natural computation" which is neither too wide nor too narrow, for in both cases the notion would become vacuous. This implies that both the notion of "observer" and of the "computational" system should satisfy certain constraints. I propose to approach this question by studying concrete *models* of computation. Without a model, we cannot precisely ask nor answer what makes a natural system "computational". Of course, assuming a model of a "real", natural system is prone to oversimplification, nevertheless, we may argue that certain models capture the relevant aspects we set out to investigate at least to some degree.

Now, we seldom encounter natural systems thought to have "computational" capacities that are readily describable by the Turing machine model of computation. It is therefore natural to look for other models of computation that are closer to the kind of systems in nature that are thought to be "computational" in some sense. An important class of such alternative models are the so-called "connectionist models". Connectionist models are basically networks of simple processing units which bring out complex computational processes by the interaction of a large number of units. Especially in neuroscience it is believed that these models may provide an account of the complex, emergent nature of "computation" in natural systems.

A simple, paradigmatic example of a connectionist model of computation are *cellular automata*. Cellular automata consist of networks of simple processing units, called cells, which by interaction among the units can produce complicated behaviour. Cellular automata have several features which make them good models of computational systems in nature, especially biological systems: such networks of cells can be identified in many natural systems, the cells follow simple rules that can be easily postulated for many interactions between cells, the memory of the system is not stored in a specific place but rather consists in the overall or partly configuration of the system and the processing is not by a single processor but rather in parallel by all or parts of the cells of the system. If biological systems do a kind of "natural computation", it is thus likely that at least certain relevant aspects of this "computation" are describable in terms of simple connectionist models such as cellular automata.

Cellular automata bring the further advantage that they can be treated both in terms of the standard theory of computation and dynamical system theory. For example, it can be shown that certain cellular automata are computationally equivalent to universal Turing machines. Whatever is computable by a universal Turing machine is thus computable by these cellular automata. At the same time, cellular automata are dynamical systems and precisely describable in terms of symbolic dynamics, the discrete version of dynamical system theory. This gives us the interesting perspective to combine and compare the computational with the general dynamical system viewpoint.

The general idea is then to describe natural systems by dynamical system theory, the broadest theory possible that deals with the dynamics of systems and then single out, according to certain criteria which will be gained through our study of cellular automata, the class of dynamical systems that show computational behaviour. As we will argue below, the system's behaviour should in particular show some form of computational universality, which in turn implies certain dynamical system properties and certain restrictions in regard to the "observer" of the computation. We thus gain a general approach to characterize "natural computation".

The paper is organized in the following manner. We first introduce a particular simple class of cellular automata, the elementary cellular automata. Then we describe their computational capacities

[1] IHPST, Paris, France, email: schuelem@gmail.com

and introduce some basic dynamical system notions which allow us to describe and classify their behaviour in dynamical system theory terms. We will then find that the class that shows interesting computational features, i.e. actual or possible computational universality, has certain dynamical system characteristics. It is then argued that these features can be postulated as an indication of "computation" for a wider class of dynamical systems. Finally, the philosophical consequences are discussed in view of our account of "computation" in nature.

2 CELLULAR AUTOMATA

Let us now introduce the elementary cellular automata and discuss their computational and dynamical system properties.

2.1 Elementary Cellular Automata

An elementary cellular automaton (ECA) consists of cells that are associated with the sites of a one-dimensional lattice. For the mathematical treatment it is useful to assume an infinite number of such cells, whereas for simulations and applications the number of cells is obviously finite. With a finite number of cells, boundary conditions have to be assumed, usually periodic boundary conditions such as in Figure 1. In the case of ECA, each cell can only be in either of two states, which are usually denoted by 0 or 1. The dynamics of the ECA is generated by the simultaneous interactions of the cells with their nearest neighbours, according to a local transition rule (see Figure 1).

