The Attractor-Basin Portrait of a Cellular Automaton

James E. Hanson
James P. Crutchfield

SFI WORKING PAPER: 1991-02-012

SANTA FE INSTITUTE

The Attractor-Basin Portrait of a Cellular Automaton

James E. Hanson
James P. Crutchfield

Department of Physics
University of California
Berkeley CA 94720 USA*

Abstract

Local space-time structures, such as domains and the intervening dislocations, dominate a wide class of cellular automata (CA) behavior. For such spatially-extended dynamics regular domains, vicinities, and attractors are introduced as organizing principles to identify the discretized analogs of attractors, basins, and separatrices: structures used in classifying dissipative continuous-state dynamical systems. We describe the attractor-basin portrait of nonlinear elementary CA rule 18, whose global dynamics is largely determined by a single regular attracting domain. The latter's basin is analyzed in terms of subbasin and portal structures associated with particle annihilation. The conclusion is that the computational complexity of such CA is more apparent than real. Transducer machines are constructed that automatically identify domain and dislocation structures in space-time, count the number of dislocations in a spatial pattern, and implement an isomorphism between rule 18 and rule 90. We use a transducer to trace dislocation trajectories, and confirm that in rule 18 isolated dislocation trajectories, as well as a dislocation gas, agree extremely well with the classical model of annihilating diffusive particles. The CA efficiently transforms randomness of an initial pattern ensemble into a random walk of dislocations in space-time.

*JPC's Internet address is chaos@gojira.berkeley.edu; JEH's is hanson@gojira.berkeley.edu.

Contents

List of Figures

Introduction

Cellular automata (CA) are models of spatially extended dynamical systems with the particular feature of being as discrete as possible: in space (cells), in site value (e.g. binary), and in time.[1] The discrete nature of CA makes the techniques of formal language and computation theories, which deal with strings of discrete symbols, particularly appropriate tools for their analysis. Computational and formal language properties of CA global behavior have been investigated during the last two decades.[2–5] Discussions of computational capability have often centered on constructing specialized cellular automata to mimic features of serial computation models: Turing machines embedded in, for example, Conway's Game of Life,[6] or digital circuitry implicit in logic gate, billiard computer, and particle flow simulations.[7] Such constructions, however, fail to address the basic question of how to identify the generic computational properties of any given CA.

Similarly, analyses of CA in the context of dynamical systems theory have tended toward specialized results, focusing for example on very small lattices and explicit or Monte Carlo enumeration of periodic orbits and their transients.[8–11] Nonetheless, the last decade has witnessed a substantial increase in our appreciation of the diversity of CA behavior. There have been extensive phenomenological studies, including classification,[12–14] identification of propagating structures,[15,16] and measurement of bulk statistical properties.[1,17–19] These unfortunately have taken advantage of neither the rich potential implicit in the well-developed theory of computation nor the geometric underpinnings of dynamical systems theory.

The following attempts a synthesis of these two approaches by considering a view of CA dynamics based on the local space-time and state-space structures that organize the system's evolution. In doing so, it introduces a rather literal analysis of CA as dynamical systems. For a range of related CA one can identify the analogs of attractors, basins and separatrix structures.[20,21] These ideas will be illustrated by focusing on a particular elementary CA, rule 18 in the conventional numbering scheme. Although our discussion is restricted to this one example, it serves to illustrate a number of properties of CA generally, but in a simple setting. A more detailed development and more complete survey of CA will appear elsewhere.[22]

The basic phenomenology of the CA we study here has been known for some time.[17,23] It organizes itself into local regions in which every other cell in space and time is 0 and the remaining cells are unrestricted. The boundaries between adjacent "phase-locked" regions are dislocations, where the spatial phase slips a cell. When evolved from a random initial condition consisting of a rarified dislocation gas, numerical results indicate that the number density d of dislocations decays in time as $d \approx (8\pi Dt)^{-\frac{1}{2}}$, with $D \approx \frac{1}{2}$. Furthermore, the excursion x of a set of initially distant dislocations grows as $\langle x^2 \rangle \approx 2Dt$. The interpretation of this is that the dislocations move as if following a space-time random walk with diffusion constant D, annihilating in pairs when their paths cross.[24] In short, starting from random configurations the dislocations behave like a diffusive gas of annihilating particles.

The focus in the present work is on the computational analysis of this behavior, on a reconstructive method for identifying domain structure, and on the implications of these in treating CA as dynamical systems. The techniques introduced below for detecting, analyzing and manipulating general structural elements such as domains and dislocations point the way toward a generally applicable computation-analytical tool.

The presentation is organized as follows. The next section gives definitions and introduces notation for elementary CA, finite automata and regular languages, and transducers. We define an operator governing the time-evolution of ensembles of CA states, and give an algorithm for calculating the iterates of such an ensemble. The following section addresses the central issue of space-time structures, giving definitions and applying them to rule 18, which serves as an illustrative example throughout. Next we describe the global structure of rule 18, and give its attractor-basin portrait. Finally, as an application of the ideas introduced, we examine the statistical properties of dislocation trajectories.

Definitions

Elementary CA and Rule 18

Definition.[12] A **cellular automaton** consists of a countable array of discrete sites or cells i and a discrete-time update rule ϕ operating in parallel on local neighborhoods of a given radius r. At each time the sites take on values in a finite alphabet $\mathcal{A}$ of primitive symbols: $\sigma_t^i \in \{0, 1, ...k-1\} \equiv \mathcal{A}$. The local site-update function is written

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

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

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

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

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

The automaton we are concerned with is an "elementary" CA for which $(k, r) = (2, 1)$; that is, nearest-neighbor interactions and binary cell values, giving a total of $k^{2r+1} = 8$ possible neighborhood patterns. Following the conventional numbering scheme,[12] we arrange these patterns in the order below, write the value obtained by applying ϕ to each pattern, and interpret the resulting string of 8 symbols as a binary number giving the rule index. This paper analyzes the CA having rule index 18, for which the rule table is as follows:

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

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

Finite Automata and Regular Languages

In this section, we briefly review the basic definitions of automata and languages we will need. For a thorough presentation of this topic, see any standard text on the subject.[25] It is important not to confuse the automata defined here, which represent ensembles of symbol strings, with cellular automata, which are the dynamical systems we are studying.

Definition. An **automaton**, or "machine", is formally defined as a quintuple $\mathrm{M} = \{Q, \Sigma, \delta, q_0, F\}$. Here Q is a set of states; Σ is an alphabet of tape symbols; $\delta : Q \times \Sigma \to Q$ is the transition rule, taking the current state and a symbol read from a tape into a new state; q_0 is the initial state; and F is the set of final or accepting states. If the set Q of states is finite, $\|Q\| < \infty$, then M is a **finite automaton.**

Finite automata can be **deterministic** or **non-deterministic** depending on whether for all states $q \in Q$, each transition from q to another state is labeled uniquely with a tape symbol. These two automaton classes are abbreviated "DFA" and "NFA", respectively. An automaton is said to **recognize** a string if, upon reading it from the tape, no disallowed transitions are seen. An automaton **accepts** a string if, after recognizing it, the automaton is in one of the final states. Strings that are not accepted are **rejected.** Two finite automata are **equivalent** if they accept identical sets of strings. Two automata are **disjoint** if they accept disjoint sets of strings.

A formal language L is a set of strings over some alphabet Σ: $L \subseteq \Sigma^* = \bigcup_{i \geq 0} \Sigma^i$. The language $L(\mathrm{M})$ of a machine M is the set of all strings, sometimes called words, accepted by that machine. This definition is independent of whether or not M is finite.

Definition. A **regular language** is that $L(\mathrm{M})$ accepted by some finite automaton M. We define **regular language space**, or just "language space", as the set $\mathcal{R}$ of all regular languages.

For each regular language L there is a unique finite automaton M with the minimal number $\|Q\|$ of states that accepts only and all of the strings in L. In this case, we denote by $\mathrm{M}(L)$

the minimal DFA accepting L. This allows us to use finite automaton and regular language representations interchangeably.

An important subclass of the languages we will be studying are process languages. That is, for all words $\omega \in L$, all subwords $\nu \in \{s^i \ldots s^{i+j} : s^k = (\omega)_k, i+j \leq \|w\|, i,j \geq 0\}$ of ω are in L. We form the subword closure sub(L) of a language L by including in sub(L) all subwords of each $\omega \in L$. Finite automata for regular process languages have the property that all states are accepting: $F = Q$.[26]

Definition. The **process graph** of a process language L is a graph formed by allowing the associated machine $\mathrm{M}(L)$ to start in any state, and minimizing the result. This is the "state transition graph" of [4]. The process graph clarifies the structure of the language by absorbing the transient states devoted to "phase-locking" of individual strings to the machine's recurrent states.[27]

Transducers

The automata defined above can be used either as input devices for classifying existing strings, or as output devices that generate strings belonging to a given language. These two modes of operation are easily combined by modifying the machine so that on each transition a symbol is emitted as well as read. The resulting "transducer" machines, also called Mealy machines,[25] are functions from strings to strings. In the information theory literature the process versions are referred to as channels.[28] In ergodic theory, they are endomorphisms of subshifts.[29] Transducers give a very compact notation for functions that map complicated sets, e.g. Cantor sets, to complicated sets. Of equal importance, they are constructive in the sense that they indicate literally how the function is to be implemented and so computed.

In this section we formally define transducers and develop a few basic tools for manipulating them. Transducers will turn out to be central to the analysis of structural properties of CA state space.

Definition. A **transducer** $\mathrm{T} = \{Q, \Sigma_{in}, \delta, q_0, F, \Sigma_{out}, \beta\}$ is an automaton where Q, δ, q_0, F are as defined above and where Σ_{in} is the previous input alphabet Σ. Σ_{out} is an output alphabet. The observation function β maps $Q \times \Sigma_{in}$ to Σ_{out}. A transducer operates as a DFA with the extra feature of writing a symbol from Σ_{out} to an output tape upon reading a symbol in Σ_{in} from the input tape.

The effect of running the transducer over a space-time pattern, say, is to recode the data by replacing original symbols with those emitted by the machine. Formally, the transducer induces a function f_{T} that maps strings over Σ_{in} to strings over Σ_{out}.

Definition. Let T be a transducer. The **output machine,** which we denote $[\mathrm{T}]_{\mathrm{out}}$, is the minimal DFA accepting the set of all strings that can be emitted from T, on all possible inputs. $[\mathrm{T}]_{\mathrm{out}}$ can be constructed by the following series of steps:

i. The input symbols on all transitions of T are disregarded, leaving only the output symbols labeling transitions.
ii. The resulting NFA is converted to a DFA using the standard algorithm.[25]
iii. The DFA is then minimized to obtain the unique machine with the minimal number $\|Q\|$ of states. This operation is also standard.[25]

We call this construction **minimization with respect to the output**.

