The Modeling and Complexity of Dynamical Systems by Means of Computation and Information Theories

Robert Logozar
Polytechnic of Varazdin
J. Krizanica 33, HR-42000 Varazdin, Croatia
robert.logozar@velv.hr

Alen Lovrencic
Faculty of Org. and Informatics, Uni. of Zagreb
Pavlinska 2, HR-42000 Varazdin, Croatia
alen.lovrencic@foi.hr

Abstract. *We present the modeling of dynamical systems and finding of their complexity indicators by the use of concepts from computation and information theories, within the framework of J. P. Crutchfield's theory of ϵ-machines. A short formal outline of the* ϵ-machines *is given. In this approach, dynamical systems are analyzed directly from the time series that is received from a properly adjusted measuring instrument. The paper serves also as a theoretical foundation for the future presentation of the DSA program that implements the ϵ-machines modeling up to the stochastic finite automata level.*
Keywords: *modeling from a time series, stochastic finite automata, deterministic and statistical complexity, ϵ-machines, DSA program.*

1 Introduction

In this work we illustrate an innovative approach in which computation theory[1] and information theory are used in analyzing and modeling of dynamical systems. These new and highly successful theories and their concepts are used for investigating the problems that belong in the domain of natural sciences. The basic idea is to build models "solely" from the data that are received from the observed systems. Specifically, here we expose the J. P. Crutchfield's theory of ϵ-machines in which hierarchically more and more complex computation entities are introduced [1]. The fundamentals of the theory applied to the chaotic systems were introduced as early as 1989 [2], and matured ever since among a group of physicists and computer scientists interested in describing the structure and complexity of natural systems in a new way. Instead of stating differential equations and using mathematical tools, we analyze the process directly from a time series emitted by a suitably adjusted measuring instrument. The model is presented by using automata with added statistical indicators, such as *Stochastic Finite Automata* in our case, or *String Production* machines on the higher level.

Another well known example of the use of computation modeling for the *spatial* dynamical systems is *cellular automata*, the computation entities that operate in 2 or more dimensional spaces [3]. These new concepts led Stephen Wolfram to postulate his Principle of Computational Equivalence, which can be shortly stated as: *All processes, whether they are produced by human effort or occur spontaneously in nature, can be viewed as computations. Many underlying natural systems, with the exception of those which are obviously simple, can perform computations up to a maximal, universal level of computational power, and are equivalent in that sense* [4].

The modeling from a time series and the theory of ϵ-machines were invented primarily to explore the nonlinear and chaotic dynamical systems, but can serve also as a general modeling scheme for a much broader range of processes that are generated by natural, technical and human systems. The new concepts emanating from the computation and information theory can shed new light on the properties and behavior of dynamical systems and thus affirm the above stated Principle. Although the theory of ϵ-machines is now two decades old, it can still be considered as a novelty. In our opinion, it deserves greater attention of broader scientific community.

Another aim of this paper is to give a theoretic foundation for the presentation of *Dynamical Systems Automata* (DSA) program that is developed to reconstruct the stochastic finite automata from the input binary time series [5]. The description of the program will be given elsewhere.

2 Modeling of dynamical systems

As we have announced in the introduction, the theory of ϵ-machines builds scientific models by seeking for structure within the data itself, and by trying to "understand" the language of a system with

[1] We use the term Computation Theory as a synonym for the Theory of Computation.

imposing as little presumptions as possible. Following the idea of the principle of the computational equivalence, we are estimating the system's computational power and present the resulting model by means of modified automata from the computation theory.

2.1 Epistemological foundation

Before giving the specifics of our approach, let us outline the general features that scientific modeling must provide. These are:

Meaning and Knowledge. The extraction of *meaning* and *knowledge* refers to our ability of recognizing different structures and substructures in the data received from a system, to which we then appoint a certain "meaning". The meaning emerges from the mutual relationships of the extracted structures. For example, if some substructure A leads to a substructure B with certainty, and never leads to some third substructure C, this behavior establishes a certain meaning of A for us. It can be described as: "A means that the following state is B, and never C". If we succeed to systematically analyze the meanings found out from system data, we are on a way to build knowledge about the system.

Compression. The *compression* of data needed to describe a system is essential for the abstraction of the important notions in our modeling. Once we can speak in the "language of the substructures", and not only in the "language of the data" itself, we will be able to describe states and behaviors of the system in a shortened and abstracted form. The compression enables saving of the (storage) space and (processing) time complexity of the calculations needed to reproduce the system generated data, and thus to reduce the model size. The compression is not possible without introduction of meaning and knowledge, and only all of these together make it possible to infer progressively better and better models.

Prediction. The *prediction* of system behavior gives us knowledge on the system future states, based on some initial conditions or presumptions. The ability to predict future states can be assumed as a part of the knowledge about the system, but is stressed separately because of its obvious importance. The scientific prediction is *accurate* in the sense that the error or tolerance margin must be known, and that the model must reproduce the behavior of the real system within that margin.

The above points also reflect the key steps in producing a scientific theory. For the sake of completeness, we shall ad a few remarks to the original epistemological theory [6]. A totally ignorant observer may end up in receiving erroneous or irrelevant data. The necessary epistemological foundation, which obviously cannot be avoided, can be summarized in the three modeling prerequisites:

1. The Basic Knowledge. Some *basic knowledge* about the system under investigation and its broader context must exist, in order to organize a proper experimental setting and to choose a relevant measuring instrument with sufficient precision. Also, the knowledge of broader context is necessary to exclude the effects that may not be a subject of our interest. On the modeling side, this basic knowledge presents the modeler's familiarity with the procedures required for extracting information out of the collected data and for building the model. In short, we cannot conduct a successful modeling from the condition of tabula rasa. We can change the modeling tools from the mathematical to the computational, but we still need to know how to operate with whatever is used. In the same time we try to make our models with as little theoretical bias as possible. The measuring can be started with some elementary experimental setup and the modeling with the simplest computation entities, so that in general we can still begin with rather basic knowledge and simple data processing.

2. Reduction. The reduction is a common principle in scientific modeling that is connected to the Occam's razor. When we decide what we shall investigate in our theory and pick an appropriate measuring instrument to collect data, more often than not we are forced to select what is and what is not relevant for the system itself, or for the aspects of the system we are interested in. This includes the elimination of the side phenomena which complicate our investigation or blur the true picture. For a successful reduction, the basic knowledge of the system and its broader context is needed.

3. Iterative Adjustment. Except providing the initial experimental setting and the basic knowledge on the processing of the received data, we must also provide an adequate feedback to enable iterative adjustment of the measuring instrument and improvement of the modeling scheme if needed. We must track the information quantity of the received data by measuring their entropy, and see if the entropy can be enlarged by adjusting the instrument. The process is not trivial, since we must try to exclude the erroneous measurements and fight the noise introduced into the communication channel, which can all enlarge the total received information. In our case, the instrument adjustment has to do with finding the right partition of the phase space (confer 2.3). Also, if the obtained model diverges, the modeling scheme should be improved to assure proper compression of the model.

It is interesting to note how close the theory of ϵ-machines is to the fundamental questions of philosophy of science. In its genuine attempt to build the model from the collected data itself, it

constantly forces us to question the steps that we make in the process. This is in contrast to the measuring and modeling process used in the established scientific areas, where one seeks the data of predefined physical values within a given theoretical framework. While this classical approach is, by all means, successful when the basic knowledge is relatively big, when the theoretical concepts are clear and the equations mathematically solvable, the proposed computation modeling can be of great interest in the opposite situations — when the basic knowledge is low, when the theoretical framework is unknown and the equations avoid analytical solution, or in other words, when all we have are "raw data" collected from a measuring instrument.

2.2 Dynamical systems

Dynamical systems and their evolution over time are usually described by *iterative processes* [7]. We start with a general deterministic iterative process with Dim dimensions for which the value of a new point $\mathbf{x}(t+\Delta t)$ in time Δt after some moment t is given in terms of the previous point $\mathbf{x}(t)$ as

$$\mathbf{x}(t+\Delta t) = \mathbf{F}(\mathbf{x}(t)),\ t,\ \Delta t \in \mathbb{R}. \tag{1}$$

$\mathbf{F}$ is a vector function that presents the equations of motions of an iterative process on its *phase space*, which is a set $\mathbb{R}_x^{\mathrm{Dim}} \subseteq \mathbb{R}^{\mathrm{Dim}}$ of all possible points $\mathbf{x}(t)$, $\mathbf{x}(t+\Delta t)$. The above expression (1) is known as a *difference equation*.

It is usual to introduce the discrete time and to change the time variable with discrete nonnegative indices, $t \to n,\ \Delta t = 1,\ n \in \mathbb{N}_0^+$. Now the new point $\mathbf{x}_{n+1}$ in the system trajectory is calculated on the basis of the previous point $\mathbf{x}_n$ as

$$\mathbf{x}_{n+1} = \mathbf{F}(\mathbf{x}_n),\ n \in \mathbb{N}_0^+. \tag{2}$$

This simplified form of the difference equation (1) is also known as the *recurrence relation*. At the time $t_0 = 0$ the dynamical system is in its initial point $\mathbf{x}_0$ called the *iteration seed*. By denoting the n-th iteration of the system as $\mathbf{F}^n$, we can calculate the n-th trajectory point as

$$\mathbf{x}_n = \mathbf{F}^n(\mathbf{x}_0). \tag{3}$$

The 0-th iteration $\mathbf{F}^0$ is an identity function $\mathbf{F}^0(\mathbf{x}) = \mathbf{x}$.

