in the Physica D (1994) special issue on the Proceedings of the Oji International Seminar
Complex Systems — from Complex Dynamics to Artificial Reality
held 5 - 9 April 1993, Numazu, Japan SFI 94-03-016

The Calculi of Emergence:

Computation, Dynamics, and Induction

James P. Crutchfield

Physics Department
University of California
Berkeley, California 94720

Abstract

Defining structure and detecting the emergence of complexity in nature are inherently subjective, though essential, scientific activities. Despite the difficulties, these problems can be analyzed in terms of how model-building observers infer from measurements the computational capabilities embedded in nonlinear processes. An observer's notion of what is ordered, what is random, and what is complex in its environment depends directly on its computational resources: the amount of raw measurement data, of memory, and of time available for estimation and inference. The discovery of structure in an environment depends more critically and subtlely, though, on how those resources are organized. The descriptive power of the observer's chosen (or implicit) computational model class, for example, can be an overwhelming determinant in finding regularity in data.

This paper presents an overview of an inductive framework — hierarchical ϵ-machine reconstruction — in which the emergence of complexity is associated with the innovation of new computational model classes. Complexity metrics for detecting structure and quantifying emergence, along with an analysis of the constraints on the dynamics of innovation, are outlined. Illustrative examples are drawn from the onset of unpredictability in nonlinear systems, finitary nondeterministic processes, and cellular automata pattern recognition. They demonstrate how finite inference resources drive the innovation of new structures and so lead to the emergence of complexity.

Contents

List of Figures . . . iii
List of Tables . . . vii
Part I INNOVATION, INDUCTION, AND EMERGENCE . . . 1
1 Emergent? . . . 1
1.1 Pattern! . . . 2
1.2 Intrinsic Emergence . . . 3
2 Evolutionary Processes . . . 4
3 What's in a Model? . . . 5
4 The Modeling Dilemma . . . 6
5 A Computational View of Nature . . . 8
6 Computational Mechanics: Beyond Statistics, Toward Structure . 9
7 Agenda . . . 9
Part II MECHANISM AND COMPUTATION . . . 10
1 Road Maps to Innovation . . . 11
2 Complexity $\neq$ Randomness . . . 13
2.1 Deterministic Complexity . . . 14
2.2 Statistical Complexity . . . 14
2.3 Complexity Metrics . . . 15
3 ϵ-Machine Reconstruction . . . 17
4 Measuring Predictability and Structure . . . 19
Part III TOWARD A MATHEMATICAL THEORY OF INNOVATION . . . 21
1 Reconstructing Language Hierarchies . . . 21
2 At Each Level in a Hierarchy . . . 23
3 The ϵ-Machine Hierarchy . . . 23
4 The Threshold of Innovation . . . 25
5 Examples of Hierarchical Learning . . . 26
5.1 The cost of chaos . . . 26
5.1.1 Intrinsic computation in the period-doubling cascade . . . 27
5.1.2 Intrinsic computation in frequency-locking route to chaos . . . 31
5.1.3 Temporal computation in deterministic chaos . . . 34
5.2 The cost of indeterminism . . . 35
5.2.1 The simplest example . . . 35
5.2.2 Stochastic counter automata . . . 39
5.2.3 Recurrent hidden Markov models . . . 39
5.2.4 The finitary stochastic hierarchy . . . 40
5.3 The costs of spatial coherence and distortion . . . 42
5.4 What to glean from the examples . . . 44
Part IV OBSERVATIONS AND HYPOTHESES . . . 45
1 Complexity as the Interplay of Order and Chaos . . . 45
2 Evolutionary Mechanics . . . 47
3 Acknowledgments . . . 50
Bibliography . . . 51

List of Figures

Figure 1 Agent-centric view of the environment: The universe can be considered a deterministic dynamical system (DS). The environment, as seen by any one agent, is a stochastic dynamical system (SDS) consisting of all the other agents. Its apparent stochasticity results from several effects — some intrinsic and some due to an agent's limited computational resources. Each agent is itself a stochastic dynamical system, since it may sample, or be plagued by, the uncontrollable randomness in its substrates and in environmental stimuli. The substrates represent the available resources that support and limit information processing, model building, and decision making. The arrows indicate the flow of information into and out of the agent. . 5

Figure 2 The discrete computation hierarchy. *Adjective legend*: 1 = one way input tape, 2 = two way input tape, D = deterministic, N = nondeterministic, I = indexed, RI = restricted I, n = nested, NE = nonerasing, CF = context free, CS = context sensitive, R = recursive, RE = R enumerable, and U = universal. *Object legend*: G = grammar, A = automata, FA = finite A, PDA = pushdown A, SA = stack A, LBA = linear bounded A, RPA = Reading PDA, TM = Turing machine, LS = Lindenmayer system, 0L = CF LS, 1L = CS LS, and RS = R set. (After .) 12

Figure 3 The Bernoulli-Turing Machine (BTM) is a deterministic Turing machine augmented by contact to an information source — a heat bath denoted as a boiling water pot. Like a Turing machine, it is a transducer that maps input tapes (0+1)* to output tapes (0+1)*. The input (output) tape cells are read (written) sequentially and once only. Any intermediate processing and storage is provided by the working tape which allows bidirectional access to its contents. The BTM defines the most general model of discrete stochastic sequential computation. 16

Figure 4 (a) Deterministic complexity — relative to (say) a deterministic universal Turing machine — is a measure of the degree of unpredictability of an information source. It indicates the degree of randomness which can be measured with the Shannon entropy rate h_μ. (b) Statistical complexity is based on the notion that randomness is statistically simple: an ideal random process has zero statistical complexity. At the other end of the spectrum, simple periodic processes have low statistical complexity. Complex processes arise between these extremes and are an amalgam of predictable and stochastic mechanisms. (After .) 17

Figure 5 Within a single data stream, morph-equivalence induces conditionally-independent states. When the templates of future possibilities — that is, the allowed future subsequences and their past-conditioned probabilities — have the same structure, then the process is in the same causal state. At t_9 and at t_{13}, the process is in the same causal state since the future morphs have the same shape; at t_{11} it is in a different causal state. The figure only illustrates the nonprobabilistic aspects of morph-equivalence. (After .) . . . 18

Figure 6 (a) Statistical complexity C_μ versus specific entropy $H(L)/L$ for the period-doubling route to chaos. Triangles denote estimated $(C_\mu, H(L)/L)$ at 193 values of the logistic map nonlinearity parameter. ϵ-machines were reconstructed using a subsequence length of $L = 16$. The heavy solid lines overlaying some of this empirical data are the analytical curves derived for C_0 versus $H_0(L)/L$. (After .) (b) At one of the critical parameter values of the period-doubling cascade in the logistic map the number $\|\mathbf{V}\|$ of inferred states grows without bound. Here $r = r_c \approx 3.5699456718695445\ldots$ and the sequence length ranges up to $L = 64$ where $\|\mathbf{V}\| = 196$ states are found. It can be shown, and can be inferred from the figure, that the per symbol density of states $\|\mathbf{V}(L)\|/L$ does not have a limiting value as $L \to \infty$. (After .) . . . 28

Figure 7 (a) Approximation of the critical ϵ-machine at the period-doubling onset of chaos. (After .) (b) The dedecorated version of the machine in (a). Here the deterministic state chains have been replaced by their equivalent strings. (After .) 29

Figure 8 (a) The finite version of Figure 7(b)'s infinite critical ϵ-machine. This is a string production machine that, when making a transition from the square states, updates two string registers with the productions A $\to$ BB and B $\to$ BA. B' is the contents of B with the last bit flipped. (b) Another finite representation of the period-doubling critical ϵ-machine — a one-way nondeterministic nested stack automaton (1NnSA in Figure 2) — that produces symbols one at a time. (After .) 30

Figure 9 (a) Statistical complexity C_μ versus specific entropy $H(L)/L$ for the quasiperiodic route to chaos. Tokens denote estimated $(C_\mu, H/L)$ at 303 values of the circle map with $\omega = \frac{1+\sqrt{5}}{2}$ and nonlinearity parameter k in three different ranges: 101 values for $k \in [0, 2]$ (triangles), 101 values for $k \in [3.5, 3.9]$ (circles), and 101 values for $k \in [4.5, 6]$ (crosses). These are ranges in which the behavior is more than simple periodic. ϵ-machine reconstruction used a tree depth of $D = 32$ and a morph depth of $L = 16$ for the first range and $(D, L) = (16, 8)$ for the second two ranges, which typically have higher entropy rates. The entropy density was estimated with a subsequence length of $L = 16$. Refer to Figure 6(a) for details of the annotations. (b) At the golden mean critical winding number (with $k = 1$) in the quasiperiodic route to chaos the number $\|\mathbf{V}\|$ of inferred states grows without bound. Here the sequence length ranges up to $L = 64$ where $\|\mathbf{V}\| = 119$ states are found. . 32

Figure 10 (a) A portion of the infinite critical machine for the quasiperiodic route to chaos at the golden mean winding number. Note that the dedecorated machine is shown — that is, the intervening states along deterministic chains have been suppressed. (b) The Fibonacci machine: the finite representation of the infinite machine in (a). 33

Figure 11 The source is a stochastic nondeterministic finite automaton — a class sometimes referred to as hidden Markov models. The hidden process consists of two states $\{\mathbf{A}, \mathbf{B}\}$ and uniform branching between them — denoted by the fractions p on the edge labels $s|p$. The observer does not have access to the internal state sequences, but instead views the process through the symbols s on the edge labels $s|p$. The inscribed circle in each state indicates that both states are start states. The fractions in parentheses give their asymptotic probabilities, which also will be taken as their initial probabilities. 36

Figure 12 The minimal machine for Figure 11's internal state process. It has a single state and equal branching probabilities. The topological and statistical complexities are zero and the topological and metric entropies are 1 bit per state symbol — a highly unpredictable, but low complexity process. That this is the correct minimal description of the internal state process follows directly from using machine reconstruction, assuming direct access to the internal state sequences $\mathbf{ABABBA}\ldots$. All state sequences are allowed and those of equal length have the same probability. 36

Figure 13 The process's topological structure is given by a deterministic finite automaton — the golden mean machine. The only rule defining the sequences is "no consecutive 0s". The number of sequences of length L is given by the Fibonacci number F_{L+2}; the growth rate or topological entropy h, by the golden mean $\phi = \frac{1}{2}\big(1+\sqrt{5}\big)$: $h = \log_2 \phi$. The numbers in parentheses give the states' asymptotic probabilities. 37

Figure 14 (a) - (d) The zeroth- through third-order causal approximations to the process of Figure 11. 38

Figure 15 The infinite causal representation of the nondeterministic process of Figure 11. The labels in the states indicate the relative weights of the original internal states $\{\mathbf{A}, \mathbf{B}\}$. The numbers in parentheses are the asymptotic state probabilities: $\Pr(v = 1\mathrm{A}i\mathrm{B}) = (i+1)2^{-i-2}$. 39

Figure 16 At a higher computational level a single state machine, augmented by a counter register, finitely describes the process of Figures 11 and 15. . . . 39

Figure 17 Stochastic deterministic automata (SDA): (a) Denumerable SDA: A denumerable ϵ-machine for the simple nondeterministic source of Figure 11. It is shown here in the (two-dimensional) 3–simplex defining its possible deterministic states (indicated with enlarged dots). Since the state probability decays exponentially, the simulation only shows a very truncated part of the infinite chain of states that, in principle, head off toward the upper vertex. Those dots correspond to the 1s backbone of Figure 15. The state on the lower lefthand vertex corresponds to the "reset" state $1\mathrm{A}0\mathrm{B}$ in that figure. (b) Fractal SDA: A nondenumerable fractal ϵ-machine shown in the 4–simplex defining the possible deterministic states. (c) Continuum SDA: A nondenumerable continuum ϵ-machine shown in the 3–simplex defining the possible deterministic states. 41

Figure 18 The computational hierarchy for finite-memory nonstochastic (below the Measure-Support line) and stochastic discrete processes (above that line). The nonstochastic classes come from Figure 2, below the Finite-Infinite memory line. Here "Support" refers to the sets of sequences, i.e. formal languages, which the "topological" machines describe; "Measure" refers to sequence probabilities, i.e. what the "stochastic" machines describe. The abbreviations are: A is automaton, F is finite, D is deterministic, N is nondeterministic, S is stochastic, MC is Markov chain, HMM is hidden Markov model, RHMM is recurrent HMM, and FMC is function of MC. . . 42

Figure 19 (a) Elementary cellular automaton 18 evolving over 200 time steps from an initial arbitrary pattern on a lattice of 200 sites. (b) The filtered version of the same space-time diagram that reveals the diffusive-annihilating dislocations obscured in the original. (After Ref. .) 43

Figure 20 (a) Elementary cellular automaton 54 evolving over 200 time steps from an initial arbitrary pattern on a lattice of 200 sites. (b) The filtered version of the same space-time diagram that reveals a multiplicity of different particle types and interactions. (From Ref. . Reprinted with permission of the author. Cf. .) 44

Figure 21 A schematic summary of the three examples of hierarchical learning in metamodel space. Innovation across transitions from periodic to chaotic, from stochastic deterministic to stochastic nondeterministic, and from spatial stationary to spatial multistationary processes were illustrated. The finite-to-infinite memory coordinate from Figure 2 is not shown. The periodic to chaotic and deterministic to nondeterministic transitions were associated with the innovation of infinite models from finite ones. The complexity (C) versus entropy (H) diagrams figuratively indicate the growth of computational resources that occurs when crossing the innovation boundaries. 45

Figure 22 Schematic diagram of an evolutionary hierarchy in terms of the changes in information-processing architecture. An open-ended sequence of successively more sophisticated computational classes are shown. The evolutionary drive up the hierarchy derives from the finiteness of resources to which agents have access. The complexity-entropy diagrams are slightly rotated about the vertical to emphasize the difference in meaning at each level via a different orientation. (Cf. Table 1.) 49

List of Tables

Table 1 A causal time-series modeling hierarchy. Each level is defined in terms of its model class. The models themselves consist of states (circles or squares) and transitions (labeled arrows). Each model has a unique start state denoted by an inscribed circle. The data stream itself is the lowest level. From it a tree of depth D is constructed by grouping sequential measurements into recurring subsequences. The next level models, finite automata (FA) with states $\mathbf{V}$ and transitions $\mathbf{E}$, are reconstructed from the tree by grouping tree nodes. The last level shown, string production machines (PM), are built by grouping FA states and inferring production rules $\mathbf{P}$ that manipulate strings in register $\mathbf{A}$. 22

Table 2 Contents of the Fibonacci machine registers A and B as a function of machine transitions. The registers contain binary strings and are modified by string concatenation: A $\rightarrow$ AB and B $\rightarrow$ A. That is, the previous contents of A are moved to B and the previous contents of B are appended to A. . . . 34

> *Order is not sufficient. What is required, is something much more complex. It is order entering upon novelty; so that the massiveness of order does not degenerate into mere repetition; and so that the novelty is always reflected upon a background of system.*
>
> A. N. Whitehead on “Ideal Opposites” in **Process and Reality**.[1]

How can complexity emerge from a structureless universe? Or, for that matter, how can it emerge from a completely ordered universe? The following proposes a synthesis of tools from dynamical systems, computation, and inductive inference to analyze these questions.

The central puzzle addressed is how we as scientists — or, for that matter, how adaptive agents evolving in populations — ever “discover” anything new in our worlds, when it appears that all we can describe is expressed in the language of our current understanding. This dilemma is analyzed in terms of an open-ended modeling scheme, called hierarchical ϵ-machine reconstruction, that incorporates at its base inductive inference and quantitative measures of computational capability and structure. The key step in the emergence of complexity is the “innovation” of new model classes from old. This occurs when resource limits can no longer support the large models — often patchworks of special cases — forced by a lower-level model class. Along the way, complexity metrics for detecting structure and quantifying emergence, together with an analysis of the constraints on the dynamics of innovation, are outlined.

The presentation is broken into four parts. Part I is introductory and attempts to define the problems of discovery and emergence. It delineates several classes of emergent phenomena in terms of observers and their internal models. It argues that computation theory is central to a proper accounting of information processing in nonlinear systems and in how observers detect structure. Part I is intended to be self-contained in the sense that the basic ideas of the entire presentation are outlined. Part II reviews computation theory — formal languages, automata, and computational hierarchies — and a method to infer computational structure in nonlinear processes. Part III, the longest, builds on that background to show formally, and by analyzing examples, how innovation and the emergence of complexity occur in hierarchical processes. Part IV is a summary and a look forward.

PART I
INNOVATION, INDUCTION, AND EMERGENCE

1 Emergent?

Some of the most engaging and perplexing natural phenomena are those in which highly-structured collective behavior emerges over time from the interaction of simple subsystems. Flocks of birds flying in lockstep formation and schools of fish swimming in coherent array abruptly turn together with no leader guiding the group.[2] Ants form complex societies whose survival derives from specialized laborers, unguided by a central director.[3] Optimal pricing of goods in an economy appears to arise from agents obeying the local rules of commerce.[4] Even in less manifestly complicated systems emergent global information processing plays a key role. The human perception of color in a small region of a scene, for example, can depend on the color composition of the entire scene, not just on the spectral response of spatially-localized retinal detectors.[5,6] Similarly, the perception of shape can be enhanced by global topological properties, such as whether or not curves are opened or closed.[7]

How does global coordination emerge in these processes? Are common mechanisms guiding the emergence across these diverse phenomena? What languages do contemporary science and mathematics provide to unambiguously describe the different kinds of organization that emerge in such systems?

Emergence is generally understood to be a process that leads to the appearance of structure not directly described by the defining constraints and instantaneous forces that control a system. Over time "something new" appears at scales not directly specified by the equations of motion. An emergent feature also cannot be explicitly represented in the initial and boundary conditions. In short, a feature emerges when the underlying system puts some effort into its creation.

These observations form an intuitive definition of emergence. For it to be useful, however, one must specify what the "something" is and how it is "new". Otherwise, the notion has little or no content, since almost any time-dependent system would exhibit emergent features.

1.1 Pattern!

One recent and initially baffling example of emergence is deterministic chaos. In this, deterministic equations of motion lead over time to apparently unpredictable behavior. When confronted with chaos, one question immediately demands an answer — Where in the determinism did the randomness come from? The answer is that the effective dynamic, which maps from initial conditions to states at a later time, becomes so complicated that an observer can neither measure the system accurately enough nor compute with sufficient power to predict the future behavior when given an initial condition. The emergence of disorder here is the product of both the complicated behavior of nonlinear dynamical systems and the limitations of the observer.[8]

Consider instead an example in which order arises from disorder. In a self-avoiding random walk in two-dimensions the step-by-step behavior of a particle is specified directly in stochastic equations of motion: at each time it moves one step in a random direction, except the one it just came from. The result, after some period of time, is a path tracing out a self-similar set of positions in the plane. A "fractal" structure emerges from the largely disordered step-by-step motion.

Deterministic chaos and the self-avoiding random walk are two examples of the emergence of "pattern". The new feature in the first case is unpredictability; in the second, self-similarity. The "newness" in each case is only heightened by the fact that the emergent feature stands in direct opposition to the systems' defining character: complete determinism underlies chaos and near-complete stochasticity, the orderliness of self-similarity. But for whom has the emergence occurred? More particularly, to whom are the emergent features "new"? The state of a chaotic dynamical system always moves to a unique next state under the application of a deterministic function. Surely, the system state doesn't know its behavior is unpredictable. For the random walk, "fractalness" is not in the "eye" of the particle performing the local steps of the random walk, by definition. The newness in both cases is in the eye of an observer: the observer whose predictions fail or the analyst who notes that the feature of statistical self-similarity captures a commonality across length scales.

Such comments are rather straightforward, even trivial from one point of view, in these now-familiar cases. But there are many other phenomena that span a spectrum of novelty from "obvious" to "purposeful" for which the distinctions are less clear. The emergence of pattern is the primary theme, for example, in a wide range of phenomena that have come to be labeled "pattern formation". These include, to mention only a few, the convective rolls of Bénard and Couette fluid flows, the more complicated flow structures observed in weak turbulence,[9] the spiral waves and Turing patterns produced in oscillating chemical reactions,[10–12] the statistical order parameters describing phase transitions, the divergent correlations and long-lived fluctuations in critical phenomena,[13–15] and the forms appearing in biological morphogenesis.[10,16,17]

Although the behavior in these systems is readily described as "coherent", "self-organizing", and "emergent", the patterns which appear are detected by the observers and analysts themselves. The role of outside perception is evidenced by historical denials of patterns in the Belousov-Zhabotinsky reaction, of coherent structures in highly turbulent fluid flows, and of the energy recurrence in anharmonic oscillator chains reported by Fermi, Pasta, and Ulam. Those experiments didn't suddenly start behaving differently once these key structures were appreciated by scientists. It is the observer or analyst who lends the teleological "self" to processes which otherwise simply "organize" according to the underlying dynamical constraints. Indeed, the detected patterns are often *assumed* implicitly by analysts via the statistics they select to confirm the patterns' existence in experimental data. The obvious consequence is that "structure" goes unseen due to an observer's biases. In some fortunate cases, such as convection rolls, spiral waves, or solitons, the functional representations of "patterns" are shown to be consistent with mathematical models of the phenomena. But these models themselves rest on a host of theoretical assumptions. It is rarely, if ever, the case that the appropriate notion of pattern is extracted from the phenomenon itself using minimally-biased discovery procedures. Briefly stated, in the realm of pattern formation "patterns" are guessed and then verified.

1.2 Intrinsic Emergence

For these reasons, pattern formation is insufficient to capture the essential aspect of the emergence of coordinated behavior and global information processing in, for example, flocking birds, schooling fish, ant colonies, financial markets, and in color and shape perception. At some basic level, though, pattern formation must play a role. The problem is that the "newness" in the emergence of pattern is always referred outside the system to some observer that anticipates the structures via a fixed palette of possible regularities. By way of analogy with a communication channel, the observer is a receiver that already has the codebook in hand. Any signal sent down the channel that is not already decodable using it is essentially noise, a pattern unrecognized by the observer.

When a new state of matter emerges from a phase transition, for example, initially no one knows the governing "order parameter". This is a recurrent conundrum in condensed matter physics, since the order parameter is the foundation for analysis and, even, further experimentation. After an indeterminant amount of creative thought and mathematical invention, one is sometimes found and then verified as appropriately capturing measurable statistics. The physicists' codebook is extended in just this way.