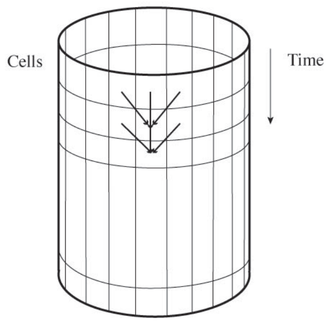


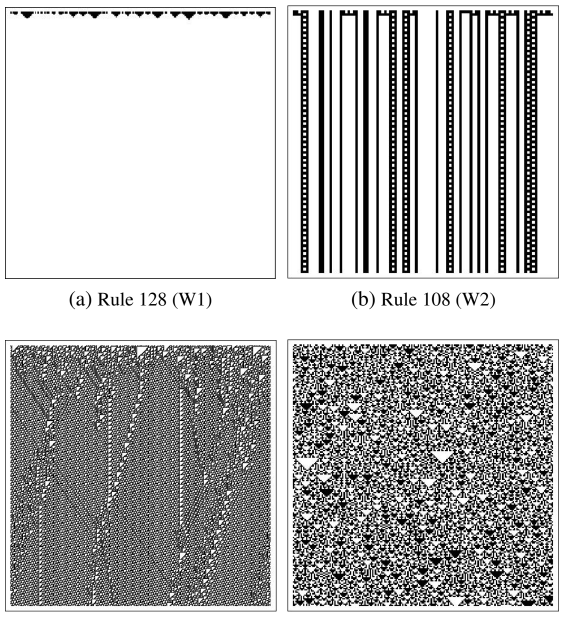
Figure 1: Schematic view of a finite elementary cellular automata with periodic boundary conditions. Cells are updated in parallel by a local rule, indicated by the arrows. Time runs from top to bottom.

The local transition rules of ECA are given in the form of a rule table, such as

111	110	101	100	011	010	001	000
0	1	1	0	1	1	1	0

The upper line shows the 8 possible configurations in the neighbourhood of an ECA cell and the lower line shows the subsequent state or value of the middle cell at the next time step, according to this specific ECA rule. It is the convention, due to Wolfram [31], that rules are referred to by the decimal number which results when the lower line, i.e. the output of the local rule, is read from left to right, in the specific example above one thus speaks of "rule 110". There are 256 such ECA rules, but due to symmetries in the rule tables (left-right and 0-1 complements), there are only 88 independent ECA rules that show different behaviour.

The whole array of cells forms a configuration. The temporal evolution of ECA is then generated by the iterated and simultaneous application of the local transition rule to the states of each cell, which yields a map that acts on the configurations. The resulting *orbit* can be visualised by so-called space-time patterns, as in Figure 2.[2]



(a) Rule 128 (W1)
(b) Rule 108 (W2)
(c) Rule 110 (W4)
(d) Rule 150 (W3)

Figure 2: Space-time patterns of four ECA rules for configurations with periodic boundary conditions. Black dots code state 1, white dots state 0. The initial configurations are chosen randomly. Time runs from top to bottom.

As we can see in Figure 2, ECA show a variety of dynamical behaviour. Of special interest are space-time patterns like the one produced by ECA rule 110 (Figure 2(c)) where localised structures arise from a "random" background and start to interact with each other. The complexity seen in such space-time patterns suggest that we might deal here with a "computational" process. Let us thus discuss now the computational capacities of ECA.

2.2 Universality in Elementary Cellular Automata

Elementary cellular automata (ECA) can easily be seen as computational devices: some input is given, via the initial configuration, which is then processed, via the local transition rule, to an output, given by some final configuration. How complex or universal can this computational process be? Well, for certain CA rules and at least one ECA rule it can be shown that they constitute computing devices that are Turing-universal, i.e., computationally equivalent to a universal Turing machine. Basically, a universal Turing machine is a Turing machine (TM) that can simulate any other Turing machine. Thus, anything computable by Turing machines is computable by a universal Turing machine. Depending on one's understanding of the Church-Turing thesis, one may thus say that there is no computational process devisable that surpasses the possibility of the universal Turing machine. Note also that universality is not only a crucial theoretical concept but also relevant in practise; in essence it is the reason why it is possible to have all-purpose computers [7].

[2] The graphics depicting space-time patterns have been generated with the software *Mathematica* by *Wolfram Research, Inc.*.

We will below argue in more detail that it is essential for any "computational" process to show some form of universality. Let us here first discuss universality in the case of CA. As mentioned, certain CA can be shown to be Turing-universal. However, there are also other forms of universality and there is no complete consensus on how to define e.g. universality in the CA case. Let us thus briefly look at some proposals regarding CA.[3]

The obvious approach is to simulate an arbitrary TM by a suitable designed CA. This has been done for many instances. For example, Kari [14] exposes a construction which implements a TM in one-dimensional CA. The input is encoded as a finite initial configuration and accepted if the CA reaches a configuration with some cell in an accepting state, i.e. in the state which corresponds to the accepting state of the TM. Under this definition, the CA is universal, if the TM it encodes is universal. Many modifications of this concept are possible.[4] A common form of universality is the following: a one-dimensional CA is universal if the decision problem whether a given finite configuration evolves into the quiescent configuration, that is, a configuration that does not change anymore, is recursively enumerable complete (r.e. complete).[5]