For 1-dimensional iterative processes we write the scalar version of (2):

$$x_{n+1} = F(x_n). \tag{4}$$

By knowing the function $\mathbf{F}(F)$, all the equations can be simply implemented on the digital computers. We can also note that the numerical solving of the system's differential (difference) equations is equivalent to the formalism of the iterative processes. As is well known, even the simplest nonlinear differential equations, like the famous logistic map

$$x_{n+1} = r x_n (1 - x_n), \tag{5}$$

cannot be solved analytically [8]. On the other hand, the observation of their iterations on computers is straightforward, and thanks to that the field of deterministic chaos flourished since the 1970s. Furthermore, according to the agenda of the theory of ϵ-machines, we shall make another step forward: the discrete computing and descrete-state machines will be used not only to simulate the evolution of dynamical systems through the above recurrence relations, but also as the basic blocks of their models.

2.3 The Measuring and Modeling Channel

The whole our endeavor of setting an experiment, adjusting the instrument and building a model as a representation of a dynamical system, as described in 2.1, can be viewed in the context of the *Measuring and Modeling Channel* (Fig. 1). In order to make the experimental and modeling process physically realistic, we must make a few generalizations of the description of deterministic dynamical systems in 2.2. The system can generally be nonstationary, and hence dependent on time t or iteration index n. Furthermore, to more realistically depict the real natural and technical systems, we must account for the noise that can be of inherent nature or added in the information channel, here formally presented by ζ_t. Finally, since the goal of our modeling is to find out the system states, instead of points $\mathbf{x}_t$ we shall define the system by its states $\mathbf{X}_t$. The change of states can then be put in correspondence to the trajectory points in the system's phase space. The equation of a general iterative process P that presents a dynamical system in a multidimensional space governed by a vector function $\mathbf{F}$ can now be written as:

$$\mathbf{X}_{t+\triangle t} = \mathbf{F}(\mathbf{X}_t, \zeta_t, t),\ t,\ \Delta t \in \mathbb{R}, \tag{6a}$$
$$\mathbf{X}_{n+1} = \mathbf{F}(\mathbf{X}_n, \zeta_n, n),\ n \in \mathbb{Z}. \tag{6b}$$

In the second equation the time is made discrete in the same way as in 2.2.

As is shown in Fig. 1, the process state $\mathbf{X}_t$ must be found out by the measuring instrument $\mathcal{I}$ as a projection $\mathcal{S}$ of the state $\mathbf{X}_t$ on the *measuring space* $\mathcal{R}^{\mathrm{Dim}}$ (e.g. Euclidean space). The dimension Dim is equal to the number of the experimental probes of the instrument $\mathcal{I}$, which is some *transducer*. As was emphasized in the previous subsection, the instrument is expected to be adjustable during the experiment.

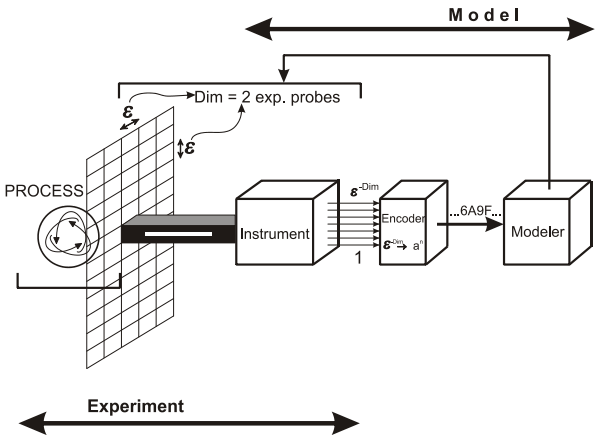

Figure 1: The Measuring and Modeling Channel. The modeling results should enable iterative adjustment of the instrument through the provided feedback (the lines going to the left). After [9].

Depending on the instrument resolution ε, we have a division $\Pi_\varepsilon(\mathrm{Dim})$ of the measuring space on the *set of cells* (*intervals*, *notches*), or simply *measuring signals*:

$$\begin{aligned} \Pi_\varepsilon(\mathrm{Dim}) &= \{\,\pi_i : \pi_i \in \mathcal{R}^{\mathrm{Dim}},\ i \in N_\pi\,\}, \qquad (7)\\ \mathrm{Dim} &\in \mathbb{N},\ N_\pi \subset \mathbb{N}\\ N_\pi &= \{0,1,\ldots,n_{\mathcal{I}}-1\},\ n_{\mathcal{I}} = \ulcorner\, \varepsilon^{-\,\mathrm{Dim}}\, \urcorner\,. \end{aligned}$$

For example if $\varepsilon \approx 10^{-3}$ and $\mathrm{Dim} = 2$, there will be $n_{\mathcal{I}} = 10^6$ cells. Every cell π_i is an equivalence class of the set of states $\mathbf{X}_t$ projected on $\mathcal{R}^{\mathrm{Dim}}$ that are unresolvable for the given instrument. N_π presents the set of all possible indices i, as a set of all possible effective measurement results.

The function of the instrument $\mathcal{I}$ is to find out the *measuring signal interval* or *notch* π_i to which the projection $\mathcal{S}(\mathbf{X}_t)$ corresponds. This can be written as:

$$\begin{aligned} \mathcal{I} &: \mathcal{R}^{\mathrm{Dim}} \to N\,, \qquad (8)\\ \mathcal{I}(\mathcal{S}(\mathbf{X}_t)) &= i\,,\ \text{with}\ \mathcal{S}(\mathbf{X}_t) \in \pi_i\,,\\ i &= 1,2,...,n_{\mathcal{I}}\,. \end{aligned}$$

2.4 Time series

The resulting output of the measuring channel is a suitably encoded value of the index i of the interval π_i. The sequence of such encoded values forms a *string of symbols*, or a *time series*. Thus, instead of tracking the evolution of a dynamical system in its phase space, we read discrete data from the instrument in a form that can be directly analyzed by computation tools like the parse tree.

The set N_π with $n_{\mathcal{I}}$ elements can be interpreted simply as a code alphabet $\mathcal{A}$ with $a = \mathrm{card}(\mathcal{A}) = n_{\mathcal{I}}$ symbols. Thus, the series of measurement values is formally translated into the series of symbols, i.e. the *time series* $\mathbf{s}$:

$$\mathbf{s} = s_0 s_1 \ldots s_i \ \ldots\,,\ \ s_i \in \mathcal{A}\,. \qquad (9)$$

Within the time series $\mathbf{s}$ the substrings $\mathbf{w}$ can be analyzed as words of some formal language, as will be done in 4. The substrings are taken out of the received strings according to some criteria, and this will result in the investigation of the language transmitted by the process. The alphabet $\mathcal{A}$ can be arbitrary, but the most elementary and the most computationally appealing choice is that of binary alphabet. In order that our instrument transmits directly in the binary form with the alphabet $\mathcal{A} = \{0,1\}$, the measuring partition of a 1-dim phase space should also be binary, with only two measuring intervals π_i, $i = 0,1$. For example, a coarse, binary, partition of the unity interval is $\{\,[0,\varepsilon)\,,\,[\varepsilon,1]\,\}$. If ε is set as the system *critical point* x_c defined by $F'(x_c) = 0$, usually the *generating partition* of the phase space is defined [5] ch. 2, [8]. For the logistic map (5) it would be $\varepsilon \simeq x_c = 1/2$. Now the time series like (9) is equivalent to the *finite* itinerary of a dynamical system.

From the course partition we get a *coarse measurement*. However, if the instrument is well tuned — that is, if ε is adjusted to reflect the generating partition — for large enough number of emitted symbols we get arbitrary accurate trajectory points. In this way the ϵ-machine theory is connected to the symbolic dynamics [7].

2.5 Process as a source of information

The measuring instrument is a source of information which can be information-theoretically analyzed. In connection with the process P we introduce the measure $\mu(\mathbf{X})$ on its configuration space, and the *entropy rate* $h_\mu(\mathbf{X})$ of the random variable $\mathbf{X}$ as a measure of information quantity produced by the process during time τ needed to accomplish the state measurement. The usual choice is $\tau = 1$. The instrument $\mathcal{I}$ that measures the process P represents the information channel through which the information $h_\mu(\mathbf{X})$ is transferred to the model builder. The capacity of that channel presents the maximal information I_P of the process that can be transmitted by the instrument:

$$I_P = H(\mathbf{Pr}(\mathbf{X}))\,. \qquad (10)$$

Here $\mathbf{Pr}(\mathbf{X})$ is the probability distribution:

$$\begin{aligned} \mathbf{Pr}(\mathbf{X}) &= (p_0, p_1, \ldots, p_{a-1})\,, \qquad (11)\\ p_i &= \mu(\pi_i)\,,\ \pi_i \in \Pi_\varepsilon(\mathrm{Dim})\,. \qquad (12) \end{aligned}$$

p_i is the probability that a single measurement from the measuring space $\mathcal{R}^{\mathrm{Dim}}$ results within the interval π_i. The measure μ must be normalized as

a probability distribution. $H(\mathbf{Pr}(\mathbf{X}))$ is the Shannon entropy:

$$H\left(\mathbf{Pr}(\mathbf{X})\right) = -\sum_{i=0}^{a-1} p_i \log_2 p_i \,. \tag{13}$$

For the instrument to transfer all the information about the process, the Shannon theorem for the noiseless coding must be valid:

$$I_P \geq h_\mu\left(\mathbf{X}\right) \,. \tag{14}$$

If the above expression is not valid, the instrument will cause the loss of information or equivocation that is equal to: $h_\mu\left(\mathbf{X}\right) - I_P > 0$. This will be the lowest limit of the noise introduced in the reconstructed model.