In the emergence of coordinated behavior, though, there is a closure in which the patterns that emerge are important *within* the system. That is, those patterns take on their "newness" with respect to other structures in the underlying system. Since there is no external referent for novelty or pattern, we can refer to this process as "intrinsic" emergence. Competitive agents in an efficient capital market control their individual production-investment and stock-ownership strategies based on the optimal pricing that has emerged from their collective behavior. It is essential to the agents' resource allocation decisions that, through the market's collective behavior, prices emerge that are accurate signals "fully reflecting" all available information.[4]

What is distinctive about intrinsic emergence is that the patterns formed confer additional functionality which supports global information processing, such as the setting of optimal prices. Recently, examples of this sort have fallen under the rubric of "emergent computation".[18] The approach here differs in that it is based on explicit methods of detecting computation embedded in nonlinear processes. More to the point, the hypothesis in the following is that during intrinsic emergence there is an increase in intrinsic computational capability, which can be capitalized on and so lends additional functionality.

In summary, three notions will be distinguished:

1. The intuitive definition of emergence: "something new appears";
2. Pattern formation: an observer identifies "organization" in a dynamical system; and
3. Intrinsic emergence: the system itself capitalizes on patterns that appear.

2 Evolutionary Processes

One arena that frames the question of intrinsic emergence in familiar terms is biological evolution, which presumes to explain the appearance of highly organized systems from a disorganized primordial soup. Unfortunately, biological evolution is a somewhat slippery and difficult topic; not the least reason for which is the less-than-predictive role played by evolutionary theory in explaining the present diversity of life forms. Due to this, it is much easier to think about a restricted world whose structure and inhabitants are well-defined. Though vastly simplified, this world is used to frame all of the later discussion, since it forces one to be clear about the nature of observers.

The prototype universe I have in mind consists of an environment and a set of adaptive observers or "agents". (See Figure 1.) An agent is a stochastic dynamical system that attempts to build and maintain a maximally-predictive internal model of its environment. The environment for each agent is the collection of other agents. At any given time an agent's sensorium is a projection of the current environmental state. That is, the environmental state is hidden from the agent by its sensory apparatus. Over time the sensory apparatus produces a series of measurements which guide the agent's use of its available resources — the "substrates" of Figure 1 — in the construction of an internal model. Based on the regularities captured by its internal model, the agent then takes actions via effectors that ultimately change the environmental state. The "better" its internal model, the more regularity in the environment the agent can take advantage of. Presumably, that advantage increases the agent's survivability. If the available inference resources are limited, then the internal model may fail to capture useful environmental states.

The basic problem facing an agent is the prediction of future sensory input based on modelling the hidden environmental states and on selecting possible actions. The problem facing the designer of such a prototype universe is how to know if the agents have adapted and how they did so. This requires a quantitative theory of how agents process information and build models.

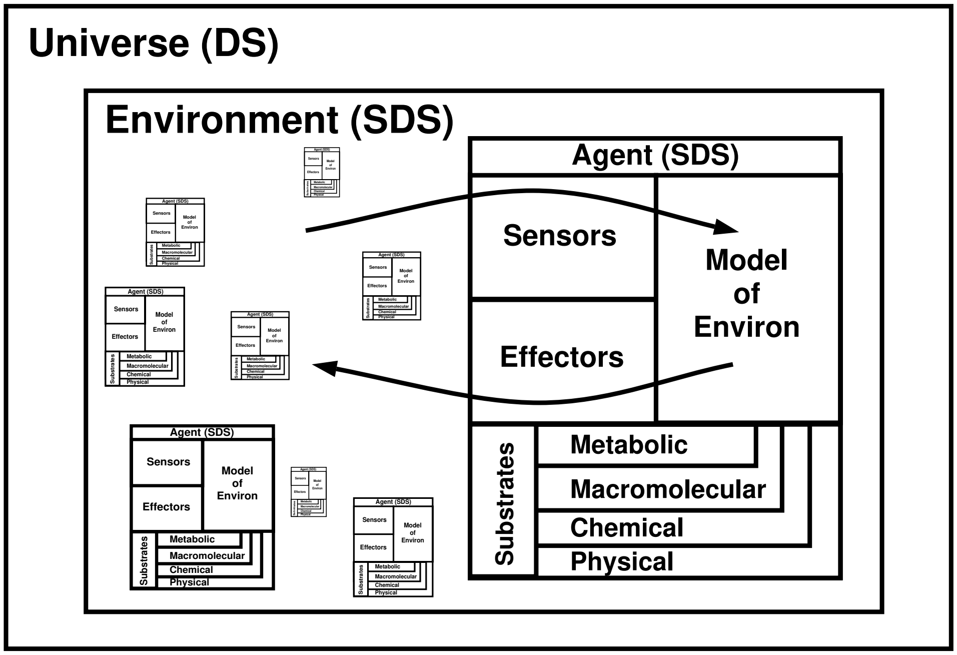


Figure 1 Agent-centric view of the environment: The universe can be considered a deterministic dynamical system (DS). The environment, as seen by any one agent, is a stochastic dynamical system (SDS) consisting of all the other agents. Its apparent stochasticity results from several effects — some intrinsic and some due to an agent's limited computational resources. Each agent is itself a stochastic dynamical system, since it may sample, or be plagued by, the uncontrollable randomness in its substrates and in environmental stimuli. The substrates represent the available resources that support and limit information processing, model building, and decision making. The arrows indicate the flow of information into and out of the agent.

3 What's in a Model?

In moving from the initial intuitive definition of emergence to the more concrete notion of pattern formation and ending with intrinsic emergence, it became clear that the essential novelty involved had to be referred to some evaluating entity. The relationship between novelty and its evaluation can be made explicit by thinking always of some observer that builds a model of a process from a series of measurements. At the level of the intuitive definition of emergence, the observer is that which recognizes the "something" and evaluates its "newness". In pattern formation, the observer is the scientist that uses prior concepts — e.g. "spiral" or "vortex"

— to detect structure in experimental data and so to verify or falsify their applicability to the phenomenon at hand. Of the three, this case is probably the most familiarly appreciated in terms of an "observer" and its internal "model" of a phenomenon. Intrinsic emergence is more subtle. The closure of "newness" evaluation pushes the observer inside the system, just as the adaptive agents are inside the prototype universe. This requires in turn that intrinsic emergence be defined in terms of the "models" embedded in the observer. The observer in this view is a subprocess of the entire system. In particular, the observer subprocess is one that has the requisite information processing capability with which to take advantage of the emergent patterns.

"Model" is being used here in a sense that is somewhat more generous than found in daily scientific practice. There it often refers to an explicit representation — an analog — of a system under study. Here models will be seen in addition as existing implicitly in the dynamics and behavior of a process. Rather than being able to point to (say) an agent's model of its environment, the designer of the prototype universe may have to excavate the "model". To do this one might infer that an agent's responses are in co-relation with its environment, that an agent has memory of the past, that the agent can make decisions, and so on. Thus, "model" here is more "behavioral" than "cognitive".

4 The Modeling Dilemma

The utility of this view of intrinsic emergence depends on answering a basic question: How does an observer understand the structure of natural processes? This includes both the scientist studying nature and an organism trying to predict aspects of its environment in order to survive. The answer requires stepping back to the level of pattern formation.

A key modeling dichotomy that runs throughout all of science is that between order and randomness. Imagine a scientist in the laboratory confronted after days of hard work with the results of a recent experiment — summarized prosaically as a simple numerical recording of instrument responses. The question arises, What fraction of the particular numerical value of each datum confirms or denies the hypothesis being tested and how much is essentially irrelevant information, just "noise" or "error"?

A fundamental point is that *any* act of modeling makes a distinction between data that is accounted for — the ordered part — and data that is not described — the apparently random part. This distinction might be a null one: for example, for either completely predictable or ideally random (unstructured) sources the data is explained by one descriptive extreme or the other. Nature is seldom so simple. It appears that natural processes are an amalgam of randomness and order. It is the organization of the interplay between order and randomness that makes nature "complex". A complex process then differs from a "complicated" process, a large system consisting of very many components, subsystems, degrees of freedom, and so on. A complicated system — such as an ideal gas — needn't be complex, in the sense used here. The ideal gas has no structure. Its microscopic dynamics are accounted for by randomness.

Experimental data are often described by a whole range of candidate models that are statistically and structurally consistent with the given data set. One important variation over this range of possible "explanations" is where each candidate draws the randomness-order distinction. That is, the models vary in the regularity captured and in the apparent error each induces.

It turns out that a balance between order and randomness can be reached and used to define a "best" model for a given data set. The balance is given by minimizing the model's size while minimizing the amount of apparent randomness. The first part is a version of Occam's dictum: causes should not be multiplied beyond necessity. The second part is a basic tenet of science: obtain the best prediction of nature. Neither component of this balance can be minimized alone, otherwise absurd "best" models would be selected. Minimizing the model size alone leads to huge error, since the smallest (null) model captures no regularities; minimizing the error alone produces a huge model, which is simply the data itself and manifestly not a useful encapsulation of what happened in the laboratory. So both model size and the induced error must be minimized together in selecting a "best" model. Typically, the sum of the model size and the error is minimized.[19–23]

From the viewpoint of scientific methodology the key element missing in this story of what to do with data is how to measure structure or regularity. Just how structure is measured determines where the order-randomness dichotomy is drawn. This particular problem can be solved in principle: we take the size of the candidate model as the measure of structure. Then the size of the "best" model is a measure of the data's intrinsic structure. If we believe the data is a faithful representation of the raw behavior of the underlying process, this then translates into a measure of structure in the natural phenomenon originally studied.

Not surprisingly, this does not really solve the problem of quantifying structure. In fact, it simply elevates it to a higher level of abstraction. Measuring structure as the length of the description of the "best" model assumes one has chosen a language in which to describe models. The catch is that this representation choice builds in its own biases. In a given language some regularities can be compactly described, in others the same regularities can be quite baroquely expressed. Change the language and the same regularities could require more or less description. And so, lacking prior God-given knowledge of the appropriate language for nature, a measure of structure in terms of the description length would seem to be arbitrary.

And so we are left with a deep puzzle, one that precedes measuring structure: How is structure discovered in the first place? If the scientist knows beforehand the appropriate representation for an experiment's possible behaviors, then the amount of that kind of structure can be extracted from the data as outlined above. In this case, the prior knowledge about the structure is verified by the data if a compact, predictive model results. But what if it is not verified? What if the hypothesized structure is simply not appropriate? The "best" model could be huge or, worse, appear upon closer and closer analysis to diverge in size. The latter situation is clearly not tolerable. At the very least, an infinite model is impractical to manipulate. These situations indicate that the behavior is so new as to not fit (finitely) into current understanding. Then what do we do?

This is the problem of "innovation". How can an observer ever break out of inadequate model classes and discover appropriate ones? How can incorrect assumptions be changed? How is anything new ever discovered, if it must always be expressed in the current language?

If the problem of innovation can be solved, then, as the preceding development indicated, there is a framework which specifies how to be quantitative in detecting and measuring structure.

5 A Computational View of Nature

Contemporary physics does not have the tools to address the problems of innovation, the discovery of patterns, or even the practice of modeling itself, since there are no physical principles that define and dictate how to measure natural structure. It is no surprise, though, that physics does have the tools for detecting and measuring complete order — equilibria and fixed point or periodic behavior — and ideal randomness — via temperature and thermodynamic entropy or, in dynamical contexts, via the Shannon entropy rate and Kolmogorov complexity. What is still needed, though, is a definition of structure and way to detect and to measure it. This would then allow us to analyze, model, and predict complex systems at the emergent scales.

One recent approach is to adapt and extend ideas from the theory of discrete computation, which has developed measures of information-processing structure, to inferring complexity in dynamical systems.[24] Computation theory defines the notion of a "machine" — a device for encoding the structures in discrete processes. It has been argued that, due to the inherent limitations of scientific instruments, all an observer can know of a process in nature is a discrete-time, discrete-space series of measurements. Fortunately, this is precisely the kind of thing — strings of discrete symbols, a "formal" language — that computation theory analyzes for structure.

How does this apply to nature? Given a discrete series of measurements from a process, a machine can be constructed that is the best description or predictor of this discrete time series. The structure of this machine can be said to be the best approximation to the original process's information-processing structure, using the model size and apparent error minimization method discussed above. Once we have reconstructed the machine, we can say that we understand the structure of the process.

But what kind of structure is it? Has machine reconstruction discovered patterns in the data? Computation theory answers such questions in terms of the different classes of machines it distinguishes. There are machine classes with finite memory, those with infinite one-way stack memory, those with first-in first-out queue memory, those with counter registers, and those with infinite random access memory, among others. When applied to the study of nature, these machine classes reveal important distinctions among natural processes. In particular, the computationally distinct classes correspond to different types of pattern or regularity.

Given this framework, one talks about the structure of the original process in terms of the complexity of the reconstructed machine. This is a more useful notion of complexity than measures of randomness, such as the Kolmogorov complexity, since it indicates the degree to which information is processed in the system, which accords more closely to our intuitions about what complexity should mean. Perhaps more importantly, the reconstructed machine describes *how* the information is processed. That is, the architecture of the machines themselves represents the organization of the information processing, that is, the intrinsic computation. The reconstructed machine is a model of the mechanisms by which the natural process manipulates information.

6 Computational Mechanics: Beyond Statistics, Toward Structure

That completes the general discussion of the problem of emergence and the motivations behind a computational approach to it. A number of concrete steps remain to implement and test the utility of this proposal. In particular, a key step concerns how a machine can be reconstructed from a series of discrete measurements of a process. Such a reconstruction is a way that an observer can model its environment. In the context of biological evolution, for example, it is clear that to survive agents must detect regularities in their environment. The degree to which an agent can model its environment in this way depends on its own computational resources and on what machine class or language it implicitly is restricted to or explicitly chooses when making a model. The second key step concerns how an agent can jump out of its original assumptions about the model class and, by induction, can leap to a new model class which is a much better way of understanding its environment. This is a formalization of what is colloquially called "innovation".

The overall goal, then, concerns how to detect structures in the environment — how to form an "internal model" — and also how to come up with true innovations to that internal model. There are applications of this approach to time series analysis and other areas, but the main goal is not engineering but scientific: to understand how structure in nature can be detected and measured and, for that matter, discovered in the first place as wholly new innovations in one's assumed representation.

What is new in this approach? Computation theorists generally have not applied the existing structure metrics to natural processes. They have mostly limited their research to analyzing scaling properties of computational problems; in particular, to how difficulty scales in certain information processing tasks. A second aspect computation theory has dealt with little, if at all, is measuring structure in stochastic processes. Stochastic processes, though, are seen throughout nature and must be addressed at the most basic level of a theory of modeling nature. The domain of computation theory — pure discreteness, uncorrupted by noise — is thus only a partial solution. Indeed, the order-randomness dichotomy indicates that the interpretation of any experimental data has an intrinsic probabilistic component which is induced by the observer's choice of representation. As a consequence probabilistic computation must be included in any structural description of nature. A third aspect computation theory has considered very little is measuring structure in processes that are extended in space. A fourth aspect it has not dealt with traditionally is measuring structure in continuous-state processes. If computation theory is to form the foundation of a physics of structure, it must be extended in at least these three ways. These extensions have engaged a number of workers in dynamical systems recently, but there is much still to do.[24–30]

7 Agenda

The remainder of the discussion focuses on temporal information processing and the first two extensions — probabilistic and spatial computation — assuming that the observer is looking at a series of measurements of a continuous-state system whose states an instrument has discretized.

The phrase “calculi of emergence” in the title emphasizes the tools required to address the problems which intrinsic emergence raises. The tools are (i) dynamical systems theory with its emphasis on the role of time and on the geometric structures underlying the increase in complexity during a system’s time evolution, (ii) the notions of mechanism and structure inherent in computation theory, and (iii) inductive inference as a statistical framework in which to detect and innovate new representations. The proposed synthesis of these tools develops as follows.

First, Part II defines a complexity metric that is a measure of structure in the way discussed above. This is called “statistical complexity”, and it measures the structure of the minimal machine reconstructed from observations of a given process in terms of the machine’s size. Second, Part II describes an algorithm — ϵ-machine reconstruction — for reconstructing the machine, given an assumed model class. Third, Part III presents an algorithm for innovation — called hierarchical ϵ-machine reconstruction — in which an agent can inductively jump to a new model class by detecting regularities in a *series* of increasingly-accurate models. Fourth, the remainder of Part III analyzes several examples in which these general ideas are put into practice to determine the intrinsic computation in continuous-state dynamical systems, recurrent hidden Markov models, and cellular automata. Finally, Part IV concludes with a summary of the implications of this approach for detecting and understanding the emergence of structure in evolving populations of adaptive agents.

PART II
MECHANISM AND COMPUTATION

Probably the most highly developed appreciation of hierarchical structure is found in the theory of discrete computation, which includes automata theory and the theory of formal languages.[31–33] The many diverse types of discrete computation, and the mechanisms that implement them, will be taken in the following as a framework whose spirit is to be emulated and extended. The main objects of attention in discrete computation are strings, or words, ω consisting of symbols s from a finite alphabet: $\omega = s_0 s_1 s_2 \ldots s_{L-1}, s_i \in \mathcal{A} = \{0, 1, \ldots, k-1\}$. Sets of words are called formal languages; for example, $\mathcal{L} = \{\omega_0, \omega_1, \ldots, \omega_m\}$. One of the main questions in computation theory is how difficult it is to “recognize” a language — that is, to classify any given string as to whether or not it is a member of the set. “Difficulty” is made concrete by associating with a language different types of machines, or automata, that can perform the classification task. The automata themselves are distinguished by how they utilize various resources, such as memory or logic operations or even the available time, to complete the classification task. The amount and type of these resources determine the “complexity” of a language and form the basis of a computational hierarchy — a road map that delineates successively more “powerful” recognition mechanisms. Particular discrete computation problems often reduce to analyzing the descriptive capability of an automaton, or of a class of like-structured automata, in terms of the languages it can recognize. This duality, between languages as sets and automata as functions which recognize sets, runs throughout computation theory.

Although discrete computation theory provides a suggestive framework for investigating hierarchical structure in nature, a number of its basic elements are antithetical to scientific practice. Typically, the languages are countable and consist of arbitrary length, but finite words. This restriction clashes with basic notions from ergodic theory, such as stationarity, and from physics, such as the concept of a process that has been running for a long time, that is, a system in equilibrium. Fortunately, many of these deficiencies can be removed, with the result that the concepts of complexity and structure in computation theory can be usefully carried over to the empirical sciences to describe how a process's behavioral complexity is related to the structure of its underlying mechanism. This type of description will be one of the main points of review in the following. Examples later on will show explicitly how nonlinear dynamical systems have various computational elements embedded in them.

But what does it mean for a physical device to perform a computation? How do its dynamics and the underlying device physics support information processing? Answers to these questions need to distinguish two notions of computation. The first, and probably more familiar, is the notion of "useful" computation. The input to a computation is given by the device's initial physical configuration. Performing the computation corresponds to the temporal sequence of changes in the device's internal state. The result of the computation is read off finally in the state to which the device relaxed. Ultimately, the devices with computational utility are those we have constructed to implement input-output mappings of interest to us. In this type of computation an outside observer must interpret the end product as useful: it involves a semantics of utility. One of the more interesting facets of useful computation is that there are universal computers that can emulate any discrete computational process. Thus, in principle, only one type of device needs to be constructed to perform any discrete computation.

In contrast, the second notion — "intrinsic" computation — focuses on how structures in a device's state space support and constrain information processing. It addresses the question of how computational elements are embedded in a process. It does not ask if the information produced is useful. In this it divorces the semantics of utility from computation. Instead, the analysis of a device's intrinsic computation attempts to detect and quantify basic information processing elements — such as memory, information transmission and creation, and logical operations.[34]

1 Road Maps to Innovation

With this general picture of computation the notion of a computational hierarchy can be introduced. Figure 2 graphically illustrates a hierarchy of discrete-state devices in terms of their computational capability. Each circle there denotes a class of languages. The abbreviations inside indicate the class's name and also, in some cases, the name of the grammar and/or automaton type. Moving from the bottom to the top one finds successively more powerful grammars and automata and harder-to-recognize languages. The interrelationships between the classes is denoted with a line: if class $\mathcal{M}_i$ is below and connected to $\mathcal{M}_j$, then $\mathcal{M}_j$ recognizes all of the languages that $\mathcal{M}_i$ does and more. The hierarchy itself is only a partial ordering of descriptive capability. Some classes are not strictly comparable. The solid lines indicate inclusion: a language lower in the diagram can be recognized by devices at higher levels, but there are languages at higher levels not recognizable at lower levels. The least powerful models, at the hierarchy's bottom, are those with finite memory — the finite automata (DFA/NFA). At the top are the universal Turing machines (UTM) which have infinite random-access tape memories. In between, roughly speaking, there are two broad classes of language: context-sensitive languages that can be recognized by machines whose infinite memories are organized in a stack, and context-sensitive languages recognized by machines whose memory accesses are limited by a linear function of the initial input's length. What is remarkable about this hierarchy is the wealth of intervening model classes and the accumulated understanding of their relative language classification powers. Figure 2 includes more detail than is necessary for the following discussion, but it does demonstrate some of the diversity of computational mechanisms that have been studied.[31]

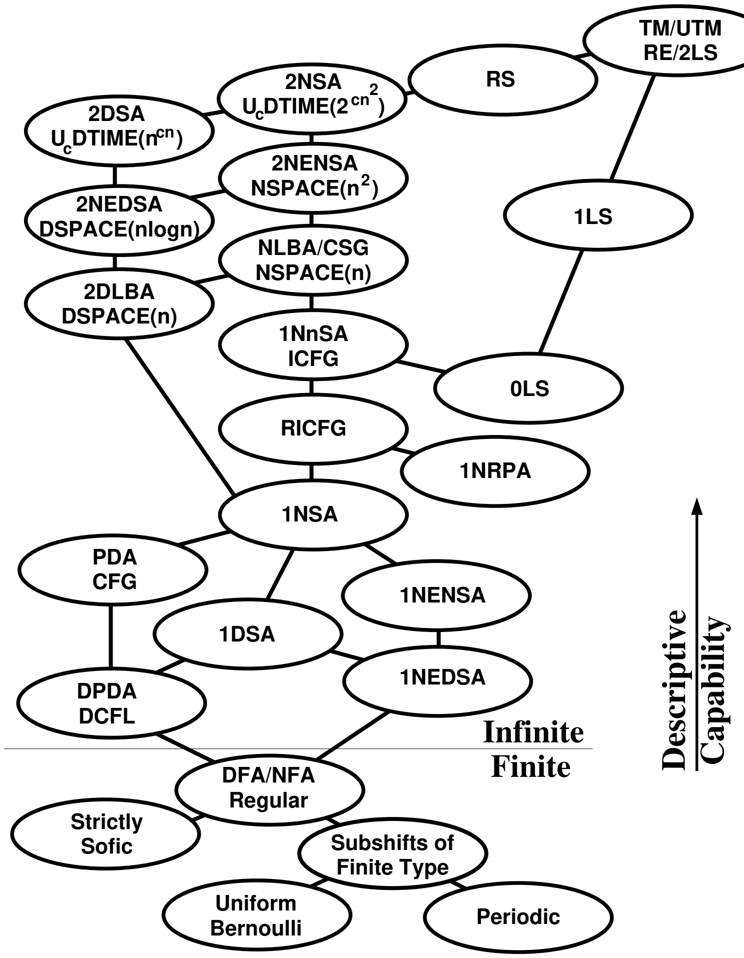

TM/UTM
RE/2LS
RS
2NSA
U_cDTIME(2^{cn^2})
2DSA
U_cDTIME(n^{cn})
2NENSA
NSPACE(n^2)
2NEDSA
DSPACE(nlogn)
NLBA/CSG
NSPACE(n)
2DLBA
DSPACE(n)
1LS
1NnSA
ICFG
0LS
RICFG
1NRPA
1NSA
PDA
CFG
1NENSA
1DSA
1NEDSA
DPDA
DCFL
Infinite
Finite
DFA/NFA
Regular
Strictly
Sofic
Subshifts of
Finite Type
Uniform
Bernoulli
Periodic
Descriptive
Capability

Figure 2 The discrete computation hierarchy. *Adjective legend*: 1 = one way input tape, 2 = two way input tape, D = deterministic, N = nondeterministic, I = indexed, RI = restricted I, n = nested, NE = nonerasing, CF = context free, CS = context sensitive, R = recursive, RE = R enumerable, and U = universal. *Object legend*: G = grammar, A = automata, FA = finite A, PDA = pushdown A, SA = stack A, LBA = linear bounded A, RPA = Reading PDA, TM = Turing machine, LS = Lindenmayer system, 0L = CF LS, 1L = CS LS, and RS = R set. (After [31,35–39].)