The well-known two-dimensional CA "Game of life" has been proven universal in the above sense [4, 11, 14]. Further, the elementary cellular automata rule 110 has been proven by Cook [5] to be universal in the following sense: For given, non-empty words $u, v, w, x \in \{0, 1\}$, i.e. for some finite strings of 0's and 1's, it is r.e. complete to decide whether the configuration $...uuwvv...$ evolves into a configuration that contains the word x. Because it is assumed that there are, to the left and right of the word w, infinitely often repeated words u and v, thus periodic sequences, the system is also called "weakly universal". At the moment, rule 110 is the only ECA rule known to be universal, though other rules, e.g. rule 54, are suspected to be universal in this sense [34].

There are however also other forms of universality devisable. For example, Delvenne et al. [9, 8] argue for the following generalised definition: A computing machine is pictured as a dynamical system together with an "observer" of that system. It is assumed that the observer is modeled by a finite state machine and that the state of the dynamical system is not exactly known. Thus, what is observed, is a set-to-set mapping effectuated by the dynamical system. The computing machine is then said to be universal, if some property about these sets is r.e. complete.[6] According to Delvenne et al., this concept of universality is to be preferred over a notion of universality relying on point-to-point properties, i.e. configuration-to-configuration properties such as the ones used in the above definitions of universality, because it avoids certain unnatural cases of universality apparent in these cases and the, in practice, unavailable infinite precision assumed therein.

A further, perhaps more natural form of universality in the CA case is *intrinsic universality*.[7] A CA is intrinsically universal, if its space-time dynamics comprises the space-time dynamics of any other CA, if it is rescaled accordingly, that is, after some grouping and shifting of cells and running the CA possibly several steps the same space-time patterns are generated.[8] Intrinsic universality is a different notion of universality than universality in the Turing machine sense: there exist Turing-universal CA that are not intrinsically universal [20]. In fact, it is not known whether the (weakly) Turing-universal ECA rule 110 is also intrinsically universal.

2.3 The Edge of Chaos

Any computational process must be sustained by some underlying dynamical process. Does the computational universality discussed above then necessitate some specific dynamical behaviour? Various authors, most prominently Langton [17], advanced the general thesis that computational capabilities can spontaneously emerge in nature in the vicinity of a phase transition, called the "edge of chaos", i.e., at some critical point between highly ordered and highly disordered behaviour. Such critical behaviour may also be self-induced by tuning the appropriate parameters, in this case one speaks of self-organised criticality [2]. Both concepts can be illustrated with cellular automata (CA).

However, although the "edge of chaos" thesis is intuitively appealing, it is too vague a notion if not made more precise. In the case of CA, this means to give a formal classification of the dynamical behavior of CA that allows to pin down where exactly the "edge of chaos" is to be found. A first classification of the dynamical behaviour of CA based on an analysis of CA simulations and some statistical measures was proposed by Wolfram in the 1980s [31, 32, 33]. According to this classification, CA behaviour falls into the following classes:

- (W1) almost all initial configurations lead to a fixed point configuration,
- (W2) almost all initial configurations lead to a periodic configuration,
- (W3) almost all initial configurations lead to random looking behaviour,
- (W4) localized structures with complex behaviour emerge.

This classes are illustrated in Figure 2.

A mathematically more precise classification than Wolfram's classification can be achieved by using the theory of topological dynamics or symbolic dynamics, i.e., the discrete version of dynamical system theory. In the case of ECA, we were able to give a complete classification of all ECA rules in terms of topological dynamics notions [26]. Let us briefly explain how this works.

The classification uses well-known concepts and notions of dynamical system theory, which we will explain here in non-technical terms. For precise definitions, see [16, 14, 26]. Usually, dynamical systems are studied on metric spaces, where a metric defines a notion of distance. In the case of CA, the so-called Cantor topology is usually assumed, which can be induced by the Cantor metric d_C, defined by

$$d_C(x, y) = \sum_{i=-\infty}^{+\infty} \frac{\delta(x_i, y_i)}{2^{|i|}} \tag{1}$$

where the discrete metric $\delta(x_i, y_i)$ is defined as

$$\delta(x_i, y_i) = \begin{cases} 1 & x_i \neq y_i \\ 0 & x_i = y_i. \end{cases} \tag{2}$$

Here, x_i and y_i mean the values of some cell i of two ECA configurations x and y. In the Cantor metric, two configurations are close if they agree with each other within a region around the origin.