To get the information received from the process we observe sequences $\mathbf{x}^l = s_0 s_1 \ldots s_{l-1}$ with l symbols, which are in the context of dynamical systems also called l-cylinders [10]. The information measure for such a source is defined as the entropy of the strings $\mathbf{x}^l$ per one symbol, or as the entropy for the next symbol after the $l-1$ previous symbols. The two quantities are the same for the stationary ergodic processes as $l \to \infty$, and are known as the source entropy rate h_μ [11]:

$$h_\mu = \lim_{l\to\infty} \frac{1}{l} H\left(\Pr\left(\mathbf{x}^l\right)\right) \,, \tag{15a}$$

$$h_\mu = \lim_{l\to\infty} \sum_{\mathbf{x}^{l-1}} \Pr\left(\mathbf{x}^{l-1}\right) H\left(\Pr\left(s_l \,|\, \mathbf{x}^{l-1}\right)\right). \tag{15b}$$

$\Pr\left(\mathbf{x}^l\right)$ denotes the probability distribution for the strings of the length l, and $\Pr\left(s_l \,|\, \mathbf{x}^{l-1}\right)$ denotes the conditional distribution for the appearance of the next symbol $s_l \in \mathcal{A}$, after the string $\mathbf{x}^{l-1}$ of $l-1$ previous symbols. H is the Shannon entropy over each distribution. The entropy rate is the "speed" at which the system emits the information, i.e. the average information per symbol.

3 Complexity measures

The metrics of computation capabilities is generally expressed through the concept of *complexity*. The automaton with greater computation capabilities has greater complexity. The source capable of producing more elaborate and intricate time series is considered to be more complex. If we observe a general object $\mathbf{x}$ described with the vocabulary of some formal language $\boldsymbol{\lambda}$, then we talk about the "presentation of $\mathbf{x}$ by $\boldsymbol{\lambda}$". This can be presented as a scalar product $\langle \mathbf{x}|\boldsymbol{\lambda}\rangle$ of the vectors $\langle \mathbf{x}|$ and $|\boldsymbol{\lambda}\rangle$ from the mutually dual vector spaces, i.e. the spaces which depict a state of the object on one hand, and the words of the chosen language on the other. Our preposition is that the former (x) can be expressed with the latter $(\boldsymbol{\lambda})$. The first can present the measurements of the system, and the second our current choice of the modeling class.

Let $M_{\min}\langle \mathbf{x}|\boldsymbol{\lambda}\rangle$ denote the minimal presentation with regard to the vocabulary $\boldsymbol{\lambda}$. Then the complexity $C\left(x\right)$ is a certain measure of it:

$$C\left(x\right) = ||\, M_{\min}\langle \mathbf{x}|\boldsymbol{\lambda}\rangle \,|| \,. \tag{16}$$

The complexity defined as above depends on the choice of language $\boldsymbol{\lambda}$, or, in other words, on the choice of the automata class applied for the modeling. Thus, for example, the presentation and the corresponding complexity will generally be of different size for a modeling based on the class of stochastic finite automata, and for a modeling based on the stack automata.

3.1 Deterministic complexity

To fix the notion of complexity, a standard representation language or computation machine is needed. The natural choice for the standard computer is *Universal Turing Machine* (UTM). In information theory the *Kolmogorov-Chaitin (K-C) complexity*, also called *algorithmic complexity*, $K\left(\mathbf{x}\right)$ is equal to the length of the shortest program on the UTM that produces the observed $\mathbf{x}$ (or some equivalent of it) as output, and stops the machine. This is written as:

$$K\left(\mathbf{x}\right) = \min_{prog\,:\,UTM(prog)=\mathbf{x}} length\left(prog\right) \,. \tag{17}$$

In our context the K-C complexity is named *deterministic complexity*, because the work of UTM is fully determined, as is the outcome of all the programs run on it. Deterministic complexity has a property that it is small for the time series which are easy to describe algorithmically. On the other hand, for randomly generated symbols, it diverges with the length of the series. The K-C complexity growth ratio converges to the entropy rate (15) for long sequences:

$$\frac{1}{l} \times K\left(s_0 s_1 \ldots s_{l-1}\right) \underset{l\to\infty}{\longrightarrow} h_\mu \,, \tag{18}$$

so that the previous statement about the K-C complexity divergence can be written as:

$$K\left(s_0 s_1 \ldots s_{l-1}\right) \underset{l\to\infty}{\longrightarrow} l \times h_\mu = H\left(\mathbf{x}\right) \,. \tag{19}$$

This is a consequence of the formal equivalence of the K-C complexity theory and the information theory based on the Shannon entropy [11]. The main conceptual novelty of the K-C complexity comparing to the entropy is that its definition is independent of the probability distribution and the system stochasticity. It can be calculated also for the deterministic systems, like computer programs, thus expanding the notion of the information quantity. But, as shown above, for stochastic

systems with internal randomness that cannot be unriddled by observer, the K-C complexity does not bring quantitatively new measure of the system information comparing to the Shannon entropy. For them the divergence of deterministic complexity in (19) may happen despite of the system's possible structural (physical) simplicity. The K-C complexity will, besides the true algorithmic complexity of the object—which generally does not depend crucially on the series length — measure also its stochastic component which is linearly dependent on the number of symbols. For big l the stochastic component of the complexity prevails over the bits contributed by the algorithmic complexity. Thus, if a simple but fully stochastic system, like the coin toss, is run for a long time, it will be immensely complex in the terms of the K-C complexity.

The problem, of course, is in the underlying deterministic concept and the requirement to search for the shortest program that will *accurately* reproduce the series $\mathbf{x}$. In the coin toss there are no deterministic components and the pure randomness prevail (see 3.2 and eq. 20). The end result is that the shortest program needed to reproduce such a series on the UTM will be the series itself, and it diverges with the sequence length.

3.2 Statistical complexity

All dynamical systems that are of interest to us are supposed to be intrinsically unknown. Their "computation program" is yet to be discovered. With a limited length time series we may never get the full knowledge of their underlying mechanisms. A system may be deterministic but very intricate, or can have some inherent source of randomness. Upon that the noise from information channel will be superimposed, introducing more stochasticity in the model. So we can ask the following question: *Can we find a measure of the system structure in which the pure random processes will not be included?*

To answer this, J. P. Crutchfield proposed *statistical complexity* [9, 12]. Its main difference from the deterministic complexity is that the amount of bits used for the computation effort to simulate randomly generated symbols in $\mathbf{x}$ are not accounted for. We can say that we give up from the (futile) effort of describing the random pattern in a precise, algorithmic, manner. But nevertheless, the random patterns are *statistically analyzed* by appropriately modified computer language tools, as are binary parse trees (see 4.1), and are simulated in the resulting model with the causal and statistical accuracy (4.2, 4.3).

There are three intuitively clear postulates that define the statistical complexity C_μ:

1. It vanishes for the trivial periodic systems, as is for example $\mathbf{x}_{fix} = 111\ldots1$, $C_\mu(\mathbf{x}_{fix}) = 0$. Such objects are fully monotone and hence *structureless*. Their statistical complexity is in compliance with the K-C complexity ($K(\mathbf{x}_{fix}) = 0$).

2. It also vanishes for completely stochastic systems $\mathbf{x}_{stoh}$:, so that $C_\mu(\mathbf{x}_{stoh}) = 0$. Such systems lack structure and are "simply" *random* (like the motion of gaseous particles). This is different from the K-C complexity, which diverges for such systems with the string length ($K(\mathbf{x}_{stoh}) \sim l \times h_\mu$).

3. For the systems different from the above two boundary cases, the statistical complexity is generally greater than 0, $C_\mu(\mathbf{x}) > 0$. It measures the system *structure*, and is, again, generally different from the K-C complexity. Furthermore, since we require $C_\mu(\mathbf{x})$ to be a continuos and nontrivial function, it must reach its maximum for certain system parameters in between the two limit cases.

The essential relationship between the deterministic and statistical complexity is given by the following approximate expression [12]:

$$K(s_0 s_1 \ldots s_{l-1}) \approx C_\mu(s_0 s_1 \ldots s_{l-1}) + l \times h_\mu . \tag{20}$$

As it was announced, to obtain the statistical complexity the "stochastic bits" must be subtracted from the K-C complexity, so that only the structurally relevant bits remain. The relationship is shown in Fig. 2 where $K(\mathbf{x})$ and $C_\mu(\mathbf{x})$ are drawn as arbitrary functions of the process stochasticity which is expressed by the entropy rate h_μ. $K(\mathbf{x})$ rises monotonously with h_μ and does not present qualitatively new information. Contrary to that, $C_\mu(\mathbf{x})$ has a maximum between the points of the zero and maximum stochasticity of the process. For the binary alphabet or a two-state process, as shown on the figure, $h_{\mu\,,\max} = 1\,\text{bit}$, which can be considered as *ideal randomness*, or *random oracle*. The maximal complexities for the l-cylinders are $l \times \log_2 a$ bit.

Another suggestive interpretation of the statistical complexity is the following: *the statistical complexity is the minimal quantity of information which is to be gathered in the system's history that is sufficient for optimal prediction of the symbols (bits) in object* $\mathbf{x}$ *with the uncertainty (and the underlying probabilities of prediction error) that is not bigger than the entropy rate* h_μ.