Figure 2 includes the formal grammar models of Chomsky and others, the associated finite and stack automata, and the arbitrary-access tape machines of Turing. Hierarchical structure should not be thought of as being limited to just these, however. Even staying within the domain of discrete symbol manipulation, there are the (Lindenmayer) parallel-rewrite[40] and queue-based[41,42] computational models. There are also the arithmetic and analytic hierarchies of recursive function theory.[43] The list of discrete computation hierarchies seems large because it is and needs to be to capture the distinct types of symbolic information processing mechanisms.

Although the discrete computation hierarchy of Figure 2 can be used to describe information processing in some dynamical systems, it is far from adequate and requires significant extensions. Several sections in Part III discuss three different extensions that are more appropriate to computation in dynamical systems. The first is a new hierarchy for stochastic finitary processes. The second is a new hierarchy for discrete spatial systems. And the third is the ϵ-machine hierarchy of causal inference. A fourth and equally important hierarchy, which will not be discussed in the following, classifies different types of continuous computation.[26,30] The benefit of pursuing these extensions is found in what their global organization of classes indicates about how different representations or modeling assumptions affect an observer's ability to build models. What a natural scientist takes from the earliest hierarchy — the Chomsky portion of Figure 2 — is the spirit in which it was constructed and not so much its details. On the one hand, there is much in the Chomsky hierarchy that is deeply inappropriate to general scientific modeling. The spatial and stochastic hierarchies introduced later give an idea of those directions in which one can go to invent computational hierarchies that explicitly address model classes which are germane to the sciences. On the other hand, there is a good deal still to be gleaned from the Chomsky hierarchy. The recent proposal to use context-free grammars to describe nonlocal nucleotide correlations associated with protein folding is one example of this.[44]

2 Complexity $\neq$ Randomness

The main goal here is to detect and measure structure in nature. A computational road map only gives a qualitative view of computational capability and so, within the reconstruction framework, a qualitative view of various types of possible natural structure. But empirical science requires quantitative methods. How can one begin to be quantitative about computation and therefore structure?

Generally, the metrics for computational capability are given in terms of "complexity". The complexity $C(x)$ of an object x is taken to be the size of its minimal representation $M_{\min}(x|\mathcal{V})$ when expressed in a chosen vocabulary $\mathcal{V}$: $C(x) = ||M_{\min}(x|\mathcal{V})||$. x can be thought of as a series of measurements of the environment. That is, the agent views the environment as a process which has generated a data stream x. Its success in modeling the environment is determined in large part by the apparent complexity $C(x)$. But different vocabularies, such as one based on using finite automata versus one based on pushdown stack automata, typically assign different complexities to the same object. This is just the modeling dilemma discussed in Part I.

Probably the earliest attempt at quantifying information processing is due to Shannon and then later to Chaitin, Kolmogorov, and Solomonoff. This led to what can be called a "deterministic" complexity, where "deterministic" means that no outside, e.g. stochastic, information source is used in describing an object. The next subsection reviews this notion; the subsequent one introduces a relatively new type called "statistical complexity" and compares the two.

2.1 Deterministic Complexity

In the mid-1960s it was noted that if the vocabulary was taken to be programs for universal Turing machines, then a certain generality obtained to the notion of complexity. The Kolmogorov-Chaitin complexity $K(x)$ of an object x is the number of bits in the smallest program that outputs x when run on a universal deterministic Turing machine (UTM).[45–47] The main deficiency that results from the choice of a universal machine is that $K(x)$ is not computable in general. Fortunately, there are a number of process classes for which some aspects of the deterministic complexity are well understood. If the object in question is a string s^L of L discrete symbols produced by an information source, such as a Markov chain, with Shannon entropy rate h_μ,[48] then the growth rate of the Kolmogorov-Chaitin complexity is

$$\frac{K\left(s^L\right)}{L} \underset{L \rightarrow \infty}{\longrightarrow} h_\mu \tag{1}$$

The growth rate h_μ is independent of the particular choice of universal machine. In the modeling framework it can be interpreted as the error rate at which an agent predicts successive symbols in s^L.

Not surprisingly, for chaotic dynamical systems with continuous state variables and for the physical systems they describe, we have

$$K\left(s_\epsilon^L\right) \underset{\substack{L \rightarrow \infty \\ \epsilon \rightarrow 0}}{\propto} h_\mu L \tag{2}$$

where the continuous variables are coarse-grained at resolution ϵ into discrete "measurement" symbols $s_\epsilon \in \left\{0, 1, 2, \ldots, \epsilon^{-d} - 1\right\}$ and d is the state space dimension.[49] Thus, there are aspects of deterministic complexity that relate directly to physical processes. This line of investigation has led to a deeper (algorithmic) understanding of randomness in physical systems. In short, $K(x)$ is a measure of randomness of the object x and, by implication, of randomness in the process which produced it.[50]

2.2 Statistical Complexity

Roughly speaking, the Kolmogorov-Chaitin complexity $K(x)$ requires accounting for all of the bits, including the random ones, in the object x. The main consequence is that $K(x)$, considered as a number, is dominated by the production of randomness and so obscures important kinds of structure in x and in the underlying process. In contrast, the statistical complexity $C_\mu(x)$ discounts the computational effort the UTM expends in simulating random bits in x. One of the defining properties of statistical complexity is that an ideal random object x has $C_\mu(x) = 0$. Also, like $K(x)$, for simple periodic processes, such as $x = 0000000 \ldots 0$, $C_\mu(x) = 0$. Thus, the statistical complexity is low for both (simple) periodic and ideal random processes. If s^L denotes the first L symbols of x, then the relationship between the complexities is simply

$$K\left(s^L\right) \approx C_\mu\left(s^L\right) + h_\mu L \tag{3}$$

This approximation ignores important issues of how averaging should be performed; but, as stated, it gives the essential idea.

One interpretation of the statistical complexity is that it is the minimum amount of historical information required to make optimal forecasts of bits in x at the error rate h_μ. Thus, C_μ is not a measure of randomness. It is a measure of structure above and beyond that describable as ideal randomness. In this, it is complementary to the Kolmogorov-Chaitin complexity and to Shannon's entropy rate.

Various complexity metrics have been introduced in order to capture the properties of statistical complexity. The "logical depth" of x, one of the first proposals, is the run time of the UTM that uses the minimal representation $M_{\min}(x)$.[51] Introduced as a practical alternative to the uncomputable logical depth, the "excess entropy" measures how an agent learns to predict successive bits of x.[52] It describes how estimates of the Shannon entropy rate converge to the true value h_μ. The excess entropy has been recoined twice, first as the "stored information" and then as the "effective measure complexity".[53,54] Statistical complexity itself was introduced in Ref. [24]. Since it makes an explicit connection with computation and with inductive inference, C_μ will be the primary tool used here for quantifying structure.

2.3 Complexity Metrics

These two extremes of complexity metric bring us back to the question — What needs to be modified in computation theory to make it useful as a theory of structures found in nature? That is, how can it be applied to, say, physical and biological phenomena? As already noted, there are several explicit differences between the needs of the empirical sciences and formal definitions of discrete computation theory. In addition to the technical issues of finite length words and the like, there are three crucial extensions to computation theory: the inclusion of probability, inductive inference, and spatial extent. Each of these extensions has received some attention in theoretical computer science, coding theory, and mathematical statistics.[23,55] Each plays a prominent role in one of the examples to come later.

More immediately the extension to probabilistic computation gives a unified comparison of the deterministic and statistical complexities and so indicates a partial answer to these questions. Recall that the vocabulary underlying K consists of minimal programs that run on a deterministic UTM. We can think of C_μ similarly in terms of a Turing machine that can guess. Figure 3 shows a probabilistic generalization — the Bernoulli-Turing machine (BTM) — to the basic Turing machine model of the discrete computation hierarchy.[56] The equivalent of the road map shown in Figure 2 is a "stochastic" computation hierarchy, which will be the subject of a later section.

With the Bernoulli-Turing machine in mind, the deterministic and statistical complexities can be formally contrasted. For the Kolmogorov-Chaitin complexity we have

$$K(x) = ||M_{\min}(x|\mathrm{UTM})|| \tag{4}$$

and for the statistical complexity we have

$$C_\mu(x) = ||M_{\min}(x|\mathrm{BTM})|| \tag{5}$$

The difference between the two over processes that range from simple periodic to ideal random is illustrated in Figure 4. As shown in Figure 4(a), the deterministic complexity is a monotonically

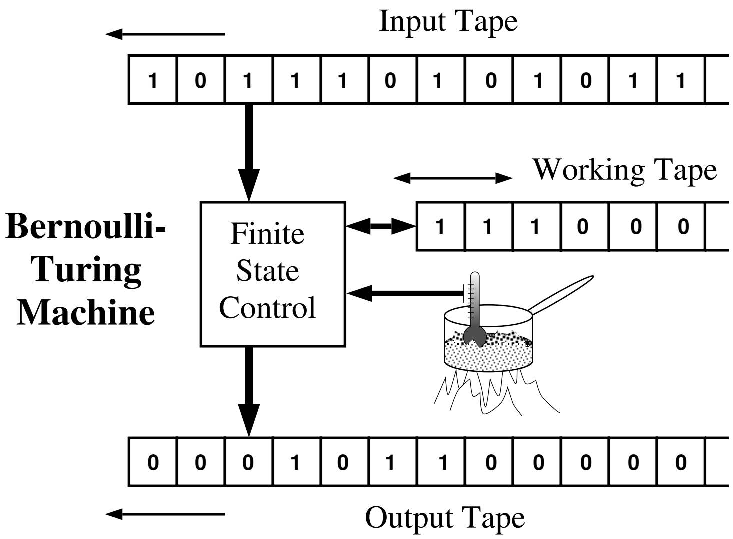


Figure 3 The Bernoulli-Turing Machine (BTM) is a deterministic Turing machine augmented by contact to an information source — a heat bath denoted as a boiling water pot. Like a Turing machine, it is a transducer that maps input tapes (0+1)* to output tapes (0+1)*. The input (output) tape cells are read (written) sequentially and once only. Any intermediate processing and storage is provided by the working tape which allows bidirectional access to its contents. The BTM defines the most general model of discrete stochastic sequential computation.

increasing function of the degree of ideal randomness in a process. It is governed by a process's Shannon entropy rate h_μ. The statistical complexity, in contrast, is zero at both extremes and maximized in the middle. (See Figure 4(b).) The "complex" processes at intermediate degrees of randomness are combinations of ordered and stochastic computational elements. The larger the number of such irreducible components composing a process, the more "complex" the process. The interdependence of randomness as measured by Shannon entropy rate and statistical complexity is a surprisingly universal phenomenon. A later section analyzes two families of dynamical systems using the complexity-entropy diagram of Figure 4(b) to describe their information processing capabilities.

It is notable, in this context, that current physical theory does not provide a measure of structure like statistical complexity. Instead one finds metrics for disorder, such as temperature and thermodynamic entropy. In a sense, physics has incorporated elements from the Kolmogorov-Chaitin framework, but does not include the elements of computation theory or of statistical complexity. There are, though, some rough physical measures of structure. These are seen in the use of group theory in crystallography and quantum mechanics. Group theoretic properties, though, only concern periodic, reversible processes or operations. Unlike ergodic theory and dynamical systems theory, contemporary physical theory is mute when it comes to quantitatively distinguishing, for example, the various *kinds* of chaotic and stochastic systems. This is what the statistical complexity is intended to provide.

The statistical complexity is a relative, not an absolute, measure of structure. It is relative to a source of ideal randomness — relative to a Random Oracle, in the parlance of computational complexity theory. A scientist might object to the use of statistical complexity, therefore, by arguing that it is important in a physical setting to account for all of the mechanisms involved in producing information. This is a fair enough comment. It acknowledges the study of randomness and it is compatible with the original spirit of Kolmogorov's program to investigate the algorithmic basis of probability. Deterministic chaos, though, has shown us that there are many sources of effective randomness in nature. One can simply use a chaotic system or appeal to the "heat bath" as an effective Random Oracle. In physics and most empirical sciences explicit accounting for random bits is neither necessary nor desirable. Ultimately, there is no contradiction between the deterministic and statistical views. Within each one simply is interested in answers to different questions.

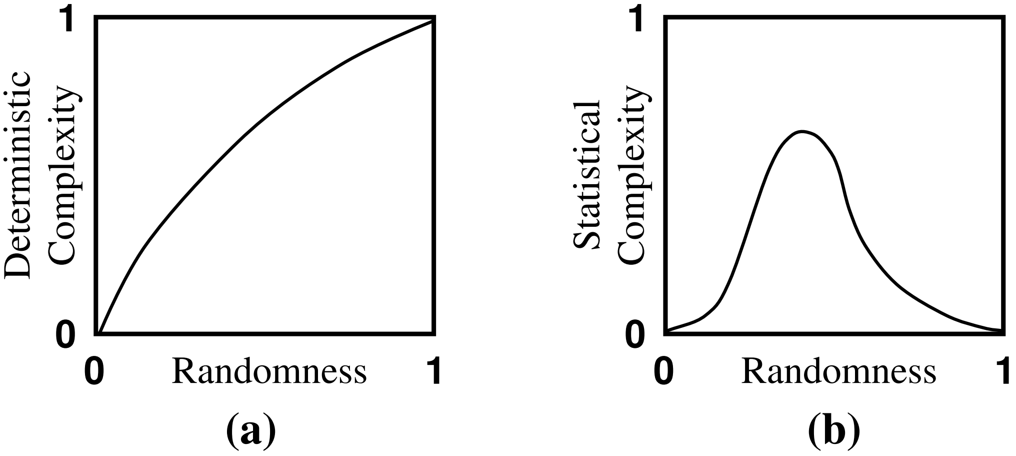


Figure 4 (a) Deterministic complexity — relative to (say) a deterministic universal Turing machine — is a measure of the degree of unpredictability of an information source. It indicates the degree of randomness which can be measured with the Shannon entropy rate h_μ. (b) Statistical complexity is based on the notion that randomness is statistically simple: an ideal random process has zero statistical complexity. At the other end of the spectrum, simple periodic processes have low statistical complexity. Complex processes arise between these extremes and are an amalgam of predictable and stochastic mechanisms. (After [56].)

The explication of the discrete computation hierarchy of Figure 2 and the two notions of deterministic and statistical complexity begins to suggest how different types of structure can be investigated. In addition to the probabilistic extension to computation theory that shed some light on the distinction between $K(x)$ and $C_\mu(x)$, another important generalization is to spatially-extended systems — those that generate "patterns" — will be the subject of later discussion. But before considering this or any other extension, the intervening sections review how complexity and randomness can be inferred from a measurement time series by an observer. The result of this will be the inductive hierarchy of ϵ-machines, which will capture the intrinsic computational structure in a process. This inductive hierarchy stands in contrast to the engineering-oriented hierarchy of Figure 2.

3 ϵ-Machine Reconstruction

How can an agent detect structure — in particular, computation — in its measurements of the environment? To answer this, let us continue with the restriction to discrete-valued time series; that is, the agent reads off a series of discrete measurements from its sensory apparatus. If one is interested in describing continuum-state systems, then this move should be seen as purely pragmatic: an instrument will have some finite accuracy, generically denoted ϵ, and individual measurements, denoted s, will range over an alphabet $\mathcal{A} = \left\{0, 1, 2, \ldots, \lceil \epsilon^{-1} \rceil - 1\right\}$. It is understood that the measurements $s \in \mathcal{A}$ are only indirect indicators of the hidden environmental states.

The goal for the agent is to detect the "hidden" states $\mathbf{S} = \{S_0, S_1, \ldots, S_{V-1}\}$ in its sensory data stream that can help it predict the environment. The states so detected will be called "causal" states. For discrete time series a causal state is defined to be the set of subsequences that render the future conditionally independent of the past. Thus, the agent identifies a state at different times in a data stream as being in identical conditions of knowledge about the future.[24] (See Figure 5 for a schematic illustration that ignores probabilistic aspects.)

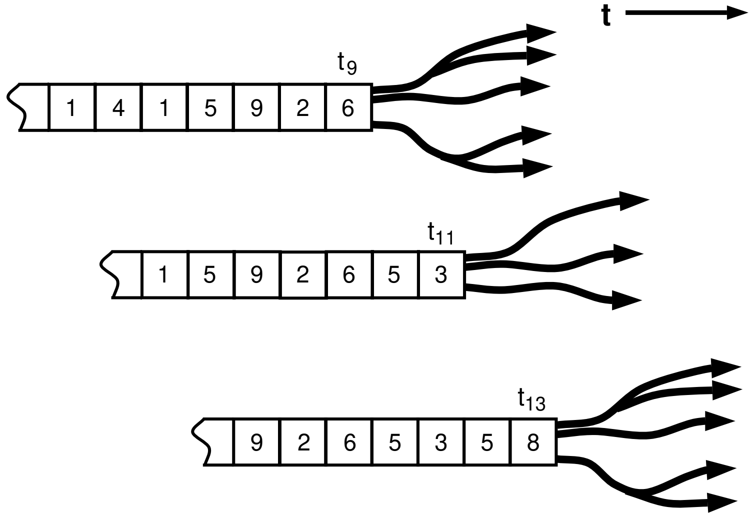


Figure 5 Within a single data stream, morph-equivalence induces conditionally-independent states. When the templates of future possibilities — that is, the allowed future subsequences and their past-conditioned probabilities — have the same structure, then the process is in the same causal state. At t_9 and at t_{13}, the process is in the same causal state since the future morphs have the same shape; at t_{11} it is in a different causal state. The figure only illustrates the nonprobabilistic aspects of morph-equivalence. (After [57].)

The notion of causal state can be defined as follows. Consider two parts of a 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. They represent the information about the future and past, respectively. Consider the joint distribution of possible forward sequences $\{\mathrm{s}^{\rightarrow}\}$ and reverse sequences $\{\mathrm{s}^{\leftarrow}\}$ over all times t:

$$\Pr(\mathrm{s}) = \Pr(\mathrm{s}^{\rightarrow}, \mathrm{s}^{\leftarrow}) = \Pr(\mathrm{s}^{\rightarrow} | \mathrm{s}^{\leftarrow}) \Pr(\mathrm{s}^{\leftarrow}) \tag{6}$$

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 wherever ω occurs in s.

Then the same causal 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,

$$t \sim t' \text{ if and only if } \Pr(\mathrm{s}^{\rightarrow} | \mathrm{s}_t^{\leftarrow}) = \Pr(\mathrm{s}^{\rightarrow} | \mathrm{s}_{t'}^{\leftarrow}) \tag{7}$$

Here "$\sim$" denotes the equivalence relation induced by equivalent future morphs. If the process generating the data stream is ergodic, then there are several comments that serve to clarify how this relation defines causal states. First, the particular sequences $\mathrm{s}_t^{\leftarrow}$ and $\mathrm{s}_{t'}^{\leftarrow}$ are typically distinct. If $t \sim t'$, Eq. (7) means that upon having seen different histories one can be, nonetheless, in the same state of knowledge 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, can 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 $\Pr(\mathrm{s}^{\rightarrow}|\mathrm{s}_{t'}^{\leftarrow})$ typically are functions over a nonempty range of "follower" sequences $\mathrm{s}^{\rightarrow}$.

This gives a formal definition to the set S of causal states as equivalence classes of future predictability: $\sim$ is the underlying equivalence relation that partitions temporal shifts of the data stream into equivalence classes. In the following the states will be taken simply as the labels for those classes. This does more than simplify the discussion. As integers ranging over $\{0, 1, 2, \ldots, ||\mathrm{S}|| - 1\}$, the states convey all of the information required to render the future conditionally independent of the past. For a given state S the set of future sequences $\left\{\mathrm{s}_S^{\rightarrow} : S \in \mathrm{S}\right\}$ that can be observed from it is called its "future morph". (Recall Fig. 5.) The set of sequences that lead to S is called its "past morph".