[3] See Ollinger [20] for a survey on universality and CA.
[4] See Kari [14].
[5] This definition of universality follows the definition of universality for Turing machines given by Davis [6]: A machine is universal if and only if its halting problem is recursively enumerable complete. A recursively enumerable problem is complete if every recursively enumerable problem is reducible to it. In particular, the halting problem of an universal Turing machine is recursively enumerable complete.
[6] Delvenne et al. [9]. There, also a more precise definition is given.
[7] This notion of universality was made strong by Ollinger [21, 20].
[8] For more precise definitions see Ollinger and references therein [21].

With this notion of "distance", we can then introduce notions of dynamical system theory, such as sensitivity, chaos, etc. First, we need the concept of an almost equicontinuous and equicontinuous CA. A CA is called almost equicontinuous, if there is at least one configuration of the CA with an orbit such that the orbits of any other configuration nearby will stay close to it. If this is the case for all configurations, the CA is called equicontinous. Such CA can be reliably simulated for any mistake in the simulation will not have a great effect on the overall temporal evolution of the CA. In contrast, for a sensitive CA, there exists a constant ϵ, such that for a given configuration there is a configuration close to it, such that the orbits starting from these configurations will eventually separate by at least ϵ. Unless the initial state is precisely known, the long term behaviour of a sensitive system can thus not be predicted by a simulation of the system. An even stronger form of sensitivity is positive expansivity. For positively expansive CA, the orbits starting at any two different configurations will always eventually differ around the origin. In the case of CA, positively expansive CA are automatically sensitive [16].

These notions can then be used to give a mathematical precise classification of the dynamical behaviour of CA.[9] It can be shown that every one-dimensional CA falls exactly in one of the following classes [15]:

(K1) equicontinuous CA,
(K2) almost equicontinuous but not equicontinuous CA,
(K3) sensitive but not positively expansive CA,
(K4) positively expansive CA.

With these notions, we can also give a precise definition of chaotic behaviour, in the sense of Devaney [10]. As it turns out, for ECA, chaos means that the ECA are sensitive and surjective. A CA is surjective if every configuration has at least one pre-image, i.e., a preceding configuration. If there is no such pre-image, one speaks of a Garden-of-Eden configuration. Thus, surjective CA are the CA with no Garden-of-Eden configurations.

The full classification of ECA according to these notions is carried out in [26] and visualized in Figure 3. In Figure 3, all 88 independent ECA are classified according to their dynamical system properties. In this classification, there is a further subclass of "eventually weakly periodic" ECA. The idea of this class is to single out from the sensitive ECA the ECA that eventually show a periodic, shifting pattern and which are thus, in this sense, not "complex".

In the light of this classification, we can localize more precisely the "edge of chaos", namely in the class of ECA that are sensitive but not chaotic. This class corresponds well to what one would intuitively regard as "complex" given the space-time patterns of ECA. In particular, the ECA rules of Wolfram's class (W4) seem to fall into this class. It would however be necessary to distinguish or exclude further subclasses in this class, such as the sensitive but "eventually weakly periodic" ECA, to understand more fully what makes an ECA rule "complex" or to be at the "edge of chaos".

3 UNIVERSALITY AND ITS DYNAMICAL SYSTEM PROPERTIES

Any natural system which is believed to be "computational" is a *dynamical system* for the simple fact that any kind of "computation" is carried out in time. However, we do not want to count any dynamical system among the "computational" system because the term "computation" would then become vacuous. For example, we suspect that certain models of biological systems show computational capacities

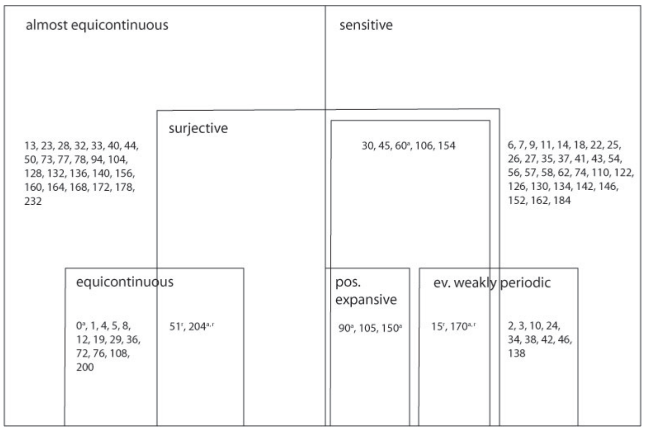


Figure 3: Classification diagram for the elementary cellular automata (ECA). The chaotic ECA are inside the double-framed box. Note that the class of sensitive and eventually weakly periodic ECA is not complete.

whereas e.g. the dynamics of planetary motion does not. Our study of elementary cellular automata (ECA), a paradigmatic class of connectionist models with a variety of dynamical behaviour, showed that the most "complex" rules display dynamics between regular or periodic and chaotic behaviour. It is believed that this kind of complex dynamics is needed to show "computation". In particular, we take as evidence for possibly dealing in this class with computational systems the fact that the Turing-universal rule 110 is in this class.