If the system entropy is $h_\mu = 0\,\text{bit}$, it is fully deterministic (case 1 above). No structure needs to be found in the past since the behavior is fully monotonous, so that we can predict without error that the system will stay in the same state forever. On the other hand, for a fully stochastic system like the coin toss (case 2), the entropy rate is $h_\mu =$

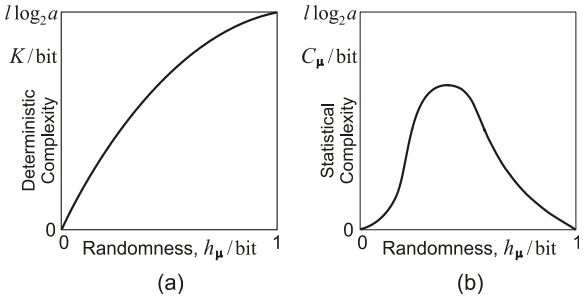

Figure 2: (a) Deterministic (Kolmogorov-Chaitin) complexity $K(h_\mu)$ and (b) Statistical complexity $C_\mu(h_\mu)$ as functions of the stochasticity of the process measured in entropy rate $h_\mu(X)$ [9, 12].

1 bit . The quantity of information that must be gathered in the past to predict the future states of the coin toss with the uncertainty of 1 bit is again $C_\mu(\mathbf{x}) = 0$, because the allowed uncertainty is already the maximal possible for the two-state system. In other words, we can give up searching the structure in the past since there is none. The system is fully random and the best we can do is to guess its state with the prediction error of $\frac{1}{2}$.

The true, structural, complexity lies in between these two extremes (case 3). If, for example, $h_\mu = 0.25$ bit, we must browse through the system's past symbols to find out its statistical complexity. The past should provide us the knowledge to guess the future state with uncertainty of at most $h_\mu = 0.25$ bit . This entropy corresponds to the two-state probability distribution $\Pr(\mathbf{x}) = \mathbf{p}(\mathbf{x}) = (0.0417,\ 0.9583)$, or the inverse one. Our prediction accuracy must be bigger than 0.9583 or, equivalently, the prediction error must be less than 0.0417.

The statistical complexity can be formally connected to the computation theory in the same manner as the K-C complexity. In order to effectively simulate the system randomness, the Turing machine is upgraded by a "guessing unit" for the generation of random symbol. The new machine is called *Bernoulli-Turing Machine* (BTM) and is shown in Fig. 3. It has a "particle-velocity probe" emerged in a heat bath as a genuine source of random numbers. Now we can define the statistical complexity $C_\mu(\mathbf{x})$ of a time series $\mathbf{x}$ as

$$C_\mu(\mathbf{x}) = \min_{prog\,:\,BTM(prog)=\mathbf{x}} length(prog) . \quad (21)$$

It is equal to the length of the shortest program on the BTM which will produce the output equivalent to $\mathbf{x}$. By introducing the vocabulary vectors $|\mathbf{UTM}\rangle$ and $|\mathbf{BTM}\rangle$ for the Universal and the Bernoulli-Turing Machine, we can summarize the expressions for deterministic and statistic complexity:

$$K(\mathbf{x}) = ||\, M_{\min} \langle \mathbf{x}|\mathbf{UTM}\rangle \,|| , \quad (22)$$

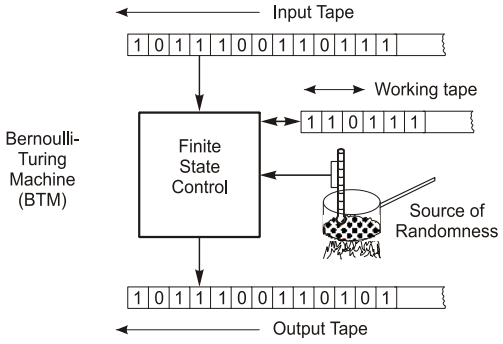

Figure 3: The Bernoulli-Turing Machine is made by adding a source of randomness to the Universal Turing Machine (UTM).

$$C_\mu(\mathbf{x}) = ||\, M_{\min} \langle \mathbf{x}|\mathbf{BTM}\rangle \,|| . \quad (23)$$

The above definitions are not immediately useful in practice except in the simple boundary cases discussed above. Only then it was easy to distinguish between the stochastic and algorithmic system components. The operational expressions for statistical complexity will emanate from the concrete reconstruction model and the corresponding stochastic matrices (sec. 4.4).

We can conclude that the statistical complexity is complementary to the K-C complexity and the Shannon entropy. It brings qualitatively and quantitatively new measure of complexity for dynamical systems, connecting them explicitly to the computationally-based models.

4 Modeling by ϵ-machines

ϵ-machines are hierarchically structured computation models that resemble the original system within an accuracy parameter ϵ. In the modern science influenced by quantum physics, the *experimental error* is not just a defect of our measuring devices or a failure of the observing procedure that could and should be mended somehow. It is a fundamental parameter of every epistemological process. Furthermore, the computation science explicates that every computing effort has its unavoidable error, since it is bounded within the finite space-time frame.

To apply the theory of formal languages and automata theory to the modeling of the real world systems, the following three new features must be added [12, 13]:

i. Stochasticity – to account for the system randomness;

ii. Inductive inferring – to include the invention step during modeling;

iii. Multidimensionality – to enable description of the spatial phenomena, if needed.

The requirement (i) is already met on the axiomatic level by introduction of the BTM, and on the concrete level by the means of *Stochastic Finite Automata* (SFA) [9, 14], with its formal explication in [5]. In short, the SFA are obtained from the standard Finite Automata (FA) by adding the probability for each possible transition. The requirement (iii) is not needed for the modeling from the time series, but in the modeling of the spatial problems. These are generally solved by the concept of *connected cells* [15], and particularly by the cellular automata [3].

To provide the inductive inferring (ii), the reconstruction of ϵ-machines is hierarchically organized in the following two steps.

I. Finding of the SFA as the basic modeling blocks, with the following levels:

0. **Data string.** This is 0-th model level on which received data represent themselves. In practice, we shall always have to limit the number of the received symbols to some $n_s \in \mathbb{N}$. The model size is also n_s (symbols, bits), the same as the length of the time series. There are in total a^{n_s} different sequences.

1. **The parse tree.** The tree of the depth (height) $D \leq n_s$ is built by extracting all the possible words $\mathbf{w}^D$ with D symbols out of the received sequence, and by feeding them into the tree. With the usual restriction to the binary symbol alphabet, the parse tree becomes a special kind of binary tree (see 4.1).

2. **Stochastic Finite Automata.** The SFA are deduced from the trees, by recognizing morphologically and stochastically similar subtrees, called *morphs*, within the parse tree from the level 1. The subtrees are of height L with the usual choice $L \approx D/2$. The size of the model is here denoted as a measure obtained from the tensor T, in which the information on all the system states and their corresponding transitions is contained (see 4.4).

II. Innovative step. If the step I results in a divergent model — despite the use of bigger and bigger computation resources and enlargement of the model (e.g. by increasing the tree depth D) — then it is necessary to make an innovation step and use a higher computation model with infinite memory capacity. However, in accordance to the principle of computational equivalence (sec. 1) and as shown by the modeling of chaotic systems [2], it need not be the UTM. It can be:

3. **String Production Machine** (PM), or a type of **Stack Automaton** (1NnSA, one-way nondeterministic nested stack automaton). These should be deduced from the SFA when the number of the finite memory automata diverges. The size of the model is proportional to the number of the string production rules [12, 13].

4.1 The parse tree

The parse tree is a first computation step in our modeling scheme. It is a tool for recognizing all the different words and for providing the statistics necessary for the next step of finding the SFA.

The binary tree of the fixed depth D is fed by the words of length D or D-cylinders

$$\mathbf{w}^D = s_0 s_1 s_2 \ldots s_{D-1}\,, \tag{24}$$

extracted from the string of n_s symbols, following the ideas already exposed in 2.4 and 2.5 [9]. Fig. 4 in sec. 5 can serve as an illustration. The modeler tries to make the model minimal by choosing $D \ll n_s$. On the other hand, D must be big enough to capture all the structure, that is, longer than the longest correlations in the time series. There are in total $n_{\mathbf{w}} = n_s - D + 1$ words of length D, each starting with the next symbol in the time series.

Together with the empty string e and the word $\mathbf{w}^D$ itself, every word defines $D+1$ prefixes. Each prefix defines exactly one tree node. The prefix e defines the root node at the zero depth, the prefix s_0 defines the left (right) node if the symbol is $0\,(1)$, on the tree depth 1. The prefix $s_0 s_1$ defines one of the four nodes on the tree depth 2, etc..., up to the word $\mathbf{w}^D$ which defines the node on the depth D. All the nodes presented by the prefixes of the word $\mathbf{w}^D$ define the *feed path* $\Theta(\mathbf{w}^D)$. Since our parse tree is always fed with the words of the same length, all the leaves are on the same depth D. There are 2^D possible different words, and the corresponding paths and leaves. Every leaf in the tree can either exist or not exist, so that there are $2^{2^D} - 1$ morphologically different trees of depth D.

During the tree feeding process, if some node on the path $\Theta(\mathbf{w}^D)$ does not exist, it is created and its counter is initialized to 1. If the node already exists, then the counter is incremented. Thus, besides the possible correlations which are recognized as different paths to the leaves and recorded in the morphology of the tree, we also capture a precise statistics of the symbol appearance.

