in **Nonlinear Ocean Waves**, A. Brandt, S. Ramberg,
and M. Shlesinger, World Scientific, Singapore (1992) 190 — 216.

SFI 91–09–034

Discovering Coherent Structures in Nonlinear Spatial Systems

James P. Crutchfield

Physics Department
University of California
Berkeley, California 94720 USA
chaos@gojira.berkeley.edu

Abstract

A synthesis of elementary computation and dynamical system theories leads to a constructive approach to discovering coherent structures in spatial systems and to quantifying a pattern's complexity. The basic technique reviewed here builds probabilistic automata from temporal and spatial data series generated by a simple nonlinear spatial system. In this way, a given pattern's unpredictability and structure are measured by the entropy rate and complexity, respectively, of the "machine" reconstructed from the pattern data. Ancillary remarks indicate how the analysis gives a global view of the high-dimensional state space structures associated with spatial systems and, in particular, the geometry of coherent structure interactions. The bulk of the review, though, emphasizes practical results on inferring coherent space-time structures and on building detectors to track particle-like objects.

also in **Complexity in Physics and Technology**,
R. Vilela-Mendes, editor, World Scientific, Singapore (1992) 1.

1. Coherent Structures

1.1 Between a Dynamical System and a Hard Turbulence

It is a well-worn dilemma that contrasts the apparent unpredictability of chaotic few-degree-of-freedom dynamical systems with the orderly and structured solitons appearing in nonlinear many-degree-of-freedom systems. Progress in mechanics over the last half-century has frustrated the intuitive view of "molecular chaos", that the complication observed in a system's behavior should be proportional to the number of subsystems that it contains. The mathematical description of nature is continually forced to the middle ground, the domain of complexity, from its simplifying extremes. The problem of finding "significant" and useful structure in nonlinear spatial systems is particularly important, but difficult since it too focuses on this intermediate regime. Coherent structures fall in the dynamical spectrum between the low-dimensional, but geometrically-tractable unpredictability studied so widely in recent applications of dynamical systems theory and the statistical simplicity of homogeneous, hard turbulence.

To date there is a veritable zoology of nonlinear waves, solitons, dislocations, solitary waves, vortices, and the like. These structures appear to play a very large role in organizing the range of behavior that spatial systems can exhibit. But how are such coherent structures discovered? Often this is simply by visual inspection of experimental systems or of numerical simulations. A process that is on the whole not objective and, given the robust nonlinearity of the governing equations, not amenable to first-principles analysis.

Similar analytic difficulties are typically encountered in nonlinear dynamical systems. Due to this, efforts in this area have taken a different, "qualitative dynamics" tack. The emphasis in this is a geometric view of solution sets and their identification with behavior classes such as basins, basin separatrices, and fixed point, limit cycle, and chaotic attractors. Decomposing a given system's behavior into this list of objects, the attractor-basin portrait, yields a picture of the global organization of its solutions. Important information is extracted in this way without recourse to exact solution of the equations of motion.[1]

This geometric view when coupled with the realization that apparent unpredictability can be generated by simple, but nonlinear dynamical systems has lead in recent years to the development of "nonlinear modeling".[2] This endeavor has many aspects, but speaking broadly the goal is to discover from given data the underlying geometric structures and possibly infer the equations of motion. It is a marked change from existing exploratory data analysis[3] in that it incorporates qualitative dynamics.

The question arises then of whether the insights into unpredictability gleaned using dynamical systems theory and the more recent results from nonlinear modeling can be adapted to the study of patterns, coherent structures, and the like found in nonlinear spatial systems. The basic observation underlying this is simply that spatial time-dependent systems are, in fact, dynamical systems. So dynamical systems theory applies to them too; in principle, at least.

1.2 Nonlinear Spatial Systems as Dynamical Systems

There are some very basic problems, though, with a literal dynamical system view of spatial systems.

First, a spatial system is seen as a very high- or infinite-dimensional system. The geometric and visualization techniques and many of the analytic methods of dynamical systems theory (DST), however, are useful largely in the situation where one performs a modal decomposition or a normal-form analysis. In doing so, the analyst imposes a host of restrictions on the range of behavior that can be explained; viz., that behavior supported by the space spanned by the assumed basis patterns.

Second, in DST the system's state is a structureless point. There is, for example, no indication of the internal spatial symmetry that constrains possible states.

Third, DST does not, in and of itself, give a notion of pattern, let alone a way to detect and quantify the structure in a pattern. There are, of course, several measures that are used to classify dynamical systems. The most widely used is the dimension of "the" attractor. This is something like the number of "active degrees of freedom". Unfortunately, it is often not clear to what structures these degrees of freedom correspond. The Kolmogorov-Sinai entropy, somewhat more usefully, measures the degree of unpredictability of the dynamical system. It quantifies the diversity of state-space trajectories. It does not, however, indicate the development of structure during spatial pattern evolution. Simply stated, high and low entropy and high and low dimension processes can generate more or less complicated spatial structures.

Finally, the bulk of DST concentrates on stationary processes. There is a particular emphasis on attractors and other invariant sets. In contrast to this, it has been argued that transients, both nonstationary and quasistationary, can dominate a spatial system's behavior for extremely long times. Complicated and unpredictable spatial patterns can give way, for example, after universe-lifetimes to simple, periodic behavior.[4] (Nonstationary transients will be the subject of investigation later on.) Thus, to be more useful in the study of spatial systems DST needs to focus more on transients and the state space structures that control them.

1.3 An Approach

The computational approach that is the subject here addresses these problems directly. In particular, it provides a way to geometrize the high-dimensional state space of a spatial system. The resulting global analysis is expressed in terms of the attractor-basin portrait; as just noted, this is one of the most basic behavioral classifications of DST. It also leads to an explicit representation of "patterns" in spatial data and to a quantitative measure of their complexity in terms of the statistical computation required to produce them. A constructive procedure is given that builds from given space-time data the effective equations of motion. Properties of coherent structures in the data are captured by and can be analyzed via the estimated model. The analysis allows for one to detect and track "particles" and other coherent structures in space-time.

The following is an overview of applying these techniques, "computational mechanics" for short, to nonlinear spatial systems. The outline is as follows. The key idea turns on the ability to build models directly from a data series. Before that can be detailed, the character of the available data and the limitations imposed by the measuring instrument are mentioned. Then the model estimation procedure, machine reconstruction, is reviewed. It consists of two steps. The first is the discovery of generalized states in the data series and the second is the estimation of the dynamic, that is, the state-to-state transitions. Several useful statistics are then defined: the entropy rate, which is a measure of the diversity of patterns in the data stream, and the (finitary) complexity, which is a measure of the amount of memory in the generating process.

With the computation analytic tools laid out, they are then applied to the complicated space-time patterns produced by a nonlinear cellular automaton. Spatial structure, pattern, and domain are defined in terms of the reconstructed machine. In this way, the machine represents the type of computation that the spatial system has embedded in the pattern over time. The global state space geometry is summarized with a few remarks on a formal analysis of the high-dimensional state space in terms of the system's attractor-basin portrait. The emphasis, though, is on the domain and dislocation structures found by reconstruction. The associated machines are used to detect and track domains and dislocations in a chaotic sea. The concluding remarks indicate how the computational analysis could be automated.

1.4 Nonlinearity versus Discovery

Before embarking on the main topics a cautionary reminder is in order concerning the problem of "discovering" structure in a nonlinear system. In the case of spatial systems this discovery problem is particularly acute. On the one hand, this is due to the large number of degrees of freedom and so, also, the rather large amount of data that must be handled. On the other hand, this is due to the human propensity to see order in spatial images where there may be none.

In simple terms, the "discovery" problem of nonlinear modeling is: Have we discovered something in our data or have we projected the newly-found structure onto it by selecting an inappropriate or biased representation? This was the main lesson of attempting to reconstruct equations of motion from a time series:[5] When it works, it works; when it doesn't, you don't know what to do; and in both cases it is ambiguous what you have learned. Even though data was generated by well-behaved, smooth dynamical systems, there was an extreme sensitivity to the assumed model class that completely swamped "model order estimation". Worse still there was no *a priori* way to select the class appropriate to the generating process. This should be contrasted with what is probably one of the more important practical results in statistical modeling: within a model class a procedure exists to find, given a finite amount of data, an optimal model that balances prediction error against model complexity.[6] Despite allusions to the contrary, this "model order estimation" procedure does not address issues of class inappropriateness and what to do when confronted with failure.