Why should computational universality, such as Turing-universality, to be taken as evidence of a "computational" system? First, we argue that a certain flexibility of the input-output behaviour is needed to speak of computation. A mere function that maps one value to another value is not of interest to computation. What is wanted is a universal capacity to deal with different, in the best case all posable problems. Second, and related to the first point, demanding a form of universality excludes cases which we don't regard as kinds of computation, such as planetary motion. If we argue for a weaker notion of computation, basically any dynamical system, i.e. any physical system with temporal dynamics, is prone to become "computational". Third, this approach agrees well with standard computation theory insofar as the Turing machine model is a model of computation because it allows for a notion of universality. As said, this is not just a theoretical issue, but also a practical one: the possibility of the all-purpose computer is grounded in the universal Turing machine model [7].[10] Fourth, although universality is a strong condition, it can come in different forms, as already seen in the CA case, thereby possibly allowing to take into account the behaviour of more "natural" systems than the constructed Turing machine model.

Seeing universality as a sign of "computation" raises several issues. First, as said, various notions of universality are devisable, as we have already seen in the CA case. It therefore takes further research to study the overlap and differences of different notions of universality. Second, it restricts somehow the role of an "observer" for the following reasons: When proving universality for CA and similar models it is important to realize that there is the actual, dynamical process which is to be proven universal, but there is also the encoding of the input instances to that process and the decoding or reading

[9] The classification was introduced by Gilman [13] and modified by Kurka [15].

[10] As long as the memory of the computer is not exhausted, which is practically always the case, conventional computers can be seen as implementations of a universal Turing machine.

out of the output, once the process has halted in some sense. Typically, the input to the systems or models are coded in such a way that they become reducible to models already known to be computationally universal. Now, the process of encoding and decoding can also be seen as a computational process. Then, this process must be of a simpler computational nature than the actual, dynamical process, such as the evolution of a CA. If not, all the computational power lies in the encoding and decoding process, whereas e.g. the CA dynamics can become completely trivial. In particular, the encoding and decoding process should not be universal itself. In the case where universality in the Turing machine sense is to be proven, this means that the model or automata describing the encoding and decoding process must be lower in the standard automata hierarchy. In more general terms it means that the computational capacities of the "observer" should be weaker compared to the computation that is being "observed".

Finally, computational universality in the Turing-machine sense, or some other sense, may only be possible if some conditions on the dynamical behaviour of the underlying system are met. Part of the problem to clarify this relation is that there is, as mentioned before, no unanimous accepted definition of computational universality for models such as CA. To different definitions of universality there might thus correspond different dynamical system properties. Despite this fact, we conjecture that, in the ECA case, sensitivity and non-chaoticity are necessary conditions of universality. This may also hold in the more general case, but further research is needed in this direction. Let us briefly point out some issues. In contrast to our findings in the ECA case, Wolfram conjectured that, for example, ECA rule 73, which is not sensitive, may be computationally universal [34]. This difference is due to the fact that our results hold generally for ECA, without any restrictions on the initial conditions, whereas Wolfram considers specific sets of initial configurations on which the rule acts. On such a restricted set of configurations, ECA rule 73 might indeed be sensitive. Also, we do not allow for "filters", such as mentioned by Mitchell et al. [19], that would filter out the complex structures out of chaotic dynamics. Further, there can also be some examples of chaotic systems constructed which are universal.[11] We think that for sufficient "natural" settings these examples lose their relevance, but the issue needs further research. Lastly, the problem of noise in natural systems must be addressed. Noise may destroy the delicate conditions to carry out computations; on the other hand, it may well be that certain systems are capable to somehow shield off this noise. Also, it may be the case that certain notions of universality, such as the one proposed by Delvenne et al. [9, 8], allow to take into account the fact that natural systems are exposed to noise.