Given a machine $\mathrm{M}(L)$ and a transducer T, one can define the composition operator $\mathrm{T} \circ \mathrm{M}(L)$ such that the resulting machine, itself a transducer, accepts as input exactly the set of strings in L, and for each such input string $s \in L$, it emits the encoded string $f_{\mathrm{T}}(s)$. In effect, $\mathrm{M}(L)$ and T are connected in series, with $\mathrm{M}(L)$ emitting strings in L which are then used as inputs to T. This corresponds to restricting the domain of f_{T} to the language L. The operator is formally expressed in the following.

Definition. Let M and T be an a finite automaton and a transducer, respectively:

$$\mathrm{M} = \left\{ Q^{\mathrm{M}}, \Sigma^{\mathrm{M}}, \delta^{\mathrm{M}}, q_0^{\mathrm{M}}, F^{\mathrm{M}} \right\}$$
$$\mathrm{T} = \left\{ Q^{\mathrm{T}}, \Sigma_{in}^{\mathrm{T}}, \delta^{\mathrm{T}}, q_0^{\mathrm{T}}, F^{\mathrm{T}}, \Sigma_{out}^{\mathrm{T}}, \beta^{\mathrm{T}} \right\} \text{ with } \Sigma_{in}^{\mathrm{T}} = \Sigma^{\mathrm{M}}$$

Then the **composition** of T and M, denoted $\mathrm{T}' = \mathrm{T} \circ \mathrm{M}$ is a transducer

$$\mathrm{T}' = \left\{ Q', \Sigma'_{in}, \delta', q'_0, F', \Sigma'_{out}, \beta' \right\}$$

such that: Q' are ordered pairs of the states in Q^{T} and Q^{M}: $Q' = Q^{\mathrm{T}} \times Q^{\mathrm{M}}$; the input alphabet is $\Sigma'_{in} = \Sigma_{in}^{\mathrm{T}} = \Sigma^{\mathrm{M}}$; the start state is $q'_0 = \left(q_0^{\mathrm{T}}, q_0^{\mathrm{M}}\right)$; the final states are $F' = \left\{ \left(q^{\mathrm{T}}, q^{\mathrm{M}}\right) \mid q^{\mathrm{T}} \in F^{\mathrm{T}}, q^{\mathrm{M}} \in F^{\mathrm{M}} \right\}$; the output alphabet is $\Sigma'_{out} = \Sigma_{out}^{\mathrm{T}}$; the transition rule $\delta' : Q' \times \Sigma'_{in} \to Q'$ is given by

$$\delta'\left(q', \sigma_{in}\right) = \left(\delta^{\mathrm{T}}\left(q^{\mathrm{T}}, \sigma_{in}\right), \delta^{\mathrm{M}}\left(q^{\mathrm{M}}, \sigma_{in}\right)\right) \text{ , where } q' = \left(q^{\mathrm{T}}, q^{\mathrm{M}}\right);$$

and the observation function $\beta' : Q' \times \Sigma'_{in} \to \Sigma'_{out}$ is

$$\beta'\left(q', \sigma_{in}\right) = \beta^{\mathrm{T}}\left(q^{\mathrm{T}}, \sigma_{in}\right), \text{ again with } q' = \left(q^{\mathrm{T}}, q^{\mathrm{M}}\right).$$

A transducer can also be used to define a "machine metric" that measures the distance of a string from a given language. The action of this transducer is to encode a given pattern into a string of 1's that represents in unary form the number of disallowed transitions observed in the pattern. We will represent the fiducial language by its corresponding automaton M. Then the metric with respect to this language is defined as follows.

Definition. Let M be a machine, and ω be any string, not necessarily in $L(\mathrm{M})$. The **machine metric** is

$$\|\omega\|_{\mathrm{M}} = \begin{cases} \|f_{\mathrm{T_M}}(\omega)\| & \|\omega\| < \infty \\ \lim\limits_{n\to\infty} \frac{\|f_{\mathrm{T_M}}(\omega_n)\|}{\|\omega_n\|} & \text{otherwise} \end{cases}$$

where $\mathrm{T_M}$ is the transducer formed from automaton M by augmenting it to output the null symbol λ on allowed transitions and adding new edges that emit a 1 on previously disallowed transitions. The new edges are "reset" edges, connected to the state representing the effect of having read the disallowed symbol when in the start state. $f_{\mathrm{T_M}}(\omega)$ is the string emitted by $\mathrm{T_M}$ upon reading the input ω. In the infinite case, ω_n denotes the string consisting of the first n symbols of ω. The finite and infinite cases are related by dividing the former by $\|\omega\|$.

In a later section, we will use a machine metric that counts dislocations in spatial patterns generated by rule 18.

CA as Regular Language Processors

The connection between CA and regular languages is readily made by identifying the CA alphabet $\mathcal{A}$ with the automaton alphabet Σ. Then a CA state $\mathrm{s}_t \in \mathcal{A}^N$ is treated as a word in some language $L \in \mathcal{R}$. It is important to note that this makes the analysis independent of lattice size, since L is a subset of the extended state space $\mathcal{A}^* \equiv \Sigma^*$, and hence may contain strings of all lengths N.

The operation of the global CA update rule Φ is to recode a parent string s_{t-1} into a child string s_t. Stated in this way, it is clear that Φ can be implemented as a transducer, which we denote T_Φ. Since a transducer reads the symbols one by one, the full definition must include a specification of the direction in which it is to be passed over the spatial array; in the following, we will always take this direction to be from left to right. The machine representation of T_Φ for an arbitrary elementary CA is shown in Figure 1. It operates as a queue with two memory registers, in which a child symbol is emitted when the final member of its parent neighborhood is read. The output symbols are given by the local update rule operating on the current parent neighborhood: $\sigma_{out}^{i-1} = \phi(\sigma_{in}^{i-2}, \sigma_{in}^{i-1}, \sigma_{in}^{i})$. This is in fact just a description of the most memory-efficient way of programming 1D CA dynamics for numerical experiments.

As defined above, the CA update rule Φ operates on individual configurations. Alternatively, we can look at how a CA processes regular language representations of spatial configurations.[4] This is analogous in dynamical systems theory to the study of the evolution of state ensembles, rather than individual orbits. To this end we define the ensemble evolution operator $\mathbf{\Phi}$ by

$$\Omega_t = \mathbf{\Phi}\Omega_{t-1} = \mathbf{\Phi}_t\Omega_0$$

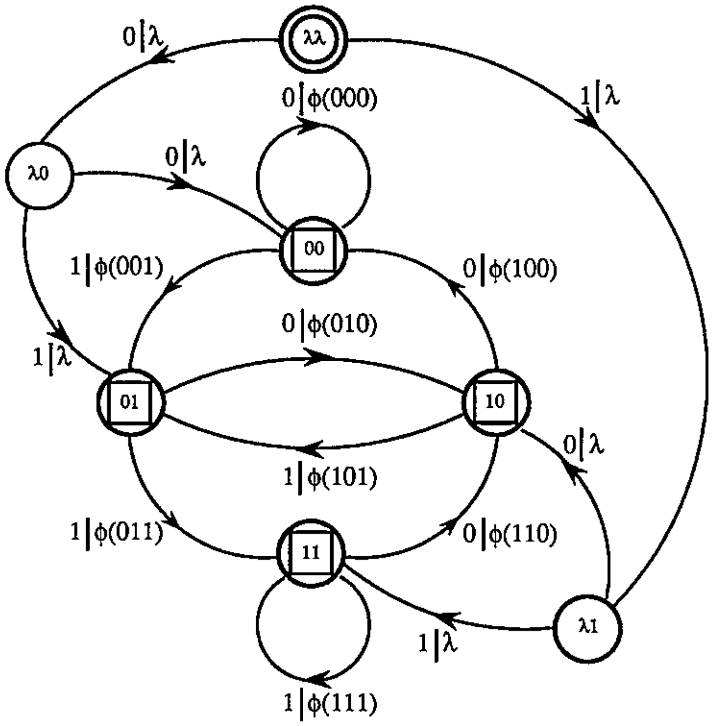


Figure 1 The CA update rule transducer T_Φ that reads a state s_{t-1} and emits its iterate $s_t = \Phi(s_{t-1})$. T_Φ is passed over the spatial array from left to right. In this and the following figures, inscribed circles and squares denote start and final states, respectively. Edges are labeled with $\sigma_{in}|\sigma_{out}$ to denote the input and output symbols. λ is the null symbol. The labels of the states correspond to the previous two symbols read. This is the machine for the infinite lattice case; finite lattices require additional constraints due to boundary conditions.

where Ω_0 is any regular language of spatial configurations used as initial conditions:

$$\Omega_t = \{s | s = \Phi s_{t-1}, \ \forall \ s_{t-1} \in \Omega_{t-1}\}$$

When $\Omega_0 = \mathcal{A}^*$, the spatial languages are regular for all finite t: $\Omega_t \in \mathcal{R}, \forall t < \infty$.[4] The FME algorithm defined below shows that the iterate of *any* regular language is regular: $\Omega_0 \in \mathcal{R} \Rightarrow \Omega_t \in \mathcal{R}, \forall t < \infty$. We also note here that, since any finite language is regular, the spatial language generated by any CA over a finite spatial lattice ($N < \infty$) is also always regular. Thus, if either space or time is finite, the set of CA configurations is a regular language.[30] As a consequence Φ is an automorphism that maps regular language space into itself: $\Phi : \mathcal{R} \rightarrow \mathcal{R}$. For infinite lattices, however, the asymptotic spatial language, found in the limit $t \rightarrow \infty$, need not be regular — or even recursively enumerable.[5] Somewhat surprisingly, this property often does not hamper our computational analysis, as will be seen in the next section.

Figure 2 shows a comparison of the operation of Φ in configuration space $\mathcal{A}^N$, where it operates on sets of configurations, and in regular language space $\mathcal{R}$, where it operates on individual languages. The important advantage of the latter approach is that it allows us to

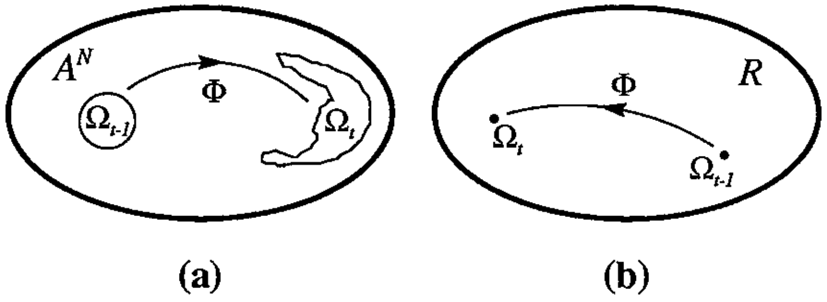


Figure 2 Schematic of the operation of the language operator Φ (a) in CA state space and (b) in regular language space. Note that case (b) subsumes all values of N simultaneously.

investigate the dynamics of ensembles of structurally similar CA states without having to deal with the highly non-trivial complications introduced by tracing the evolution of individual states.