One way to address the model class discovery dilemma is hierarchical machine reconstruction. This approach starts at the lowest level of representation, the given discrete data, and builds adaptively a series of models within a series of model classes of increasing computational capability until a finite causal model is found.[7] It has been useful in pointing out some general principles for detecting hierarchies of structure and connections with phase transitions. Both of these undoubtedly will have important implications in the future for spatially-extended systems. The following will take a substantially less abstract view of the modeling process, viz. modeling focussed on a single level of representation. But it will not be forgotten altogether, since when analyzing spatial systems, hierarchical reconstruction suggests a way to "factor out" layers of structure from very complicated pattern data.

With this attempt to mollify the muse of objectivity out of the way, the particular modeling methodology can be introduced.

2. Computational Mechanics

The overall goal is to infer from a series of measurements of a process a model of the generating mechanism. Additionally, the model is to indicate the process's computational structure. This refers not only to its statistical properties, such as the decay of correlation, but also to the amount of memory it contains and, for example, whether or not it is capable of producing the digit string of (say) $\sqrt{3}$. The extraction of these properties from the model determines the utility of the model class beyond mere temporal prediction of the process's behavior.

The first subsection starts off by defining the character of the data from which the model is built. The next two subsections then review machine reconstruction and the statistical mechanics of the inferred stochastic machines. The section closes with some comments on complexity and causality.

2.1 The Measurement Channel

The universe of discourse for nonlinear modeling consists of a process P, the measuring apparatus $\mathcal{I}$, and the modeler itself. The goal is for the modeler, by taking advantage of its available resources, to make the "best" representation of the nonlinear process.

The process P, the object of the modeler's ultimate attention, is the unknown, but hopefully knowable, variable in this picture. And so there is little to say, except that it can be viewed as governed by stochastic evolution equations

$$\vec{X}_{t+\Delta t} = \vec{F}\left(\vec{X}_t, \vec{\xi}_t, t\right) \tag{1}$$

where $\vec{X}_t$ is the configuration, e.g. a finite dimensional state or a spatial pattern, at time t, $\vec{\xi}_t$ some noise process, and $\vec{F}$ the governing equations of motion. The following discussion also will have occasion to refer to the process's probability distribution $\mu\left(\vec{X}\right)$ on its configuration space and the entropy rate $h_\mu\left(\vec{X}\right)$ at which it produces information.

The measuring apparatus is a transducer that maps $\vec{X}_t$ to some accessible states of an instrument $\mathcal{I}$. This instrument has a number of characteristics, most of which should be under the modeler's control. The primary interaction between the instrument and the process is through the measurement space $\mathcal{R}^{\mathrm{D}}$ which is a projection $\mathcal{P}$ of $\vec{X}_t$ onto (say) a Euclidean space whose dimension is given by the number D of experimental probes. The instrument, which distinguishes the projected states to within resolution ϵ, partitions the measurement space into a set $\Pi_\epsilon(\mathrm{D}) = \left\{\pi_i : \pi_i \subset \mathcal{R}^{\mathrm{D}}, i = 0, \ldots \epsilon^{-\mathrm{D}}\right\}$ of cells. Each cell π_i is the equivalence class of projected states that are indistinguishable using that instrument. The instrument represents the event of finding $\mathcal{P}\left(\vec{X}_t\right) \in \pi_i$ by the cell's label i. With neither loss of generality nor information, these indices are then encoded into a time-serial binary code. As each measurement is made its code is output into the data stream. In this way, a time series of measurements made by the instrument becomes a binary string, the data stream s available to the modeler. This is a discretized set of symbols $\mathrm{s} = \ldots s_{-4}s_{-3}s_{-2}s_{-1}s_0s_1s_2s_3s_4\ldots$ where in a single measurement made by the modeler the instrument returns a symbol $s_t \in \mathcal{A}$ in an alphabet $\mathcal{A}$ at time index $t \in \mathbf{Z}$. Here we take a binary alphabet $\mathcal{A} = \{0, 1\}$.

The modeler is presented with s, the bit string, some properties of which were just given. The modeler's concern is to go from it to a useful representation, which will be called a model. To do this the modeler needs a notion of the process's effective state and its effective equations of motion. Having built a model consisting of these two components, any residual error or deviation from the behavior described by the model can be used to estimate the process's effective noise level. This level is determined by the amount of data "unexplained" by the inferred model. It should be clear when said this way that the noise level and the model's sophistication depend directly on the data and on the modeler's resources. Finally, the modeler may have access to experimental control parameters. And these can be used to aid in obtaining different data streams useful in improving the model by (say) concentrating on behavior where the effective noise level is highest.

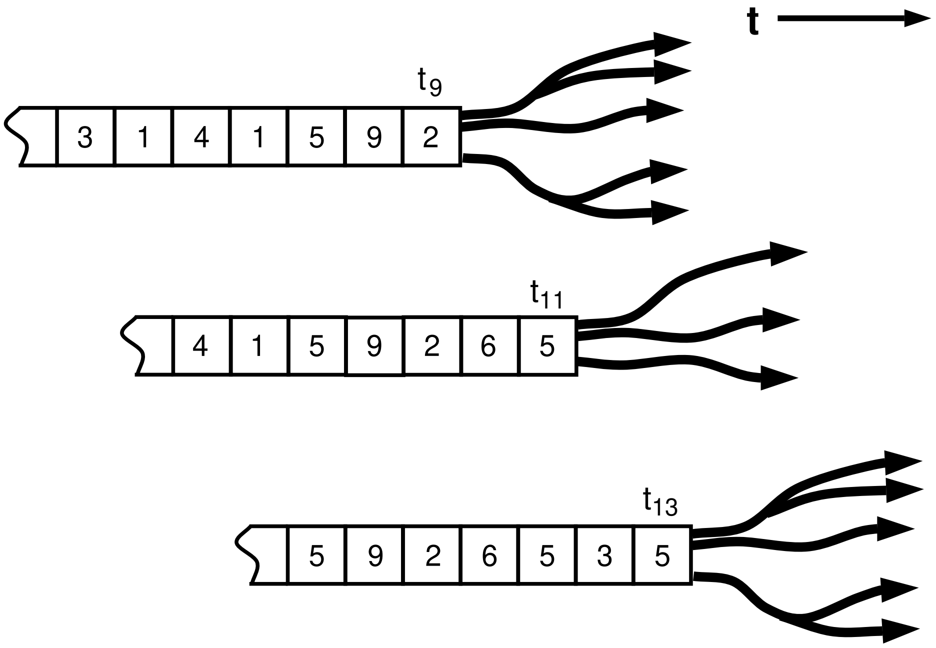

Fig. 1 Morph-equivalence induces conditionally independent states. When the template of future possibilities, i.e. allowed future subsequences and their past-conditioned probabilities, has the same structure then the process is in the same generalized state. At t_9 and at t_{13}, the process is in the same state; at t_{11} it is in another different state.

The central problem of nonlinear modeling now can be stated. Given an instrument, some number of measurements, and fixed *finite* inference resources, how much regularity and computational structure in the underlying process can be extracted?

2.2 Computation from a Time Series

On what sort of structure in the data stream should the models be based? If part of the goal is prediction, then a natural object to reconstruct from the data series is a representation of the process's instantaneous state. Unfortunately, as already noted, individual measurements are only indirect representations of the state. Indeed, the instrument simply may not supply data of a quality sufficient to discover the true states, independent of the amount of data. So how can the process's "effective" states be accessed?

For discrete time series a state is defined to be the set of subsequences that render the future conditionally independent of the past.[8] This notion of state is widespread; appearing in various guises in early symbolic dynamics and ergodic and automata theories. It is close to the basic notion of state in Markov process theory. And so, here, an observer identifies a state at different times in the data stream as its being in identical conditions of ignorance about the future. (See Fig. 1.)