With these caveats, our findings in the ECA case are in accordance with the "edge of chaos" thesis advanced by Langton [17] and others: It is the systems with neither too simple nor chaotic dynamical behaviour that are the computationally relevant systems for physics and biology. It is particularly important to realize that it is not the chaotic systems, as it is sometimes assumed, but the so-called intermittent systems that are the most complex. Such intermittent systems can also be characterised formally as having the largest complexity in the sense that their behaviour is the hardest to predict [30]. Alternatively, if computation is measured as a reduction of complexity [29], the intermittent systems may then also be said to provide the complexity needed for efficient computations. It is intuitively clear, as pointed out by Langton [17] and others, why this kind of complexity is needed for computation: it provides a balance between the capability to process and to store information, i.e., to have a kind of memory. Sensitivity thereby provides the flexibility to process different inputs, whereas non-chaoticity ensures that there are certain localized, but stable patterns which allow to store information for a certain time.

[11] See e.g. Delvenne et al. [9].

Where are such computational systems to be found in nature? Again, we may take the CA case and say that whenever a system can be modeled by CA, and the corresponding CA is at the "edge of chaos", which seems to mean that the CA is sensitive but not chaotic, the CA is prone to show universality, which is a sign of computation. As mentioned before, the system sustaining computation in this sense could also tune itself to such a critical point where computational properties can emerge spontaneously. Such self-organized criticality [2] has now been found in many natural systems. It seems that biological systems are the most likely candidates for such behaviour, for the reversibility and thermodynamic equilibrium usually assumed in physical systems hinder the emergence of the kind of dissipative structures that are needed here.

4 CONCLUSION

Let us summarize. Any natural system showing "computation" is a dynamical system, because there must be an underlying, dynamical process sustaining the "computation". We may thus take the broad class of dynamical systems, which are described by the well-developed theory of dynamical systems, and ask which systems, out of all dynamical systems, are the "computational" ones. A convenient way to study this question is by looking at cellular automata (CA), which are networks of simple processing units capable to produce the complicated dynamical behaviour expected to provide for computational processes. As we have argued that computational universality, in the Turing machine or some other sense, is a sign of a computational system, we are especially interested in CA that show a dynamical behaviour which can be shown to allow for computational universality. For the particular simple class of the elementary cellular automata (ECA), it turns out that the ECA rules with the most complex behaviour are sensitive but not chaotic, in the dynamical systems theory sense. It is in this class that the ECA rule known to be Turing-universal, rule 110, is to be found. With certain caveats, these findings agree well with the "edge of chaos" thesis advanced by Langton [17] and others, which says that the systems with dynamics between simple periodic and chaotic behaviour are the ones capable of bringing out emergent computational processes.

Building on this case, we may now generalize these findings to the claim that "natural computation" needs to show the following characteristic features: simulation, hierarchy and universality. Let us expand on this further.

By simulation, we mean that computation has a representational character. It needs signs to represent objects, events or processes. These signs or symbols are then processed by the computational process, thereby simulating another process. The notion of sign or symbol is thereby taken in a wide sense. The case of analog computation may serve here as an example. Typically, the problem is to solve a differential equation. The differential equation describes a physical system, for example planetary motion. The analog computer then emulates the same differential equation. Physically, this is possible because the laboratory system, i.e. the analog computer, as a physical system, can show the same behaviour, in other words, it is describable by the same equations. While planetary motion is not susceptible to manipulation, the system in the laboratory can be manipulated, thereby allowing to compute the behaviour of the inaccessible system. Computation is about solving problems by simulating the problem with a more accessible system. In order to simulate one system by another, we need a map that relates the two, thus we need a notion of "representation".

With this, the notion of an "observer" comes into play. There must be some entity or process which provides this "representation". In this sense, computation is observer-relative. It is however beneficial to separate here clearly semantical issues from syntactical ones. Although the "observer" provides the semantics for the computational system, the notion of "observer" as a syntactical or computational entity or process must be a weak one. In particular, if there are computational capacities attributed to an "observer", they must be weaker than the observed computation. We call this feature "hierarchy": the process of mapping a problem or process to another problem or process which allows for a simulation must be simpler than the actual process carrying out the simulation or "computation". If this were not the case, we could have trivial processes carrying out very complex computational task, because the difficult task is already carried out when encoding and decoding the computational system, that is, when translating one system to another. Because the notion of "observer" is in this sense restricted, the notion of computation proposed here becomes observer-independent in the sense that the computational systems must display certain "objective" properties, first of all a form of universality.