Definition. The **Finite Machine Evolution (FME) algorithm.** Given the CA update transducer T_Φ for a specific CA rule Φ, and a machine $M_{t-1} = M(\Omega_{t-1})$, we calculate $M_t = M(\Omega_t) = M(\Phi\Omega_{t-1})$ by using machine-composition and minimizing the resulting transducer with respect to the output symbols:

$$M_t = [T_\Phi \circ M_{t-1}]_{\text{out}}$$

Structures in Language Space

The global machine $M_t = M(\Omega_t)$, when $\Omega_t = \Phi_t(\mathcal{A}^*)$, describes the time-evolution of the ensemble of all possible initial conditions up to some time t. It necessarily captures all of the information processing accessible to the CA, which includes correlations and information transmission as well as computation. Many previous investigations of CA language properties have tended to focus on the global machine.[4,31]

In terms of dynamical systems theory, though, the global machine describes the entire attractor-basin portrait.[20] In many cases, this description is seen to be prohibitively difficult to construct because, for example, M_t grows too large with time. In fact, such complete global descriptions are rarely pursued in dynamical systems theory. The main exceptions to this come in the study of flows on manifolds of constant negative curvature, uniform hyperbolic systems, and some very simple periodic systems: all highly idealized systems. Similarly for CA, the complexity of the global machine is reflected in the fact that the asymptotic language can be non-regular, as noted above. Even for elementary CA, there are many cases in which size of M_t increases extremely rapidly in time. This is true in particular of rule 18: $M(\Omega_3)$ has $\|Q\| = 143$ states; $M(\Omega_4)$ is estimated to have $\|Q\| > 20,000$.[4] Furthermore, there are other CA rules for which the growth is even faster.[31]

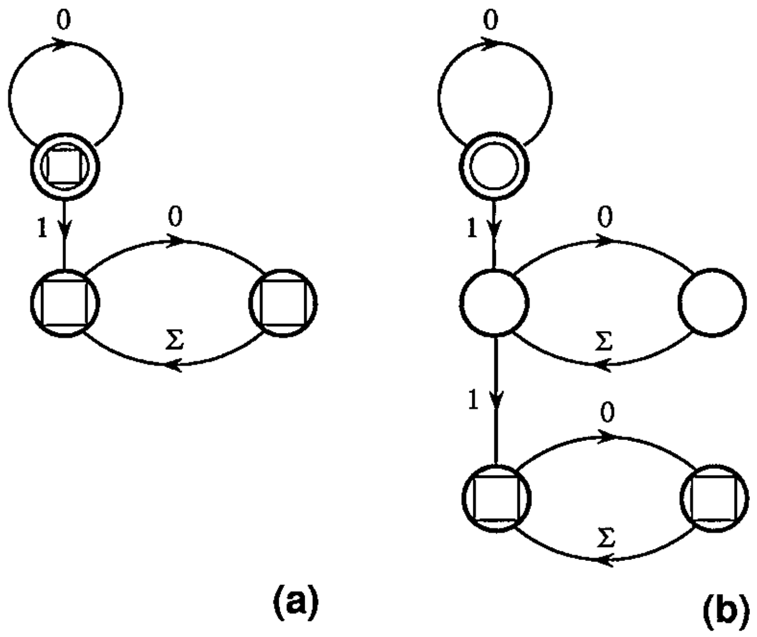


Figure 4 Machines (a) $M(\Lambda^0)$ for single domain language Λ^0 and (b) $M(\Lambda^{0,0})$ for domain-wall-domain language $\Lambda^{0,0}$.

to the output, the final machine $M(\Phi\Lambda^0) = [T_{18} \circ M(\Lambda^0)]_{\text{out}}$ is seen to be identical to $M(\Lambda^0)$. (ii) Spatial homogeneity: this follows immediately since the process graph of Λ^0, shown in Figure 3, is strongly connected. ▌

As the next proposition shows, the two-domain language $\Lambda^{0,0}$ is also time-invariant, with the following qualification. There exist a few states in $\Lambda^{0,0}$ that eventually map into the null language 0^*. The latter consists of the "quiescent" states for all lattice sizes in which all sites are 0. Since 0^* is in Λ^0, it is necessary to excise these by hand. We do this by defining the language

$$\tilde{\Lambda}^{0,0} \equiv \Lambda^{0,0} - \Lambda^{0,0} \cap \mathcal{B}(0^*)$$

where $\mathcal{B}(0^*)$ denotes the set of all states that eventually map to 0^*, i.e. the basin of 0^*. In a later section, we will define the basin of a language more formally. For odd and infinite lattices, $\Lambda^{0,0} \cap \mathcal{B}(0^*) = s_1 \bigcup s_2$ up to spatial reflection, where $s_1 = (01)^*1(01)^*$ and $s_2 = (0010)^*0(1000)^*$. Furthermore,

1. $\Phi(s_1) = 0^*$ and $\Phi(s_2) = s_1$.
2. $||s_1||_{M(\Lambda^0)} = ||s_2||_{M(\Lambda^0)} = 1$: they have a single dislocation.
3. Although s_2 is not a Garden of Eden state, the first and second preimages of s_2 exhaust all of its predecessors and have an infinite number of dislocations. Thus, all preimages of s_2 are not in $\Lambda^{0,0}$.

Domains

Definition. A **regular domain** or "domain language" $\Lambda \in \mathcal{R}$ of a CA is a language representing a set of spatial regions with the following two properties:
(i) *Temporal invariance:* Λ is mapped onto itself by the dynamic: $\Phi\Lambda = \Lambda$; and
(ii) *Spatial homogeneity:* The process graph of Λ is strongly connected.[34] This is similar to statistical stationarity of the spatial pattern.

For general CA there can be any number of different domains Λ^i, each with its own computational structure.

Below we generalize this to more complicated dynamical behavior than single iteration invariance. A domain can be defined as a nonwandering orbit of languages. This includes those whose temporal behavior is periodic or even chaotic. In each case, all languages in the orbit must be spatially homogeneous.

A domain describes a computationally-homogeneous *region* of the spatial array. In this way we take "domain" to mean the domain language Λ or, alternatively, any configuration $s \in \Lambda$.

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

Definition. The **phase** of site i in a domain Λ^j is the state q of the machine $M(\Lambda^j)$ when s^i is read in a chosen direction.

In the case of rule 18, the observed behavior described above defines a domain language Λ^0. All spatial states are made up of concatenations of words $\omega \in \Lambda^0$ separated by dislocations. In particular, the two-domain language $\Lambda^{0,0}$ consists of all states in which there is a single dislocation. The process graph of Λ^0 is shown in Figure 3. The DFAs for both Λ^0 and $\Lambda^{0,0}$ are shown in Figure 4.

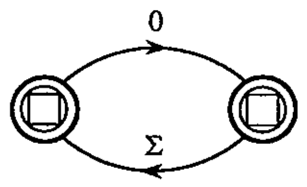


Figure 3 The process graph of Λ^0 in which both states are start and final states. The symbol Σ denotes the symbol set $\{0, 1\}$.

Proposition. Λ^0 is a regular domain.

Proof: (i) Temporal invariance: by explicit construction using the FME algorithm. Figure 5 shows an intermediate stage of the construction of the first iterate of $M(\Lambda^0)$, immediately after composing the machine with the rule 18 update transducer T_{18}. After minimizing with respect

The main point is that the patterns excluded from $\Lambda^{0,0}$ are very few in number.

Proposition. $\tilde{\Lambda}^{0,0}$ is a time-invariant (fixed-point) regular language, although it is not a domain.

Proof: We begin by calculating the image of $M(\Lambda^{0,0})$ under rule 18 by means of the FME algorithm. Figure 6 shows the transducer $T_{18} \circ M(\Lambda^{0,0})$. After finishing the FME algorithm, we find that the resulting machine is slightly different from $M(\Lambda^{0,0})$. This is due to two factors: (i) the set $\Lambda^{0,0} \cap \mathcal{B}(0^*)$ of states that map to the null state, as mentioned above, and (ii) a set of input strings accepted by $M(\Lambda^{0,0})$ whose images under rule 18 are accepted by $M(\Lambda^{0})$. Since $\tilde{\Lambda}^{0,0}$ by definition does not include the states that map to 0^*, only the second factor need be addressed. This a spurious effect due to the spatial inhomogeneity of $M(\Lambda^{0,0})$ in combination with the spatial correlation of the input and output strings of the CA update rule transducer. The input to $T_{18} \circ M(\Lambda^{0,0})$ consists of some portion of the spatial state s_{t-1}, starting an arbitrary distance from the dislocation. For input strings of the form $(01)^*10\Sigma0\Sigma...$, the observed region before the parent dislocation maps to a sequence of adjacent 0's. There is no nonzero cell to the left of the child dislocation to fix the spatial phase, and so the dislocation does not show up. Since, as we discuss in the next section, dislocations can only disappear in pairs, the dislocation is still present, but merely unseen. We remove this "phase-ambiguity" artifact by requiring that the machine observe the dislocation before it can enter a final state. After the result is minimized, the final machine is seen to be identical to $M(\Lambda^{0,0})$, shown in Figure 4. ▮

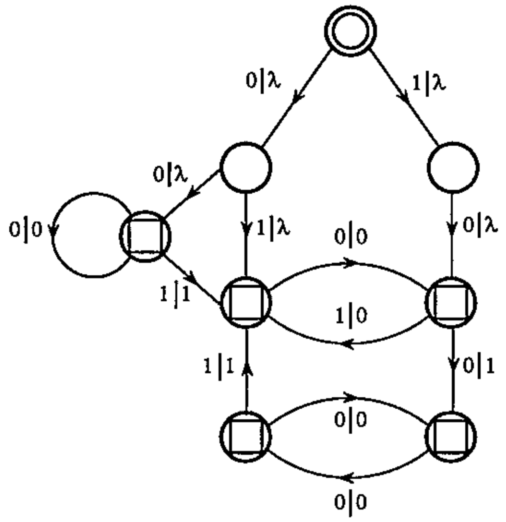


Figure 5 Λ^0 is a regular domain: Machine $T_{18} \circ M(\Lambda^0)$ obtained after composing the rule 18 update transducer with the single-domain machine of Figure 4(a). Minimization with respect to the output results in a machine identical to that shown in Figure 4(a).

An essential issue that arises here is the question of discovering domains in the first place. Even with unlimited space-time data, the domain language(s) for a given CA may be sufficiently

Another shortcoming of the global approach is that machines for qualitatively similar CA can be vastly different. Comparison of the space-time patterns generated by rule 18 and rule 90 shows a marked similarity; indeed, for an important class of initial conditions they are identical. Yet the global machine for rule 90, far from exhibiting the radical growth seen in rule 18, has just one state for all t.

The apparent intractability due to unbounded growth, and the contrast between global machines for rules with such similar behavior, suggests that an alternative view is required. It is not necessary, for example, to use $\Omega_0 = \mathcal{A}^*$; nor can we expect it to be particularly useful, as we just noted. Characterizing the entire attractor-basin portrait of any dynamical system with a single expression is implausible for anything but the simplest dynamics. Instead, one standard approach is to identify the *structures* that dominate and organize the system's behavior. The most basic of these for dissipative systems are attractors, basins, and separatrices. The state space is then understood in terms of those structures. In the case of CA, we want to find the dominant, dynamically-homogeneous space-time patterns and their associated *languages*.