Consider two parts of the data stream $\mathrm{s} = \ldots s_{-2}s_{-1}s_0s_1s_2\ldots$. The one-sided forward sequence $\mathrm{s}_t^{\rightarrow} = s_t s_{t+1} s_{t+2} s_{t+3}\ldots$ and one-sided reverse sequence $\mathrm{s}_t^{\leftarrow} = \ldots s_{t-3}s_{t-2}s_{t-1}s_t$ are obtained from s by splitting it at time t into the forward- and reverse-time semi-infinite subsequences. Consider the joint distribution of possible forward sequences $\{\mathrm{s}^{\rightarrow}\}$ and reverse sequences $\{\mathrm{s}^{\leftarrow}\}$ over all times t

$$Pr(\mathrm{s}^{\rightarrow}, \mathrm{s}^{\leftarrow}) = Pr(\mathrm{s}^{\rightarrow}|\mathrm{s}^{\leftarrow})Pr(\mathrm{s}^{\leftarrow}) \tag{2}$$

The conditional distribution $Pr(\mathrm{s}^{\rightarrow}|\omega)$ is to be understood as a function over all possible forward sequences $\{\mathrm{s}^{\rightarrow}\}$ that can follow the particular sequence ω where ever ω occurs in s.

Then the same state $S \in \mathbf{S}$ is associated with all those times $t, t' \in \{t_{i_1}, t_{i_2}, t_{i_3} \ldots : i_k \in \mathbf{Z}\}$ such that past-conditioned future distributions are the same. That is,

$$t \sim t' \text{ if and only if } Pr(\mathrm{s}^{\rightarrow}|\mathrm{s}_t^{\leftarrow}) = Pr(\mathrm{s}^{\rightarrow}|\mathrm{s}_{t'}^{\leftarrow}) \tag{3}$$

If the source generating the data stream is ergodic, then there are several comments that serve to clarify how this relation defines states. First, the sequences $\mathrm{s}_t^{\leftarrow}$ and $\mathrm{s}_{t'}^{\leftarrow}$ are typically distinct. If $t \sim t'$, Eq. (3) means that upon having seen different histories one can be, nonetheless, in the same state of anticipation or ignorance about what will happen in the future. Second, $\mathrm{s}_t^{\leftarrow}$ and $\mathrm{s}_{t'}^{\leftarrow}$, when considered as particular symbol sequences, will each occur in s many times other than t and t', respectively. Finally, the conditional distributions $Pr(\mathrm{s}^{\rightarrow}|\mathrm{s}_t^{\leftarrow})$ and $Pr(\mathrm{s}^{\rightarrow}|\mathrm{s}_{t'}^{\leftarrow})$ are functions over a nontrivial range of "follower" sequences $\mathrm{s}^{\rightarrow}$.

This gives a formal definition to the set $\mathbf{S}$ of states as equivalence classes of future predictability: $\sim$ is the underlying equivalence relation that partitions temporal shifts of the data stream into equivalence classes. The states are labels for these classes. For a given state S the set of future sequences $\left\{\mathrm{s}_S^{\rightarrow} : S \in \mathbf{S}\right\}$ that can be observed from it is called its future morph. Once these states are found, the temporal evolution of the process is given by a mapping from states to states, $T : \mathbf{S} \rightarrow \mathbf{S}$; that is, $S_{t+1} = TS_t$.

The available model building algorithms infer the states $\mathbf{S}$ via various approximations of the equivalence class conditions specified in (3).[7, 8, 9] These procedures are referred to generically as "machine reconstruction". The result of machine reconstruction, then, is the discovery of the underlying process's "hidden" states. This should be contrasted with the *ad hoc* methods employed in hidden Markov modeling in which a set of states and a transition structure are imposed by the modeler at the outset[10] and common in "transform" analysis using Fourier, Laplace, or wavelet basis functions.

The overall procedure has two steps. This first is to identify the states $\mathbf{S}$ and the second is to infer the transformation T. In the following I review the simplest implementation since this affords the most direct means of commenting on certain properties of the resulting models and derivative statistics.

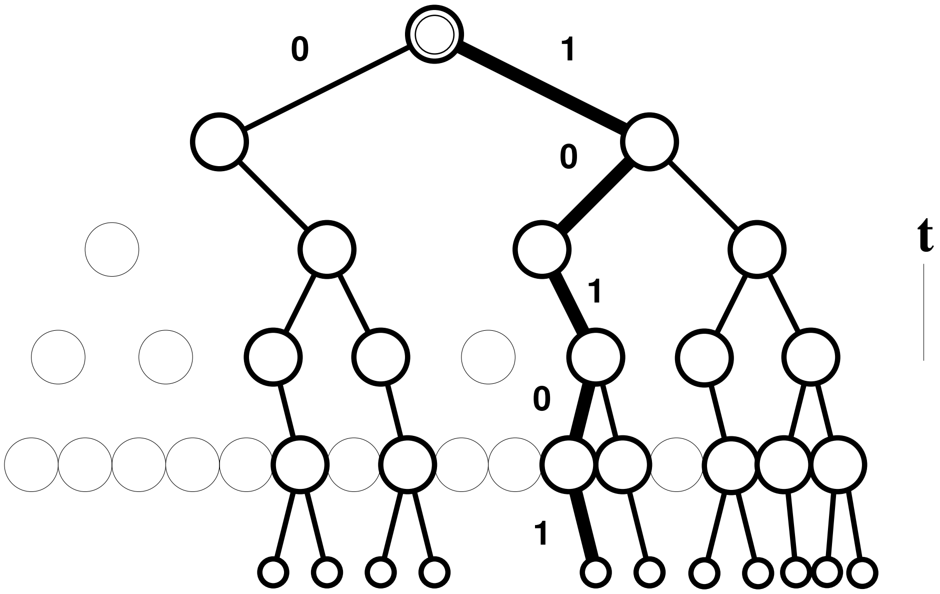

Fig. 2 The parse tree for the process "every other symbol is a 1". The tree is a hierarchical representation of the subsequences produced by the process. Here the tree depth is $D = 5$. If s = $\ldots 1010101110101\ldots$, then the subsequence $s_t^5 = 10101$ will be seen. This is put into the parse tree, as shown by the bold line. Note that time goes down the tree. When counts of the subsequence that lead to a given tree node are accumulated, the tree gives a hierarchical representation of the infinite-sequence probability distribution.

Initially, a parse tree is built from the data stream s. A window of width D is advanced through s one symbol at a time. If s $= \ldots 1010101110101\ldots$ then at some time t the $D = 5$ subsequence $s_t^5 = 10101$ is seen, followed by $s_{t+1}^5 = 01010$. Each such subsequence is represented in the parse tree as a path. The tree has depth $D = 5$. (See Fig. 2).

Counts are accumulated at each tree node as each subsequence is put into to the parse tree. If the associated path is not present in the tree, it is added; new nodes each begin with a count of 1 and counts in existing nodes are incremented. If s has length N, then a node probability, which is also the probability of the sequence leading to it, is estimated by its relative frequency

$$Pr(n) = \frac{c_n}{N - D} \tag{4}$$

where c_n is the count accumulated at node n. The node-to-node transition probabilities $Pr(n \to n')$ are estimated by

$$Pr\left(n \to n'\right) = Pr\left(s|s^k\right) = \begin{cases} \frac{Pr\left(ss^k\right)}{Pr\left(s^k\right)} \approx \frac{c_{n'}}{c_n} & c_n > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where the length k sequence s^k leads to node n and the length $k+1$ sequence $s^{k+1} = ss^k$ leads to node n' on symbol $s \in \mathcal{A}$.