Note that a state and its morphs are the contexts in which an individual measurement takes on semantic content. Each measurement is anticipated or "understood" by the agent *vis á vis* the agent's internal model and, in particular, the structure of the states. This type of measurement semantics is discussed elsewhere.[34]

Once the causal states are found, the temporal evolution of the process — its symbolic dynamic — is given by a mapping $T : \mathrm{S} \rightarrow \mathrm{S}$ from states to states; that is, $S_{t+1} = TS_t$. The pair $\mathrm{M} = (\mathrm{S}, T)$ is referred to as an ϵ-machine; where ϵ simply reminds us that what we have reconstructed (i) is an approximation of the process's computational structure and (ii) depends on the measuring instrument's characteristics, such as its resolution. The procedure that begins with a data stream and estimates the number of states and their transition structure and probabilities is referred to as ϵ-machine reconstruction.[24]

What do these reconstructed machines represent? First, by the definition of future-equivalent states, the machines give the minimal information dependency between the morphs. It is in this respect that they represent the causal structure of the morphs considered as events. The machines capture the information flow within the given data stream. If state $\mathbf{B}$ follows state $\mathbf{A}$ then, as far as the observer is concerned, $\mathbf{A}$ is a cause of $\mathbf{B}$ and $\mathbf{B}$ is one effect of $\mathbf{A}$. Second, ϵ-machine reconstruction produces minimal models up to the given prediction error level. The effective error level is determined by the available inference resources. Minimality guarantees that there are no other events (morphs) that intervene, at the given error level, to render $\mathbf{A}$ and $\mathbf{B}$ independent. In this case, we say that information flows from $\mathbf{A}$ to $\mathbf{B}$. The amount of information that flows is the negative logarithm of the connecting transition probability: $-\log_2 p_{\mathbf{A} \rightarrow \mathbf{B}}$. Finally, time is the natural ordering captured by ϵ-machines.

4 Measuring Predictability and Structure

With the modeling methodology laid out, several statistics can be defined that capture how information is generated and processed by the environment as seen by an agent. A useful coordinate-independent measure of information production has already been introduced — the Shannon entropy rate h_μ.[48] If the agent knows the distribution $\Pr(\omega)$ over infinite measurement sequences ω, then the entropy rate is defined as

$$h_\mu = \lim_{L \to \infty} \frac{H\big(\Pr(s^L)\big)}{L} \tag{8}$$

in which $\Pr\big(s^L\big)$ is the marginal distribution, obtained from $\Pr(\omega)$, over the set of length L sequences s^L and H is the average of the self-information, $-\log_2 \Pr\big(s^L\big)$, over $\Pr\big(s^L\big)$. In simple terms, h_μ measures the rate at which the environment appears to produce information. Its units are bits per symbol. The higher the entropy rate, the more information produced, and the more unpredictable the environment appears to be.

Typically, the agent does not know $\Pr(\omega)$ and so the definition in Eq. (8) is not directly applicable. Assuming that the agent has observed a "typical" data stream s and that the process is ergodic, the entropy becomes

$$h_\mu = H(\Pr(s_{t+1}|\mathrm{s}_t^{\leftarrow})) \tag{9}$$

where $\Pr(s_{t+1}|\mathrm{s}_t^{\leftarrow})$ is the conditional distribution of the next symbol s_{t+1} given the semi-infinite past $\mathrm{s}_t^{\leftarrow}$ and H averages the conditional distribution over $\Pr(s^{\leftarrow})$. Using the agent's current set $\mathbf{S}$ of inferred causal states and finding the one to which $\mathrm{s}_t^{\leftarrow}$ leads, the agent can estimate the entropy in a much simpler way using

$$h_\mu = H(\Pr(s|S)) \tag{10}$$

in which $\Pr(s|S)$ is the conditional distribution of the next symbol s given the current state $S \in \mathbf{S}$.

Thinking about quantifying unpredictability in this way suggests there are other, perhaps more immediate, measures of the environment's structure. The topological complexity C_0 of a process is simply given in terms of the minimal number of causal states in the agent's model

$$C_0 = \log_2 ||\mathbf{S}|| \tag{11}$$

It is an upper bound on the amount of information needed to specify which state the environment is in. There is also a probabilistic version of the "counting" topological complexity. It is formulated as follows. The $||\mathbf{S}|| \times ||\mathbf{S}||$ transition probability matrix T determines the asymptotic causal state probabilities as its left eigenvector

$$p_{\mathbf{S}} T = p_{\mathbf{S}} \tag{12}$$

in which $p_{\mathbf{S}}$ is the causal states' asymptotic probability distribution: $\sum_{S \in \mathbf{S}} p_S = 1$. From this we have an informational quantity for the machine's size

$$C_\mu = H(p_{\mathbf{S}}) \tag{13}$$

This is the statistical complexity. If, as provided by machine reconstruction, the machine is minimal, then C_μ is the amount of memory (in bits) required for the agent to predict the environment at the given level "ϵ" of accuracy.[24]

Let's step back a bit. This section reviewed how an agent can build a model from a time series of measurements of its environment. If one considers model building to be a dynamic process, then during model construction and refinement there are two quantities, entropy rate and statistical complexity, that allow one to monitor the effectiveness and size, respectively, of the agent's model. Since the absolute difference between the environment's actual entropy rate and that of the agent's internal model determines the agent's rate of incorrect predictions, the closer the model's entropy is to that of the environment, the higher the agent's chance for survival. This survivability comes at a cost determined by the resources the agent must devote to making the predictions. This, in turn, is measured as the model's statistical complexity.

PART III

TOWARD A MATHEMATICAL THEORY OF INNOVATION

1 Reconstructing Language Hierarchies

Complexity, entropy, and ϵ-machine reconstruction itself concern incremental adaptation for an agent: the agent's "development" or its "interim" evolution when survival is viewed as an optimization and the environmental statistics are quasi-stationary. In contrast, innovation is associated with a change in model class. One would expect this change to correspond to an increase in computational sophistication of the model class, but it need not be. Roughly, innovation is the computational equivalent of speciation — recall that the partial ordering of a computational hierarchy indicates that there is no single way "up" in general. In concrete terms, innovation is the improvement in an agent's *notion* of environmental (causal) state. However it is instantiated in physical and biological processes, innovation seems to be an active process given the demonstrated robustness and creativity of life in the face of adversity. Innovation, in the narrow sense used here, should be distinguished from the passive, random forces of evolutionary change implied by mutation and recombination.

The computational picture of innovation, shown schematically in Table 1, leads to an enlarged view of the evolutionary dynamic. This can be described from the agent's view in terms of hierarchical ϵ-machine reconstruction as follows.[28,58]

1. Start at the lowest level of the computational hierarchy by building stochastic finite automata via ϵ-machine reconstruction. There are, in fact, transitions over three levels implicit in the previous introduction of ϵ-machine reconstruction; these are shown explicitly as levels 0 through 2 in Table 1. These go from the data stream (Level 0) to trees (Level 1) and then to stochastic finite automata (Level 2).
2. At any given level, if the approximations continue increasing in size as more data and resources are used in improving the model's accuracy, then "innovate" a new class when the current representation hits the limits of the agent's computational resources.

The innovation step is the evolutionary dynamic that moves from less to more capable model classes by looking for similarities between state-groups within the lower level models. This is how the agent's notion of causal state changes: from states to state-groups. The effective dynamic is one of increasing abstraction. The process is open-ended, though a possible first four levels are shown in Table 1.

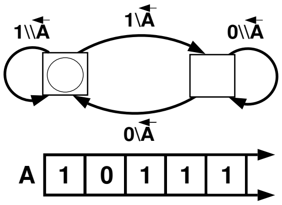
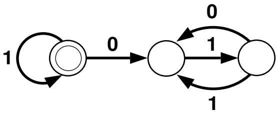
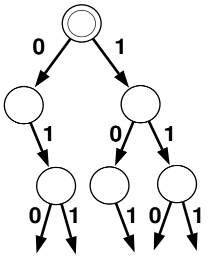
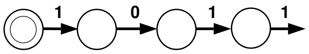
Level	Model Class	Machine	Model Size, if class is appropriate	Equivalence Relation
...	...	...	...	
3	String Production		$\mathcal{O}(\Vert\mathbf{V}\Vert + \Vert\mathbf{E}\Vert + \Vert\mathbf{P}\Vert)$	Finitary-Recursive Conditional Independence
2	Finite Automata		$\mathcal{O}(\Vert\mathbf{V}\Vert + \Vert\mathbf{E}\Vert)$	Conditional Independence
1	Tree		$\mathcal{O}\left(\Vert\mathcal{A}\Vert^{D}\right)$	Block Independence
0	Data Stream		m	Measurement

Table 1 A causal time-series modeling hierarchy. Each level is defined in terms of its model class. The models themselves consist of states (circles or squares) and transitions (labeled arrows). Each model has a unique start state denoted by an inscribed circle. The data stream itself is the lowest level. From it a tree of depth D is constructed by grouping sequential measurements into recurring subsequences. The next level models, finite automata (FA) with states $\mathbf{V}$ and transitions $\mathbf{E}$, are reconstructed from the tree by grouping tree nodes. The last level shown, string production machines (PM), are built by grouping FA states and inferring production rules $\mathbf{P}$ that manipulate strings in register $\mathbf{A}$.

Consider a data stream s of m measurements. If the source is periodic, then Level 0, the data itself, gives a representation that depends on m. In the limit $m \to \infty$ Level 0 produces an infinite representation. Level 0, of course, is the most accurate model of the data, though it is largely unhelpful and barely worth the label "model". In contrast, a depth D tree will give a finite representation, though, of a data stream with period $\leq D$, even if the data stream is infinite in length. This tree has paths of length D given by the source's period. Each of these paths corresponds to a distinct phase of the repeating pattern in s.

If s is nonperiodic, then the tree model class will no longer be finite and independent of m. Indeed, if the source has positive entropy ($h_\mu > 0$) then the tree's size will grow exponentially, $\approx ||\mathcal{A}||^{D h_\mu}$, as D is increased to account for subsequences in s of increasing length D.

If the source has, roughly speaking, correlations that decay fast enough over time, then the next level of (stochastic) finite automata, will give a finite representation. The number of states $||\mathbf{V}||$ indicates the amount of memory in the source and so the typical time over which correlations can exist between the measurements in s. But it could very well be, and examples will show this shortly, that Level 2 does not give a finite representation. Then yet another level (Level 3) will be required.

The next section gives a more precise statement of this picture. And later sections will go through several examples in detail to illustrate the dynamic of increasing abstraction. But briefly the idea is to move up the hierarchy in search of a representation that gives a finite model of the environment with optimal prediction of the environment's behavior.

2 At Each Level in a Hierarchy

To be more precise about the innovation step, let's review the common aspects across the levels in the hierarchy of Table 1; and, for that matter, in the computational hierarchy of Figure 2. At each level in a hierarchy there are a number of elements that can be identified, such as the following.

1. **Symmetries** reflecting the agent's assumptions about the environment's structure. These determine the semantic content of the model class $\mathcal{M}$, which is defined by equivalence relations $\{\sim\}$ corresponding to each symmetry.
2. **Models** M, in some class $\mathcal{M}$, consisting of states and transitions observed via measurements.
3. **Languages** being the ensembles of finitely representable behaviors.
4. **Reconstruction** being the procedure for producing estimated models. Formally, reconstruction of model $\mathrm{M} \in \mathcal{M}$ is denoted as $\mathrm{M} = \mathrm{s}/\sim$. That is, reconstruction factors out a symmetry from a data stream s.
5. **Complexity** of a process being the size of the minimal reconstructed model M with respect to the given class $\mathcal{M}$: $C(\mathbf{s}|\mathcal{M}) = ||\mathrm{M}_{\min}||$.
6. **Predictability** being estimated with reference to the distinguishable states as in Eq. (10).

It is crucial that reconstructed models $\mathrm{M} \in \mathcal{M}$ be minimal. This is so that M contains no more structure than and no additional properties beyond those in the environment. The simplest example of this is to note that there are many multiple-state representations of an ideal random binary string. But if the size of representation is to have any meaning, such as the amount of memory, only the single state process can be allowed as the model from which complexity is computed.

3 The ϵ-Machine Hierarchy

At this level of analysis — namely, discussing the structure of a hierarchy of model classes — the relativity of information, entropy, and complexity becomes clear. They all depend on the agent's assumed representation. Indeed, the representation's properties determine what their values can mean to the agent.

ϵ-machine reconstruction was introduced above as a way for the agent to detect causal states. Although causal states as formulated here can be related to notions of state employed in other fields, it should be clear now that there is an inductive hierarchy delineated by different notions of state. Once this is appreciated, the full definition of an ϵ-machine can be given. An ϵ-machine is that

> *minimal* model at the
> *least* computationally powerful level yielding a
> *finite* description.

The definition builds in an adaptive notion that the agent initially might not have the correct model class. How does it find a better representation? Moving up the inductive hierarchy can be associated with the innovation of new notions of causal state and so new representations of the environment's behavior. In formal terms, an ϵ-machine is reconstructed at some level in the computational hierarchy when hierarchical reconstruction — considered as an operator on representations — falls onto a fixed point. One can envision a procedure, analogous to the schematic view in Table 1, that implements this incremental movement up the hierarchy as follows.

1. At the lowest level, the data stream is its own, rather degenerate and uninformative, model: $\mathrm{M}_0 = \mathrm{s}$. Initially set the hierarchy level indicator to one step higher: $l = 1$.
2. Reconstruct the level l model M_l from the lower level model by factoring out the regularities — equivalence classes — in the state transition structure of the lower level model M_{l-1}: $\mathrm{M}_l = \mathrm{M}_{l-1}/\sim$, where $\sim$ denotes the equivalence relation defining the level l causal-state equivalence classes. Literally, one looks for regularities in groups of states in M_{l-1}. The groups revealing regularity in M_{l-1} become the causal states of M_l; the transitions between the M_{l-1}-state groups become the transitions in M_l.
3. Test the parsimony of the l-level class's descriptive capability by estimating successively more accurate models. As before, the degree of approximation is generally denoted ϵ, with $\epsilon \rightarrow 0$ being the limit of increasingly accurate models.
4. If the model complexity diverges, $\|\mathrm{M}_l\| \underset{\epsilon \rightarrow 0}{\rightarrow} \infty$, then set $l \leftarrow l + 1$ and go back to 2 and move up another level.
5. If $\|\mathrm{M}_l\| \underset{\epsilon \rightarrow 0}{<} \infty$, then the procedure has found the first level that is the least computationally powerful and that gives a finite description. An ϵ-machine has been reconstructed. Quit.

The essential idea in moving up the hierarchy is that the symmetries assumed by the agent are broken by the data when reconstruction leads to an infinite model at some level of representation. The process of going from step 4 back to step 2 — i.e. of jumping up the hierarchy to a new model class — is what has been referred to as "innovation". The key step in innovating a new model class is the discovery of new equivalence relations. A large part of this, though, is simply a reapplication of ϵ-machine reconstruction: discovering new structure is done by grouping lower-level states into equivalence classes of the same future morph. These equivalence classes then become the notion of causal state at the new higher level. A series of increasingly-accurate lower level models are, in this sense, a data stream — $\mathrm{M}_{l-1}(\epsilon), \mathrm{M}_{l-1}(\frac{\epsilon}{2}), \mathrm{M}_{l-1}(\frac{\epsilon}{4}), \mathrm{M}_{l-1}(\frac{\epsilon}{8}), \ldots$ — for reconstruction at the next higher level M_l. A section to follow shortly will show that, for example, at the onset of chaos hierarchical ϵ-machine reconstruction goes across four levels — data, trees, finite automata, and stack automata — before finding a finite representation. The details in Table 1 were selected in anticipation of those results.

There is an additional element beyond the grouping of states according to their transition (morph) structure, though. This will be seen shortly in the section on hidden Markov models as the innovation of a resettable counter register,[59] at the onset of chaos as the innovation of string productions,[56] and in discrete spatial processes as the innovation of regular domains, domain walls, and particles.[60] It is also seen in the innovation of local state machines to break away from cellular automata look-up table representations; an example of this can be found elsewhere.[29] In each case it is quite straightforward to find the additional structural element riding on top of the higher-level causal states. But since, as far as is known, no one has delineated an exhaustive and ordered spectrum of basic computational elements, innovation must contain a component, albeit small, of undetermined discovery.

The meta-reconstruction algorithm results in a hierarchy of computation classes — the ϵ-machine hierarchy. Unlike the generative hierarchy of Chomsky,[31] this is a causal hierarchy for inductive inference. It takes into account the possibility, for example, that causal recognition might be distinct from the complexity of the generating process.

4 The Threshold of Innovation

When should innovation occur? A basic premise here is that an agent can only call upon finite resources. The answer then is straightforward. Innovation should occur as the agent's modeling capacity, denoted $||\mathbf{A}||$, is approached by the complexity of the agent's internal model M. That is, the threshold of innovation is reached when $||\mathrm{M}|| \approx ||\mathbf{A}||$. To be more explicit about what is happening, one can use a diagnostic for innovating a new model class. Let $C_\mu^l(\epsilon)$ denote the complexity of one model $M_l(\epsilon)$ in the increasing-accuracy series. Then the innovation rate $\mathcal{I}_l$ at the given level is defined

$$\mathcal{I}_l = \lim_{\epsilon \to 0} - \frac{2^{C_\mu^l(\epsilon)}}{\log_2 \epsilon} \tag{14}$$

The innovation rate monitors the increase in model size. If $\mathcal{I}_l > 0$ the model size at level l diverges and the agent will have to innovate a new model class at the first accuracy threshold ϵ' where $C_\mu^l(\epsilon') > ||\mathbf{A}||$. Failure to do so is tantamount to precluding the use of an enhanced notion of environmental state to represent new forms of regularity. The ultimate result of failing to innovate is that some deterministic aspect of the environment will appear forever random. The consequence may be, nonetheless, a perfectly appropriate balance of evolutionary forces; there is a reason why houseflies and humans coexist in the same environment.

It turns out that $\mathcal{I}_l$ has a simpler interpretation. First, note that from Eq. (14), it can be rewritten

$$\mathcal{I}_l = \lim_{\epsilon \to 0} \left(2^{C_\mu^l(\epsilon/2)} - 2^{C_\mu^l(\epsilon)} \right) \tag{15}$$

Expanding this, one finds

$$\mathcal{I}_l = \lim_{\epsilon \to 0} \sum_{\substack{v \in \mathbf{V}(\epsilon/2) \\ v' \in \mathbf{V}(\epsilon)}} p_v \log_2 \frac{p_v}{p_{v'}} \tag{16}$$

where $\mathbf{V}(\epsilon)$ is the sets of states in $\mathrm{M}_l(\epsilon)$. Thus, $\mathcal{I}$ is the information gain in going from one model to a more accurate one. Under ϵ-machine reconstruction the states $v' \in \mathbf{V}\left(\frac{\epsilon}{2}\right)$ of the more accurate model come from the "splitting" of states $v' \in \mathbf{V}(\epsilon)$ in the less accurate model.

One might be tempted to define a single number χ for hierarchical complexity, such as

$$\chi = l.\hat{\mathcal{I}}_l \in [0, \infty) \tag{17}$$

where l is the (integer) level above the raw data stream at which an ϵ-machine is reconstructed and $\hat{\mathcal{I}}_l = 1 - 2^{-C^l} \in [0, 1]$ is the fractional complexity at that level. Although in some circumstances this could be useful, it is ultimately doomed, since there is no linear order of computational capability. The hierarchies are only partial orderings.

Casting innovation in this formal light emphasizes one important consequence: When confronted with hierarchical processes, finite computational resources fuel the drive toward higher complexity — toward agents with internal models of increasing computational power.

5 Examples of Hierarchical Learning

The preceding sections laid out an abstract framework for computation, dynamics, and innovation. The intention was to show how the different calculi of emergence are related and how together they address the problem of inadequate representations both qualitatively and quantitatively. The discussion was couched in terms of an agent that learns models of an environment via a data stream of sensory measurements.

The following sections take a more concrete approach and demonstrate how several of these general ideas are put into practice. In a sense, the following examples put us in the position of the agents above. The examples analyze the intrinsic computation in a wide range of processes: continuous-state dynamical systems, hidden Markov models, and cellular automata. The intention here is not only to be explicit, but to also broaden the notion of computation that has been used up to this point.

5.1 The cost of chaos

The following three subsections review how intrinsic discrete computation is embedded in two well-known continuous-state dynamical systems. The connection between discrete computation and the continuous states is made via symbolic dynamics. In this approach a continuous-state orbit is observed through an instrument that produces very coarse, in fact binary, measurements. To detect the intrinsic computation the resulting binary data stream is fed into ϵ-machine reconstruction to produce a minimal computational model. The resulting ϵ-machine describes the intrinsic computational capability of the observed process — dynamical system plus instrument. Due to the choice of a particular type of instrument, the ϵ-machine also describes the computational capability of the hidden dynamical system.

Intrinsic computation in the period-doubling cascade

The first dynamical system to be analyzed for computational structure is the logistic map and, in particular, its period-doubling route to chaos. The data stream used for reconstructing models is derived from a trajectory of the logistic map when it is started with an initial condition on its attractor. This makes the observed process stationary. The trajectory is generated by iterating the map

$$x_{n+1} = f(x_n) \tag{18}$$

with the logistic function $f(x) = rx(1-x)$, with nonlinearity parameter $r \in [0,4]$ and initial condition $x_0 \in [0,1]$. Note that the map's maximum occurs at $x_c = \frac{1}{2}$. The orbit $\mathbf{x} = x_0 x_1 x_2 x_3 \ldots$ is converted to a discrete sequence by observing it via the binary partition

$$\mathcal{P} = \{x_n \in [0, x_c) \Rightarrow s = 0, x_n \in [x_c, 1] \Rightarrow s = 1\} \tag{19}$$

This partition is "generating" which means that sufficiently long binary sequences come from arbitrarily small intervals of initial conditions. Due to this, the information processing in the logistic map can be studied using the "coarse" measuring instrument $\mathcal{P}$.

Many investigations of the logistic map concentrate on how its time-asymptotic behavior, its attractor, changes with the nonlinearity parameter r. Here, however, the interest is in how its various information processing capabilities are related to one another. The two basic measures of this that can be directly taken from the reconstructed ϵ-machines were introduced above. The first was the statistical complexity C_μ, which is the size of the reconstructed ϵ-machine or, equivalently, the effective amount of memory in the logistic map. The second measure of information processing is the entropy rate h_μ, which is the rate in bits per time step at which information is produced. The net result of using just the complexity and entropy rate is that the original equations of motion and the nonlinearity parameter are simply forgotten. All that is of interest is how the complexity C_μ of the data stream depends on the rate h_μ of information production.