Yet previous investigations from the viewpoint of pure dynamical systems theory have tended to overlook computationally equivalent space-time behavior. For example, the study of periodic orbits of very small lattices,[8–11] perturbation growth rates as the analog of Lyapunov characteristic exponents,[32] and information transmission in space or in time,[33] have failed to provide a formulation for attractors, basins, and separatrices that integrates them with a description of the intrinsic computational capability of the underlying CA.

This is a reflection of a basic problem in the literal application of dynamical systems theory to CA and other spatially-extended systems. The theory does not provide a formalism for the investigation of a state's internal structure. A state of a dynamical system is a structureless point. But it is exactly the internal, now spatial, structure of a CA configuration that is of interest. This deficit can be addressed by interpolating between the literal notion of a CA state as a position-dependent configuration of cells and the intuitive notion of a translation-independent pattern. Generally speaking, a pattern is (i) a position-independent configuration and (ii) a typical sequence generated by the process, where "typical" is used in the sense of the Shannon-McMillan theorem of information theory.[28] The use of a regular language or finite automaton description of processes captures exactly these features.

For CA rule 18, the dominant structure is the *domain* described above: every other site has value 0, while the remaining sites can be either 0 or 1.[17,23] The iterate of a domain consists of a domain shifted by one site. This observation is consistent with the intuitive notion of a domain as a dynamically-homogeneous region of the spatial array. We formalize the preceding comments in the following subsections. First we consider the spatial structure via the notion of a regular domain. Then we address temporal structure and stability by defining regular vicinity and regular attractor.

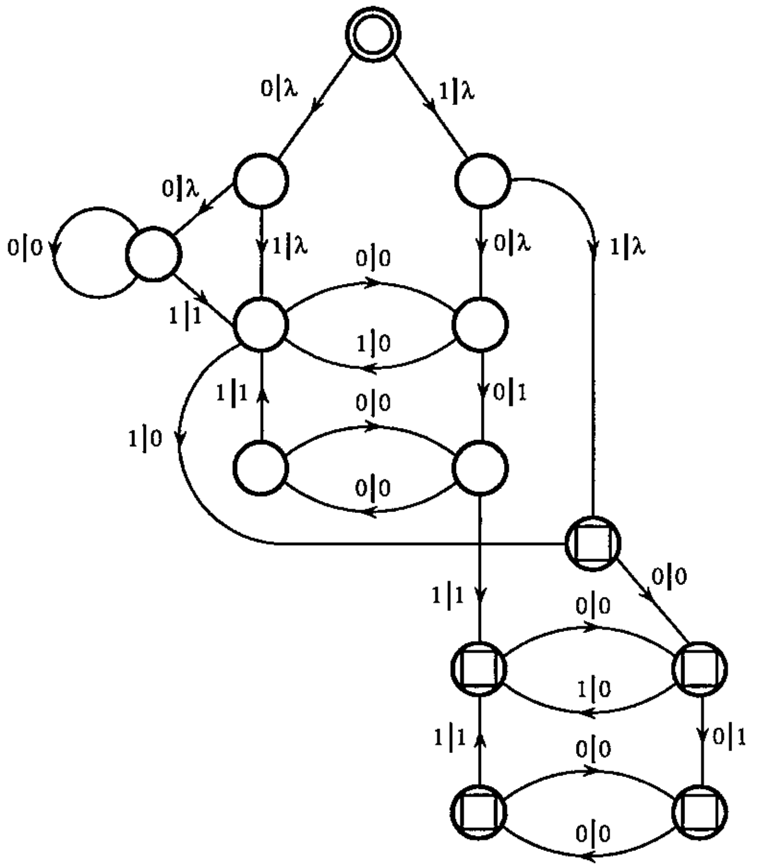


Figure 6 $\tilde{\Lambda}^{0,0}$ is a fixed-point language: Machine $T_{18} \circ M(\Lambda^{0,0})$ obtained after composing the rule 18 update transducer with the two-domain machine of Figure 4(b). On completion of the remaining steps of the proof, Figure 4(b) is found.

complex as to defy visual identification. For rule 18, the dominant structure had already been found; but this is not true in general. One consequence of the foregoing analysis is that a general approach to identifying these structures, once developed, would provide a basis for automated dynamical systems investigation of CA computational capability. A promising answer to this problem is what we have used here: the machine reconstruction technique.[26,35] As a starting point to this study we applied machine reconstruction to the state reached by 10^5 iterations of rule 18 on a periodic array of 10^5 sites, starting from a random initial condition. The results of a series of reconstructions were trivial variants of the two-domain machine $M(\Lambda^{0,0})$ shown in Figure 4. The variations consisted of missing edges or non-deterministic transitions to "dangling" states from which there was no exiting edge, both of which are common signatures of incomplete statistics. With machine reconstruction it was quite straightforward to identify the regular domains and attractors and proceed with the dynamical systems analysis. Plans include an expert-system-like environment to facilitate the discovery of domains and to perform automated construction of vicinities and basins and proofs of invariance and attraction. The knowledge base for such a system consists of the properties of and rules of manipulation and theorems for regular languages and finite automata. It also incorporates basic notions from dynamical systems, such as those used here.

Vicinities

Before turning to the discussion of regular attractors, it is necessary to develop the notion of the *vicinity* of a language. Dynamical systems theory requires that an attractor be stable to small perturbations. But it is necessary to define the precise meaning of "small" in our context, since the discrete nature of CA state space precludes carrying along the standard notion that demands continuity of site values. To this end we make the following observation.

Remark. Regular language space induces a topology on the extended CA state space.

Proof: The set $\mathcal{R}$ of regular languages satisfies the two requirements of the basis of a topology on $\mathcal{A}^*$:
(i) $\bigcup_{L \in \mathcal{R}} L = \mathcal{A}^*$, since $\mathcal{A}^*$ is itself a regular language; and
(ii) $L \cap L' \in \mathcal{R}, \ \forall\, L, L' \in \mathcal{R}$ by the closure of regular languages under intersection.[25] ∎

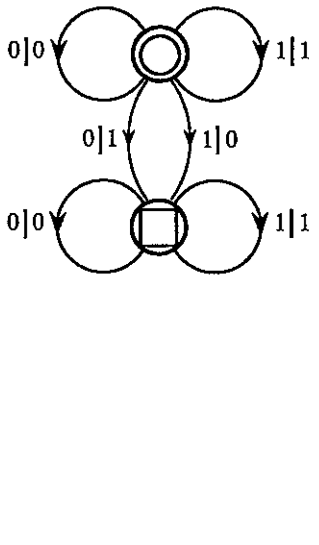
Thus, the extended CA state space possesses the necessary topological structure. All regular languages are open sets in this topology. This allows us to define the regular vicinity of a language as an open set containing that language. The idea is to use the vicinity of a language the way ϵ-balls are used in standard dynamical systems theory.

Definition. A **regular vicinity** $V(L)$ of a language L is (i) a regular language and (ii) an open set in $\mathcal{A}^*$ such that $L \subset V(L)$.

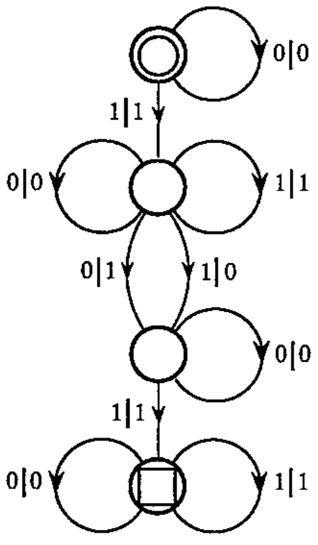
Definition. The **Hamming vicinity** $V_H(L)$ of a regular language L is the regular vicinity generated by adding a unit Hamming distance perturbation each string in L: $V_H(L) = L \dot{+} \delta L$ where $\|\delta L\|_{\text{Hamming}} = 1$, that is, $\delta L = \{0^*10^*\}$, and $\dot{+}$ is sum modulo the alphabet size. Figure 7(a) shows the Hamming vicinity transducer T_{V_H}.

Some problems arise with perturbations of patterns on infinite lattices. For rules with a quiescent state 0^*, one can define "finite initial conditions"[36] as the set of states having a finite nonzero support. Then the effect of the perturbation on these states may depend sensitively on whether it occurs within the nonzero region or outside it. For example, a single bit flip inside the nonzero support of a finite pattern in Λ^0 generates 0 or 2 dislocations, whereas a bit flip outside the nonzero region can create either 0 or 1 dislocation. It is convenient to restrict the vicinity to include only perturbations in the nonzero region. Note that the new vicinity type can be used with finite lattices with no additional consequences, since in that case the support is the whole finite lattice.

Definition. The **restricted Hamming vicinity** $V_{H_{restricted}}(L)$ of a regular language L is the regular vicinity generated by adding a unit Hamming distance perturbation somewhere in the region of nonzero support of each string in L: $V_{H_{restricted}}(L) = L \dot{+} \delta_r L$ where $\|\delta_r L\|_{\text{Hamming}} = 1$ and the perturbation δ_r is restricted to operate in the nonzero support of each string in L. Figure 7(b) shows the restricted Hamming vicinity transducer $\mathrm{T}_{V_{H_r}}$.



(a)



(b)

Figure 7 (a) The Hamming vicinity transducer T_{V_H}. (b) The restricted version $\mathrm{T}_{V_{H_r}}$.

Lemma. $V^0 \equiv V_H(\Lambda^0)$, shown in Figure 8(a), and $V^0_{\text{supp}} \equiv V_{H_{restricted}}(\Lambda^0)$, shown in Figure 8(b), are the regular Hamming vicinity and the restricted Hamming vicinity of Λ^0, respectively. The subscript supp denotes the perturbations' restriction to the region of nonzero support.

Proof: By explicit construction using the definitions: $V^0 = [\mathrm{T}_{V_H} \circ \Lambda^0]_{\text{out}}$ and $V^0_{\text{supp}} = [\mathrm{T}_{V_{H_r}} \circ \Lambda^0]_{\text{out}}$.

Note that V^0_{supp} consists of two disjoint parts: (i) states that remain in Λ^0, and (ii) states in which the perturbation generates two dislocations separated by an even number of 0's.

Attractors

Having set up the necessary geometrical prerequisites, it is now possible to define the regular attractor of a CA. First we define the necessary temporal behavior, and then we combine that with a stability analysis using regular vicinities, to arrive at the final definition.