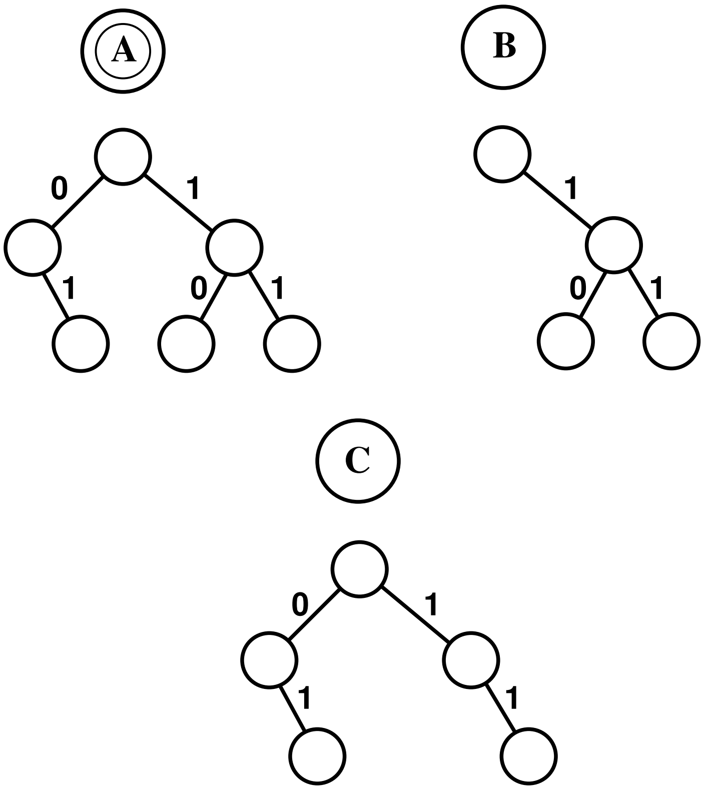

Fig. 3 The morph depth $L = 2$ subtrees found below tree nodes down to depth 3 in Fig. 2's parse tree. Each morph, or subtree, has been labeled by its associated machine state (cf. Fig. 4).

To infer the set of states one looks for a set of similar future-morphs. On the parse tree the latter are distinct-shape subtrees. By way of incremental approximation, the morph similarity is limited to future subsequences of length L. On the parse tree, equivalently, one looks for subtrees of morph depth L. Naturally, $L \leq D$ and typically one takes $2L = D$. Investigating the parse tree of Fig. 2 using depth $L = 2$ subtrees one finds the three morphs shown in Fig. 3. In this way, three "machine" states $\mathbf{S} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$ have been discovered for the "every other symbol is a 1" process of Fig. 2 up to the approximation implied by setting $D = 5$ and $L = 2$. A moment's reflection shows that the latter parameter settings are sufficient to infer the process's correct and minimal model.

The state-to-state transition structure is obtained by looking at how the morphs change into one another upon moving down the parse tree, i.e. upon reading $s = 0$ or $s = 1$. The resulting transformation T is represented graphically by the machine shown in Fig. 4. (Ignore for the moment the transition probabilities mentioned there.) This should be compared to the parse tree (Fig. 2). There the morph below the top node, which is associated with machine state A, makes a transition on $s = 1$ to a tree node with a morph below it in the same equivalence class (machine state A). On $s = 0$, though, the top tree node makes a transition to a tree node with a morph associated with machine state

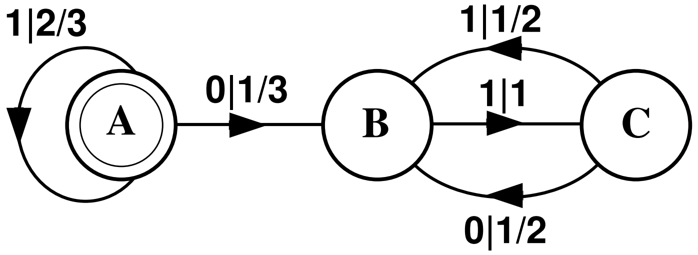

Fig. 4 The reconstructed stochastic machine for the process "every other symbol is a 1". The state with an inscribed circle is the start state, in which the machine begins reading or generating symbols in a string. Each edge is labeled $s|p$ where $s \in \mathcal{A}$ and $p \in [0, 1]$ is the probability of making the transition between the states connected by the edge with label s.

B. In just this way the machine of Fig. 4 summarizes the morph to morph transition structure T on the parse tree.

This exposition only considered topological reconstruction: subtrees were compared only up to the subsequences $\{\overrightarrow{\mathrm{s}}_t\}$ which were observed. Of course, what is needed is a set of states that are not only topologically distinct, but also distinct in probability as indicated by (3). Details for this and for the transition probabilities can be found elsewhere.[8]

2.3 Statistical Mechanics

Machine reconstruction gives then a set of states, that will be associated with a set $\mathbf{V} = \{v\}$ of vertices, and a set of transitions, that will be associated with a set

$$\mathbf{E} = \left\{ e : e \sim v \underset{s}{\rightarrow} v', \quad v, v' \in \mathbf{V}, s \in \mathcal{A} \right\} \tag{6}$$

of labeled edges. The graphical depiction of $\mathrm{M} = \{\mathbf{V}, \mathbf{E}, \mathcal{A}\}$ is a labeled directed graph as seen above in Fig. 4. The full probabilistic structure is described by a set of input-alphabet labeled transition matrices

$$\left\{ T^{(s)} : \left(T^{(s)} \right)_{vv'} = p_{v \underset{s}{\rightarrow} v'}, \quad v, v' \in \mathbf{V}, s \in \mathcal{A} \right\} \tag{7}$$

where $p_{v \underset{s}{\rightarrow} v'}$ denotes the conditional probability to make a transition to state v' from state v on observing symbol s.

Given the current state $v_t \in \mathbf{V}$ at time t the future is conditionally independent of the past

$$\begin{aligned} Pr(\mathrm{s}) &= Pr\big(\overleftarrow{\mathrm{s}}_t \overrightarrow{\mathrm{s}}_{t+1}\big) \\ &= Pr\big(\overrightarrow{\mathrm{s}}_{t+1} | \overleftarrow{\mathrm{s}}_t\big) Pr\big(\overleftarrow{\mathrm{s}}_t\big) \\ &= Pr\big(\overrightarrow{\mathrm{s}}_{t+1} | v_t\big) Pr(v_t) \end{aligned} \tag{8}$$

where $\mathrm{s}_t^{\leftarrow}$ is the historical sequence leading to v_t. By factoring the joint distribution $Pr(\mathrm{s})$ over the observed subsequences, the discovered states vastly reduce the dimensionality of the process's representation. In the case of strings of length N, the representation changes from requiring an N-dimensional probability vector for the joint distribution to a set of $\|\mathbf{E}\|$ transition probabilities.

Rather than simply looking up the probability of a given sequence $\mathrm{s}^N = s_0 s_1 s_2 \ldots s_{N-1}, \quad s_i \in \mathbf{A}$, in the table of joint probabilities, its probability is recovered from the machine by computing the telescoping product of conditional transition probabilities

$$Pr\left(\mathrm{s}^N\right) = p_{v_0} p_{v_0 \underset{s_0}{\rightarrow} v_1} p_{v_1 \underset{s_1}{\rightarrow} v_2} \cdots p_{v_{N-1} \underset{s_{N-1}}{\rightarrow} v_N} \tag{9}$$

Here v_0 is the unique start state. By the nature of machine reconstruction it is the state of total ignorance: before the first measurement is made $p_{v_0} = 1$. The sequence $v_0, v_1, v_2, \ldots, v_{N-1}, v_N$ are those states through which the sequence s^N drives the machine.

A machine is a compact and informative representation of the underlying source. In order to appreciate the properties that it captures, there are several statistics that can be computed from a given machine. Each of these is, naturally, a simplification in one way or another of the structure captured by the machine. Up to this point there has been no restriction on the number $\|\mathbf{V}\|$ of states. For the sake of simplicity in the following the number of states will be assumed to be finite. This restricts the general model class to that of finitary machines or, more commonly, stochastic finite automata.

One useful reduction of a machine M is to ask for its equivalent Markov process. This is described by the stochastic connection matrix

$$T = \sum_{s \in \mathcal{A}} T^{(s)} \tag{10}$$

where $(T)_{vv'} = p_{v \rightarrow v'}$ is the state to state transition probability, unconditioned by the measurement symbols. By construction every state has an outgoing transition. This is reflected in the fact that T is a stochastic matrix: $\sum\limits_{v' \in \mathbf{V}} p_{vv'} = 1$. It should be clear from dropping the input-alphabet transition labels from the machine that the detailed, I call it "computational", structure of the input data stream has been lost. All that is retained in T is the state transition structure and this is a Markov chain. The interesting fact is that Markov chains are a proper subset of finitary machines.