The complexity-entropy plot of Figure 6(a) summarizes this relationship by showing the results of reconstructing ϵ-machines from data streams produced at different parameter values. For each data set produced, an ϵ-machine is reconstructed and its statistical complexity C_μ and entropy rate h_μ are estimated. In order to show the full range of behavior, from periodic to chaotic, the latter is estimated as $h_\mu(L) = H(L)/L$ where $H(L)$ is the Shannon information of length L sequences. Figure 6(a) is simply a scatter plot of the estimated complexity-entropy pairs, in emulation of Figure 4(b).

There are a number of important features exhibited by the complexity-entropy diagram. (Details are given in Refs. [24] and [56].) The first is that the extreme values of entropy lead to zero complexity. That is, the simplest periodic process at $H(L)/L = 0$ and the most random one at $H(L)/L = 1$ are statistically simple. They both have zero complexity since they are described by ϵ-machines with a single state. Between the extremes the processes are noticeably more complex with an apparent peak about a critical entropy value denoted H_c. Below this entropy, it turns out, all of the data streams come from parameters at which the logistic map

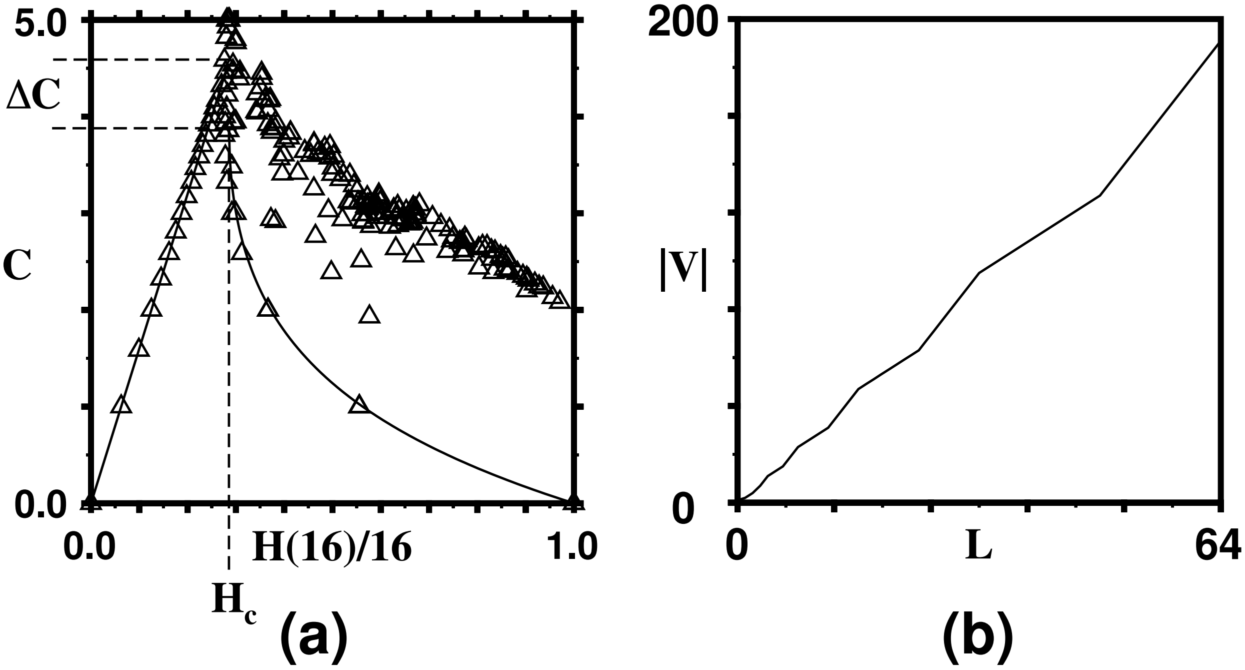


Figure 6 (a) Statistical complexity C_μ versus specific entropy $H(L)/L$ for the period-doubling route to chaos. Triangles denote estimated $(C_\mu, H(L)/L)$ at 193 values of the logistic map nonlinearity parameter. ϵ-machines were reconstructed using a subsequence length of $L = 16$. The heavy solid lines overlaying some of this empirical data are the analytical curves derived for C_0 versus $H_0(L)/L$. (After [24].) (b) At one of the critical parameter values of the period-doubling cascade in the logistic map the number $\|\mathbf{V}\|$ of inferred states grows without bound. Here $r = r_c \approx 3.5699456718695445\ldots$ and the sequence length ranges up to $L = 64$ where $\|\mathbf{V}\| = 196$ states are found. It can be shown, and can be inferred from the figure, that the per symbol density of states $\|\mathbf{V}(L)\|/L$ does not have a limiting value as $L \rightarrow \infty$. (After [56].)

is periodic — including parameters within the "periodic windows" found in the map's chaotic regime. The data sets with $H(L)/L > H_c$ are produced at chaotic parameter values.

A theory was developed in Ref. [56] to explain the emergence of high computational capability between the ordered and disordered regimes. For processes with $H(L)/L < H_c$ the entropy and complexity are equivalent

$$C_\mu = H \tag{20}$$

This is shown as a solid straight line on the left portion of Figure 6(a). For processes with $H(L)/L > H_c$ the dependence of complexity on entropy is more interesting. In fact, the solution is given in terms of the dependence of the entropy on the topological complexity. The result, a lower bound, is that

$$H(L) = C_0 + \log_2 \left(2^{L2^{-C_0}} - 2^{-1} \right) \tag{21}$$

The curved solid line in Figure 6(a) shows the relevant portion of Eq. (21).

Comparing the periodic and chaotic analyses — i.e. Eqs. (20) and (21) — provides a detailed picture of the complexity-entropy phase transition. The critical entropy H_c at each sequence length L is given

$$H_c(L) = C'(L) + \log_2 \left(by - 2^{-1} \right) \tag{22}$$

where $C'(L) = \log_2 L - \log_2 \log_2 y$ is the complexity on the high entropy side at H_c, $y \approx 2.155535$ is the solution of $y \log_e y - y + 2^{-1} = 0$, and $1 \leq b \leq 3 \cdot 2^{-\frac{3}{2}} \approx 1.06066$ is a constant. From Eq. (20) it follows immediately that the complexity C'' on the low-entropy side of the transition is itself $H_c \cdot L$. The difference is a finite constant — the latent complexity of the transition $\Delta C = C'' - C' \approx 0.7272976$ bits. The latent complexity is independent of the sequence length.

This analysis of the interdependence of complexity and entropy is nonasymptotic in the sense that it applies at each sequence length L. If, as done for Figure 6(a), this length is fixed at $L = 16$, the preceding results predict the transition's location. The critical entropy there, for example, is $H_c \approx 0.286205$. But for any L the overall behavior is universal. All behaviors with specific entropy densities $H(L)/L < H_c$ are periodic. All behaviors with higher entropy densities are chaotic. The functional forms in Eqs. (20) and (21) are general lower bounds. The statistical complexity is maximized at the border between the predictable and unpredictable "thermodynamic phases". It is important to emphasize that the complexity-entropy diagram makes no explicit reference to the system's nonlinearity parameter. The diagram was defined this way in order to show those properties which depend only on the intrinsic information production and intrinsic computational structure.

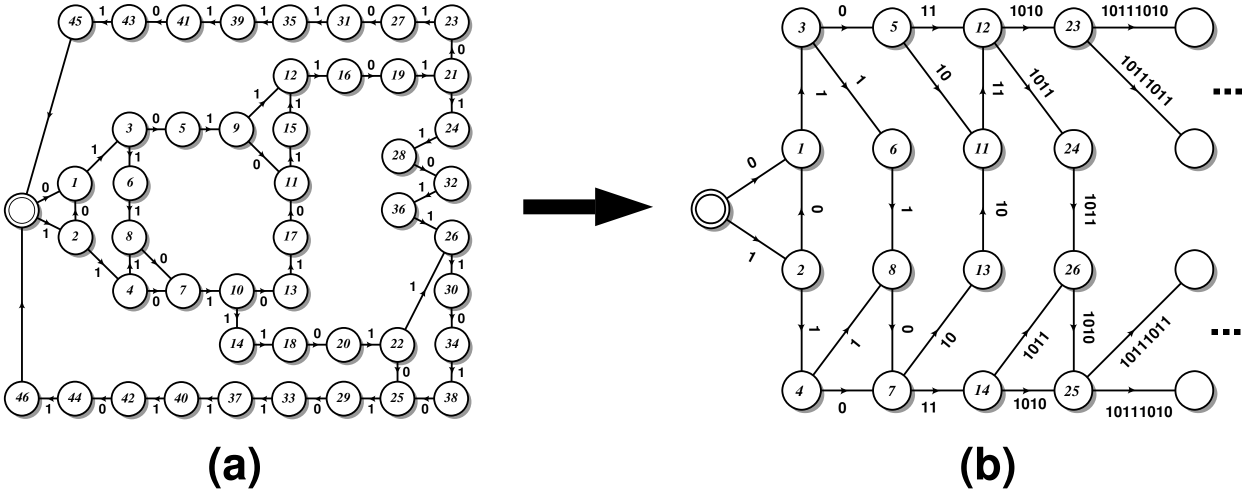


Figure 7 (a) Approximation of the critical ϵ-machine at the period-doubling onset of chaos. (After [24].) (b) The dedecorated version of the machine in (a). Here the deterministic state chains have been replaced by their equivalent strings. (After [56].)

Up to this point the overall interplay between complexity and entropy for the period-doubling cascade has been reviewed. But what happens at the phase transition; i.e. at the critical entropy density H_c? One parameter value, out of the many possible, corresponding to $H(L)/L = H_c$ is the first period-doubling onset of chaos at $r = r_c \approx 3.5699456718695445\ldots$. Figure 7(a) shows the 47 state ϵ-machine reconstructed with window size $L = 16$ at this parameter setting. An improved approximation can be attempted by increasing the window length L to take into account structure in longer subsequences. Figure 6(b) shows the result of doing just this: at the onset of period-doubling chaos the number $\|\mathbf{V}\|$ of states for the reconstructed ϵ-machines grows without bound.

The consequence is that the data stream produced at the onset of chaos leads to an infinite machine. This is consonant with the view introduced by Feigenbaum that this onset of chaos can be viewed as a phase transition at which the correlation length diverges.[61] The computational analog of the latter is that the process intrinsically has an infinite memory capacity. But there is more that the computational analysis yields. As will now be shown, for example, the infinite memory is organized in a particular way such that the logistic map is not a universal Turing machine, but instead is equivalent to a less powerful stack automaton.

The "explicit state" representation of Figure 7(a) does not directly indicate what type of information processing is occurring at the phase transition. Nor does the unbounded growth of machine size shown in Figure 6(b) give much help. A simple transformation of the 47 state machine in 7(a) goes some distance in uncovering what is happening. Replacing the unbranched "chains" in the machine with the corresponding sequences produces the "dedecorated" critical machine of Figure 7(b). In this representation is it evident that the branching states are quite regularly organized. Beyond the discovery of this higher-order regularity, there is an additional element that consists of manipulating the intervening strings between the branching states.

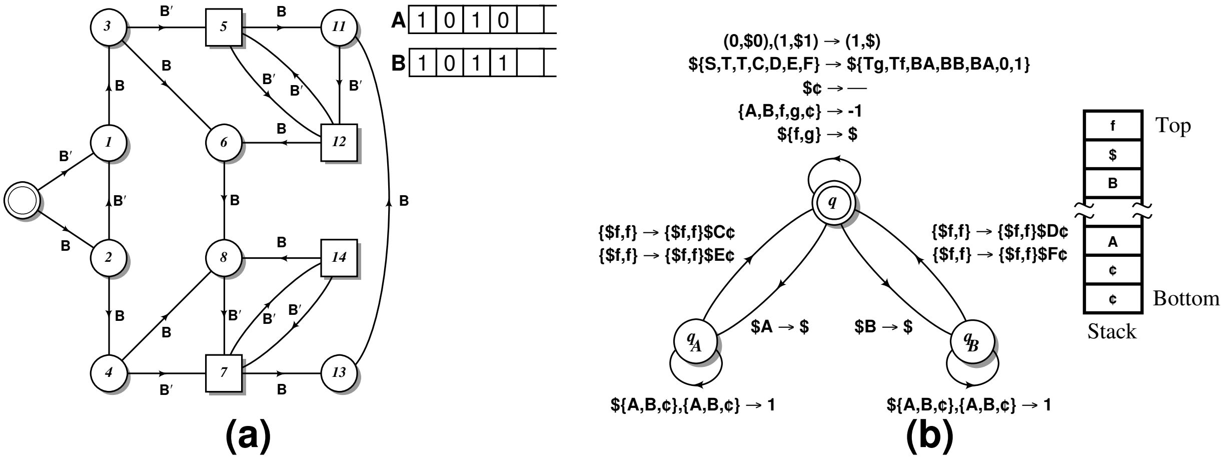


Figure 8 (a) The finite version of Figure 7(b)'s infinite critical ϵ-machine. This is a string production machine that, when making a transition from the square states, updates two string registers with the productions **A** $\rightarrow$ **BB** and **B** $\rightarrow$ **BA**. **B'** is the contents of **B** with the last bit flipped. (b) Another finite representation of the period-doubling critical ϵ-machine — a one-way nondeterministic nested stack automaton (1NnSA in Figure 2) — that produces symbols one at a time. (After [56].)

By following in detail the increasing-accuracy modeling experiment shown in Figure 6(b), one can ask *how* the machines in a series of successively-improved models grow in size. The result, as disclosed by the dedecorated machine, is that only the branching states and "string productions" are needed to describe the regularity in the growth of the machines. This in turn leads to the innovation, shown in Figure 8(a), of a finite machine with two kinds of states (the new type is denoted with squares) and two registers **A** and **B** that hold binary strings. Simple inspection of the dedecorated machine shows that the string manipulations can be described by appending a copy of **A**'s contents onto **B** and replacing the contents of **A** with two copies of **B**'s contents. These string productions are denoted **A** $\rightarrow$ **BB** and **B** $\rightarrow$ **BA**. At the outset, register **A** contains "0" and **B** contains "1".

One problem with the string production machine of Figure 8(a) is that the length of strings in the registers grows exponentially fast, which contrasts sharply with the sequential production of symbols by the logistic map. Figure 8(b) gives an alternative, but equivalent, serial machine that produces a single symbol at a time. It is called a one-way nondeterministic nested stack automaton and was denoted 1NnSA in Figure 2. The memory in this machine is organized not as string registers, but as a pushdown stack. The latter is a type of memory whose only accessible element is on the top. In fact, the automaton shown has a slightly more sophisticated stack that allows the finite control to begin a new "nested" stack within the existing one. The only restriction is that the automaton cannot move on to higher levels in the outer stack(s) until it is finished with its most recently created stack.

The net effect of these constructions is that a finite representation has been discovered from an infinite one. One of the main benefits of this, aside from producing a manageable description and the attendant analytical results it facilitates, is that the type of information processing in the critical "state" of the logistic map has been made transparent.

Intrinsic computation in frequency-locking route to chaos

The second route to chaos of interest, which also has received extensive study, is that through quasiperiodicity. In the simplest terms, this route to chaos and the models that exhibit it describe the coupling of two oscillators whose periods are incommensurate — the ratio of periods is not rational. The ratio of the number of periods of one oscillator to the other in order to complete a full cycle for both is called the winding number $\hat{\omega}$. This is a key parameter that controls the entire system's behavior: when $\hat{\omega}$ is rational the two oscillators are phase-locked. Quasiperiodic behavior is common in nature and underlies such disparate phenomena as cardiac arrhythmia, the stability of the solar system, and the puzzling synchronization of two mechanical clocks located in close proximity.

The simplest model of two "competing" oscillators is the discrete-time circle map

$$\begin{aligned} \phi_{n+1} &= f(\phi_n) \mod 1 \\ \text{where} \quad f(\phi) &= \omega + \phi + \frac{k}{2\pi}\sin 2\pi\phi \end{aligned} \tag{23}$$

The map's name derives from the fact that the $\bmod\ 1$ operation keeps the state ϕ_n on the unit circle. One thinks of ϕ_n then as a phase — or, more properly, the relative phase of the two original oscillators. There are two control parameters, ω and k. The former directly sets the phase advance and the latter the degree of nonlinearity, which can be roughly interpreted as the coupling strength between the two oscillators.

As a function of the nonlinearity parameter the behavior makes a transition to chaos. Like the logistic map, there is a signature to the path by which chaotic behavior is approached from periodic behavior. Furthermore, the circle map's signature has the basic character of a phase transition.[62]

The following will investigate one arc through (ω, k)-space that exhibits just such a phase transition to chaos. This is a path that includes the golden mean circle map — so-called since its winding number is the golden mean $\hat{\omega} = \frac{1+\sqrt{5}}{2}$. The easiest way to implement this is to set $\omega = \hat{\omega}$. Varying $k \in [0, 6]$ then gives a wide sample of behavior types on the quasiperiodic route to chaos. $k = 1$ is the threshold of nonlinear behavior, since the map for larger values becomes many-to-one; $k > 1$ is also a necessary, but not sufficient condition for deterministic chaos.

The measuring instrument uses three types of partition depending on the parameter range: $k = 0$, $k \in (0, 1]$, and $k > 1$. Generally, the instrument is a binary partition that labels $\phi_n \in (\phi', \phi'']$ with $s = 0$ and $\phi_n = (\phi'', \phi']$ with $s = 1$. For $k = 0$, $\phi' = \frac{1}{2}$ and $\phi'' = 0$; for $k \in (0, 1]$, $\phi' = \frac{1}{2}$ and $\phi'' = f^{-1}\left(\frac{1}{2}\right)$; and, for $k > 1$, ϕ' is the larger and ϕ'' the smaller value of $(2\pi)^{-1} \cos^{-1}\left(-k^{-1}\right)$ on the interval. By iterating the map many times on an initial condition a time series $\boldsymbol{\phi} = \phi_0\phi_1\phi_2\ldots$ is produced. When observed with an instrument the time series is converted to a binary string $\mathrm{s} = s_0s_1s_2\ldots$ of coarse measurements $s_i \in \{0, 1\}$.

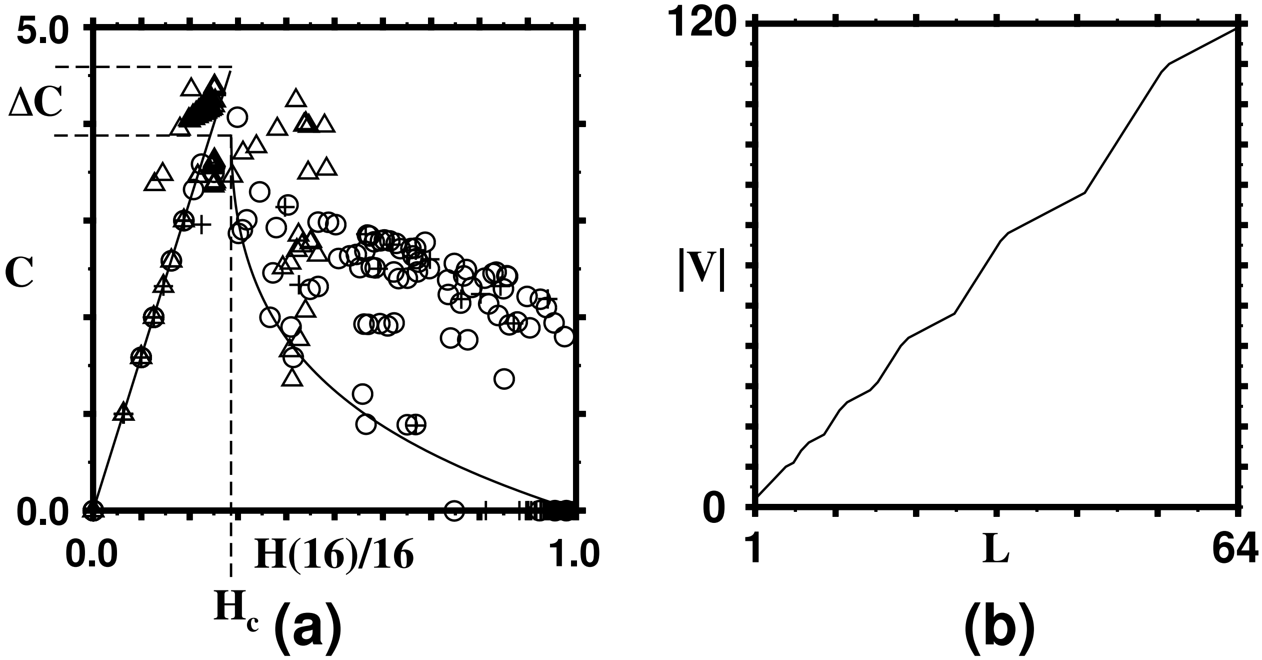


Figure 9 (a) Statistical complexity C_μ versus specific entropy $H(L)/L$ for the quasiperiodic route to chaos. Tokens denote estimated $(C_\mu, H/L)$ at 303 values of the circle map with $\omega = \frac{1+\sqrt{5}}{2}$ and nonlinearity parameter k in three different ranges: 101 values for $k \in [0, 2]$ (triangles), 101 values for $k \in [3.5, 3.9]$ (circles), and 101 values for $k \in [4.5, 6]$ (crosses). These are ranges in which the behavior is more than simple periodic. ϵ-machine reconstruction used a tree depth of $D = 32$ and a morph depth of $L = 16$ for the first range and $(D, L) = (16, 8)$ for the second two ranges, which typically have higher entropy rates. The entropy density was estimated with a subsequence length of $L = 16$. Refer to Figure 6(a) for details of the annotations. (b) At the golden mean critical winding number (with $k = 1$) in the quasiperiodic route to chaos the number $\|\mathbf{V}\|$ of inferred states grows without bound. Here the sequence length ranges up to $L = 64$ where $\|\mathbf{V}\| = 119$ states are found.

Figure 9(a) shows the complexities and entropies estimated for the quasiperiodic route to chaos at several hundred settings along the chosen parameter arc. As with period-doubling, the quasiperiodic behavior with entropies $H(L)/L < H_c$ are periodic. All those with higher entropies are unpredictable. The statistical complexity is maximized at the border between the ordered and chaotic "thermodynamic phases". The lower bounds, Eqs. (20) and (21), are shown again as solid lines for both phases. The circle map clearly obeys them, as did the logistic map, though the scatter differs. For example, there is a cluster of points just below H_c at high complexity. These are all due to the "irrational" quasiperiodic behavior that is predictable. The complexity derives from the fact that the map essentially "reads out" the digits of their irrational winding number. This leads to data streams that require large ϵ-machines to model. There is also some scatter at high entropy and low complexity. This is due to highly intermittent behavior that results in all subsequences being observed, but with an underlying probability distribution that is far from uniform. The result is that ϵ-machine reconstruction approximates the behavior as a biased coin — zero complexity, since it has a single state, and entropy less than unity.