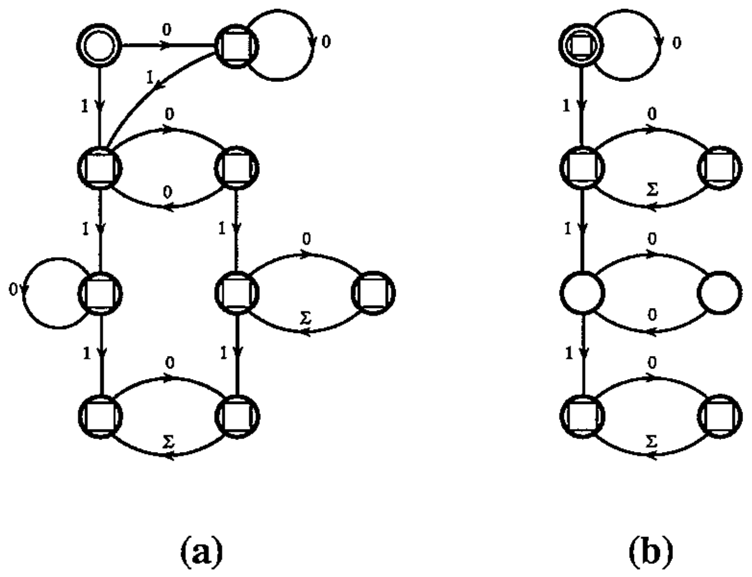


Figure 8 (a) The Hamming vicinity V^0 of the regular domain Λ^0, shown in Figure 4(a). (b) The machine for the language V^0_{supp}, formed from Λ^0 by flipping a single bit in the nonzero region of each state.

Definition. An **invariant language** L is one that is mapped onto itself by the dynamic: $\Phi(L) = L$. A **periodic language** is a language L such that $\Phi_p(L) = L$ for some finite integer period p. A **language orbit** Γ is the set of iterates of a language: $\Gamma = \{\Phi_t(L) : t \geq 0\}$.

Note that since any finite union of regular languages is itself a regular language, any periodic regular language orbit can be turned into a regular invariant language by taking the union over the orbit's elements. Aperiodic language orbits, which can occur only on infinite lattices, may be non-regular, however.

Definition. An invariant language L is a **fixed-point regular attractor** if it is a regular language and there exists a regular vicinity $V(L)$ such that

1. All states in V remain in V forever: $\Phi_t(V) \subseteq V$, $\forall\, t \geq 0$; and
2. All states in V, except for a set of zero spatial entropy, eventually collapse down onto L: $\Phi_t(V) \underset{t\to\infty}{\longrightarrow} L$, for almost all s $\in V$.

The qualifiers "zero entropy" and "almost all" in the definition are absolutely essential to domain-wall dominated CA, as will be seen below. They ensure that the set of states in the vicinity which are not attracted is vanishingly small on sufficiently large lattices. If the nonattracting language is regular, its entropy is given by the logarithm of the maximal eigenvalue of its equivalent DFA connection matrix.[37] If it is not, then one can look at the entropy of equivalent DFAs at each finite lattice size. Then we require that this entropy vanish with increasing lattice size. Altogether these conditions give an operational definition of the caveats.

One of the benefits of this type of definition is that the simple Hamming vicinity is an adequate tool in establishing the existence of attractors. An alternative formulation would have introduced a machine metric that implicitly excluded such nonattracting states; the net result would be the same.

In the following, we take as the regular vicinity of Λ^0 the restricted Hamming vicinity V^0_{supp}, and $V^{0,0}_{\text{supp}} \equiv V_{H_{\text{restricted}}}(\Lambda^{0,0})$ as the regular vicinity of $\Lambda^{0,0}$. Recall that all states in V^0_{supp} have either 0 or 2 dislocations. In the next section we will show that the number of dislocations is nonincreasing with time and dislocations annihilate in pairs, which means that all states in V^0_{supp} remain in V^0_{supp} for all t. Similarly, $V^{0,0}_{\text{supp}}$ consists of states with either 1 or 3 dislocations, and all states in $V^{0,0}_{\text{supp}}$ remain in $V^{0,0}_{\text{supp}}$ for all t.

Proposition. On a finite lattice of sufficiently large size N with periodic boundary conditions, Λ^0 is a regular attractor if N is even, and $\Lambda^{0,0}$ is a regular attractor if N is odd.

Proof: For even N, the periodic boundary conditions require that there be an even number of dislocations in the state at all times. Since $\Lambda^{0,0}$ has a single dislocation, it will never be present on an even lattice. Similarly, Λ^0 will never exist on a lattice with N odd. Thus, one or the other may be present, but never both together. To prove attraction, it is sufficient to show that the set of states that do *not* fall into Λ^0 or $\Lambda^{0,0}$ has zero size for sufficiently large N. Simulations, to be discussed in a later section, indicate that the upper bound on the fraction of states asymptotically not in Λ^0 rapidly decays. The upper bound is estimated to obey a power law $N^{-0.42}$. Furthermore, the number of nonattracted states is already below 1% on lattices with as few as 200 cells. A rigorous proof will appear elsewhere. ▌

Proposition. On an infinite lattice with finite initial conditions, Λ^0 and $\Lambda^{0,0}$ are regular attractors.

Proof: See [36]. All states in which the initial number n_d of dislocations is even fall into Λ^0. If n_d is odd, then the state will fall into $\Lambda^{0,0}$. If we denote the set of finite initial conditions by F, this means in particular that all states in $V^0_{\text{supp}} \cap F$ fall into Λ^0. More formally,

$$\forall s \in V^0_{\text{supp}} \cap F, \exists t < \infty \text{ such that } \Phi_t s \in \Lambda^0$$

Similarly, all states in $V^{0,0}_{\text{supp}} \cap F$ fall into $\Lambda^{0,0}$. ▌

Conjecture. On an infinite lattice with infinite initial conditions, Λ^0 and $\Lambda^{0,0}$ are regular attractors.

Discussion: We consider only Λ^0 in detail. The discussion for $\Lambda^{0,0}$ is quite similar. We have already seen that Λ^0 is a regular attractor for even-sized finite periodic lattices. Further, all states in V^0_{supp} on an infinite lattice with finite support eventually fall into Λ^0. The only remaining case is an infinite lattice with initial conditions of infinite support. The regular vicinity in this case is $V^0_{\text{supp}} - V^0_{\text{supp}} \cap F$. It breaks up into three disjoint subsets.

i. $\{s|\Phi_t s \in \Lambda^0 \text{ for some } t < \infty\}$. This is by far the largest set. Indeed, half of the states in $V^0_{\text{supp}} - V^0_{\text{supp}} \cap F$ are already in Λ^0 at $t = 0$.

ii. $\left\{s|\Phi_t s \underset{t\to\infty}{\longrightarrow} \text{temporally periodic state not in } \Lambda^0\right\}$. The temporal periodicity implies that s is spatially periodic. This is because of the expanding nature of rule 18: local site information is transmitted along the lattice in both directions at the speed of light. If the state as a whole is to be temporally periodic, it must be due to an underlying spatial periodicity. A spatially periodic state of period p evolving on an infinite lattice is equivalent to a primitive period of that state evolving on a finite periodic lattice of p cells, so this set is of vanishing size by a preceding proposition.

iii. $\left\{s|\Phi_t s \underset{t\to\infty}{\longrightarrow} \text{temporally aperiodic state not in } \Lambda^0\right\}$. This the "quasiperiodic" case in which the two dislocations either diverge forever from each other or march along the lattice roughly in parallel, their velocities having the same sign. It is not yet clear whether such states even exist. At the very least, if they do, they are extremely rare, and we ignore them.

The net result is that at $t = \infty$ at most a set of vanishing size remains outside of Λ^0. This implies that all the states in V^0_{supp}, except for a set of vanishing size, eventually fall into Λ^0.

Remark. We expect to be able to use the language-theoretic apparatus we have set up to prove the attracting nature of Λ^0 and $\Lambda^{0,0}$. This will be left for presentation elsewhere. The strategy to be used is as follows. Define $\tilde{V}^0 = V^0_{\text{supp}} - \Lambda^0$, and evolve this language forward in time using the FME algorithm, at each time step excising those states which have fallen into Λ^0. In this way, examine the time series of languages

$$\tilde{V}^0_t = \Phi\left(\tilde{V}^0_{t-1}\right) - \Lambda^0, \text{ with } \tilde{V}^0_0 = V^0_{\text{supp}} - \Lambda^0$$

This is the set of vicinity patterns that have not fallen onto the attractor. Our goal is to find sufficient regularity in them to make it possible to bound the decay in the size of $\tilde{V}^0_t$ as $t \to \infty$. If the asymptotic size is zero, the proposition is proved. The conjecture is that there is a monotonic decrease in upper bound on the size of the nonattracted set and that this decay rate can be estimated by a functional renormalization equation determined by the CA rule acting on the defined ensemble. The spatial rescaling should reveal with increased lattice size a "fractal" structure. The critical exponent associated with the self-similarity will describe that set and determine the size decay rate. All we require is monotonic decrease, a fairly lenient property compared to the existing algebraic and combinatorial analyses.

Proposition. Λ^0 is a regular attracting domain.

Proof: This follows immediately from Λ^0 being both a regular domain and a regular attractor. ▌

This establishes the original claim that the attracting domain Λ^0 organizes the state space. $\Lambda^{0,0}$ is the lowest "energy" fluctuation above the vacuum state Λ^0. Like analogous situations with topological solitons the dislocation in $\Lambda^{0,0}$ is stabilized by the global topology of either finite odd-size or infinite lattices. And so $\Lambda^{0,0}$ is an attractor, although not a domain.

Other Structures

Finally, we note that we can use the above definition of regular attractor as a basis for defining other structures in language space, namely basins and separatrices. These definitions mirror those found in standard dynamical systems theory.

Definition. The set of all preimages of a regular attractor Λ is its **basin:** $\mathcal{B}(\Lambda) = \bigcup_{t \geq 0} \Phi_{-t}(\Lambda)$.
The set of states that are not in basins, but still in the accessible region of configuration space make up the **separatrices**: $\mathcal{S} = \mathcal{A}^N - \bigcup_{\Lambda \in \{\Lambda^i\}} \mathcal{B}(\Lambda)$, where $\{\Lambda^i\}$ is the set of all attractors of the system. Both CA basins and separatrices may contain states with no predecessors. These so-called Garden-of-Eden configurations occur only as initial configurations in $\mathcal{A}^N$. The list of attractors, basins, and separatrices of a CA is its **attractor-basin portrait**.

The attractor basin portrait of rule 18 is the subject of the next section.

Global Structure of Rule 18

Having defined some of the fundamental structural objects in language space, we can now put them together with a discussion of certain behavioral properties determined by the global organization of the state space, to come up with a fairly complete picture of the overall behavior of rule 18. The main question is the following: What usually happens to a typical initial configuration as it evolves in time?

As a first step, we note again that we can treat any initial state s as a concatenation of domains of various lengths, separated by dislocations. Then part of the question is to decide what happens to these dislocations as the state evolves in time.

Lemma.[36] The number of dislocations in an arbitrary CA state evolving under rule 18 decreases or remains constant in time. Furthermore, dislocations always annihilate in pairs.