4.2 Causal states

Before moving on to the definition of ϵ-machines, the notion of the *system* and *conditional states* must be explicated. The encoded measurements received from an instrument in the form of a time

series $\mathbf{s} = s_0 s_1 \ldots s_i \ldots$, provide the data from which the model states ξ_i:

$$\xi_i \in \Xi = \{\xi_1, \xi_1, \ldots, \xi_v\}, \tag{25}$$

are to be inferred. Ξ is the set of all found states, with $v = \mathrm{card}(\Xi)$. This is our model representation of the system deterministic and noise components $\mathbf{X}_n$, ζ_n from 2.3. In our modeling up to the level of SFA, the states ξ_i will correspond to some subtrees of the given depth, found within the main parse tree.

The goal of the modeler is to find out:

i. The unknown system states from the series of measurements $\mathbf{s} = s_0 s_1 s_2 \ldots$, by assigning the corresponding automaton states to them;

ii. The possible transitions between the system states by establishing the equivalent transitions between the automaton states;

iii. The probabilities for the state transitions.

The faithful model will be the one in which the assigned automaton states, the transitions between them, and the probabilities for the transitions, correspond well to the real states, or at least simulate them in a way good enough to produce the same or similar (analogous) output strings. Of course, the real states for closed systems may never be truly discovered, but, by comparison of the predictions given by the model to the behavior of the real system we can get the ϵ-estimate of the model quality. This provides a foundation for the necessary rigor in the modeling process.

What follows next is a short review of the crucial definitions of the theory of ϵ-machines: *causal states, equivalent states, morphs, and morphological equivalence of states.* The original elaboration of the topic can be found in [9, 10, 12, 14]. Another formal approach to this is given in [5] ch. 4.

Causal states. For each discrete moment in time t in the observed time series $\mathbf{s} = \ldots s_{t-2}\, s_{t-1}\, s_t\, s_{t+1}\, s_{t+2} \ldots$, we define the *forward string* of symbols $\mathbf{s}_t^{\rightarrow} = s_{t+1}\, s_{t+2}\, s_{t+3} \ldots$, presenting the future of the process, and the *backward string* $\mathbf{s}_t^{\leftarrow} = \ldots\, s_{t-2}\, s_{t-1}\, s_t$, presenting its past. For the backward string $\mathbf{s}_t^{\leftarrow}$, we say that it is a causal state for the forward string $\mathbf{s}_t^{\rightarrow}$.

$\delta-$equivalence of states. Two causal states $\overline{\mathbf{s}} = \mathbf{s}_t^{\leftarrow}$ and $\overline{\mathbf{s}}' = \mathbf{s}_{t'}^{\leftarrow}$ defined by the times $t, t' \in \mathbb{Z}$, are δ-equivalent for some $\delta \in \mathbb{R}$, $\delta > 0$, if their conditional probability distributions are equal within the chosen δ parameter:

$$\begin{aligned} \overline{\mathbf{s}} \underset{\delta}{\sim} \overline{\mathbf{s}}' \quad &\Longleftrightarrow \quad |\Pr(\mathbf{s}_t^{\rightarrow} | \, \overline{\mathbf{s}}) - \Pr(\mathbf{s}_t^{\rightarrow} | \, \overline{\mathbf{s}}')| \leq \delta\,, \\ \forall\, \mathbf{s}_t^{\rightarrow} \quad &\in \quad \mathcal{A}^*. \end{aligned} \tag{26}$$

Morphs. The set $M_{\xi_i}^{\rightarrow}$ of all forward strings $\mathbf{s}_i^{\rightarrow}$ which are possible from the state $\xi_i \in \Xi$,

$$M_{\xi_i}^{\rightarrow} = \{\, \mathbf{s}_{i+1}^{\rightarrow} : \Pr(\mathbf{s}_{i+1}^{\rightarrow} | \, \mathbf{s}_i^{\leftarrow}) > 0 \,\},$$

is called *future morph*, or simply *morph* and is denoted as M_{ξ_i}. The set of all backward sequences $\mathbf{s}_i^{\leftarrow}$ which led to the state $\xi_i \in \Xi$:

$$M_{\xi_i}^{\leftarrow} = \{\mathbf{s}_i^{\leftarrow} : \Pr(\mathbf{s}_{i+1}^{\rightarrow} \, | \, \mathbf{s}_i^{\leftarrow}) > 0 \,\}.$$

is called the *past morph.*

Morphologically equivalent states. The states which in general need not to be δ-equal according to (26), but have the equal structure of nodes, or simply, equal morphs, are *morphologically equivalent*:

$$\xi_i \underset{L}{\sim} \xi_j\,. \tag{27}$$

L is the characteristic length parameter of the morph presentation and its approximation level. It presents the depth of a corresponding subtree within the main parse tree (see 5).

4.3 Formal definition of ϵ-machines

Once we have established all the different causal states — by checking the equivalence relation between them — the temporary evolution of the process is defined by transitions from one state to another. This is described by operator $\mathcal{T}$:

$$\mathcal{T} : \Xi \underset{\mathbf{s}_i^{\rightarrow}}{\longrightarrow} \Xi\,,\ \xi_{t+1} = \mathcal{T}\ (\xi_t\,, \mathbf{s}_i^{\rightarrow})\,. \tag{28}$$

$\mathcal{T}$ is found out by the morph analysis. By reading the conditional probabilities $\Pr(\mathbf{s}_t^{\rightarrow} | \, \mathbf{s}_t^{\leftarrow})$, the full insight into the possible transitions and their probabilities can be obtained. The operator can be connected to the transition tensor $\mathbf{T}$, which is defined by the probabilities for the possible transitions described for the SFA class (see e.g. [9], [5] ch. 3).

The formal definition is now simple: *ϵ-machine is an ordered pair $(\Xi\,, \mathcal{T}\,)_\epsilon$, with parameter ϵ reminding us that it is:*

i. A construction dependent on the measuring instruments and its resolution or accuracy (ε), and the number of experimental probes Dim;

ii. An approximation of the computing structure of the process by means of a certain computation automaton.

4.4 Complexity measures from the modeling results

Here we shall briefly sketch how to find out the complexity measures from the parameters of the reconstructed ϵ-machines. The *state topological complexity* or simply *topological complexity* C_0 of the

process can be taken as a first approximation. It is derived simply from the size of the state set [9, 14]:

$$C_0 = \log_2 \|\Xi\| = \log_2 v \,, \tag{29}$$

where $\|\Xi\| = \mathrm{card}\,(\Xi)$. The topological complexity has the meaning of the average information needed to establish the process state if the states' probability distribution is uniform. According to the theorem of maximal entropy, this is also the maximal information for the set with v members.

The ϵ-machine reconstruction gives us the operator $\mathcal{T}$ and the corresponding transition tensor $\mathbf{T}$ with elements $T^{\,s}_{qq'}$. The elements denote a probability for transition from the SFA state with index q to the state with index q', $q, q' = 1, 2, \ldots, v$, upon reception (emission) of the symbol $s \in \mathcal{A}$, and satisfy the standard normalizing property:

$$T^{\,s}_{qq'} = p_{q \underset{s}{\to} q'} \;,\; \sum_{q'} \sum_{s \in \mathcal{A}} p_{q \underset{s}{\to} q'} = 1 \;. \tag{30}$$

Fig. 6 and 7 in sec. 5 illustrate such SFAs with transitions marked as *directed labeled edges* from one state to another, together with the triggering symbols and transition probabilities. With the usual choice of the binary alphabet the tensor $\mathbf{T}$ is simply two matrices: $T^{\,s=0}$ and $T^{\,s=1}$.

The nonzero elements from $\mathbf{T}$ form the set E of the SFA graph edges,

$$E = \left\{ e : e \sim p_{q \underset{s}{\to} q'} > 0 \right\} \;. \tag{31}$$

From the number of edges we can derive the *transition topological complexity* C^e :

$$C^e = \log_2 \|E\| \;. \tag{32}$$

By summing the tensor elements $T^{\,s}_{qq'}$ over all symbols of the alphabet $\mathcal{A}$, we get the elements $T_{qq'}$ of the *transition matrix* T. The elements are the total probabilities $T_{q\,q'} = p_{q \to q'}$ for a transition $q \to q'$ regardless of the initiating symbol. The analogous matrix operation is summing of the $T^{\,s}$ matrices. This can be summarized as follows:

$$T = [T_{qq'}] \;,\; T_{qq'} = \sum_{s \in \mathcal{A}} T^{\,s}_{qq'} \,,\; T = \sum_{s \in \mathcal{A}} T^{\,s} \,. \tag{33}$$

Obviously, the matrix T presents a Markov process transition matrix of dimension $v \times v$, with the row normalization $\sum_{q'} T_{qq'} = \sum_{q'} p_{q \to q'} = 1$.

Another "rough" complexity measure can be defined from the structure of the machine by disregarding its probability details. In tensor $\mathbf{T}$ we exchange the nonzero elements corresponding to the graph edges with unities, and get the *connection tensor* $\mathbf{T}_0$ with elements:

$$(\mathbf{T}_0)^{\,s}_{qq'} = \begin{cases} 1, & p_{q \underset{s}{\to} q'} > 0 \\ 0\,, & T^{\,s}_{qq'} = 0\,. \end{cases} \;,\; s \in \mathcal{A} \;. \tag{34}$$

From $\mathbf{T}_0$ we can get the *connection matrix*

$$T_0 = \sum_{s \in \mathcal{A}} (\mathbf{T}_0)^{\,s}_{qq'} \;. \tag{35}$$

Now the *connection topological entropy* h_{T_0} is calculated from the principal eigenvalue $\lambda_{\max}$ of T_0 as

$$h_{T_0} = \log_2 \lambda_{\max}\,(T_0) \;. \tag{36}$$

h_{T_0} shows the grow rate of the number of sequences produced (accepted) by our ϵ-machine [9, 14].