What happens at the quasiperiodic onset at $k = 1$? The metric entropy is zero here, since the number of length L subwords increases strictly linearly: $N(L) = L + 1$. The single symbol entropy is high, $H(1) \approx 0.959419$ bits, since the frequency of isolated zeros is $\lim_{L \to \infty} \frac{F_{L-2}}{F_L} = \hat{\omega}^{-2} \approx 0.381966$, where F_L is the Lth Fibonacci number.

ϵ-machine reconstruction applied to this "critical" data stream does not lead to a finite state machine. In fact, just as for the logistic map at the onset of chaos, the machine size keeps diverging. (See Figure 9(b).) A finite approximation to the presumably infinite "critical" machine is shown in Figure 10(a).

Notably, the intrinsic computation in quasiperiodicity can be finitely represented at a next higher level. When the average winding number is the golden mean, one finds the "Fibonacci" machine shown in Figure 10(b). There is a two state finite control automaton shown at the top portion of Figure 10(b) that determines copying operations on two registers, **A** and **B**, holding binary strings. The finite control is started in the left-most, double-circled state, **A** begins with "1", and **B** with "0". The finite control machine's edges are labelled with the actions to be taken on each state-to-state transition. The first symbol on each edge label is a zero or one read from the input data stream that is to be recognized. The symbol read determines the edge taken when in a given state. The backward slash indicates that a string production is performed on registers **A** and **B**. This consists of copying the previous contents of **A** to **B** and appending the previous contents of **B** to **A**. The string productions are denoted **A** $\rightarrow$ **AB** and **B** $\rightarrow$ **A**. They are applied simultaneously. If there are two backward slashes, then two "Fibonacci" productions are performed. The input string must match the contents of register **A**, when register **A** is read in reverse. The latter is denoted by the left-going arrow above **A** in the edge label. Table 2 shows the temporal development of the contents of **A** and **B**.

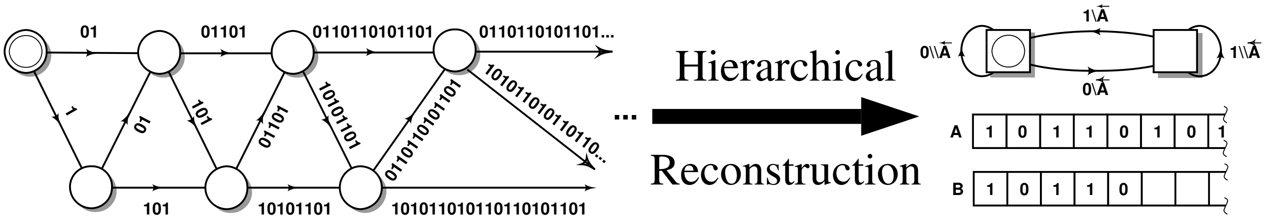


Figure 10 (a) A portion of the infinite critical machine for the quasiperiodic route to chaos at the golden mean winding number. Note that the dedecorated machine is shown — that is, the intervening states along deterministic chains have been suppressed. (b) The Fibonacci machine: the finite representation of the infinite machine in (a).

The basic computation step describing the quasiperiodic critical dynamics employs a pair of string productions. The computational class here is quite similar to that for period-doubling behavior — that is, nested stack automata. It is at this higher level that a finite description of the golden mean critical behavior is found. This is demonstrated, as for period-doubling, by noting that the productions are context-free Lindenmayer productions and that these can be mapped first to an indexed context-free grammar and then to nested stack automaton.[31] Thus, rather than Figure 10(b) the Fibonacci machine can be represented with a stack automaton analogous to that shown in Figure 8(b) for the period-doubling onset of chaos.

t	A	B	$\|\mathbf{A}\|$
1	1	0	1
2	10	1	2
3	101	10	3
4	10110	101	5
5	10110101	10110	8

Table 2 Contents of the Fibonacci machine registers **A** and **B** as a function of machine transitions. The registers contain binary strings and are modified by string concatenation: **A** → **AB** and **B** → **A**. That is, the previous contents of **A** are moved to **B** and the previous contents of **B** are appended to **A**.

The required length of the Fibonacci machine registers grows as a function of the number of applications of the production at an exponential rate which is the golden mean, since the string length grows like the Fibonacci numbers — an observation directly following from the productions. Thus, with very few transitions in the machine input strings of substantial length can be recognized.

Another interpretation of the recognition performed by the Fibonacci machine in Figure 10(b) is that it phase locks to the quasiperiodic data stream. That is, the Fibonacci machine can jump in at any point in the "critical" string, not necessarily some special starting time, and, from that symbol on, determine if the subword it is reading is in the language of all Fibonacci subwords.

Temporal computation in deterministic chaos

This investigation of the computational structure of two well-known routes to chaos show that away from the onset of chaos there are (at least) finite memory processes. Finite memory processes are all that is found below the onset — that is, with periodic processes. Above the onset the situation is much more interesting. There is a universal lower bound that the primary band-merging sequence obeys. But above this there can be more complex and highly unpredictable processes. These examples make it clear how to construct processes in this region of the complexity-entropy plane. Take a nonminimal representation of the all-sequences process, (say) one with 16 states. Add transition probabilities randomly to the outgoing edges, observing the need to have them sum to unity for each state. Typically, this machine will be minimal. And if the range of probabilities is restricted to be near 1/2, then the entropy will be high and by construction the process has a statistical complexity of about 4 bits. Now an entire family of high complexity, moderate entropy machines can be constructed by applying the period-doubling operator to the high entropy machine just created. This results in processes of lower and lower entropy and higher and higher complexity. These move down to the onset of chaos. Finally, note that the analysis of this family's complexity versus entropy dependence is not so different from that for the lower bound.

The preceding subsections also showed that to get a simple model that captures the system's *true* computational capability, as determined by observations, it is sometimes necessary to jump up to a more powerful computational class. At both onsets of chaos the computational analysis identified structures that were higher than finite memory devices. The onset of chaos led to infinite memory and, just as importantly, to memory that is organized in a particular way to facilitate some types of computation and to proscribe others. The logistic and circle maps at their respective onsets of chaos are far less than Turing machines, especially ones that are universal. At the onset the information processing embedded in them jumps from the finitary level to the level of stack automata. One practical consequence of failing to change to a more powerful representation for these critical systems is that an observer will conclude that they are more random, less predictable, and less complex, than they actually are. More generally, appreciating how infinite complexity can arise at the onset of chaos leads one to expect that highly nonlinear systems can perform significant amounts of and particular forms of information processing.

5.2 The cost of indeterminism

This section explores the possible ill-effects of measurement distortion: the apparent complexity can diverge if the "wrong" instrumentation is used. (This section follows Ref. [63].) Along the way a new class of processes will be considered — the stochastic nondeterministic finite automata, often called hidden Markov models. One of the main conclusions will be that an agent's sensory apparatus can render a simple environment apparently very complex. Thus, in an evolutionary setting the effects described here indicate that there should be a strong selection pressure on the quality of measurements produced by an agent's sensory apparatus.

The simplest example

Returning to the logistic map, let's fix its parameter to $r = 4$ — where its attractor fills the interval and has the maximal entropy rate of $h_\mu = 1$. The probability density function for the invariant measure over "internal" real-valued states $x \in [0, 1]$ is

$$\Pr(x) = \frac{1}{\pi\sqrt{x - x^2}} \tag{24}$$

Then, we associate a state $\mathbf{A}$ with the event $x_t \in [0, x_c)$ and a state $\mathbf{B}$ with the event $x_t \in [x_c, 1]$; recalling that $x_c = \frac{1}{2}$ is the map's maximum. Finally, we use a sliding-block code on the resulting $\mathbf{A} - \mathbf{B}$ stream that outputs $s = 1$ when the length 2 subsequences $\mathbf{AA}$, $\mathbf{AB}$, or $\mathbf{BB}$ occur, and $s = 0$ when $\mathbf{BA}$ occurs. The $0 - 1$ data stream that results is produced by the machine shown in Figure 11 — a stochastic nondeterministic finite automaton (SNFA).

That Figure 11 gives the correct model of this source is seen by first noting that the intermediate states $\mathbf{A}$ and $\mathbf{B}$ have the asymptotic probabilities

$$\Pr(\mathbf{A}) = \int_0^{x_c} dx \Pr(x) = \frac{1}{2} \tag{25}$$

and, by symmetry, $\Pr(\mathbf{B}) = \frac{1}{2}$. The two inverse iterates of x_c, $x_\pm = x_c \pm \frac{1}{2\sqrt{2}}$, delimit the interval segments corresponding to the occurrence of $\mathbf{A} - \mathbf{B}$ pairs. These then give the four state transition probabilities, such as

$$\Pr(\mathbf{A} \to \mathbf{A}) = \frac{\Pr(\mathbf{AA})}{\Pr(\mathbf{A})} = 2 \int_0^{x_-} dx \Pr(x) \tag{26}$$

It turns out they are all equal to $\frac{1}{2}$.

With the use of the pairwise $\mathbf{A}-\mathbf{B}$ coding this construction might seem somewhat contrived. But it can be reinterpreted without recourse to an intermediate code. It turns out that the $0-1$ data stream comes directly from the binary partition

$$\mathcal{P} = \{x_n \in [0, x_+) \Rightarrow s = 1, x_n \in [x_+, 1] \Rightarrow s = 0\} \quad (27)$$

This is a partition that is not much more complicated than the original. The main difference is that the "decision point", originally at x_c, has been moved over to x_+.

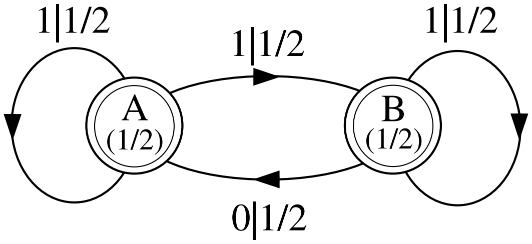


Figure 11 The source is a stochastic nondeterministic finite automaton — a class sometimes referred to as hidden Markov models. The hidden process consists of two states $\{\mathbf{A}, \mathbf{B}\}$ and uniform branching between them — denoted by the fractions p on the edge labels $s|p$. The observer does not have access to the internal state sequences, but instead views the process through the symbols s on the edge labels $s|p$. The inscribed circle in each state indicates that both states are start states. The fractions in parentheses give their asymptotic probabilities, which also will be taken as their initial probabilities.

The result is that the environment seen by the agent is described by the two-state stochastic process shown in Figure 11. There are two internal states $\{\mathbf{A}, \mathbf{B}\}$. Transitions between them are indicated with labeled, directed edges. The labels $s|p$ give the probability p of taking the transition. When the transition is taken the agent receives the measurement symbol $s \in \{0, 1\}$. In effect, the agent views the internal state dynamics through the instrument defined by the particular association of the measurement symbols and the transitions. The agent assumes no knowledge of the start state and so the environment could have started in either $\mathbf{A}$ or $\mathbf{B}$ with equal likelihood.

Figure 12 shows the minimal machine for the environment's internal state dynamics. It is the single state Bernoulli process $\mathrm{B}\left(\frac{1}{2}, \frac{1}{2}\right)$ — a fair coin. From Eqs. (10) and (12) it is evident that the metric entropy is $h_\mu = 1$ bit per symbol, as is the topological entropy h. From Eqs. (11), (12), and (13) both the topological complexity and statistical complexities are zero. It is a very random, but simple process.

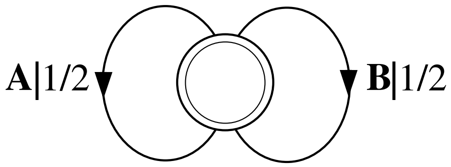


Figure 12 The minimal machine for Figure 11's internal state process. It has a single state and equal branching probabilities. The topological and statistical complexities are zero and the topological and metric entropies are 1 bit per state symbol — a highly unpredictable, but low complexity process. That this is the correct minimal description of the internal state process follows directly from using machine reconstruction, assuming direct access to the internal state sequences $\mathbf{ABABBA}\ldots$. All state sequences are allowed and those of equal length have the same probability.

The goal, of course, is for the agent to learn the causal structure of this simple process from the $\{0, 1\}$ data stream. It has no knowledge of Figure 11, for example. The overall inference procedure is best illustrated in two steps. The first is learning a model of the "topological" process that produces the set of sequences in the data stream, ignoring the probabilities with which they occur. The second step is to learn a model that gives the sequences' probabilities.

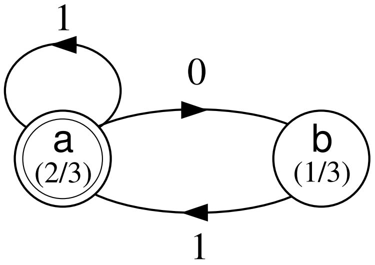


Figure 13 The process's topological structure is given by a deterministic finite automaton — the golden mean machine. The only rule defining the sequences is "no consecutive 0s". The number of sequences of length L is given by the Fibonacci number F_{L+2}; the growth rate or topological entropy h, by the golden mean $\phi = \frac{1}{2}\big(1 + \sqrt{5}\big)$: $h = \log_2 \phi$. The numbers in parentheses give the states' asymptotic probabilities.

The first step is relatively straightforward and can be explained briefly in words. Inspection of the stochastic automaton's output symbols in Figure 11 shows that if $s = 0$ is observed, then $s = 1$ must follow. Further reflection shows that this is the only restriction: consecutive 0s are not produced. All other binary sequences occur.

The automaton, again "topological", that captures this property is shown in Figure 13. This automaton is also what machine reconstruction generates. There are several things to notice. First, the state **a** has a circle inscribed in it. This denotes that **a** is the start state; and it happens to be the unique start state. The reconstructed ϵ-machine has removed the first element of non-causality in the original process: ignorance of the start state. Second, the automaton is deterministic — a term used here as it is in formal language theory and which does *not* refer to probabilistic elements. Determinism means that from each state a symbol selects a unique successor state.

Note that the original process (Figure 11) with its measurement labeling is not deterministic. If the process happens to be in state $\mathbf{A}$ and the observer then sees $s = 1$, then at the next time step the internal process can be in either state $\mathbf{A}$ or $\mathbf{B}$. This ambiguity grows as one looks at longer and longer sequences. Generally, indeterminism leads to a many-to-one association between internal state sequences and measurement sequences. In this example, the observation of 0110 could have been produced from either the internal state sequence $\mathbf{BAABA}$ or $\mathbf{BABBA}$.

The consequences of indeterminism, though, become apparent in the second inference step: learning the observed sequences' probabilities. To implement this, a series of increasingly-accurate machines approximating the process of Figure 11 is reconstructed; these are shown in Figure 14. Each gives a systematically better estimate of the original process's sequence distribution. The machine resulting from full reconstruction is shown in Figure 15. It has an infinite number of causal states. All of their transitions are deterministic. Note that the infinite machine preserves the original process's reset property: when $s = 0$ is observed the machine moves to a unique state and from this state $s = 1$ must be seen. But what happened,

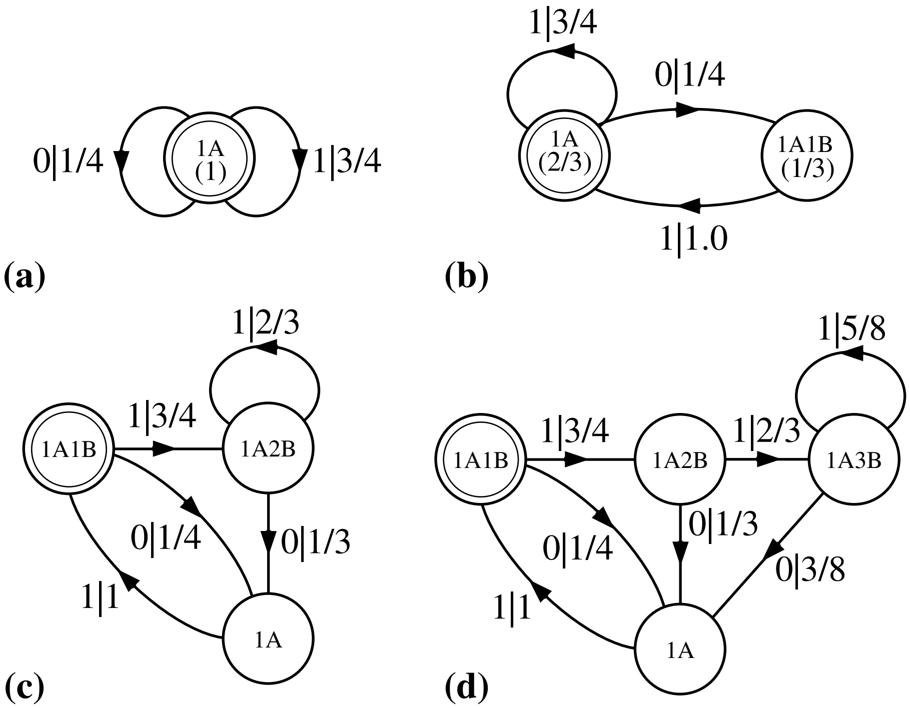


Figure 14 (a) - (d) The zeroth- through third-order causal approximations to the process of Figure 11.

in comparison to the finite machine of Figure 13, to produce the infinite machine in Figure 15? The indeterminism mentioned above for state **A** has lead to a causal representation that keeps track of the number of consecutive 1s since the last $s = 0$. For example, if 01 has been observed, then $\Pr(s = 0) = \frac{1}{4}$ and $\Pr(s = 1) = \frac{3}{4}$. But if 011 has been observed, $\Pr(s = 0) = \frac{1}{3}$ and $\Pr(s = 1) = \frac{2}{3}$. In this way the causal representation accounts for the agent's uncertainty in each internal states' contribution to producing the next symbol. The result is that as more consecutive 1s are seen the relative probability of seeing $s = 0$ or $s = 1$ continues to change — and eventually converges to a fair coin. This is reflected in the change in transition probabilities down the machine's backbone. Causal machine reconstruction shows exactly what accounting is required in order to correctly predict the transition probabilities. But it gives more than just optimal prediction. It provides an estimate of the process's complexity and a complete representation of the distribution $\Pr(\omega)$ over infinite sequences.

Interestingly, even if the agent has knowledge of Figure 11, the infinite causal machine of Figure 15 represents in a graphical way the requirements for achieving optimal predictability of the original process. There is no shortcut to computing, for example, the original process's entropy rate and complexities, since the machine in Figure 15, though infinite, is minimal. That is, there is no smaller (causal) machine that correctly gives $\Pr(\omega)$. From the topological machine it follows that the topological entropy is $h = \log_2 \phi \approx 0.694242$ and from Eqs. (10) and (12) that the metric entropy is $h_\mu \approx 0.677867$ bits per symbol. Recall that the original process's topological and statistical complexities were zero. From Eqs. (11), (12), and (13) the causal machine's topological complexity is infinite, $C_0 = \log_2 ||\mathbf{S}||$, and its statistical complexity is $C_\mu \approx 2.71147$ bits. These are rather large changes in appearance due to the instrumentation.

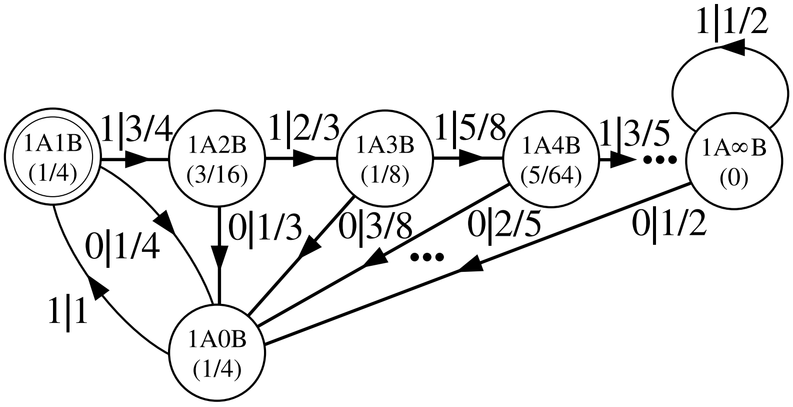


Figure 15 The infinite causal representation of the nondeterministic process of Figure 11. The labels in the states indicate the relative weights of the original internal states $\{\mathbf{A}, \mathbf{B}\}$. The numbers in parentheses are the asymptotic state probabilities: $\Pr(v = 1\mathrm{A}i\mathrm{B}) = (i+1)2^{-i-2}$.

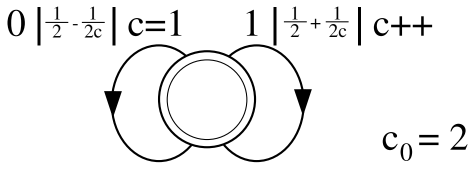

$0 \,|\, \frac{1}{2} - \frac{1}{2c} \,|\, c{=}1$
$1 \,|\, \frac{1}{2} + \frac{1}{2c} \,|\, c{+}{+}$
$c_0 = 2$

Figure 16 At a higher computational level a single state machine, augmented by a counter register, finitely describes the process of Figures 11 and 15.

In this example, the agent can be considered to have simply selected the wrong instrument. The penalty is infinite complexity. Thus, the logistic map can appear to have an infinite number of causal states and so infinite topological complexity. In contrast to the preceding sections, which illustrated infinite intrinsic complexity, this example illustrates measurement-induced complexity.

Stochastic counter automata

The apparent infinite complexity of the deterministic denumerable-state machine of Figure 15 gives way to a finite representation once the regularity of the change in transition probabilities is discovered. The resulting model — in the class of stochastic counter automata for this one example — is shown in Figure 16. The structural innovation is a counter, denoted **c**, that begins with the value 2. **c** can be either incremented by one count or reset to 1. When $s = 0$ is observed, the counter is reset to 1. As long as $s = 1$ is observed, the counter is incremented. The nondeterminism of the original process is simulated in this deterministic representation using the counter to modify the transition probabilities: it keeps track of the number of consecutive 1s. The transition probabilities are calculated using the value stored in the counter: $\Pr(s = 0|c) = \frac{1}{2} - \frac{1}{2c}$ and $\Pr(s = 1|c) = \frac{1}{2} + \frac{1}{2c}$. The finite control portion of the machine is simply a single state machine, and so its complexity is zero. But the required register length grows like $\log_2 L$. The cost of nondeterminism in this example is this increment-reset counter.