Proof: (Outline) From the CA rule table. There are three cases, each concerning a spatially-local configuration: (i) no dislocations, (ii) an isolated dislocation, and (iii) two or more nearby dislocations that may or may not collide during the next time step. The first case has been addressed by showing that Λ^0 is an invariant language: no dislocations are created in time. The single-dislocation case is proven by enumerating all local neighborhoods that contain one dislocation. The result is that the dislocation moves to the left or right, but neither disappears or multiplies. Thus, when Ω_0 consists of configurations in Λ^0 or $\Lambda^{0,0}$, no dislocations are created in time. Case (iii) is proven as follows. The local action of the update rule ensures that unless the dislocations' closest boundaries are within $2r$ cells of each other, the dislocations do not interact, and can be treated under case (ii) above. The sole exception to this is when two dislocations and the cells between them are mapped to a region of consecutive 0's, in which case they annihilate and the number of dislocations decreases by two. Enumeration of all 5-cell neighborhoods shows that the number of dislocations either remains the same or decreases by an even number. ▌

Since, in addition, the domain walls appear to execute a random walk on the lattice, as we discuss in detail in the next section, they will eventually all annihilate as far as possible.[24] Thus we have the following statistical characterization that complements the earlier language theoretic analysis.

Proposition. If, given sufficiently large N, the dislocations travel diffusively, then almost every initial configuration in $\mathcal{A}^N$ eventually goes to either the regular attracting domain Λ^0, if N is even, or the two-domain regular attractor $\Lambda^{0,0}$, if N is odd.

Proof: The assumption of diffusive motion ensures that almost all pairs of dislocations will eventually collide and annihilate. If N is even, then there are necessarily an even number of dislocations in any initial condition, and, with probability 1, the state will eventually evolve onto Λ^0. If, on the other hand, N is odd, the number of dislocations is always odd, and almost all initial conditions will eventually fall onto $\Lambda^{0,0}$. The same argument holds for the case when N is infinite, but the initial condition has a finite number of dislocations. For infinite lattices with an infinite number of dislocations, the random walk implies that the spatial density of dislocations vanishes asymptotically.[24] This, in turn, implies that the limiting set of states with a positive dislocation density has vanishing size relative to the full state space with increasing lattice size. Simulation results to be discussed in a later section indicate that the fraction of nonattracted patterns in the attractors' vicinities decays rapidly. The upper bound behaves like $N^{-0.42}$. ▌

The central conclusion to be drawn from this discussion is that dislocations annihilate, leading eventually to simpler, *not* more complex, spatial patterns. The regular language complexity in the conventional "global" analysis[4] blows up due to the complexity of the motion of all possible dislocations implicit in the initial ensemble of all possible configurations. There are many routes of evolution onto the attractors, thus very large machines will persist for arbitrarily long times. The exact topological analysis necessarily includes these, but is unfortunately also dominated by them.[†] Consequently, these machines obscure important features such as the *typicalness* of the patterns observed and the simple decomposition into domains and walls. The simple, observed fact is that the number of dislocations is almost always decreasing. This means that ultimately the language, for all initial conditions but a vanishingly small set, asymptotically becomes quite simple.

[†]We note that the direct probabilistic generalization of the FME algorithm also obscures this structure, since it carries along the full topology in the support and also is sensitive to all variations in the initial configuration distribution.[38]

There is undoubtedly a deep connection between the apparent diffusivity of the dislocations and the structure of the attractors Λ^0 and $\Lambda^{0,0}$. The exact nature of that connection, and the particular mechanism by which the randomness of the initial ensemble is transformed into space-time random walks, remain open questions. Nevertheless, the greater part of the behavior of rule 18 is described by the picture of computationally simple domains bounded by dislocations that wander about at random, annihilating when they collide.

Transducers for Global Structure Analysis

Given this identification of the gross behavior of rule 18, we can organize our discussion around the observed domains and the dislocations between them. In doing so, we will find use for three different transducers, all based on the domain language Λ^0: the machine metric $||s||_{M(\Lambda^0)}$ that counts the number of dislocations in a spatial state s, an intercalation operator $T_{\mathrm{intercalate}}(\Lambda^0)$, and a domain isomorphism $T_{\mathrm{decimate}}(\Lambda^0)$. The machine metric was introduced above in a more general context. The remaining two transducers are defined here.

The site-intercalation transducer $T_{\mathrm{intercalate}}(\Lambda^0)$, shown in Figure 9, is a machine for removing dislocations by adding extra sites to the lattice. Within a domain it is the identity on all transitions, i.e. 0|0 and 1|1, but when a wall is detected we put 1|01. The result is a many-to-one mapping from arbitrary patterns to patterns in Λ^0. By inserting the additional 0, the wall is effectively removed. The mapping $f_{T_{\mathrm{intercalate}}(\Lambda^0)} : \mathcal{A}^* \rightarrow \Lambda^0$ is equivalent to the global linearization operator $G(s)$ of [36]. Since the evolution of Λ^0 is identical in rule 18 and rule 90, this operator linearizes rule 18 in the sense that the rule becomes equivalent to the linear, superposition preserving dynamics of rule 90.

This type of global linearization applies only to patterns consisting of domains of the same type, i.e. the same regular attractor. For CA with more than one domain type $\{\Lambda^i\}$ such global linearization does not exist over the entire state space. Rather it appears only on subspaces composed of configurations with domains of the same type.

The domain isomorphism transducer $T_{\mathrm{decimate}}(\Lambda^0)$, shown in Figure 10(a), demonstrates in a transparent manner that rule 18 acting on Λ^0 is equivalent to rule 90 acting on $\mathcal{A}^*$. Its effect is to drop the (predictable) 0 transition from the input pattern. This can also be considered an encoding and is, in fact, a compression of the input pattern. The corresponding function $f_{T_{\mathrm{decimate}}(\Lambda^0)} : \Lambda^0 \rightarrow \mathcal{A}^*$ maps the domain patterns of rule 18 onto the full configuration space. It is important to point out that in general decimation of space-time patterns requires both a space-like machine to decimate spatial configurations and a time-like machine to decimate in time. The proper framework for this requires introducing space-time machines. We defer this discussion to later,[35] since pursuing it here would take us too far afield. In this particular example the issue can be effectively skirted because the space, time, and space-time machines are identical.

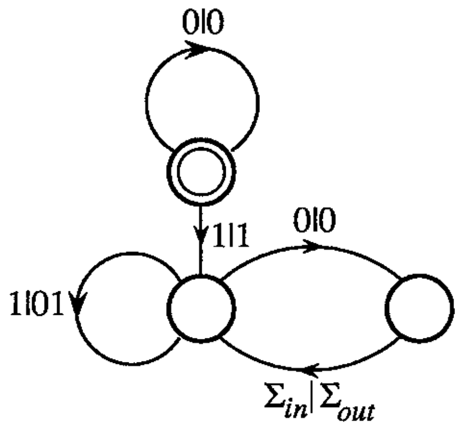


Figure 9 Transducer $\mathrm{T}_{\mathrm{intercalate}}(\Lambda^0)$ for inserting new sites at dislocations. The result of such an operation is a global linearization of rule 18.[36] Note that when a dislocation is detected two symbols, rather than one, are emitted. The transducer takes length N strings to length $(N+d)$ strings, where d is the number of dislocations in the input string.

One must also look at the effect of $f_{\mathrm{T}_{\mathrm{decimate}}(\Lambda^0)}$ on the rule table. From this we find the machine commutation relation

$$\mathrm{T}_{\Phi_{90}} \circ \mathrm{T}_{\mathrm{decimate}(\Lambda^0)} = \mathrm{T}_{\mathrm{decimate}(\Lambda^0)} \circ \mathrm{T}_{\Phi_{18}}$$

In terms of languages, direct comparison of the space-time patterns shows that the action of rule 18 on neighborhoods of size five after two time steps is identical to rule 90 acting on neighborhood of size three after one time step, when one drops intervening zeros:

$$\begin{array}{ccccccccc} \cdots & 0 & s_t^{i-2} & 0 & s_t^i & 0 & s_t^{i-2} & 0 & \\ \cdots & s_{t+1}^{i-3} & 0 & s_{t+1}^{i-1} & 0 & s_{t+1}^{i+1} & 0 & s_{t+1}^{i+3} & \cdots \\ \cdots & 0 & s_{t+2}^{i-2} & 0 & s_{t+2}^i & 0 & s_{t+2}^{i+2} & 0 & \cdots \end{array} \xrightarrow[\mathrm{T}_{\mathrm{decimate}}]{} \begin{array}{ccccc} \cdots & s_t^{i-2} & s_t^i & s_t^{i+2} & \cdots \\ \cdots & s_{t+2}^{i-2} & s_{t+2}^i & s_{t+2}^{i+2} & \cdots \end{array}$$

where $s \in \mathcal{A}$. In other words, we have the language commutation relation

$$\Phi_{90} \circ f_{\mathrm{T}_{\mathrm{decimate}(\Lambda^0)}}(\Lambda^0) = f_{\mathrm{T}_{\mathrm{decimate}(\Lambda^0)}}(\Lambda^0) \circ \Phi_{18}(\Lambda^0)$$

Thus, not only is there an equivalence on patterns, but also on the action of the rules. The transducer implements this directly, being the projection operator for the associated commuting diagram

$$\begin{tikzcd} \Lambda^0 \arrow[r, "\Phi_{18}"] \arrow[d, "f_{\mathrm{T}_{\mathrm{decimate}}(\Lambda^0)}"'] & \Lambda^0 \arrow[d, "f_{\mathrm{T}_{\mathrm{decimate}}(\Lambda^0)}"] \\ \mathcal{A}^* \arrow[r, "\Phi_{90}"'] & \mathcal{A}^* \end{tikzcd}$$

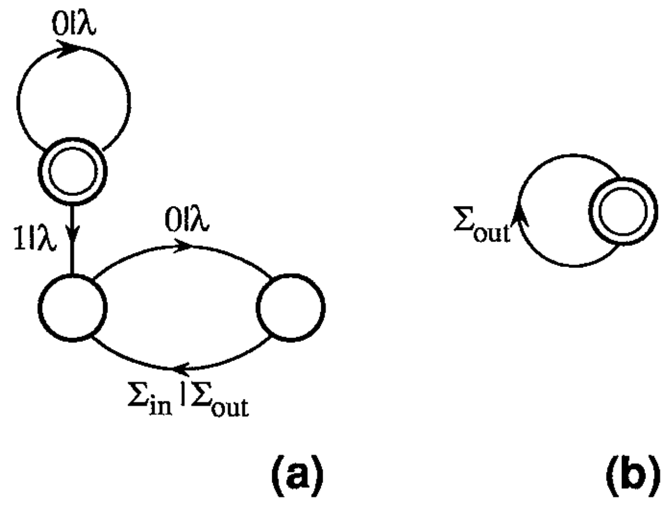


Figure 10 (a) Transducer $T_{\text{decimate}}(\Lambda^0)$ for removing, in fact compressing or encoding, the patterns in Λ^0 into $\mathcal{A}^*$. (b) The transducer has been minimized with respect to its output. The resulting automaton accepts $\mathcal{A}^*$.