The stationary state probabilities $\vec{p}_{\mathbf{V}} = \left\{ p_v \in [0,1] : \sum\limits_{v \in \mathbf{V}} p_v = 1, v \in \mathbf{V} \right\}$ are given by the left eigenvector of T

$$\vec{p}_{\mathbf{V}} T = \vec{p}_{\mathbf{V}} \tag{11}$$

The entropy rate of the Markov chain is then[11]

$$h_\mu(T) = -\sum_{v\in\mathbf{V}} p_v \sum_{v'\in\mathbf{V}} p_{v\rightarrow v'} \log_2 p_{v\rightarrow v'} \tag{12}$$

Being the transition uncertainty averaged over all the states, it measures the information production rate in bits per time step. It is the entropy rate $h_\mu(\mathrm{M})$ of the stochastic machine M. The entropy rate gives the growth rate of the Shannon information in subsequences

$$h_\mu = \lim_{L\rightarrow\infty} \frac{H(L)}{L} \tag{13}$$

where $H(L) = -\sum_{\omega\in\{s^L\}} Pr(\omega)\log_2 Pr(\omega)$. That is, h_μ is a scaling exponent: $H(L) \underset{L\rightarrow\infty}{\propto} h_\mu L$. Once a machine is reconstructed from a data stream, its entropy is an estimate of the underlying process's entropy rate.

The complexity* quantifies the information in the state-alphabet sequences

$$C_\mu(\mathrm{M}) = H(\vec{p}_{\mathrm{V}}) = -\sum_{v\in\mathbf{V}} p_v \log_2 p_v \tag{14}$$

It measures the amount of memory in the generating process.

The entropy h_μ, as a measure of the diversity of patterns, and the complexity C_μ, as a measure of memory, have been taken as the two elementary "information processing" coordinates with which to analyze a range of nonlinear dynamical systems.[9]

Although it falls somewhat outside of the present discussion, it is worthwhile noting that there is a general thermodynamics of ϵ-machines. This describes the process of optimal model estimation as one in which the observer comes to "inferential" equilibrium with the data stream. Additionally, it indicates the detailed structure of invariant subsequences in terms of generalized thermodynamic potentials.[12]

2.4 Complexity and Causality

It is useful at this stage to stop and reflect on some properties of the models whose reconstruction and structure have just been described. Consider two extreme data sources. The first, highly predictable, produces a stream of 1s; the second, highly unpredictable, is an ideal random source of binary symbols. The parse tree of the predictable source is a single path of 1s. And there is a single distinct subtree, at any depth. As a result the machine has a single state and a single transition on $s = 1$: a simple model of a simple source. For the ideal random source the parse tree, again to any depth, is the full binary tree. All paths appear in the parse tree since all binary subsequences are produced by the

---

* Within the reconstruction hierarchy this is actually the finitary complexity, since discussion is restricted to processes with a finite number of states.

source. There is a single subtree, of any morph depth, at all parse tree depths: the full binary subtree. And the machine has a single state with two transitions: one on $s = 1$ and one on $s = 0$. The result is a simple machine, even though the source produces the widest diversity of binary sequences. Thus, these two zero entropy and maximal entropy sources have zero complexity.

A simple gedanken experiment serves to illustrate how complexity is a measure of a machine's memory capacity. Consider two observers **A** and **B**, each with the same model M of some process. **A** is allowed to start machine M in any state and uses it to generate binary strings that are determined by the edge labels of the transitions taken. These strings are passed to observer **B** which traces their effect through its own copy of M. On average how much information about M's state can **A** communicate to **B** via the binary strings? If the machine describes (say) a period three process, e.g. it outputs strings like $101101101\ldots$, $011011011\ldots$, and $110110110\ldots$; it has $\|\mathbf{V}\| = 3$ states. Since **A** starts M in any state, **B** can learn only the information of the process's phase in the period 3 cycle. This is $\log_2 \|\mathbf{V}\| = 1.584\ldots$ bits of information on average about the process's state, if **A** chooses the initial states with equal probability. However, if the machine describes an ideal random binary process, by definition **A** can communicate no information to **B**, since there is no structure in the sequences to use for this purpose. This is reflected in the fact, as already noted above, that the corresponding machine has a single state and its complexity is $C_\mu(\mathrm{M}) = \log_2 1 = 0$. In this way, a process's complexity is the amount of information that someone controlling its start state can communicate to another.

These examples serve to highlight one of the most basic properties of complexity, as I use the term. Both predictable and random sources are simple in the sense that their models are small. Complex processes in this view have large models. In computational terms, complex processes have, as a minimum requirement, a large amount of memory as revealed by many internal states in the reconstructed machine. Most importantly, this memory is structured in particular ways that support different types of computation.

There are two additional points to be brought out concerning what reconstructed machines represent. First, by the definition of future-equivalent states, they give the minimal information dependency between the morphs. In this respect, they represent the causality of the morphs considered as events. If state B follows state A then A is a cause of B and B is one effect of A. Second, the machines capture the information flow within the given data stream. Machine reconstruction produces minimal models up to the given approximation level; that is, up to the amount of data available for the estimation and up to the setting of the parameters (D, L). This minimality guarantees that there are no other events (morphs) that intervene between successive states, at the given error level, to render A and B independent. In this and only this case, can one unambiguously say that information flows from A to B, under the chosen data stream parsing direction. The amount of information that flows is given by the mutual information $I(\mathrm{A};\mathrm{B}) = H(\mathrm{B}) - H(\mathrm{B}|\mathrm{A})$ of observing the state-event A followed by state-event B.[11] This criterion for information flow also extends to spatial systems for which it gives a way to trace out iso-causality contours.

3. Cellular Automata: Prototype Spatial Systems

The preceding sections reviewed a model estimation method that reconstructs a stochastic automaton from a data series. The explanations were facilitated by thinking of the individual measurements as ordered by time. This, of course, need not be the case. The remainder will show, in fact, an application to building models of spatial patterns using machine reconstruction. Along the way, a number of interpretations and uses for the resulting models will be pointed out.

This analysis will be done for what are probably the simplest deterministic models of spatial dynamical systems: cellular automata (CA). They have the notable feature of being as discrete as possible: in space (cells), in site value (e.g. binary), and in time.[13] The particular example selected, rule 18 to be defined below, is itself the simplest nonlinear CA. It is specificed by a very simple rule, but obeys no linear superposition principle as do other periodic and chaotic CA.

CA might seem to be too simple to be a useful indicator of nonlinear spatial modeling for continuum, nonlinear wave systems. However, despite their simplicity, or perhaps because of it, CA have been used recently as a simulation technique for Navier-Stokes fluid flows and for nonlinear reaction-diffusion partial differential equations, such as the shock-wave forming Burghers equation.[14] Though these lattice-gas CA will not be further discussed here, the techniques that follow apply in principle to them and so form an interesting area for their application.

The following reviews a synthesis of the dynamical systems view of a CA as a mapping of a state space and the computation theoretic view of a CA as a parallel algorithm that produces complicated bit strings.[15] CA dynamics will be analyzed in terms of local space-time and state-space "coherent" structures that organize the system's evolution. The approach introduces a rather literal analysis of CA as dynamical systems. Using it a wide range of related CA, those dominated by domains and walls, can be analyzed in terms of the analogs of attractors, basins, and separatrix structures; the centrals object of study in DST. The remainder of the discussion briefly reviews results presented elsewhere that bear on the question of detecting coherent structures.[15]

3.1 Definitions

A cellular automaton consists of a countable array of discrete sites or cells i and a discrete-time update rule ϕ operating in parallel on local neighborhoods of a given radius r.[16] At each time the sites take on values in a finite alphabet $\mathcal{A}$ of primitive symbols:

$\sigma_t^i \in \{0, 1, ...k-1\} \equiv \mathcal{A}$. The local site-update function is written

$$\sigma_{t+1}^i = \phi(\sigma_t^{i-r}, \ldots, \sigma_t^{i+r}) \tag{15}$$

The state s_t of the CA at time t is the configuration of the finite or infinite spatial array: $\mathrm{s}_t \in \mathcal{A}^N$, where $\mathcal{A}^N$ is the set of all possible site value configurations on a lattice of N cells. The "extended state space", denoted $\mathcal{A}^*$, is the union of all states of any N:

$$\mathcal{A}^* = \bigcup_{N \geq 0} \mathcal{A}^N \tag{16}$$

with $\mathcal{A}^0 = \emptyset$.