Recurrent hidden Markov models

This example is just one from a rich class of processes called — depending on the field — recurrent hidden Markov models, stochastic nondeterministic finite automata, or functions of

Markov chains. The difficulty of finding the entropy rate for these processes was first noted in the late 1950's.[64] In fact, many questions about this class are very difficult. It is only recently, for example, that a procedure for determining the equivalence of two such processes has been given.[65] The awareness that this problem area bears on the complexity of observation and the result that finite complexity processes can appear infinitely complex is also recent.[59]

The new notion of state here that needs to be innovated involves the continuity of real variables. The causal states are no longer discrete. More precisely, an ϵ-machine state for a hidden Markov model is a *distribution* over the hidden states. Since this distribution is a vector of real numbers, the causal states are continuous. In addition, this vector is normalized since the components are state probabilities. The result is that the ϵ-machine states for these processes can be graphically represented as a point in a simplex. The simplex vertices themselves correspond to δ-distributions concentrated on one of the hidden Markov model states; the dimension of the simplex is one less than the number of hidden states. In the simplex representation the ϵ-machine models are stochastic deterministic automata (SDA). Only a subset of the simplex is recurrent. This subset corresponds to an attracting set under the dynamics of the agent's step-by-step prediction of the hidden state distribution based on the current data stream. Figure 17 shows the causal state simplices for three example processes.

The infinite, but countable state machine of Figure 15(a) is shown in this simplex representation in Figure 17(a) as a discrete point set in the 3-simplex. Each point, somewhat enlarged there, corresponds to one of the states in Figure 15. Two more examples of ϵ-machines are given in Figures 17(b) and 17(c). They suggest some of the richness of stochastic finite processes. In Figure 17(b), for example, the ϵ-machine has a partial continuum of states; the causal states lie on a "fractal". Figure 17(c) shows an ϵ-machine whose causal states limit on a full continuum of states.

The different classes shown in Figure 17 are distinguished by a new complexity measure of the ϵ-machines's state structure — the ϵ-machine dimension $d_{\epsilon M}$. $d_{\epsilon M}$ is the information dimension of the state distribution on the simplex. In the case of the countable state ϵ-machine of Figure 17(a) C_μ is finite due to the strong localization of the state distribution over the earlier states. But, since the states are a discrete point set, $d_{\epsilon M} = 0$. For the fractal and continuum SDAs the statistical complexity diverges, but $d_{\epsilon M} > 0$. $d_{\epsilon M}$ is noninteger in the first case and $d_{\epsilon M} = 2$ in the second. Further results will be presented elsewhere.[59]

The finitary stochastic hierarchy

The preceding examples of stochastic finitary processes can be summarized using the computational model hierarchy of Figure 18. This hierarchy borrows the finite memory machines of Figure 2 and indicates their stochastic generalizations. As before each ellipse denotes a model class. As one moves up the diagram classes become more powerful in the sense that they can finitely describe a wider range of processes than lower classes. A class below and connected to a given one can finitely describe only a subset of the processes finitely described by the higher one. Again, this hierarchy is only a partial ordering of descriptive capability. It should be emphasized that "descriptive capability" above the Measure-Support line refers to finitely representing a *distribution* over the sequences; not just the distribution's supporting formal languages.

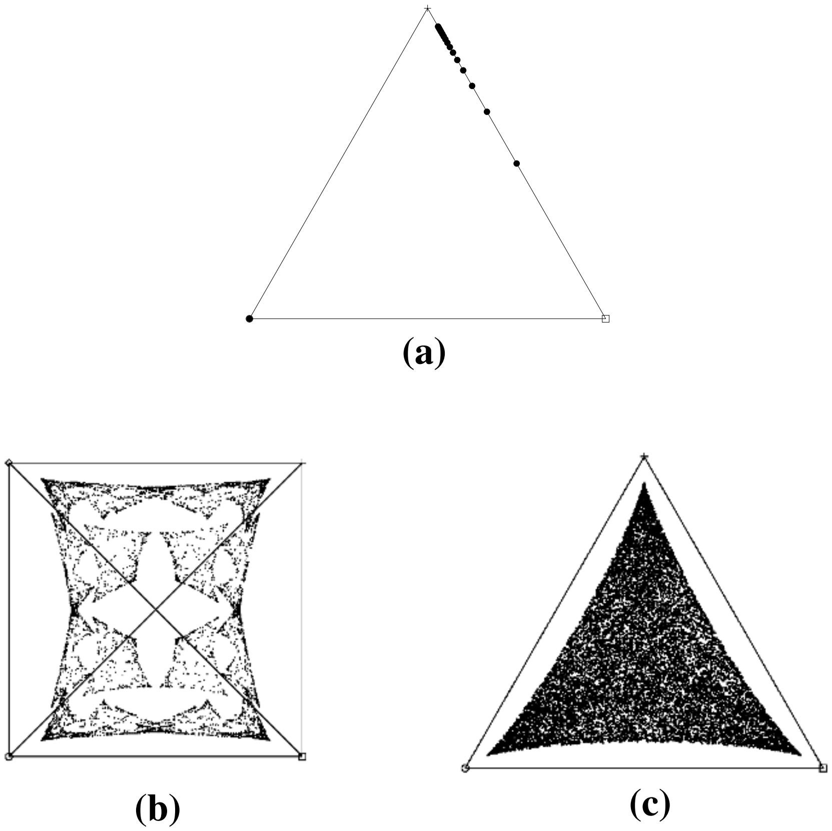


Figure 17 Stochastic deterministic automata (SDA): (a) Denumerable SDA: A denumerable ϵ-machine for the simple nondeterministic source of Figure 11. It is shown here in the (two-dimensional) 3–simplex defining its possible deterministic states (indicated with enlarged dots). Since the state probability decays exponentially, the simulation only shows a very truncated part of the infinite chain of states that, in principle, head off toward the upper vertex. Those dots correspond to the $1s$ backbone of Figure 15. The state on the lower lefthand vertex corresponds to the "reset" state $1A0B$ in that figure. (b) Fractal SDA: A nondenumerable fractal ϵ-machine shown in the 4–simplex defining the possible deterministic states. (c) Continuum SDA: A nondenumerable continuum ϵ-machine shown in the 3–simplex defining the possible deterministic states.

In formal language theory it is well-known that deterministic finite automata (DFA) are as powerful as nondeterministic finite automata (NFA).[31] This is shown in the hierarchy as both classes being connected at the same height. But the equivalence is just topological; that is, it concerns only the descriptive capability of each class for sets of observed sequences. If one augments these two classes, though, to account for probabilistic structure over the sequences, the equivalence is broken in a dramatic way — as the above example for the "mismeasured" logistic map demonstrated. This is shown in Figure 18 above the Measure-Support line. The class of SNFA is higher than that of the stochastic deterministic finite automata (SDFA). Crudely speaking, if a DFA has transition probabilities added to its edges, one obtains the single class of SDFA. But if transition probabilities are added to an NFA, then the class is qualitatively more powerful and, as it turns out, splits into three distinct classes. Each of these classes is more powerful than the SDFA class. The new causal classes — called stochastic deterministic automata (SDA) — are distinguished by having a countable infinity, a fractional continuum, or a full continuum of causal states: the examples of Figure 17.

Initially, the original logistic map process as represented in Figure 11 was undistinguished as an SNFA. Via the analysis outlined above its causal representation showed that it is equivalent

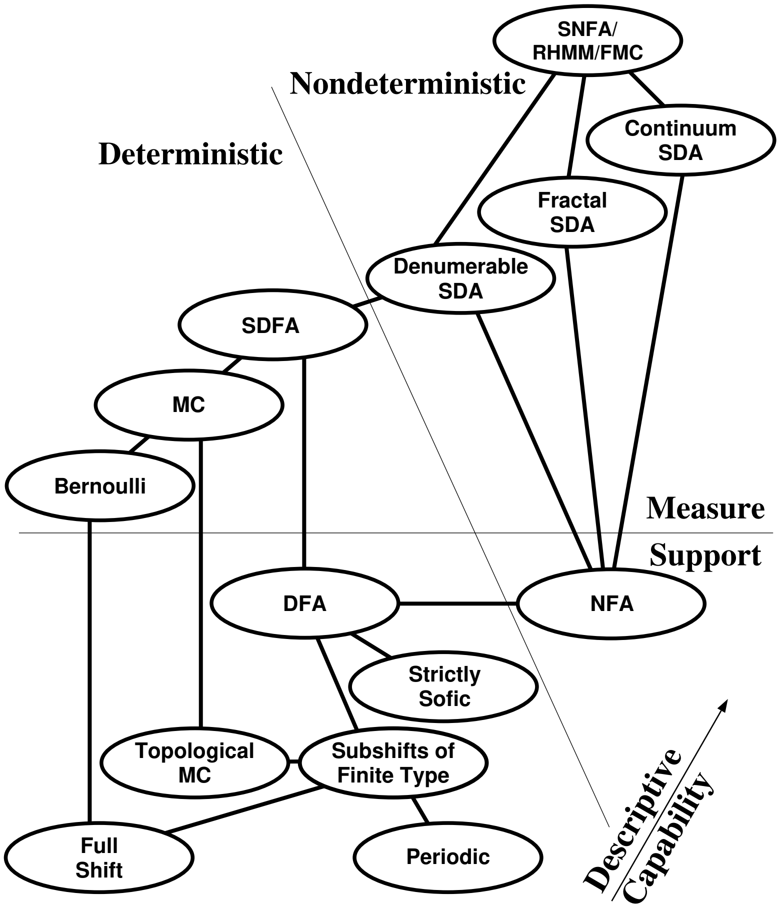

SNFA/
RHMM/FMC
Nondeterministic
Deterministic
Continuum
SDA
Fractal
SDA
Denumerable
SDA
SDFA
MC
Bernoulli
Measure
Support
DFA
NFA
Strictly
Sofic
Topological
MC
Subshifts of
Finite Type
Full
Shift
Periodic
Descriptive
Capability

Figure 18 The computational hierarchy for finite-memory nonstochastic (below the Measure-Support line) and stochastic discrete processes (above that line). The nonstochastic classes come from Figure 2, below the Finite-Infinite memory line. Here "Support" refers to the sets of sequences, i.e. formal languages, which the "topological" machines describe; "Measure" refers to sequence probabilities, i.e. what the "stochastic" machines describe. The abbreviations are: A is automaton, F is finite, D is deterministic, N is nondeterministic, S is stochastic, MC is Markov chain, HMM is hidden Markov model, RHMM is recurrent HMM, and FMC is function of MC.

to a denumerable stochastic deterministic automaton (DSDA). Generally, in terms of descriptive power $\mathrm{DSDA} \subset \mathrm{SNFA}$ as Figure 18 emphasizes. Recall that we interpret the SNFA as the environment, which is a Markov chain (MC), viewed through the agent's sensory apparatus. So the computational class interpretation of the complexity divergence is that $\mathrm{MC} \rightarrow \mathrm{DSDA}$ under measurement distortion. That is, MC and DSDA are qualitatively different classes and, in particular, $\mathrm{MC} \subset \mathrm{DSDA}$, as shown in the hierarchy of Figure 18. The representational divergence that separates them is characteristic of the transition from a lower to a higher class.

5.3 The costs of spatial coherence and distortion

As an example of hierarchical structure and innovation for spatial processes this section reviews an analysis of two cellular automata (CA). CA are arguably the simplest dynamical systems with which to study pattern formation and spatio-temporal dynamics. They are discrete in time, in space, and in local state value. The two examples considered have two local states

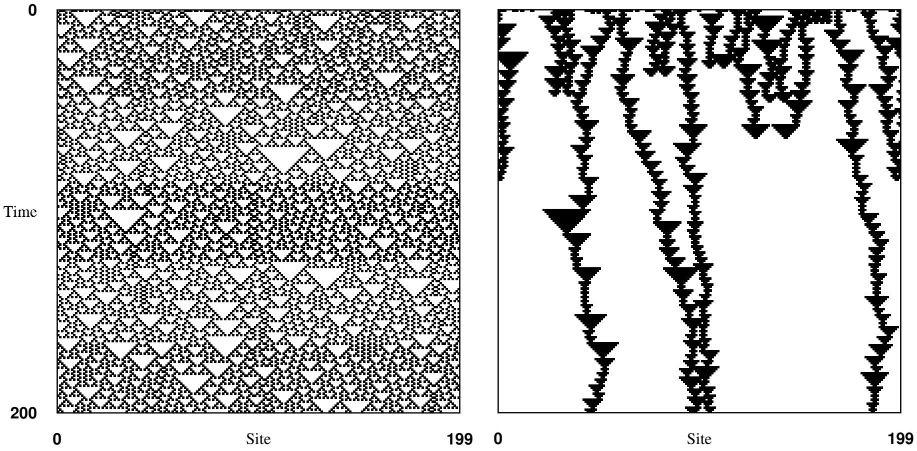


Figure 19 (a) Elementary cellular automaton 18 evolving over 200 time steps from an initial arbitrary pattern on a lattice of 200 sites. (b) The filtered version of the same space-time diagram that reveals the diffusive-annihilating dislocations obscured in the original. (After Ref. [66].)

at each site of a one-dimensional spatial lattice. The local state at a site is updated according to a rule that looks only at itself and its two nearest neighbors.

Figure 19(a) shows the temporal development of an arbitrary initial pattern under the action of elementary CA 18 (ECA 18). This rule maps all of the neighborhood patterns to 0; except 001 and 100, which are mapped to 1. A space-time diagram for elementary CA 54 (ECA 54) is shown in Figure 20(a).

ϵ-machine reconstruction applied to the patterns reveals much of the internal structure of these systems' state spaces and the intrinsic computation in their temporal development of spatial patterns.[60] An important component of CA patterns are domains. CA domains are dynamically homogeneous regions in space-time defined in terms of the same set of pattern regularities, such as "every other site value is a 0". From knowledge of a CA's domains nonlinear filters can be constructed to show the motion and interaction of domains, walls between domains, and particles. The result of this filtering process — shown in Figures 19(b) and 20(b) — is a higher level representation of the original space-time diagrams in Figures 19(a) and 20(a). This new level filters out chaotic (ECA 18) and periodic (ECA 54) backgrounds to highlight the various propagating space-time structures — dislocations, particles, and their interactions that are hidden in the unfiltered space-time diagrams.

In turns out that ECA 18 is described by a single type of chaotic domain — "every other local state is 0, otherwise they are random" — and a single type of domain wall — a dislocation that performs a diffusive annihilating random walk.[66] At the level of particles one discovers an irreducibly stochastic description — the particle motion has vanishing statistical complexity. Because the description of ECA 18 is finite at this level, one can stop looking further for intrinsic computational structure. The correct ϵ-machine level, that providing a finite description, has been found by filtering the chaotic domains.

In ECA 54 there is a single periodic domain with a diversity of particles that move and interact in complicated — aperiodic, but nonstochastic — ways. ECA 54 is more complex that ECA 18, in the sense that even at the level of particles ECA 54's description may not be finite. Thus for ECA 54, one is tempted to look for a higher level representation by performing

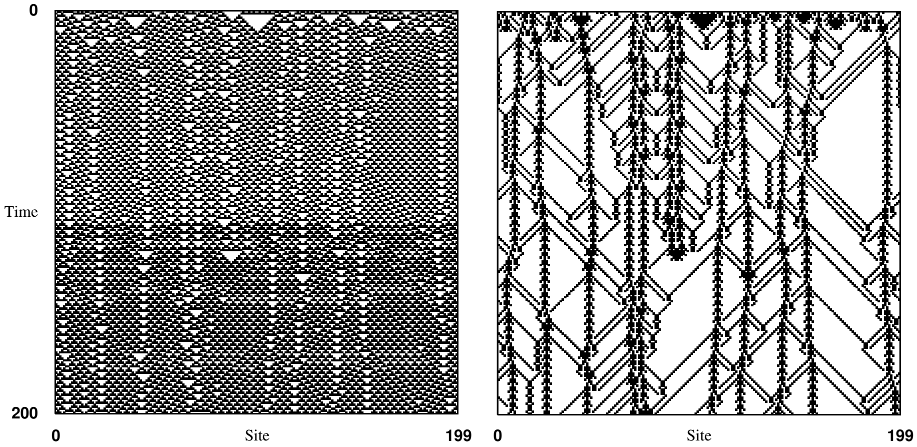


Figure 20 (a) Elementary cellular automaton 54 evolving over 200 time steps from an initial arbitrary pattern on a lattice of 200 sites. (b) The filtered version of the same space-time diagram that reveals a multiplicity of different particle types and interactions. (From Ref. [67]. Reprinted with permission of the author. Cf. [68].)

ϵ-machine reconstruction at this "particle" level to find further regularity or stochasticity and ultimately to obtain a finite description.

These hierarchical space-time structures concern different levels of information processing within a single CA. In ergodic theory terms, the unfiltered spatial patterns are nonstationary — both spatially and temporally. With the innovation of domains and particles, the new level of representation allows for stationary elements, such as the domains, to be separated out from elements which are nonstationary. For ECA 18, discovering the dislocations lead to a stochastic model based on a diffusive annihilating random walk that could be analyzed in some detail and which explained the nonstationary aspects of the dislocation motion. For ECA 54 the description at this level is still not completely understood.

Intrinsic computation in spatio-temporal processes raises a number of interesting problems. Aside from the above, there is an analog for spatio-temporal processes of the apparent complexity explosion in the stochastic nondeterministic processes. This led to the introduction of a new hierarchy for spatial automata, in which CA are the least capable, that accounts for spatial measurement distortion.[29] This spatial discrete computation hierarchy is expressed in terms of automata rather than grammars.[63]

5.4 What to glean from the examples

Understanding the hierarchical structure embedded in the preceding examples required crossing a model class boundary: from determinism to indeterminism, from finitary support to finitary measure, from predictability to chaos, from undifferentiated patterns to domains and particles, and from observed states to hidden internal states. Each of these transitions is a phase transition in the sense that there is a divergence in a macroscopic observable — the representation's complexity — and in that the finitely-describable behaviors on either side of the divergence are qualitatively distinct. Figure 21 attempts to summarize in a very schematic way their relationship.

The point that the examples serve to make is that innovation of new computational structures is required at each step. Just what needs to be innovated can be determined in very large measure by the state-grouping step in hierarchical ϵ-machine reconstruction. As was stated before, beyond the state-grouping it seems that there will always be some undetermined aspect during the innovation step required of the agent. The undetermined aspect has been isolated by the algorithmic portion of hierarchical ϵ-machine reconstruction that groups lower level states into a higher level state. To the extent it is undetermined, though, this step can be the locus of a now highly-constrained search over a space of new mechanisms. In any evolutionary process, such a space would be largely circumscribed by the agents's internal structure and the search, by how the latter can be modified. In this way, hierarchical ϵ-machine reconstruction reduces the search space for innovation by a substantial fraction and indicates where random search could be most informative and effective.

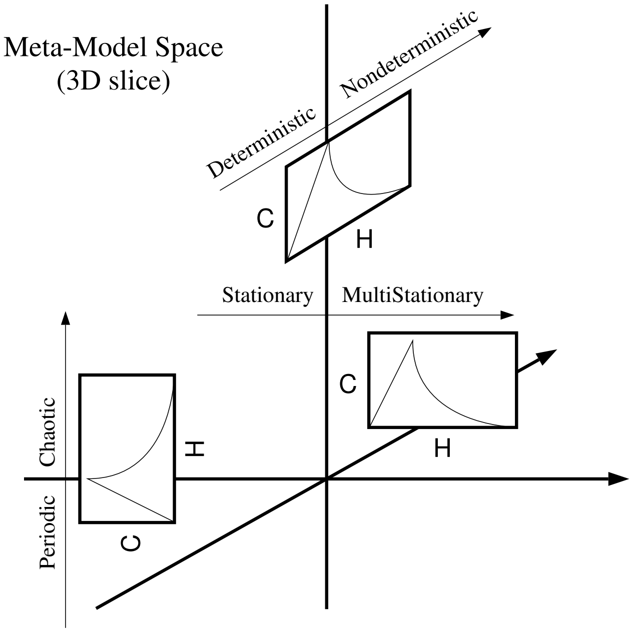


Figure 21 A schematic summary of the three examples of hierarchical learning in metamodel space. Innovation across transitions from periodic to chaotic, from stochastic deterministic to stochastic nondeterministic, and from spatial stationary to spatial multistationary processes were illustrated. The finite-to-infinite memory coordinate from Figure 2 is not shown. The periodic to chaotic and deterministic to nondeterministic transitions were associated with the innovation of infinite models from finite ones. The complexity (C) versus entropy (H) diagrams figuratively indicate the growth of computational resources that occurs when crossing the innovation boundaries.

PART IV
OBSERVATIONS AND HYPOTHESES

1 Complexity as the Interplay of Order and Chaos

Neither order nor randomness are sufficient in and of themselves for the emergence of complexity. Nor is their naive juxtaposition. As Alfred North Whitehead and many before and after him knew, order and randomness are mere components in an ongoing process. That process *is* the subtlety in the problem of emergence. For Heraclitus, the persistence of change was the by-product of an “attunement of opposite tensions”.[69] Natural language, to take one ever-present example, has come to be complex because utility required it to be highly informative *and* comprehensibility demanded a high degree of structure. For the process that is human language, emergence is a simple question: How do higher semantic layers appear out of this tension?

As far as the architecture of information processing is concerned, these questions for natural language have direct analogues in adaptation, evolution, and even in the development of scientific theories.[70] While conflating these fields suggests a possible general theory of the emergence of complexity, at the same time it reveals an ignorance of fundamental distinctions. Due to the sheer difficulty of the nonlinear dynamics involved and to the breadth of techniques that must be brought to bear, one must be particularly focused on concrete results and established theory in discussing “emergence”.

In spite of receiving much popular attention, the proposals for “computation at the edge of chaos”,[71] “adaptation toward the edge of chaos”,[72] and “life at the edge of chaos”[73] are recent examples of the problems engendered by this sort of blurring of basic distinctions. Despite the preceding sections’ review of the complexity versus entropy spectrum and their analyses of processes in which higher levels of computation arise at the onset of chaos, there is absolutely no general need for high computational capability to be near an “edge of chaos”. The infinite-memory counter register functionality embedded in the highly chaotic hidden Markov model of Figure 11 is a clear demonstration of this simple fact. More to the point, that stability and order are necessary for information storage, on the one hand, and that instability is necessary for the production of information and its communication, on the other hand, are basic requirements for any nontrivial information processing. The trade-off between these requirements is much of what computation theory is about. Moreover, natural systems are not constrained in some general way to move toward an “edge of chaos”. For example, the very informativeness ($h_\mu > 0$) of natural language means, according to Shannon, that natural language is very far and must stay away from any “edge of chaos” ($h_\mu = 0$).