If we minimize it with respect to the output alphabet, then we obtain $\mathcal{A}^*$, which is invariant under rule 90. One consequence is that rule 18 is linear on the domain in the sense that $\Phi(\Lambda^0)$ obeys a linear superposition in space-time. (See also[36].)

We can use the inverse decimation operator $f^{-1}_{T_{\text{decimate}}(\Lambda^0)}$ to enumerate a cascade of unstable periodic sublanguages embedded in Λ^0. If we think of the latter as defined by the rule "every other site is arbitrary and the rest are zero", then these sublanguages are "every $(2^k)th$ site is arbitrary and the rest are zero", assuming, of course, that $k > 0$. These are regular languages, which are denoted Λ^0_k, $k > 0$. In this notation $\Lambda^0 = \Lambda^0_1$. Clearly $\Lambda^0_{k+1} \subset \Lambda^0_k$. They are the successive preimages of Λ^0 under $f_{T_{\text{decimate}}(\Lambda^0)}$; that is,

$$\Lambda^0_{k+1} = f^{-1}_{T_{\text{decimate}}}(\Lambda^0_k),\ k > 0$$

For all positive k, Λ^0_k is a periodic language with temporal period 2^{k-1}. Each language orbit

$$\left\{\Phi_t(\Lambda^0_k)\ :\ 0 \le t < 2^{k-1},\ k > 0\right\}$$

is easily shown to be a domain. When the orbit recurs the origin is shifted by 2^{k-1} sites. Finally, $\Lambda^0_k, k > 1$, are unstable to unit Hamming perturbations and decay to Λ^0.

Basins and Separatrix Structure

The basin of Λ^0 consists of configurations, aside from Λ^0 itself, that have an even number of dislocations, but are neither temporally periodic nor quasiperiodic. That is,

$$\mathcal{B}(\Lambda^0) = \Lambda^0 \bigcup \left\{ \mathrm{s} \in \mathcal{A}^N : \|\mathrm{s}\|_{\mathrm{M}(\Lambda^0)} = 2n, n = 1, 2, \ldots,\ \text{and}\ \lim_{t\to\infty} \|\Phi_t \mathrm{s}\|_{\mathrm{M}(\Lambda^0)} = 0 \right\}$$

Similarly, for $\Lambda^{0,0}$ we have

$$\mathcal{B}(\Lambda^{0,0}) = \Lambda^{0,0} \bigcup \left\{ \mathrm{s} \in \mathcal{A}^N : \|\mathrm{s}\|_{\mathrm{M}(\Lambda^0)} = 2n+1, n = 1, 2, \ldots, \text{ and } \lim_{t \to \infty} \|\Phi_t \mathrm{s}\|_{\mathrm{M}(\Lambda^0)} = 1 \right\}$$

The preceding analysis indicated that Λ^0 and $\Lambda^{0,0}$ are the only two important attractors. The global separatrix $\mathcal{S}$ then consists of the set of spatial states that do not evolve into Λ^0 and $\Lambda^{0,0}$. We can formally write this in set theoretic notation as

$$\mathcal{S} = \mathcal{A}^N - \mathcal{B}(\Lambda^0) - \mathcal{B}(\Lambda^{0,0})$$

It is also the set of states for which the number of dislocations remains greater than 1 forever. As we noted above, one set, perhaps the largest, for which this is true is the set of temporally periodic states with more than one dislocation. Examples of this are readily obtained by looking at periodic lattices with only a few cells. For instance, the spatial configuration $(1000010000100010)^*$, with two dislocations in the basic pattern, when iterated on periodic lattices with $N = 16m$, $m \geq 1$, or N infinite, has temporal period $p = 14$. However, all such languages are also *spatially* periodic, and hence have zero spatial entropy. For an infinite lattice there are also quasiperiodic orbits, in which, for example, two dislocations (i) travel away from each other forever or (ii) translate in parallel with a roughly constant or periodic separating distance.

We identify three disjoint subsets of the separatrix $\mathcal{S} = \mathcal{S}_{\mathrm{even}} \bigcup \mathcal{S}_{\mathrm{odd}} \bigcup \mathcal{S}_{\infty}$. $\mathcal{S}_{\mathrm{even}}$ contains those periodic orbits, and their preimages, with an even number of dislocations greater than one; $\mathcal{S}_{\mathrm{odd}}$ has an odd number greater than one. $\mathcal{S}_{\infty}$ contains all the aperiodic, quasiperiodic orbits.

The foregoing analysis specifies the attractor-basin portrait. The remaining problems include identifying the complexity class of the basins and separatrices and giving their finite descriptions, if such exist. We leave these to the future.

The Attractor-Basin Portrait

We gather together the preceding results for an essentially graphical display of the global view of the attractor basin (AB) portrait of rule 18. We present two views. The first is somewhat skeletal and is the picture provided by regular language space. The second is schematic and gives a cartoon of the state space of arbitrary, but high dimension.

From the viewpoint of regular language space the AB portrait is relatively simple. We can identify the basic sets in a skeletal AB portrait in which the vertices of the graph are various languages. A skeletal picture of rule 18's attractor-basin portrait as seen in regular language space is given in Figure 11. The various symbols indicate the formal languages identified in the computation theoretic analysis: attractors $\{\Lambda^0, \Lambda^{0,0}\}$, basins $B^0 = \mathcal{B}(\Lambda^0)$ and $B^{0,0} = \mathcal{B}(\Lambda^{0,0})$, and separatrices $\mathcal{S} = \{\mathcal{S}_{\mathrm{even}}, \mathcal{S}_{\infty}, \mathcal{S}_{\mathrm{odd}}\}$. The temporal evolution from one language to another goes down the figure. The context of this diagram is a lattice of arbitrary size N, either infinite or finite with periodic boundary conditions. If N is odd, then only the portion to the right of the right vertical dashed line is observed. If N is even, then only that to the left of the left dashed line is relevant. In an infinite lattice the entire diagram is present. The vertical dashed lines also denote the extent to which small perturbations can affect states in the attracting languages. A unit Hamming distance perturbation $\delta_\tau\Lambda$ from the attractor drives the state to its basin or, at worst, to a periodic orbit in the associated subset of the separatrix. In fact, topological constraints prohibit bit-flip perturbations within the nonzero support from driving the configuration to the other attractor. If the lattice is infinite and the initial patterns are restricted to a finite support with nonzero site values, then there are no separatrices, since all states eventually collapse onto one or the other attractor in finite time.[36] Unit Hamming distance perturbations outside the region of nonzero sites then can introduce single dislocations and so move from B^0 to $B^{0,0}$, and visa versa. The subseparatrices $\mathcal{S} = \{\mathcal{S}_{\rm even}, \mathcal{S}_{\rm odd}, \mathcal{S}_\infty\}$ are disjoint. $\mathcal{S}_{\rm even}$ and $\mathcal{S}_{\rm odd}$ consist of temporally periodic patterns with an even and odd number of dislocations greater than one asymptotically in time, respectively. $\mathcal{S}_\infty$ contains all aperiodic patterns with more than one dislocation, and all patterns with an infinite number of dislocations, in the limit $t \rightarrow \infty$.

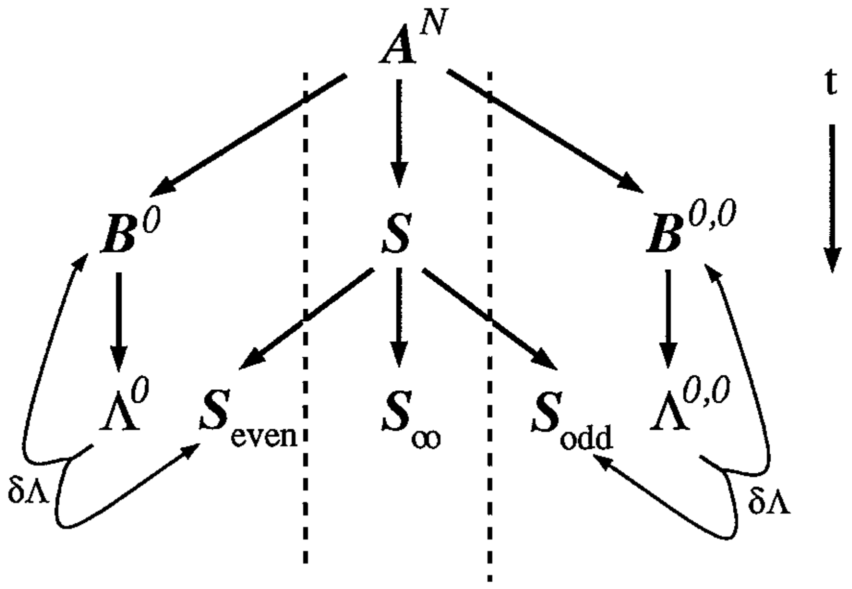


Figure 11 A skeletal picture of rule 18's attractor-basin portrait as seen in language space. The various symbols indicate the formal languages identified in the computation theoretic analysis. The temporal evolution from one language to another goes down the page. The thin lines with arrows indicate the relaxation of perturbations $\delta\Lambda$ to the attractor.

But there is much more that can be said about the configuration space itself. Generally, local particle annihilation dynamics induces a hierarchical structure of subbasins and portals.[39] The subbasins are subspaces in which the number of dislocations is constant: $||\mathrm{s}||_{\mathrm{M}(\Lambda^0)}$ = constant. Thus they are level sets of this metric: $b_d = \left\{ \mathrm{s} : ||\mathrm{s}||_{\mathrm{M}(\Lambda^0)} = d \right\}$. They are connected by portals which are defined solely in terms of the particle annihilation geometry. Since the number of dislocations cannot increase, there is a flow directed from higher to lower dimensional subspaces. Here by "dimension" we refer to the number of "active" degrees of freedom which we take to be the number of dislocations. In the transition from a d-subbasin b_d, caused by two dislocations annihilating, the number of dislocations decreases by two. And so the state moves from a d-subbasin to a $(d-2)$-subbasin. The subbasin-portal structure is shown schematically in Figure 12.