The CA global update rule $\Phi : \mathcal{A}^N \rightarrow \mathcal{A}^N$ applies ϕ in parallel to all sites in the lattice: $\mathrm{s}_t = \Phi \mathrm{s}_{t-1}$. For finite N it is also necessary to specify a boundary condition. The following will use periodic boundary conditions.

A pattern ensemble $\Lambda \subseteq \mathcal{A}^*$ is a collection of strings each of which is an observable configuration at a given time. The machine representation of a pattern ensemble Λ will be denoted $\mathrm{M}(\Lambda)$.

3.2 Rule 18

The automaton of interest here is an "elementary" CA for which $(k, r) = (2, 1)$; that is, nearest-neighbor interactions and binary cell values, giving a total of $k^{2r+1} = 8$ possible neighborhood patterns. Following the conventional numbering scheme,[16] one arranges these patterns in the order below, writes the value obtained by applying ϕ to each pattern, and interprets the resulting string of 8 symbols as a binary number giving the rule index. The following analyzes the CA having rule index 18, for which the rule table is as follows:

Neighborhood	111	110	101	100	011	010	001	000	$\sigma_t^{i-1}\sigma_t^{i} \ \sigma_t^{i+1}$
Next site value	0	0	0	1	0	0	1	0	σ_{t+1}^{i}

(17)

As is readily seen from the rule table this CA is dissipative in the sense that the global update rule is many-to-one: $\Phi \mathcal{A}^N \subset \mathcal{A}^N$.

The basic phenomenology of rule 18 has been known for some time.[17] It organizes itself into local regions in which every other cell in space and time is 0 and the remaining cells are unrestricted. The boundaries between adjacent "phase-locked" regions are dislocations, where the spatial phase slips a cell. These are the coherent structures in rule 18. (The reader may wish jump ahead to the space-time diagram of Fig. 7. This shows rule 18 evolving from a random initial pattern.) When evolved from a random initial condition, numerical results indicate that the number density d of dislocations decays in time as $d \approx (8\pi Dt)^{-\frac{1}{2}}$, with $D \approx \frac{1}{2}$. Furthermore, the excursion x of a "plume" of dislocations grows as $\langle x^2 \rangle \approx 2Dt$. The interpretation of this is that the dislocations move as if following a space-time random walk with diffusion constant D, annihilating in pairs when their paths cross. In short, starting from a random configuration the dislocations behave like a diffusive gas of annihilating particles.

3.3 Transducers

The ϵ-machines defined above can be used either as input devices for classifying existing strings or as output devices that generate strings belonging to a given language. These two modes of operation are easily combined by modifying the machine so that on each transition a symbol is emitted as well as read. The resulting "transducer" machines are functions from strings to strings. Transducers give a very compact notation for functions that map complicated sets, e.g. Cantor sets, to complicated sets. Of equal importance, they are constructive in the sense that they indicate literally how the function is to be implemented and so computed.

A transducer $\mathrm{T} = \{\mathbf{V}, \Sigma_{in}, \mathbf{E}, \Sigma_{out}, \beta\}$ is an automaton where $\{\mathbf{V}, \mathbf{E}\}$ are as defined above and where Σ_{in} is the previous input alphabet $\mathcal{A}$. Σ_{out} is an output alphabet. The observation function β maps $\mathbf{V} \times \Sigma_{in}$ to Σ_{out}. A transducer operates as an ϵ-machine with the extra feature of writing a symbol in Σ_{out} to an output string upon reading a symbol in Σ_{in} from the input string. In the context of spatial systems, the effect of running the transducer over a pattern is to recode the data by replacing original symbols with those emitted by the machine.

4. Coherent Structures

With these definitions in hand, this section begins to address the problems of spatially-extended systems alluded to in the introduction. The first task is to motivate several definitions of coherent structures. The following section then applies these to structure detection.

The global machine $\mathrm{M}_t = \mathrm{M}(\Omega_t)$, when $\Omega_t = \mathbf{\Phi}_t(\mathcal{A}^*)$, describes the time-evolution of the ensemble of all possible initial conditions up to some time t. It necessarily captures all of the information processing accessible to the CA, which includes correlations and information transmission, as well as computation. Previous investigations of CA patterns have focused on the global machine M_t.[18, 19]

In terms of dynamical systems theory, though, the global machine describes the entire attractor-basin portrait: the collection of all invariant sets and transients, including attractors and separatrices.[1] In many cases, this description is seen to be prohibitively difficult to construct because, for example, M_t grows too large with time. In fact, such complete global descriptions are rarely pursued in dynamical systems theory. Even for elementary CA, there are many cases in which the size of M_t increases extremely rapidly in time. This is true in particular of rule 18: $\mathrm{M}(\Omega_3)$ has $\|\mathbf{V}\| = 143$ states; $\mathrm{M}(\Omega_4)$ is estimated to have $\|\mathbf{V}\| > 20,000$.[18] Furthermore, there are other CA rules for which the growth is even faster.[19]

An alternative approach is to identify the *structures* that dominate and organize the spatial system's behavior. The most basic of these for dissipative systems are attractors, basins, and separatrices.[1] The state space is then understood in terms of those structures. In the case of CA, the goal is to find the dominant, dynamically-homogeneous space-time patterns and their associated machines.

The introductory comments noted that a state of a dynamical system is a structureless point. In contrast, it is exactly the internal, now spatial, structure of a CA configuration s_t that is of interest. This deficit can be addressed by interpolating between the literal notion of a CA state as a position-dependent configuration of cells and the intuitive notion of a translation-independent pattern. Generally speaking, a pattern is (i) a position-independent configuration and (ii) a typical sequence generated by the process, where "typical" is used in the sense of the Shannon-McMillan theorem of information theory.[20] The use of a finitary machine description of processes captures exactly these features.

For CA rule 18, the dominant structure is the domain already described: every other site has value 0, while the remaining sites can be either 0 or 1. The iterate of a domain consists of a domain shifted by one site. This observation is consistent with the intuitive notion of a domain as a dynamically-homogeneous region of the spatial array.

A domain Λ of a CA is an ensemble representing a set of spatial regions with the following properties:
(i) *Temporal invariance:* Λ is mapped onto itself by the dynamic: $\mathbf{\Phi}\Lambda = \Lambda$; and
(ii) *Spatial homogeneity:* The recurrent states of $\mathrm{M}(\Lambda)$ are strongly connected. This is similar to statistical stationarity of the spatial pattern.

A domain describes a computationally-homogeneous *region* of the spatial array. In this way we take "domain" to mean the domain language Λ or, alternatively, any configuration $\mathrm{s} \in \Lambda$. For general CA there can be any number of different domains Λ^i, each with its own computational structure. Additional particle-like coherent structures can now be defined with respect to domains.

A wall is a boundary between domains. Two domains Λ^i and Λ^j separated by a wall will be denoted $\Lambda^{i,j}$. Walls may consist of any number of cells, including zero. A dislocation is a wall separating two identical domains and whose minimum width is zero.

The phase of site *i* in a domain Λ^j is the state $v \in \mathbf{V}$ of the machine $\mathrm{M}(\Lambda^j)$ when s^i is read in a chosen direction.

In the case of rule 18, the observed behavior described above defines a domain language Λ^0. All spatial states are made up of concatenations of words $\omega \in \Lambda^0$ separated by dislocations. In particular, the two-domain language $\Lambda^{0,0}$ consists of all states in which there is a single dislocation. The (minimal) machines for both Λ^0 and $\Lambda^{0,0}$ are shown in Figure 5.