The full information-theoretic analysis of our model starts by finding out the full entropy of the reconstructed SFA. This is the *transition* or *edge entropy* $H\,(\mathbf{Pr}_e)$ of the edge probability distribution $\mathbf{Pr}_e = \left(p_1 p_{1 \underset{s}{\to} 1}, \ldots, p_v\, p_{v \underset{s}{\to} v'}\right)$ for every existing edge, where p_q, $q = 1, \ldots, v$, are the state probabilities (to be determined below). Now the edge entropy is [9, 10]:

$$H\,(\mathbf{Pr}_e) = -\sum_{q} \sum_{q'} \sum_{s \in \mathcal{A}} p_q p_{q \underset{s}{\to} q'} \,\log_2\, p_q\, p_{q \underset{s}{\to} q'} \,. \tag{37}$$

This can be expanded to an information-conservation expression:

$$H\,(\mathbf{Pr}_e) = H\,(\mathbf{Pr}_q) + h_\mu \,, \tag{38}$$

where $H\,(\mathbf{Pr}_q)$ is the entropy of the distribution of the SFA states, and h_μ is the source entropy rate (15b). In the terms of our reconstructed machine h_μ is a conditional entropy for the next state q' and the accepted (emitted) symbol s , given the previous state q :

$$h_\mu = -\sum_{q} p_q \sum_{q'} \sum_{s \in \mathcal{A}} p_{q \underset{s}{\to} q'} \,\log_2\, p_{q \underset{s}{\to} q'} \;. \tag{39}$$

The probability distribution $\mathbf{Pr}_q$ that gives the probabilities p_q for the SFA states is most naturally recognized as the *stationary* (asymptotic) *probability distribution* $\mathbf{p}_{stat}$ for the time invariant Markov processes. It is the crucial statistical parameter defined by the condition of stationarity:

$$\mathbf{p}_{stat}\, T = \mathbf{p}_{stat} \,. \tag{40}$$

Here $\mathbf{p}_{stat} = (p_1, p_2, \ldots, p_v)$ is a left eigenvector of T with eigenvalue 1 , with its components being normalized to 1: $\sum_{i=1}^{v} p_i = 1$.

The entropy $H\,(\mathbf{p}_{stat})$ of this distribution was shown to represent the structural complexity of the reconstructed ϵ-machine [2, 10], i.e. the statistical complexity C_μ , as it was introduced in 3.2:

$$C_\mu = H\,(\mathbf{p}_{stat}) \;. \tag{41}$$

C_μ quantifies the information stored in the model's states, or in other words, it gives the amount of the model's memory.

In analogy to the statistical complexity, the full entropy $H(\mathbf{Pr}_e)$ of the model can be named *edge complexity* C_μ^e. Now from eq. (38) we have:

$$C_\mu^e = C_\mu + h_\mu \ . \tag{42}$$

For the Markov chain of states presented by the transition matrix (33), the entropy (rate) $h_\mu(T)$ is

$$h_\mu(T) = -\sum_q p_q \sum_{q'} p_{q\to q'} \log_2 \ p_{q\to q'} \ . \tag{43}$$

It can be shown that the total entropy rate h_μ from (39) can be expressed via $h_\mu(T)$ as

$$h_\mu = h_\mu(T) + h_\mu(\alpha_s) \ . \tag{44}$$

The last term presents the remaining uncertainty from the tensor $\mathbf{T}$, that is, the uncertainty for the same transition $q \to q'$ to be triggered by more than one symbol $s \in \mathcal{A}$, with the averaging being done over all possible state pairs:

$$h_\mu(\alpha_s) = \\ = -\sum_q p_q \sum_{q'} p_{q\to q'} \sum_{s\in\mathcal{A}} a_{q\underset{s}{\to}q'} \log_2 \ a_{q\underset{s}{\to}q'} \ . \tag{45}$$

In (45) $\alpha_{q\underset{s}{\to}q'}$ is the weight factor that connects the probabilities from the tensor $\mathbf{T}$ and matrix T :

$$p_{q\underset{s}{\to}q'} = \alpha_{q\underset{s}{\to}q'} \, p_{q\to q'} \ , \ \sum_{s\in\mathcal{A}} \alpha_{q\underset{s}{\to}q'} = 1 \, . \tag{46}$$

$h_\mu(\alpha_s)$ may by interpreted as the "symbol-per-transition" entropy, with the upper boundary of $\log_2 a$ bit. This entropy vanishes if every transition $q \to q'$, $q, q' = 1, 2, \ldots, v$, is triggered by only one symbol, as is often the case (see sec. 5). A careful reader will note that then the connection matrix T_0 in (35) will have elements not larger than 1.

After eq. 44 the edge complexity (42) can be rewritten as

$$C_\mu^e = C_\mu + h_\mu(T) + h_\mu(\alpha_s) \ . \tag{47}$$

This equation generally expresses the conservation of information quantities, and in our case, it is the conservation of complexity measures and entropy rates.

The modeling by ϵ-machines insists on the minimality of the model. If the reconstruction is successful regarding this criterion, and if the model size $||\,\Xi\,||$ does not diverge with the enlargement of the cylinder length l, then C_μ presents the memory needed for the modeler to predict the process state with the given ϵ-accuracy.

5 A few simple examples

To illustrate the above theoretical concepts and the building of an ϵ-machine up to the SFA level, we start with a simple but quite general process that includes both, a structure with nonzero complexity, and a source of randomness. It will then be contrasted with a few elementary processes which generate the periodic sequences, and the purely random time series.

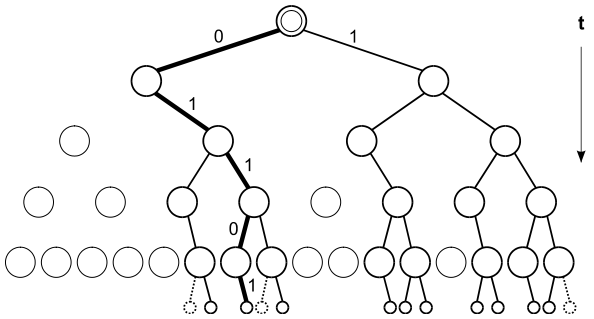

Figure 4: Feeding of the binary tree. The state of the (ϵ) tree of depth $D = 5$, after it was fed by the time series $\mathbf{s} = 01101011011110110101...$, generated by the rule "no consecutive zeros". The paths that are allowed by the rule, but are not traversed by words from the string $\mathbf{s}$ are denoted by dotted lines.

5.1 Structure within randomness

Let us consider the time series $\mathbf{s}$ emitted by a process "generate 1 if 0 preceded, otherwise randomly generate 0 or 1", or shortly "**no consecutive** 0s**",**

$$\mathbf{s} = 01101011011110110101 \ldots \ . \tag{48}$$

Its regular expression is

$$(0 + e)(1 + 10)^* \, , \tag{49}$$

where $+$ denotes the choice of equal probability, and asterisk is the Kleene closure.[2] This and similar time series can be generated by the DSA program introduced in section 1.

Here we shall take the tree depth $D = 5$. From the 20 symbols in $\mathbf{s}$ we can extract in total 16 words of length 5 and parse them through the binary tree of Fig. 4. The first word defines the path $\Theta(01101)$ which is marked with the bold line. The second word 11010 defines the 5-th leaf from the right. The last 5-letter word is 10101. In the true modeling process, much longer symbol sequences would be required in order to obtain sufficient statistical relevance.

The subtrees of depth $L = 2$ will enable tracking of only short correlations, but long enough for the simple rule of our system (Fig. 5). The first subtree or morph growing from the root of the main tree

[2] For some word w consisting of the symbols s from the alphabet $\mathcal{A}$, the Kleene closure is defined as $w^* = \sum_{i=0}^{\infty} w^i$. $w^0 = e$ is the empty string (word), with no symbols. $\mathcal{A}^*$ is the set of all possible words of the alphabet $\mathcal{A}$.

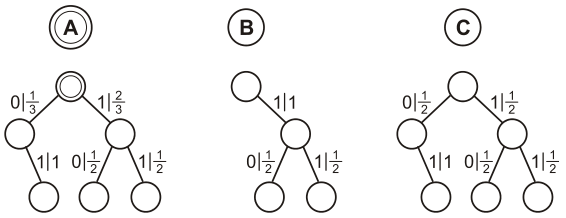

Figure 5: Subtrees of depth $L = 2$ found as unique morphs (states) in the parse tree in Fig. 4. A and C are morphologically equivalent, but with different transition probabilities upon generation of 0 and 1.

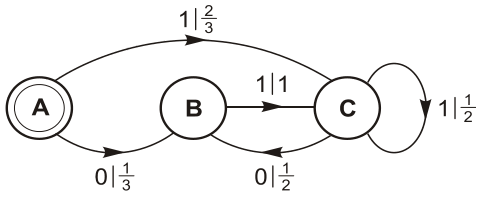

Figure 6: Stochastic Finite Automata for the process "no consecutive zeros". The states are circles, the initial state has additional inner circle. A transition between states is labeled with the symbol that initiates it, and with the transition probability.

is A. As a first morph, it is always unique and it defines a new equivalence class. After emitting 0 with probability 1/3 the system goes from A to B. B has a form different from A, and it also defines a new equivalence class. After emitting 1 with probability 2/3 the process goes from A to a morphologically equal state because the two subtrees of depth 2 are equal in their structure. However, the two morphs have different transition probabilities, as can be proved by a simple argumentation (see e.g. [5] ch. 4). Hence, that is a new state C.