A critique of the first two “edge of chaos” proposals, which concern cellular automata, can be found elsewhere.[74] Aside from the technical issues discussed there, all three proposals exhibit a fatal conceptual difficulty. To the extent that they presume to address the emergence of complexity, with reference neither to intrinsic computation nor to the innovation of new information-processing architectures, they can provide no grounding for the key concepts — “order”, “chaos”, “complexity”, “computation” — upon which their arguments rely.

The preceding development of statistical complexity, hierarchical ϵ-machine reconstruction, and various extensions to discrete computation theory is an alternative. The presentation gave a broad overview of how three tool sets — the calculi of emergence: computation, dynamics, and innovation — interrelate in three different problem areas: various routes to chaos, measurement distortion and nondeterminism, and cellular automaton pattern formation. It suggested a way to frame the question of how structure appears out of the interplay of order and randomness. It demonstrated methods for detecting and metrics for quantifying that emergence. It showed how the information processing structure of nonlinear processes can be analyzed in terms of computational models and in terms of the effective information processing.

2 Evolutionary Mechanics

The arguments to this point can be recapitulated by an operational definition of emergence. A process undergoes emergence if at some time the architecture of information processing has changed in such a way that a distinct and more powerful level of intrinsic computation has appeared that was not present in earlier conditions.

It seems, upon reflection, that our intuitive notion of emergence is not captured by the "intuitive definition" given in Part I. Nor is it captured by the somewhat refined notion of pattern formation. "Emergence" is a groundless concept unless it is defined within the context of processes themselves; the only well-defined (nonsubjective) notion of emergence would seem to be intrinsic emergence. Why? Simply because emergence defined without this closure leads to an infinite regress of observers detecting patterns of observers detecting patterns .... This is not a satisfactory definition, since it is not finite. The regress must be folded into the system, it must be immanent in the dynamics. When this happens complexity and structure are no longer referred outside. No longer relative and arbitrary, they take on internal meaning and functionality.

Where in science might a theory of intrinsic emergence find application? Are there scientific problems that at least would be clarified by the computational view of nature outlined here?

In several ways the contemporary debate on the dominant mechanisms operating in biological evolution seems ripe. Is anything ever new in the biological realm? The empirical evidence is interpreted as a resounding "yes". It is often heard that organisms today are more complex than in earlier epochs. But how did this emergence of complexity occur? Taking a long view, at present there appear to be three schools of thought on what the guiding mechanisms are in Darwinian evolution that produce the present diversity of biological structure and that are largely responsible for the alteration of those structures.

The Selectionists hold that structure in the biological world is due primarily to the fitness-based selection of individuals in populations whose diversity is maintained by genetic variation.[75] The second, anarchistic camp consists of the Historicists who hold fast to the Darwinian mechanisms of selection and variation, but emphasize the accidental determinants of biological form.[76,77] What distinguishes this position from the Selectionists is the claim that major changes in structure can be and have been nonadaptive. Lastly, there are the Structuralists whose goal is to elucidate the "principles of organization" that guide the appearance of biological structure. They contend that energetic, mechanical, biomolecular, and morphogenetic constraints limit the infinite range of possible biological form.[16,73,78–81] The constraints result in a relatively small set of structural attractors. Darwinian evolution serves, at best, to fill the waiting attractors or not depending on historical happenstance.

What is one to think of these conflicting theories of the emergence of biological structure? The overwhelming impression this debate leaves is that there is a crying need for a theory of biological structure and a qualitative dynamical theory of its emergence.[82] In short, the tensions between the positions are those (i) between the order induced by survival dynamics and the novelty of individual function and (ii) between the disorder of genetic variation and the order of developmental processes. Is it just an historical coincidence that the structuralist-selectionist dichotomy appears analogous to that between order and randomness in the realm of modeling? The main problem, at least to an outsider, does not reduce to showing that one or the other view is correct. Each employs compelling arguments and often empirical data as a starting point. Rather, the task facing us reduces to developing a synthetic theory that balances the tensions between the viewpoints. Ironically, evolutionary processes themselves seem to do just this sort of balancing, dynamically.

The computational mechanics of nonlinear processes can be construed as a theory of structure. Pattern and structure are articulated in terms of various types of machine classes. The overall mandate is to provide both a qualitative and a quantitative analysis of natural information-processing architectures. If computational mechanics is a theory of structure, then innovation via hierarchical ϵ-machine reconstruction is a computation-theoretic approach to the transformation of structure. It suggests one mechanism with which to study what drives and what constrains the appearance of novelty. The next step, of course, would be to fold hierarchical ϵ-machine reconstruction into the system itself, resulting in a dynamics of innovation, the study of which might be called "evolutionary mechanics".

The prototype universe of Figure 1 is the scaffolding for studying an abstract "evolutionary mechanics", which is distinguished from chemical, prebiotic, and biological evolution. That is, all of the "substrates" indicated in Figure 1 are thrown out, leaving only those defined in terms of information processing. This more or less follows the spirit of computational evolution.[83,84] The intention is to expunge as much disciplinary semantic content as possible so that if novel structure emerges, it does so neither by overt design nor interpretation but (i) via the dynamics of interaction and induction and (ii) according to the basic constraints of information processing. Additionally, at the level of this view mechanisms for genotypic and phenotypic change are not delineated. In direct terms evolutionary mechanics concerns the change in the information-processing architecture of interacting adaptive agents. The basic components that guide and constrain this change are the following.

1. *Modeling*: Driven by the need for an encapsulation of environmental experience, an agent's internal model captures its knowledge, however limited, of the environment's current state and persistent structure.
2. *Computation*: Driven by the need to process sensory information and produce actions, computation is the adaptive agent's main activity. The agent's computational resources delimit its inferential, predictive, and semantic capabilities. They place an upper bound on the maximal level of computational sophistication. Indirectly, they define the language with which the agent expresses its understanding of the environment's structure. Directly, they limit the amount of history the agent can store in its representation of the environment's current state.
3. *Innovation*: Driven by limited observational, computational, and control resources, innovation leads to new model classes that use the available resources more efficiently or more parsimoniously.

The computational mechanics approach to emergence attempted to address each component in turn, though it did not do so in the fully dynamic setting suggested by the prototype universe. But evolutionary mechanics is concerned specifically with this dynamical problem and, as such, it leads to a much wider range of questions.

At the very least, the environment appears to each agent as a hierarchical process. There are subprocesses at several different scales, such as those at the smallest scale where the agents are individual stochastic dynamical systems and those at larger scales at which coordinated global behavior may emerge. Each agent is faced with trying to detect as much structure as possible within this type of environment. How can an evolutionary system adapt when confronted with this kind of hierarchical process?

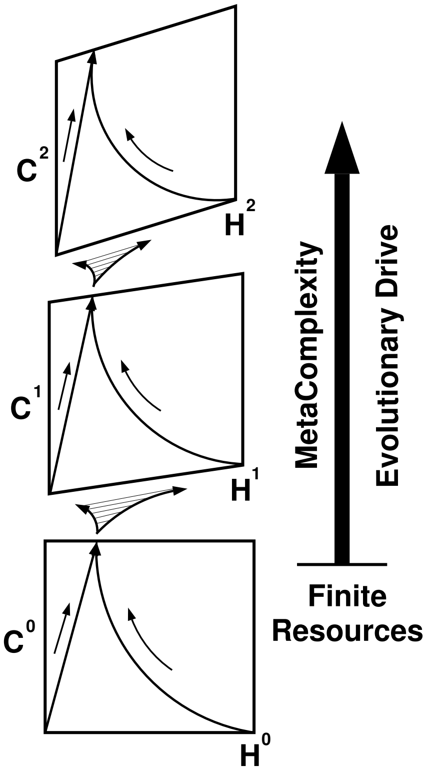


Figure 22 Schematic diagram of an evolutionary hierarchy in terms of the changes in information-processing architecture. An open-ended sequence of successively more sophisticated computational classes are shown. The evolutionary drive up the hierarchy derives from the finiteness of resources to which agents have access. The complexity-entropy diagrams are slightly rotated about the vertical to emphasize the difference in meaning at each level via a different orientation. (Cf. Table 1.)

Figure 22 gives a very schematic summary of one of the consequences following from this framework in terms of successive changes in information-processing architecture. The underlying representation of the entire prototype universe, of course, is the realization space of nonlinear processes — the orbit space of dynamical systems. The figure shows instead a temporal analog of the discrete computation hierarchy of Figure 2 in which the trade-offs within each computational level are seen through the complexity-entropy plane. The overall evolutionary dynamic derives from agent interaction and survival. Species correspond in the realization space to metastable invariant languages: they are temporary (hyperbolic) fixed points of the evolutionary dynamic. Each species is dual to some subset of environmental regularities — a niche — whose defining symmetries some subpopulation of agents has been able to discover and incorporate into their internal models through innovation. Innovation then manifests itself as a kind of speciation. The invariant sets bifurcate into computationally distinct agents with new, possibly-improved capabilities. The statistical mechanical picture of this speciation is that of a phase transition between distinct computational "thermodynamic" phases. Through all of this the macro-evolutionary observables — entropy rate h_μ, statistical complexity C_μ, and innovation $\mathcal{I}$ — monitor changes in the agents' and the universe's information-processing architecture.

Evolutionary mechanics seems to suggest that the emergence of natural complexity is a manifestly open-ended process. One force of evolution appears in this as a movement up the inductive hierarchy through successive innovations. This force of evolution itself is driven at each stage by the limited availability of resources.

> *Some new principle of refreshment is required. The art of progress is to preserve order amid change, and to preserve change amid order. Life refuses to be embalmed alive.*
>
> A. N. Whitehead in **Process and Reality**.[1]

Acknowledgments

Many thanks are due to Lisa Borland, Don Glaser, Jim Hanson, Blake LeBaron, Dan McShea, Melanie Mitchell, Dan Upper, and Karl Young for helpful discussions and comments on early drafts. This work was partially funded under AFOSR 91-0293 and ONR N00014-92-4024. For the opportunity to present this work thanks also are due to the organizers of the Oji International Seminar on *Complex Systems — from Complex Dynamics to Artificial Reality* held 5 - 9 April 1993 Numazu, Japan; to the organizers of the Santa Fe Institute *Integrative Themes of Complex Adaptive Systems* Meeting held on 8 July 1992 in Santa Fe, New Mexico; and to the organizers of the *Artificial Life 2* conference, held during 5 - 9 February 1990.

Bibliography

1. A. N. Whitehead. *Process and Reality*. The Free Press, New York, corrected edition, 1978.
2. C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21:25 – 34, 1987.
3. B. Holldobler and E. O. Wilson. *The Ants*. Belknap Press of Harvard University Press, Cambridge, Massachusetts, 1990.
4. E. F. Fama. Efficient capital markets II. *J. Finance*, 46:1575 – 1617, 1991.
5. E. Land. The retinex. *Am. Scientist*, 52:247 – 264, 1964.
6. B. A. Wandell. Color appearance: The effects of illumination and spatial pattern. *Proc. Nat. Acad. Sci.*, 10:2458 – 2470, 1993.
7. I. Kovacs and B. Julesz. A closed curve is much more than an incomplete one - effect of closure in figure ground segmentation. *Proc. Nat. Acad. Sci.*, 90:7495 – 7497, 1993.
8. J. P. Crutchfield, N. H. Packard, J. D. Farmer, and R. S. Shaw. Chaos. *Sci. Am.*, 255:46 – 57, 1986.
9. H. L. Swinney and J. P. Gollub, editors. *Hydrodynamic Instabilities and the Transition to Turbulence*, Berlin, 1981. Springer Verlag.
10. A. M. Turing. The chemical basis of morphogenesis. *Trans. Roy. Soc., Series B*, 237:5, 1952.
11. A. T. Winfree. *The Geometry of Biological Time*. Springer-Verlag, Berlin, 1980.
12. Q. Ouyang and H. L. Swinney. Transition from a uniform state to hexagonal and striped Turing patterns. *Nature*, 352:610 – 612, 1991.
13. H. E. Stanley. *Introduction to Phase Transitions and Critical Phenomena*. Oxford University Press, Oxford, 1971.
14. P. Bak and K. Chen. Self-organized criticality. *Physica A*, 163:403 – 409, 1990.
15. J. J. Binney, N. J. Dowrick, A. J. Fisher, and M. E. J. Newman. *The Theory of Critical Phenomena*. Oxford University Press, Oxford, 1992.
16. D. W. Thompson. *On Growth and Form*. Cambridge University Press, Cambridge, 1917.
17. H. Meinhardt. *Models of Biological Pattern Formation*. Academic Press, London, 1982.
18. S. Forrest. Emergent computation: Self-organizing, collective, and cooperative behavior in natural and artificial computing networks: Introduction to the Proceedings of the Ninth Annual CNLS Conference. *Physica D*, 42:–11, 1990.
19. J. G. Kemeny. The use of simplicity in induction. *Phil. Rev.*, 62:391, 1953.
20. C. S. Wallace and D. M. Boulton. An information measure for classification. *Comput. J.*, 11:185, 1968.
21. J. Rissanen. Modeling by shortest data description. *Automatica*, 14:462, 1978.
22. J. P. Crutchfield and B. S. McNamara. Equations of motion from a data series. *Complex Systems*, 1:417 – 452, 1987.
23. J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, Singapore, 1989.
24. J. P. Crutchfield and K. Young. Inferring statistical complexity. *Phys. Rev. Let.*, 63:105–108, 1989.

25. S. Wolfram. Computation theory of cellular automata. *Comm. Math. Phys.*, 96:15, 1984.
26. L. Blum, M. Shub, and S. Smale. On a theory of computation over the real numbers. *Bull. AMS*, 21:1, 1989.
27. M. G. Nordahl. Formal languages and finite cellular automata. *Complex Systems*, 3:63, 1989.
28. J. E. Hanson and J. P. Crutchfield. The attractor-basin portrait of a cellular automaton. *J. Stat. Phys.*, 66:1415 – 1462, 1992.
29. J. P. Crutchfield. Unreconstructible at any radius. *Phys. Lett. A*, 171:52 – 60, 1992.
30. C. Moore. Real-valued, continuous-time computers: A model of analog computation, part I. Technical Report 93-04-018, Santa Fe Institute, 1993.
31. J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, 1979.
32. H. R. Lewis and C. H. Papadimitriou. *Elements of the Theory of Computation*. Prentice-Hall, Englewood Cliffs, N.J., 1981.
33. J. G. Brookshear. *Theory of computation: formal languages, automata, and complexity*. Benjamin/Cummings, Redwood City, California, 1989.
34. J. P. Crutchfield. Semantics and thermodynamics. In M. Casdagli and S. Eubank, editors, *Nonlinear Modeling and Forecasting*, volume XII of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 317 – 359, Reading, Massachusetts, 1992. Addison-Wesley.
35. A. V. Aho. Indexed grammars - an extension of context-free grammars. *J. Assoc. Comp. Mach.*, 15:647, 1968.
36. A. V. Aho. Nested stack automata. *J. Assoc. Comp. Mach.*, 16:383, 1969.
37. B. Weiss. Subshifts of finite type and sofic systems. *Monastsh. Math.*, 77:462, 1973.
38. A. Walker. Adult languages of L Systems and the Chomsky hierarchy. In G. Rozenberg and A. Salomaa, editors, *L Systems*, volume 15, page 201, 1974.
39. A. Lindenmayer and P. Prusinkiewicz. Developmental models of multicellular organisms: A computer graphics perspective. In C. G. Langton, editor, *Artificial Life*, page 221, Redwood City, California, 1989. Addison-Wesley.
40. G. Rozenberg and A. Salomaa, editors. *The Book of L*. Springer-Verlag, Berlin, 1986.
41. E. Allevi, A. Cherubini, and S. Crespi-Reghizzi. Breadth-first context-free grammars and queue automata. *Lect. Notes in Comp. Sci.*, 324:162, 1988.
42. A. Cherubini, C. Citrini, S. Crespi-Reghizzi, and D. Mandrioli. Quasi-real-time fifo automata, breadth first grammars, and their relations. *Theo. Comp. Sci.*, 85:171 – 203, 1991.
43. H. Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York, 1967.
44. D. B. Searls. The linguistics of DNA. *Am. Sci.*, 80:579 – 591, 1992.
45. G. Chaitin. On the length of programs for computing finite binary sequences. *J. ACM*, 13:145, 1966.
46. A. N. Kolmogorov. Three approaches to the concept of the amount of information. *Prob. Info. Trans.*, 1:1, 1965.
47. R. J. Solomonoff. A formal theory of inductive control. *Info. Control*, 7:224, 1964.

48. C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, Champaign-Urbana, 1962.
49. A. A. Brudno. Entropy and the complexity of the trajectories of a dynamical system. *Trans. Moscow Math. Soc.*, 44:127, 1983.
50. J. Ford. How random is a coin toss? *Physics Today*, April 1983.
51. C. H. Bennett. Dissipation, information, computational complexity, and the definition of organization. In D. Pines, editor, *Emerging Syntheses in the Sciences*. Addison-Wesley, Redwood City, 1988.
52. J. P. Crutchfield and N. H. Packard. Symbolic dynamics of noisy chaos. *Physica*, 7D:201 – 223, 1983.
53. R. Shaw. *The Dripping Faucet as a Model Chaotic System*. Aerial Press, Santa Cruz, California, 1984.
54. P. Grassberger. Toward a quantitative theory of self-generated complexity. *Intl. J. Theo. Phys.*, 25:907, 1986.
55. D. Angluin and C. H. Smith. Inductive inference: Theory and methods. *Comp. Surveys*, 15:237, 1983.
56. J. P. Crutchfield and K. Young. Computation at the onset of chaos. In W. Zurek, editor, *Entropy, Complexity, and the Physics of Information*, volume VIII of *SFI Studies in the Sciences of Complexity*, pages 223 – 269, Reading, Massachusetts, 1990. Addison-Wesley.
57. J. P. Crutchfield. Knowledge and meaning ... chaos and complexity. In L. Lam and V. Naroditsky, editors, *Modeling Complex Phenomena*, pages 66 – 101, Berlin, 1992. Springer-Verlag.
58. J. P. Crutchfield. Reconstructing language hierarchies. In H. A. Atmanspracher and H. Scheingraber, editors, *Information Dynamics*, pages 45 – 60, New York, 1991. Plenum.
59. J. P. Crutchfield and D. R. Upper. In preparation, 1994.
60. J. P. Crutchfield and J. E. Hanson. Turbulent pattern bases for cellular automata. *Physica D*, 69:279 – 301, 1993.
61. M. J. Feigenbaum. Universal behavior in nonlinear systems. *Physica*, 7D:16, 1983.
62. L. P. Kadanoff, M. J. Feigenbaum, and S. J. Shenker. Quasiperiodicity in dissipative systems: a renormalization group analysis. *Physica*, 5D:370, 1982.
63. J. P. Crutchfield. Observing complexity and the complexity of observation. In H. Atmanspacher, editor, *Inside versus Outside*, pages 235 – 272, Berlin, 1994. Springer-Verlag.
64. D. Blackwell and L. Koopmans. On the identifiability problem for functions of Markov chains. *Ann. Math. Statist.*, 28:1011, 1957.
65. H. Ito, S.-I. Amari, and K. Kobayashi. Identifiability of hidden Markov information sources and their minimum degrees of freedom. *IEEE Info. Th.*, 38:324, 1992.
66. J. P. Crutchfield and J. E. Hanson. Attractor vicinity decay for a cellular automaton. *CHAOS*, 3(2):215–224, 1993.
67. J. E. Hanson. *Computational Mechanics of Cellular Automata*. PhD thesis, University of California, Berkeley, 1993. Published by University Microfilms Intl, Ann Arbor, Michigan.

68. N. Boccara, J. Nasser, and M. Roger. Particle-like structures and their interactions in spatio-temporal patterns generated by one-dimensional deterministic cellular automaton rules. *Phys. Rev. A*, 44:866 – 875, 1991.
69. B. Russell. *A History of Western Philosophy*. Simon and Schuster, New York, 1945.
70. T. S. Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, Chicago, 1962.
71. C. G. Langton. Computation at the edge of chaos: Phase transitions and emergent computation. In S. Forrest, editor, *Emergent Computation*, page 12. North-Holland, Amsterdam, 1990.
72. N. H. Packard. Adaptation toward the edge of chaos. In A. J. Mandell J. A. S. Kelso and M. F. Shlesinger, editors, *Dynamic Patterns in Complex Systems*, pages 293 – 301, Singapore, 1988. World Scientific.
73. S. A. Kauffman. *Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, New York, 1993.
74. M. Mitchell, P. Hraber, and J. P. Crutchfield. Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems*, 7:89 – 130, 1993.
75. J. Maynard-Smith. *Evolutionary Genetics*. Oxford University Press, Oxford, 1989.
76. J. Monod. *Chance and Necessity: An Essay on the Natural Philosophy of Modern Biology*. Vintage Books, New York, 1971.
77. S. J. Gould. *Wonderful Life*. Norton, New York, 1989.
78. C. H. Waddington. *The Strategy of the Genes*. Allen and Unwin, London, 1957.
79. B. Goodwin. Evolution and the generative order. In B. Goodwin and P. Sanders, editors, *Theoretical Biology: Epigenetic and Evolutionary Order from Complex Systems*, pages 89 – 100, Baltimore, Maryland, 1992. Johns Hopkins University Press.
80. W. Fontana and L. Buss. What would be conserved if the tape were played twice? *Proc. Nat. Acad. Sci.*, 91:757 – 761, 1994.
81. W. Fontana and L. Buss. "The Arrival of the Fittest": Toward a theory of biological organization. *Bull. Math. Bio.*, 56:1 – 64, 1994.
82. B. Goodwin and P. Sanders, editors. *Theoretical Biology: Epigenetic and Evolutionary Order from Complex Systems*, Baltimore, Maryland, 1992. Johns Hopkins University Press.
83. L. J. Fogel, A. J. Owens, and M. J. Walsh, editors. *Artificial Intelligence through Simulated Evolution*. Wiley, New York, 1966.
84. J. H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, 1992. Second edition (First edition, 1975).