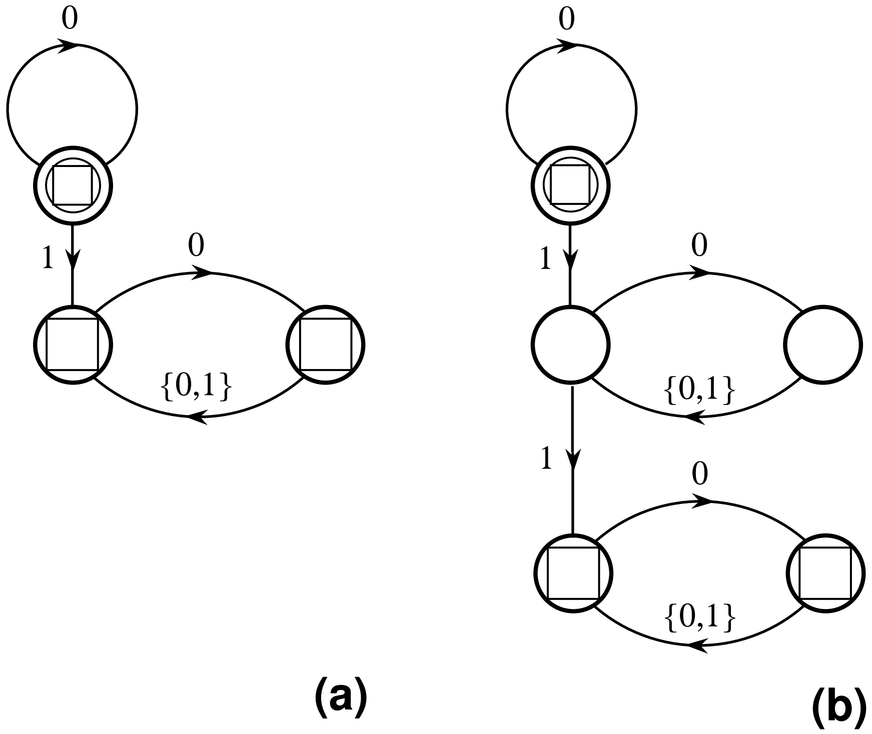

Fig. 5 $\mathrm{M}(\Lambda^0)$ for single domain language Λ^0 and (b) $\mathrm{M}(\Lambda^{0,0})$ for domain-wall-domain language $\Lambda^{0,0}$. Inscribed squares indicate the accept states: finite strings are accepted if and only if they leave the machine in one of the states. Edges labeled $\{0, 1\}$ indicate two transitions, one on $s = 0$ and one on $s = 1$, are allowed. Transition probabilities are uniform and are not indicated.

Starting from the above definition of a domain and a dislocation, definitions can be developed for neighborhoods, attractors, basins, and separatrices. The global structure of rule 18's state space can be laid out in terms of an attractor-basin portrait, almost wholly independent of the lattice size or, equivalently, the dimension of the state space. With the identification of the coherent structures, even more geometry can be extracted concerning the organization of the basins into subbasins connected by portals. The number of dislocations is constant in a subbasin. The portals are determined by the annihilation conditions operating when dislocations collide; that is, when the number of dislocations decreases. From the view of average properties, it is then seen that there are two statistically significant attractors, viz. those given in Fig. 5. Formally, it can be shown that (i) Λ^0 is a domain; (ii) $\Lambda^{0,0}$ is a time-invariant (fixed-point) ensemble. Additionally, it can be shown that on an infinite lattice with finite initial conditions and on an overwhelmingly large set of finite lattices, Λ^0 and $\Lambda^{0,0}$ are attractors. These results established the geometric picture that the attracting domain Λ^0 organizes the state space. $\Lambda^{0,0}$ appears as the lowest "energy" fluctuation above the vacuum state Λ^0. Like analogous situations with topological solitons the dislocation in $\Lambda^{0,0}$ is stabilized by the global topology of either finite odd-size or infinite lattices. And so $\Lambda^{0,0}$ is an attractor, although not a domain.

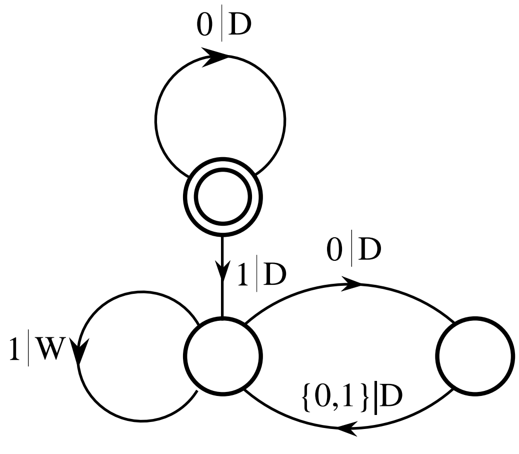

Fig. 6 Transducer $T_{\mathrm{domain}}(\Lambda^0)$ for recognizing dislocations in rule 18 domain structure Λ^0.

5. Structure Detection

An essential issue that arises here is the question of discovering domains in the first place. Even with unlimited space-time data, the domain language(s) for a given CA may be sufficiently complex as to defy visual identification. For rule 18, the dominant structure had already been found; but this is not the case for arbitrary CA. One consequence of the foregoing analysis is that a general approach to identifying these structures, once developed, would provide a basis for automated structure discovery and detection, as will now be shown. The contention here is that machine reconstruction is the central technique facilitating this type of analysis.[21]

As a starting point in the analysis of rule 18 machine reconstruction was applied to the state reached by 10^5 iterations on a periodic array of 10^5 sites, starting from a random initial condition. The results of a series of reconstructions were trivial variants of the two-domain machine $M(\Lambda^{0,0})$ shown in Fig. 5. With machine reconstruction it was quite straightforward to identify the domains, dislocations, and attractors with their "vicinity" neighborhoods and proceed with the dynamical systems analysis. Current plans include an interactive expert-system-like environment to facilitate the discovery of domains and to perform automated construction of vicinities and basins and proofs of invariance and attraction. The knowledge base for such a system consists of the properties, rules of manipulation and theorems for finite automata.[22] It also incorporates basic notions from dynamical systems, such as those defining the attractor-basin portrait.

Given the coherent structures for rule 18, space-time data can be recoded to isolate them. Dislocation trajectories, for example, are detected by modifying the domain-machine $M(\Lambda^0)$ to be a transducer $T_{\mathrm{domain}}(\Lambda^0)$ that emits a (domain) symbol D on allowed transitions of Λ^0, and a (wall) symbol W on disallowed transitions. This dislocation recognizer is shown in Fig. 6. For general CA with some set $\{\Lambda^i\}$ of domain types, it is straightforward to construct a transducer that will encode the data to show each of the domain and wall structures.

Fig. 7 CA rule 18 space-time diagram showing 251 sites over 251 time steps starting with a random initial condition. The 251 sites are a window into a 1000 lattice. Sites with a 1 are black; with 0, white.

Figure 7 shows a typical evolution of patterns generated by rule 18. The space-time diagram consists of 251 sites taken out of a 1000 site simulation over 251 time steps. The initial pattern was a random binary string. In the space-time diagram sites whose state is a 1 are black; those with 0, white.

Figure 8 illustrates the effect of recoding the preceding figure with the transducer $\mathrm{T}_{\mathrm{domain}}(\Lambda^0)$. As seen there, domains and dislocations are easily detected within the chaotic domain background. There all sites in an "even"-site domain are colored white; those in an "odd"-site domain, gray. The dislocations are shown as black cells on the domain borders. This type of display indicates in a transparent way the temporal evolution of the coherent domain and dislocation structures. This should be compared to the original space-time plot of the preceding figure.

There are two additional benefits of using $\mathrm{T}_{\mathrm{domain}}(\Lambda^0)$ worth noting. The first is a marked increase in display speed of one or two orders of magnitude, which would be even more significant for patterns of higher spatial dimension or a wider dynamic range of site values than binary. The second is the very substantial data compression that can be achieved by storing only the trajectories of the coherent structures. For example, Fig. 7 can be stored in 63 kilobits as a raw pattern; using the domain detector to factor out the "every other cell a zero" regularity this can be reduced to 32 kbits; and, giving up exact retrieval of the space-time pattern and storing only the initial condition and the relative moves along the dislocation trajectories, Fig. 8 can be stored in about 3 kbits.

The transducer is run over the spatial data in a particular direction: in the present example, from left to right. This introduces a spatial asymmetry into the recoded picture. For example, the visible asymmetry in the trajectory at the top of a triangle of white space (Figure 8) is due to the fact that as $\mathrm{T}_{\mathrm{domain}}(\Lambda^0)$ is moved to the right over the data, it identifies a dislocation only by reading 1 in the wrong place. Thus, a 1 followed by an even number of consecutive 0's shows a dislocation at the right side, when the next 1 is encountered. This has the effect of a constraint on the motion of the observed dislocation: all moves to the left are a single cell, but moves to the right may be any odd-valued distance. Further, a move left may be followed by a move either to the right or left, but a move right must be followed by one to the left. More complicated transducers can be defined to evade this directionality. For example, the transducer could label all blocks $1(00)^n1, \;\; n = 0, 1, 2, \ldots,$ as walls. With this detector, the dislocation becomes delocalized as it enters a triangle of 0's, as seen in Fig. 7.