To connect this with the theory from 4.2, let us repeat that each morph defines a class of equivalence for the past-independent causal state from which it emanates, by having determined transition possibilities (the morph shape) and their probabilities. By proclaiming the found morphs (subtrees) to be the system states, the system model in the form of a SFA can be deducted (Fig. 6).

The above model can also be automatically reconstructed by the DSA program. Besides showing the engaged nodes and paths, it provides the full statistics of every node of its parse tree. Also, it enables showing of the found subtrees. If long enough time series are read, the δ parameter from 4.2 can go very small while still exactly reproducing the process (e.g. $\delta < 10^{-4}$, [5] ch. 6).

Formally, we can say that the process "no consecutive zeros" is presented by an ϵ-machine $(\Xi, \mathcal{T})_\epsilon$, with the set of states $\Xi = \{A, B, C\}$, and the operator $\mathcal{T}$, which is fully defined by the corresponding SFA graph in Fig. 6. The system starts in the initial state A corresponding to the subtree A. After 0 is emitted the system goes to the state B, from where only the deterministic transition to C is allowed upon emission of 1. From C the system can emit both symbols with equal probabilities, going either to B after emitting 0, or back to C after emitting 1.

The tensor $\mathbf{T}$ is presented by the two matrices:

$$T^{s=0} = \begin{pmatrix} 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \end{pmatrix}, \; T^{s=1} = \begin{pmatrix} 0 & 0 & \frac{2}{3} \\ 0 & 0 & 1 \\ 0 & 0 & \frac{1}{2} \end{pmatrix}.$$

E.g. the probability for the transition triggered by 0 from A to B is $T_{12}^{s=0} = \frac{1}{3}$, for the transition triggered by 1 from B to C is $T_{23}^{s=1} = 1$, etc.

From the number of states and the number of the SFA edges (the nonzero elements in tensor $\mathbf{T}$), we get the topological complexity C_0 and the transition topological complexity C^e:

$$\begin{aligned} C_0 &= \log_2 ||\,\Xi\,|| = \log_2 3 = 1.5850\,\text{bit}\,, \\ C^e &= \log_2 5 = 2.3219\,\text{bit}\,. \end{aligned}$$

The topological matrix

$$T_0 = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

gives the connection topological entropy h_{T_0} from its maximal eigenvalue of $\lambda_{\max} = \frac{1}{2}\sqrt{5} + \frac{1}{2}$:

$$h_{T_0} = \log_2 1.6180 = 0.6942\,\text{bit}\,.$$

It is easy to note that A is a *transient state*, since there is no transition and no "connection" to its 1st column in $\mathbf{T}$ and T_0 above. The system leaves the state after emitting the first symbol and never returns back to it. Generally, a system leaves the transient states as soon as it "catches the phase". Then it turns into the *recurrent states* that describe its *stationary behavior*. In our case the recurrent states B and C form the set $\Xi_r = \{B, C\}$. The corresponding *recurrent topological complexity* is $C_{0,r} = \log_2 2 = 1\,\text{bit}$.

The statistical properties are summarized in the transition matrix T obtained from eq. (33):

$$T = \begin{pmatrix} 0 & \frac{1}{3} & \frac{2}{3} \\ 0 & 0 & 1 \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}. \qquad (50)$$

Here the information about the triggering symbols is lost, and we track just the state transitions. A straightforward calculation gives the stationary probability distribution (the left eigenvector):

$$\mathbf{p}_{stat} = \left(0, \frac{1}{3}, \frac{2}{3}\right). \qquad (51)$$

Again, as a consequence of A being a transient state, its probability in the stationary probability distribution $\mathbf{p}_{stat}$ is zero: $p_1 = \Pr(A) = 0$. So, we can exclude the state A from the matrix T and present the system with the recurrent state transition matrix:

$$T_r = \begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} . \quad (52)$$

T_r has effectively the same eigenvector: $\mathbf{p}_{stat,\,r} = \left(\frac{1}{3}, \frac{2}{3}\right)$, because the transient states do not contribute to the statistical complexity of the system.

The statistical complexity is now:

$$\begin{aligned} C_\mu(\Xi, \mathcal{T}) &= H(\mathbf{p}_{stat}) = H(\mathbf{p}_{stat,\,r}) \\ &\approx 0.91830\,\text{bit} . \end{aligned} \quad (53)$$

After having found the state probabilities, the edge probability distribution follows:

$$\begin{aligned} \mathbf{Pr}_e &= \left(p_A\, p_{A \overset{}{\to}_0 A}, \ldots, p_C\, p_{C \overset{}{\to}_1 C}\right) \\ &= \left(0 \times \tfrac{2}{3},\, 0 \times \tfrac{1}{3},\, \tfrac{1}{3} \times 1,\, \tfrac{2}{3} \times \tfrac{1}{2},\, \tfrac{2}{3} \times \tfrac{1}{2}\right) \\ &= \left(0,\, 0,\, \tfrac{1}{3},\, \tfrac{1}{3},\, \tfrac{1}{3}\right) . \end{aligned} \quad (54)$$

The corresponding entropy (edge complexity) is:

$$C_\mu^e = H(\mathbf{Pr}_e) = \log_2 3 = 1.5850\,\text{bit} . \quad (55)$$

The transitions emanating from the transient states with zero-probability — in our case the two transitions from the state A — do not contribute to the entropy. The joint state-transition probabilities for the rest 3 transitions from the two recurrent states give a uniform distribution and the complexity C_μ^e that is the same as the topological complexity C_0.

The entropy rate $h_\mu(T)$ follows directly from the transition matrix T (or T_r) according to (43):

$$h_\mu(T) = \tfrac{1}{3} \times H(0,1) + \tfrac{2}{3} \times H\left(\tfrac{1}{2}, \tfrac{1}{2}\right) = \tfrac{2}{3}\,\text{bit} . \quad (56)$$

The symbol-per-transition entropy $h_\mu(\alpha_s)$ from (45) is zero, so that, according to (44) we have:

$$h_\mu = h_\mu(T) \approx 0.6667\,\text{bit} \quad (57)$$

The same result could be obtained from the limes $h_\mu = \lim_{l \to \infty} H\left(\Pr\left(\mathbf{x}^l\right)\right) / l$ (15a). The probability distributions $\Pr\left(\mathbf{x}^l\right)$ on the tree depth l follow from the probabilities of the nodes in the parse tree according to the system rules (or by inspecting them in the the parse tree of the DSA program). From there one calculates the entropies for the l-cylinders. With $l = 1, 2, 3, 4, 5, 6$, the values are (.9183, .7925, .7505, .7296, .7170, .7086) bit, showing the convergence to the value in (56). Similarly, by applying (15b) and by calculating the conditional entropies for the next symbol after the $(l-1)$ previous ones in the same way — that is, directly from the parse tree — the result $h_\mu = 2/3\,\text{bit}$ is obtained immediately for $l \geq 2$, showing that the system has the first order memory. These results are easily verifiable directly from the system rule, showing the consistency of the reconstructed machine.

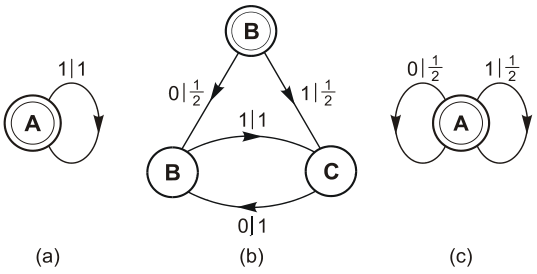

Figure 7: Stochastic Finite Automata for: (a) Period 1 process, (b) Period 2 process, (c) Bernoulli time series.

Now we can confirm the complexity-entropy rate conservation law (42, 47):

$$C_\mu^e = C_\mu + h_\mu = 1.5850\,\text{bit} . \quad (58)$$

In this system the statistical complexity prevails over the entropy rate: $C_\mu > h_\mu$. Namely, the former is close to 1 bit which is the maximal value for the two states. And the latter is diminished by the fact that the second row in T (the first in T_r) is a degenerate distribution with zero entropy because of the certainty in the $B \to C$ transition. Thus the average in (56) is lowered.

By recognizing the deterministic transition, our model correctly distinguished the system's inner rule (organization) from the inherent random processes. It also gave us the statistical complexity $C_\mu \approx 0.92$ bit as a measure of the system's memory. Computationally, we need 1 bit of memory to record the present recurrent state, and statistically, a bit less because of the prevalence of the state C.

5.2 Structure from periodicity

Period 1 process is the simplest periodic process, as was already mentioned in 3.2. It can be described as 0^* (or 1^*), and it generates a fully monotonous sequence $0000\ldots$ (or $1111\ldots$). Such a string defines only one path in the parse tree, filling only the left most (right most) leaf. All the subtrees that originate from the lower nodes are identical to the whole, degenerate, tree. The system has just one morph and the corresponding single state. Also, it is fully deterministic. The SFA of the system is shown in Fig. 7a. The trivial transition tensor and matrix $T = (1)$ lead to zero topological, statistical, and all other complexities and entropies that are listed in Table 1.

Period 2 process is described by the regular expression $(01)^*$ or $(10)^*$ and generates the time series $0101\ldots$ or $1010\ldots$. The subsequences from either

Table 1: The SFA statistics for: (i) periodical dynamical systems with period n, (ii) for the systems with a source of randomness, "no consecutive 0s" [(0+e)(1+10)*], and for the Bernoulli series of random 0s and 1s [(0+1)*]. The complexities and entropies are in bits. "Logaritmus dualis" is abbreviated as $\log_2 = \mathrm{ld}$.