By universality, we mean that a computational system should display some form of universality in its computational behaviour. A system that simply allows to map one value to another value is not doing a computation but a *calculation*. Such calculating systems abound in nature. For example, one may say that the planets "calculate" their own motions, but they do not compute because they lack the universality that allows to deal with different input or problem instances. Universality means the capacity to deal with different, in the best case all posable problems. As was pointed out in the CA case, there are however different forms or notions of universality devisable, which may bring about that different systems can be called "computational".

A clear case of universality is exhibited by the Turing machine model of computation. The universal Turing machine is capable to simulate any other Turing machine. Universality makes thereby a computational system powerful, at the same time, it brings about its limitation. We may ask the system a question, in a perfectly adequate manner, but the answer cannot be given. This relative nature of computation can also be found with simpler models. For example, the problem of squaring the circle can be clearly formulated in the terms of pure Euclidean geometry, but it cannot be solved within this theory. What the Turing concept distinguishes from these earlier conceptions with their corresponding "unsolvability" theorems is that it is, by the "Church-Turing thesis", thought of as an absolute concept. If this is right, the power of universal representation, as formalised by the concept of the universal Turing machine, entails the power of self-reference and this leads inevitably to principally unsolvable problems. Computation is both a relative and universal notion: it should be universal in respect to the questions that may be asked and may turn out to be relative in respect to the answers it can provide.

Now, not every system is capable of showing universality. As we claimed above, dynamical systems must show certain characteristics to possibly sustain complicated computational capacities such as universality. In particular, based on the ECA case, these systems seem to be at the "edge of chaos", but further research is needed to substantiate this claim.

Let us conclude by briefly putting the notion of computation characterised here in perspective to some of the ideas and concepts of computation proposed in the philosophical literature. In a series of papers [22, 23, 24], Piccinini has argued against the "semantic view of computation", attributed e.g. to Fodor [12], and stressed a "functional" and "mechanistic" view on computation. I agree, for the reasons stated above, that it is reasonable to understand computation in syntactical terms. However, Piccinini's account seems to boil down to a vindication of the standard models of computation, such as finite state machines and Turing machines, thereby explicitly excluding other models of computation, such as analog computers [22] or neural networks that have continuous-valued input and outputs [24]. Such an understanding of "computation" seems to me both too wide and too narrow: too wide, because it attributes computational capacities to very simple process, which merely calculate but not compute according to the view developed here, and too narrow, because the characteristics of computation we have identified above may also apply to e.g. analog models of computation or neural networks with continuous-valued input.

On the other hand, Shagrir maintains that computation is a relative notion that is evoked to explain how certain systems, such as the brain, can perform "semantic tasks" [27, 28]. Shagrir substantiates this view by a concise analysis of the practise in the brain and cognitive sciences to explain "computational" features by "information-processing mechanisms". In this account, planetary systems, stomachs, etc. do not compute because the processes involved are not defined in terms of representational content [27]. At the same time, the concept of analog computation fits neatly into this framework, as it exemplifies the modeling aspect of computation, that is characterised, for Shagrir, by the isomorphic nature of the representation function with respect to the (computational) modeling functions and the mathematical relations modeled [28]. While I agree with Shagrir that the notion of computation in neuroscience usually involves semantic aspects, I think that it is reasonable, as explained above, to distinguish the syntactic and semantic aspects of computation. I reserve the notion of computation for the former, whereas the latter should be covered by the analysis of notions such as "representation", "information" or "sign" which I intend to discuss elsewhere. It seems to me that the modeling aspect of computation in Shagrir's account belongs to the syntactic side, whereas the question of representational content is ultimately a semantical issue, involving general semiotic and epistemological problems which can to a certain degree be treated separately to the question of "natural computation".

ACKNOWLEDGEMENTS

This contribution is building on work in my unpublished thesis [25]. I thank Nazim Fatès for helpful comments.

REFERENCES

[1] J. Albert and K. Culik II, 'A Simple Universal Cellular Automaton and Its One-Way and Totalistic Version', *Complex Systems*, **1**, 1–16, (1987).

[2] P. Bak, C. Tang, and K. Wiesenfeld, 'Self-Organized Criticality: An Explanation of the 1/f Noise', *Physical Review Letters*, **59**(4), 381–384, (1987).

[3] E.R. Banks, 'Universality in Cellular Automata', in *IEEE Conference Record of 1970 Eleventh Annual Symposium on Switching and Automata Theory*, pp. 194–215. IEEE, (1970).

[4] E.R. Berlekamp, J.H. Conway, and R.K. Guy, *Winning Ways for Your Mathematical Plays. Vol. 2.*, AK Peters, 2004.