Figure 9 demonstrates that some detectors can suggest misleading interpretations of coherent structures. A common CA recoding method uses block transformations in which every other cell is removed from the display. Notice in Fig. 9, where this recoding is used on rule 18, that the even-site domains are in phase with the decimation. That is, the even-site domains' random cells are deleted, leaving the cells that must be zero. In the odd-site domains, however, the decimation is out of phase. It deletes the cells which must be zero in the odd-domain, leaving the arbitrary cells. Previous investigations often concluded from block-transformed space-time diagrams that complicated-looking propagating structures, such as the misidentified domains in Fig. 9, are coherent structures distinct from the other (all-white) domains. It has been suggested based on this that such propagating structures are the carriers of useful information processing. The computational analysis suggests an alternative. There is only a single domain in rule

Fig. 8 Domains and dislocations detected in rule 18. Space-time diagram as in preceding figure and starting from the same initial condition. Here all sites in an “even”-site domain are colored white; those in an “odd”-site domain, gray. The dislocations are shown as black cells on the domain borders.

18. Furthermore, domains themselves are not useful for high levels of computation. Rather the dislocations are the computationally useful structures.

Given a transducer that will recode space-time data to isolate dislocations, it is possible to examine the encoded data for other, higher-order regularities. By designing the transducer properly, one can use this process to identify structures of increasing

Fig. 9 Misleading structure detection using a blocking transformation that decimates every other site. The space-time diagram is the same as in preceding figure; again including the same initial condition.

complexity. The result is the original pattern dynamics “modulo” the coherent structures captured by the transducer machine. In this all the regularity captured by the transducer is excised, giving a clearer picture of residual coherent structures supporting longer-range correlation and higher-level computation. The method allows for more accurate and incisive statistical analysis of coherent structures, even in the presence of noisy backgrounds. The overall procedure should be viewed as an incremental one in which as more structure is discovered in a nonlinear spatial system, more regularity is factored out. The remaining machine "filtered" pattern is then investigated for further coherent structure.

6. Concluding Remarks

This overview addressed one aspect of the general problem of discovering and analyzing space-time structures. By way of contrast, consider the class of coherent structures called solitons. Historically, as in the case of a soliton in a shallow water channel, the recognition of a propagating structure was done intuitively via visual inspection. In some cases, such as with solitons, this identification eventually develops a formal expression. The propagating entity has a mathematically defined shape expressed in closed form. With it one can search for them by convolving the space-time pattern with the soliton kernel. In a literal sense, the shapes form a basis for the space of recognizable patterns. The method of transducer machine recognition introduced above gives a different, but complementary, computational approach to space-time pattern recognition, as we have just described it, that is both general and constructive. It, along with machine reconstruction, suggests an automated method of identifying coherent structures and tracking them in space-time. Although the present approach starts with discretized data, note that rather than a liability this is the domain of many important problems in time series and image processing. The study of domain-wall dominated CA strongly suggests similar analyses for a wider class of spatially-extended systems, such as map lattices, where each site has a continuum of states, and soliton-bearing PDEs.

The explosive complexity found in the global machine representation of rule 18's temporal evolution was circumvented by studying subspaces of initial configurations. These were represented by taking initial pattern ensembles Ω_0 to be various ensembles other than $\mathcal{A}^*$. Implicit in this is a set of "designer" ensemble methods in which pattern ensembles are constructed to facilitate the study of specific geometric properties in high-dimensional spatially-extended systems. This analysis for CA was motivated, in fact, by techniques used in earlier work on continuous-state lattice dynamical systems[23] and their extremely long transients.[24, 4] And these too will be helpful in analyzing a wide range of similar problems posed by nonlinear spatial systems.

Acknowledgements

Thanks to Jim Hanson for making Figs. 7, 8, and 9. Many thanks to the Santa Fe Institute, where the author was supported by a Robert Maxwell Foundation Visiting Professorship, for the warm hospitality during the writing of the present essay. Funds from ONR contract N00014-90-J-1774 and NASA-Ames University Interchange NCA2-488 contributed to this work.

REFERENCES

1. D. R. J. Chillingworth, *Differential Topology with a View to Applications*. London: Pitman, 1976.
2. M. Casdagli and S. Eubank, eds., *Nonlinear Modeling*, SFI Studies in the Sciences of Complexity, (Reading, Massachusetts), Addison-Wesley, 1992.
3. J. W. Tukey, *Exploratory Data Analysis*. Reading, Massachusetts: Addison-Wesley, 1977.
4. J. P. Crutchfield and K. Kaneko, "Are attractors relevant to turbulence?," *Phys. Rev. Let.*, vol. 60, p. 2715, 1988.
5. J. P. Crutchfield and B. S. McNamara, "Equations of motion from a data series," *Complex Systems*, vol. 1, p. 417, 1987.
6. J. Rissanen, *Stochastic Complexity in Statistical Inquiry*. Singapore: World Scientific, 1989.
7. J. P. Crutchfield, "Reconstructing language hierarchies," in *Information Dynamics* (H. A. Atmanspracher and H. Scheingraber, eds.), (New York), p. 45, Plenum, 1991.
8. J. P. Crutchfield and K. Young, "Inferring statistical complexity," *Phys. Rev. Let.*, vol. 63, p. 105, 1989.
9. J. P. Crutchfield and K. Young, "Computation at the onset of chaos," in *Entropy, Complexity, and the Physics of Information* (W. Zurek, ed.), vol. VIII of *SFI Studies in the Sciences of Complexity*, (Reading, Massachusetts), p. 223, Addison-Wesley, 1990.
10. L. R. Rabiner, "A tutorial on hidden Markov models and selected applications," *IEEE Proc.*, vol. 77, p. 257, 1989.
11. C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Champaign-Urbana: University of Illinois Press, 1962.
12. J. P. Crutchfield and K. Young, "ϵ-machine spectroscopy." preprint, 1992.
13. S. Wolfram, *Theory and Applications of Cellular Automata*. Singapore: World Scientific Publishers, 1986.
14. B. M. Boghosian, "Lattice gases," in *1989 Lectures in Complex Systems* (E. Jen, ed.), vol. Lect. Vol. II of *SFI Studies in the Sciences of Complexity*, (Reading, Massachusetts), Addison-Wesley, 1990.
15. J. E. Hanson and J. P. Crutchfield, "The attractor-basin portrait of a cellular automaton," *J. Stat. Phys.*, vol. 66, p. 1415, 1992.
16. S. Wolfram, "Statistical mechanics of cellular automata," *Rev. Mod. Phys.*, vol. 55, p. 601, 1983.
17. P. Grassberger, "New mechanism for deterministic diffusion," *Phys. Rev. A*, vol. 28, p. 3666, 1983.

18. S. Wolfram, “Computation theory of cellular automata,” *Comm. Math. Phys.*, vol. 96, p. 15, 1984.
19. L. P. Hurd, “Appendix. table 10. regular language complexities,” in *Theory and Applications of Cellular Automata* (S. Wolfram, ed.), Singapore: World Scientific, 1986.
20. R. E. Blahut, *Principles and Practice of Information Theory*. Reading, Massachusetts: Addison-Wesley, 1987.
21. J. P. Crutchfield and J. E. Hanson, “Computational mechanics of cellular automata: Space-time machines.” in preparation, 1992.
22. J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*. Reading: Addison-Wesley, 1979.
23. J. P. Crutchfield and K. Kaneko, “Phenomenology of spatio-temporal chaos,” in *Directions in Chaos* (H. Bai-lin, ed.), p. 272, Singapore: World Scientific Publishers, 1987.
24. J. P. Crutchfield, “Subbasins, portals, and mazes: Transients in high dimensions,” *J. Nucl. Phys. B*, vol. 5A, p. 287, 1988.