D y n a m i c a l S y s t e m :	Period n				Regular Expression	
Stochastic Finite Automaton Statistics	1	2	3	n	$x(1+10)^*$	$(0+1)^*$
Number of states $v = \|\| \Xi \|\|$	1	3	5	$2n-1$	3	1
Number of transient states $v - \|\| \Xi_r \|\|$	0	1	2	$n-1$	1	0
Number of recurrent states $\|\| \Xi_r \|\|$	1	2	3	n	2	1
Topological complexity C_0 / bit	0	1.585	2.322	$\mathrm{ld}\, v$	1.585	0
Topolog. complex. of recurr. states $C_{0,\,r}$ / bit	0	1.000	1.585	$\mathrm{ld}\, n$	1.000	–
Number of SFA edges $n_e = \|\| E \|\|$	1	4	7	$3n-2$	5	2
Transition topological complexity C^e / bit	0	2.000	2.807	$\mathrm{ld}\, n_e$	2.322	1
Topological entropy h_{T_0} / bit	0	0	0	0	0.694	1
Statistical complexity C_μ / bit	0	1.000	1.585	$\mathrm{ld}\, n$	0.918	0
Markov chain entropy rate $h_\mu(T)$ / bit	0	0	0	0	0.667	0
Symbol-per-transition entr. rate $h_\mu(\alpha_s)$ / bit	0	0	0	0	0	1
Total entropy rate $[h_\mu = h_\mu(T) + h_\mu(\alpha_s)]$ / bit	0	0	0	0	0.667	1
SFA full edge complexity $\left[C^e_\mu = C_\mu + h_\mu\right]$ / bit	0	1.000	1.585	$\mathrm{ld}\, n$	1.585	1

of these fill two paths in the parse tree. There are in total three states corresponding to three morphologically different morphs (subtrees): the first growing from the root node, and the other two growing from the left and the right node (Fig. 7b). After the model catches the phase (decides which of the two series it receives), it leaves the transient state A and oscillates deterministically between the two recurrent states B and C. Its topological state and edge complexities are $C_0 = \log_2 3 = 1.5850\,\mathrm{bit}$, $C^e = \log_2 4 = 2\,\mathrm{bit}$. The transition matrix

$$T = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{59}$$

leads to the left eigenvalue $\mathbf{p}_{stat} = \left(0, \frac{1}{2}, \frac{1}{2}\right)$ and the statistical complexity $C_\mu = 1\,\mathrm{bit}$. The reduced topological complexity for the recurrent states is the same, $C_{0,r} = 1\,\mathrm{bit}$. The full edge complexity is also $C^e_\mu = C_\mu = 1\,\mathrm{bit}$, since the entropy rate $h_\mu = 0$. The topological entropy is $h_{T_0} = 0\,\mathrm{bit}$ because the topological matrix T_0 has the principal eigenvalue 1. The vanishing h_{T_0} entropy tells us that the growth rate of the number of sequences emitted from the system is zero, since there are just two different "starting-phase versions" of one and the same periodic sequence. All these results can be easily verified by following the procedure given for the first example.

The analogous conclusions are valid for the general period n processes. From them we observe the repeating blocks of n symbols $w_n = (0)_k\,(1)_m$, with k 0s and m 1s, $1 \leq k,\, m < n$, $k + m = n$, in which the symbols can be permuted in such a way that the period is not reduced. The SFA statistics and complexities for this generalized case are presented in the fourth column of the Table 1, after our two simple examples and the period-3 system. The proof of the results and further consideration of the periodic systems will be omitted from here.

5.3 Pure randomness

Our last elementary example is the **Bernoulli series** — the randomly generated 0s and 1s with equal probability, described as $(0+1)^*$ (confer also 3.2). By feeding the words of length D from such a series into the parse tree, all the tree nodes on the same level will have equal probability, and from every node the probability that the next symbol is either 0 or 1 is 1/2. This means that the resulting SFA has just one state as is shown in Fig. 7c. The system topological complexity $C_0 = 0\,\mathrm{bit}$.

The transition tensor can be described with its two matrices $\mathbf{T} = \left(T^0_{11} = \left(\frac{1}{2}\right), T^1_{11} = \left(\frac{1}{2}\right)\right)$, which give the topological matrix $T_0 = (2)$ and the topological entropy $h_{T_0} = 1\,\mathrm{bit}$. So, although there is just one state, the system's growth rate of the number of sequences it can produce is maximal for the binary alphabet.

The transition matrix $T = (1)$ yields the eigenvector $\mathbf{p}_{stat} = (1)$ and the statistical complexity $C_\mu = 0\,\mathrm{bit}$. We see no structure in this system, just the randomness. It is the symbol-per-transition entropy $h_\mu(\alpha_s) = 1\,\mathrm{bit}$ that solely contributes to the entropy rate h_μ and to the full edge entropy C^e_μ of the model because there is just one state, and the Markov chain entropy $h_\mu(T) = 0$. The SFA statistics of this "random oracle" can be compared to the statistics of the "no consecutive zeros" model in the last two columns of the Table 1.

The advanced information-theoretic elaboration of similar and several other examples can be found in the references in [1].

6 Conclusion

Building of a dynamical system model from a time series by means of computation theory machines is an iterative and inductive process. It requires an appropriate measuring instrument which is adjusted in a way to provide the transmission of relevant and sufficient data, coded in a suitable way. Although the basic idea is to build the model from scratch, without many theoretical presumptions, it is obvious that the basic knowledge is needed for both, the instrument setup and adjustment, and for the computation of the model parameters. From the (partial) modeling results the entropy of the received data should be checked. If needed, the measuring instrument should be readjusted to obtain the maximal quantity of information from the system.

The information received in the form of a time series is then used to build the hierarchical ϵ-machine model. The build starts with parsing of the symbol strings through the binary tree of the fixed depth (modeling level 1). From there the stochastic finite automaton follows (modeling level 2) for all the systems with finite internal memory. To enable a proper minimization of the model through finding of the equivalent states that are probabilistically equal within some δ parameter, the modeler must provide sufficient computing resources. Within the framework of ϵ-machines a new complexity measure is introduced — the statistical complexity. Besides that, several other information-theoretic quantities follow from the SFA model.

The theoretical foundations of the ϵ-machines modeling that are laid down in this paper should also serve to the future exposition of the DSA program capabilities and computational aspects.

7 Acknowledgments

The first author wishes to thank J. P. Crutchfield for his valuable comments, and L. Budin for mentoring and supporting this work.

References

[1] J. P. Crutchfield's Web Address: http://www.santafe.edu/~jpc/ .

[2] J. P. Crutchfield, K. Young, Inferring Statistical Complexity, Physical Review Letters 63 (1989) 105-108.

[3] S. Wolfram, Computation theory of cellular automata, *Comm. Math. Phys.,* 96:15, 1984.

[4] S. Wolfram, *A New Kind of Science (www.wolframscience.com)*, Wolfram Media, Inc. (www.wolfram-media.com), Champaign, 2002.

[5] R. Logozar, Modeliranje dinamičkih sustava s pomoću stohastičkih konačnih automata (Modeling of Dynamical Systems by Stochastic Finite Automata), Master Thesis, Faculty of Electrical Engineering and Computing, University of Zagreb, 1999.

[6] J. P. Crutchfield, Is Anything Ever New? (Considering Emergence), in *Integrative Themes*, G. Cowan, D. Pines, and D. Melzner, editors, vol. XIX

[7] R. L. Devaney, *An Introduction to Chaotic Dynamical Systems*, Second Edition, Addison-Wesley, Redwood City, Calif. 1989. *Santa Fe Institute Studies in the Sciences of Complexity*, Addison-Wesely, Reading Massachusetts, 1994.

[8] R. L. Devaney, *A First Course in Chaotic Dynamical Systems*, Addison-Wesley, Reading, 1992.

[9] J. P. Crutchfield, Knowledge and Meaning, in *Modeling Complex Phenomena*, L. Lam and V. Naroditsky, editors, Springer, Berlin (1992) 66 - 101.

[10] J. P. Crutchfield and Karl Young, Computation at the Onset of Chaos, in *Complexity, Entropy and the Physics of Information*, W. H. Zurek. editor, vol. VIII of *Santa Fe Institute Studies in the Sciences of Complexity*, Addison-Wesely, Reading Massachusetts, 1990.

[11] T. M. Cover, *Elements of Information Theory,* John Wiley & Sons, New York 1991.

[12] J. P. Crutchfield, The Calculi of Emergence: Computation, Dynamics, and Induction; in the Physica D (1994) special issue on the Proceedings of the Oji International Seminar: *Complex Systems — from Complex Dynamics to Artificial Reality*, Numazu, Japan.

[13] J. P. Crutchfield, Observing Complexity and the Complexity of Observation, in *Series in Synergetics*, H. Atmanspacher, editor, Springer, Berlin, 1993, 235 - 272.

[14] J. P. Crutchfiled, Semantics and Thermodynamics, in *Nonlinear Modeling and Forecasting,* M. Casdagli and S. Eubank, editors, vol XII of *Santa Fe Institute Studies in the Sciences of Complexity*, Addison-Wesely, Reading Massachusetts, 1992.

[15] F. C. Moon, *Chaotic and Fractal Dynamics*, John Wiley & Sons, New York, 1992.