[5] M. Cook, 'Universality in Elementary Cellular Automata', *Complex Systems*, **15**(1), 1–40, (2004).

[6] M. Davis, 'A Note on Universal Turing Machines', *Automata studies*, 167–175, (1956).

[7] M. Davis, *The universal computer: the Road from Leibniz to Turing*, CRC Press, 2012.

[8] J.C. Delvenne, 'What is a Universal Computing Machine?', *Applied Mathematics and Computation*, **215**(4), 1368–1374, (2009).

[9] J.C. Delvenne, P. Kurka, and V. Blondel, 'Decidability and Universality in Symbolic Dynamical Systems', *Fundamenta Informaticae*, **74**(4), 463–490, (2006).

[10] R.L. Devaney, *An Introduction to Chaotic Dynamical Systems*, Westview Press, 2003.

[11] B. Durand and R. Zsuzsanna, 'The Game of Life: Universality Revisited', *Cellular Automata: a Parallel Model*, **460**, 51, (1999).

[12] J.A. Fodor, 'The Mind-Body Problem', *The Scientific American*, (1981).

[13] R.H. Gilman, 'Classes of Linear Cellular Automata', *Ergodic Theory & Dynamical Systems*, **7**(105-118), (1987).

[14] J. Kari, *Cellular Automata: Lecture Notes*, http://users.utu.fi/jkari/ca/, Online; accessed end of 2009.

[15] P. Kurka, 'Languages, Equicontinuity and Attractors in Cellular Automata', *Ergodic Theory and Dynamical Systems*, **17**(02), 417–433, (2001).

[16] P. Kurka, *Topological and Symbolic Dynamics*, Société Mathématique de France, Paris, 2003.

[17] C.G. Langton, 'Computation at the Edge of Chaos: Phase Transitions and Emergent Computation', *Physica D: Nonlinear Phenomena*, **42**(1), 12–37, (1990).

[18] J. Mazoyer and I. Rapaport, 'Inducing an Order on Cellular Automata by a Grouping Operation', *Discrete Applied Mathematics*, **91**(1), 177–196, (1999).

[19] M. Mitchell, P.T. Hraber, and J.P. Crutchfield, 'Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations', *arXiv preprint adap-org/9303003*, (1993).

[20] N. Ollinger, 'Universalities in Cellular Automata; A (Short) Survey', in *Proceedings of the First Symposium on Cellular Automata, 'Journées Automates Cellulaires'*, pp. 102–118, (2008).

[21] N. Ollinger, 'The Intrinsic Universality Problem of One-Dimensional Cellular Automata', *STACS 2003, Lecture Notes in Computer Science*, **2607**, 632–641, (Springer, 2003).

[22] G. Piccinini, 'Computing Mechanisms', *Philosophy of Science*, **74**(4), 501–526, (2007).

[23] G. Piccinini, 'Computation without Representation', *Philosophical Studies*, **137**(2), 205–241, (2008).

[24] G. Piccinini, 'Some Neural Networks compute, others don't', *Neural networks*, **21**(2-3), 311–321, (2008).

[25] M. Schüle, *Natural Computation*, PhD Thesis, ETH Zurich, 2012.

[26] M. Schüle and R. Stoop, 'A Full Computation-Relevant Topological Dynamics Classification of Elementary Cellular Automata', *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **22**(4), 043143, (2012).

[27] O. Shagrir, 'Why we view the Brain as a Computer', *Synthese*, **153**(3), 393–416, (2006).

[28] O. Shagrir, 'Computation, San Diego Style', *Philosophy of Science*, **77**(5), 862–874, (2010).

[29] R. Stoop and N. Stoop, 'Natural Computation Measured as a Reduction of Complexity', *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **14**(3), 675–679, (2004).

[30] R. Stoop, N. Stoop, and L. Bunimovich, 'Complexity of Dynamics as Variability of Predictability', *Journal of Statistical Physics*, **114**(3), 1127–1137, (2004).

[31] S. Wolfram, 'Cellular Automata', *Los Alamos Science*, **9**(2-21), 42, (1983).

[32] S. Wolfram, 'Statistical Mechanics of Cellular Automata', *Reviews of Modern Physics*, **55**(3), 601–644, (1983).

[33] S. Wolfram, 'Universality and Complexity in Cellular Automata', *Physica D: Nonlinear Phenomena*, **10**(1-2), 1–35, (1984).

[34] S. Wolfram, *A New Kind of Science*, Wolfram Media, 2002.