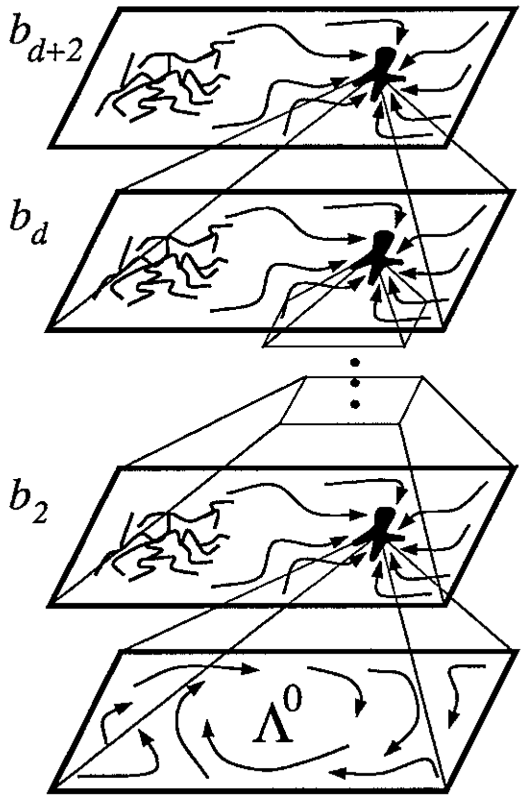


Figure 12 The state-space view of the attractor-basin portrait of rule 18 for an even length lattice or, equivalently, the left-most portion of Figure 11. The diagram illustrates the subbasin-portal structure of the state space.[39] The subbasins correspond to each horizontal plane. Within each subbasin the number of dislocations is constant. The portal is indicated by the solid dark blob. When pairs of particles annihilate the state passes through the portal. Its shape is determined by the geometric constraints of annihilation. That portion of the separatrix consisting of configurations with the subbasin's number of dislocations is shown as a filamentary structure to denote its vanishing relative size. The general flow of time is down the figure. Asymptotically in time and lattice size, the overwhelming fraction of initial patterns in V^0 end up on the attractor Λ^0, all of the dislocations having annihilated.

The intercalation operator $f_{T_{\text{intercalate}}(\Lambda^0)}$ maps the configuration space of a N site lattice containing d dislocations to the domain subspace of a $(N+d)$ lattice on which the rule is linear.[36] This gives an additional hierarchical constraint to the state space.

It is important to point out that the state space structures implied by the portal concept and the intercalation operator are not the same. Portals, on the one hand, are local in the state space and their geometric structure is entirely determined by the local "real" space constraints on particle annihilation. The intercalation operator, on the other hand, is global in both the state and "real" spaces. The result is that the portals still organize the state space of multiple distinct-domain dynamics, a situation in which the global intercalation operator does not exist. Indeed, the underlying views behind each are complementary. Understanding their relationship is certain to lead to an even more detailed picture of the state space geometry of domain-wall dominated CA than that presented in this section.

Dislocation Trajectories

This section presents a concrete application of the preceding ideas. We give a construction procedure that builds detectors for propagating space-time structures in domain-wall dominated behavior. The application uses these transducers to find dislocations in rule 18 in order to investigate their statistical behavior in detail. We are interested specifically in the diffusive behavior of an isolated dislocation and of a dislocation gas. The results strongly suggest that rule 18 is very close to an ideal diffusion process with annihilating particles. This study of the statistical properties along with the dynamical systems analysis gives a very complete picture of rule 18.

Detection

We can recode space-time data to show the dislocation trajectories generated by rule 18 by taking the domain-machine $M(\Lambda^0)$ and defining a transducer $T_{\text{domain}}(\Lambda^0)$ that emits a (domain) symbol D on allowed transitions of Λ^0, and a (wall) symbol W on disallowed transitions. This dislocation recognizer is shown in Figure 13. In general, for an arbitrary set of domain types $\{\Lambda^i\}$, it is straightforward to construct a transducer that will encode the data to show each of the domain and wall structures.

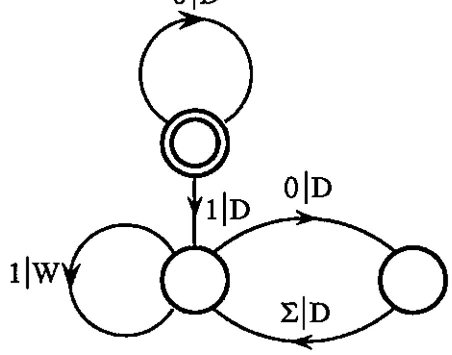


Figure 13 Transducer $T_{\text{domain}}(\Lambda^0)$ for recognizing dislocations in rule 18 domain structure Λ^0.

For rule 18 running $T_{\text{domain}}(\Lambda^0)$ over the space-time data gives an encoded picture that shows only the dislocation trajectories. Figures 14, 15, and 16 are typical of simulations starting with a random initial condition. Each of these pictures shows a different space-time region of the same simulation. Note the progressively reduced scale in Figures 14 and 15 used to make the diffusive behavior more apparent. The thickness of the lines is due to the figure's finite resolution. In Figure 16, blackened squares correspond to dislocations and hollow squares show the nonzero cells of the original pattern.

The transducer is run over the spatial data in a particular direction: in our case, from left to right. This introduces a spatial asymmetry into the picture. For example, the visible asymmetry in the trajectory at the top of a triangle of white space (Figure 16) is due to the fact that as $T_{\text{domain}}(\Lambda^0)$ is moved to the right over the data, it identifies a dislocation only by reading 1 in the wrong place. Thus, a 1 followed by an even number of consecutive 0's shows a dislocation at the right side, when the next 1 is encountered. This has the effect of a constraint on the motion of the observed dislocation: all moves to the left are a single cell, but moves to the right may be any odd-valued distance. Further, a move left may be followed by a move either to right or left, but a move right must be followed by one to the left. More complicated transducers can be defined to evade this directionality. For example, the transducer could label all blocks $1(00)^n1$, $n = 0, 1, 2, \ldots$, as walls. With this detector, the dislocation becomes delocalized as it enters a triangle of 0's. The associated detector for bounding the delocalization region is more complex, requiring an infinite queue (FIFO) memory, if it is to not only recognize dislocations but also to label all participating sites appropriately.

Given a machine that will encode the space-time data to recognize dislocations, it is possible to examine the encoded data for other, higher-order regularities. By designing the transducer properly, one can use this process to identify structures of arbitrary regular language complexity. The result is the original language "modulo" the transducer machine. In this all the encoded structures are excised, giving a clearer picture of long-range structures. This technique is applicable to a large class of systems, including those which possess regular domains. It also generalizes to structures of higher complexity, using recognizers based (say) on stack transducers that implement context-free or context-sensitive grammars for recognition.

Figure 14 Trajectories of dislocations evolving from a random initial configuration. The time and space scales are highly contracted to show the diffusive behavior. Every fifth iteration is displayed. The square indicates the region displayed in Figure 15.

Figure 15 Expanded view of a part of Figure 14. Every iteration is displayed. The tick mark at $t = 52000$ shows the location of Figure 16. The apparent break in the trajectory just below that point shows the recognized dislocation making a large lateral jump to the right in a single time step.

Figure 16 Trajectory of a single dislocation recognized by $\mathrm{T}_{\mathrm{domain}}(\Lambda^0)$. This is an expanded view of a part of Figure 15, located at the tick mark on the vertical axis of that figure. The region shown is in fact smaller than the width of the tick.

Figure 17 The dislocation plume: Evolution of the initial ensemble $\Lambda^{0,0}$ consisting of configurations with a single dislocation. The gray scale intensity of a space-time cell is proportional to the number of dislocations visiting there. It is normalized at each time step to run from white to black so the distribution tails are more apparent.

Figure 18 Dislocation histogram at times $t = 1, \; 10,$ and 100.

Figure 19 Standard deviation of the dislocation histogram as a function of time. The measured slope of the asymptotic line is 0.501 ± 0.004 for $t > 100$. Logarithms are base 2. The straight line shown has slope $\frac{1}{2}$.

Diffusion

Having recoded the space-time data according to the preceding scheme, it is simple to measure the statistical properties of an ensemble of dislocations. Consider the set of dislocation trajectories originating from a uniformly distributed ensemble of initial conditions belonging to $\Lambda^{0,0}$. We translate the states so that the dislocations lie initially at the same site. This initial distribution is formed by using $\mathrm{M}(\Lambda^{0,0})$ as a pattern generator, taking uniform transition probability at branchings. The result obtained by evolving N_{total} members of this ensemble forward in time is the "dislocation plume" shown in Figure 17. There the gray scale darkness of a space-time cell (i, t) is proportional to the number $N_d(i, t)$ of dislocations visiting it. To improve contrast it has been normalized so that $\max_i \{p_d(i)\} = 1$ at each time t, where $p_d = N_{\mathrm{total}}^{-1} N_d(i, t)$. Note the "checkerboard" pattern of the plume. This is due to the fact that in a single time step a dislocation can move in space by only an odd number of cells. Figure 18 shows the plume histogram at times $t = 1, \; 10,$ and 100, in which the checkerboard pattern of the histogram has been factored out. The histogram's asymmetry, noticeable at the early time steps, is due to right-going dislocation detection. The relaxation from the early time binomial process to the Gaussian is clearly demonstrated.

The plume's width $\sigma(t) \equiv \sqrt{\langle x^2 \rangle}$, plotted against time in Figure 19, quickly approaches the time dependence of a random walk. As the straight line in Figure 19 indicates we find $\sigma(t) \approx (2Dt)^{-\alpha}$ with a least-squares fit for $t > 512$ giving $D = .506$ and $\alpha = 0.501 \pm 0.004$. The systematic deviation from Gaussian diffusion seen at early times is again due to the recognition asymmetry which gives $2^{-\frac{i}{2}}$ tails to the distribution and to the binomial structure of the dislocation movement. The result of the exact analysis of these effects is that, for example, $\log \sigma(1) = \frac{3}{2}$. The simulations give $\log \sigma(1) = 1.52 \pm 0.05$.

To complete the statistical analysis of rule 18 diffusion dynamics facilitated by the dislocation detection transducer, we re-measure the temporal decay rate of the dislocation number density. Grassberger[23] estimated it using a rarefied dislocation gas on a very large lattice; that is, initial patterns with widely separated dislocations of the form $(\Sigma 0)^*11(0\Sigma)^*$. Additionally, dislocations after $t = 0$ were detected only as isolated pairs of 1s; i.e. as $(\Sigma 0)^*11(0\Sigma)^*$ again. Thus, dislocations were not counted in any other form, such as $(\Sigma 0)^*10^{2n}1(0\Sigma)^*, n = 1, 2, \ldots$. The consequence is that statistics could only be gathered relatively infrequently and for a limited type of initial ensemble. Here we avoid these problems entirely and collect substantially better statistics on unrestricted initial conditions by using the domain metric to estimate the dislocation number density

$$d_N(t) = N^{-1} \left\| \Phi_t\left(\mathcal{A}^N\right) \right\|_{\mathrm{M}(\Lambda^0)}$$

We count the density $d_N(t)$ of dislocations at each time in a Monte Carlo sample of $\Omega_0 = \mathcal{A}^N$. The result is graphed in Figure 20, where we see $d_{10^5}(t)$ plotted against time. As the straight line suggests, we find $d(t) = \frac{1}{\sqrt{8\pi D}} t^{-\alpha}$ with least-squares analysis giving values of $D = 0.54$ and $\alpha = 0.49 \pm 0.04$. The estimate is from a fit over data at all times.

Figure 20 Dislocation number density versus time on a periodic lattice of 10^5 sites. The initial pattern was random. The density is plotted at each of $100,000$ time steps. Again, logarithms are base 2. The straight line has slope $\frac{1}{2}$.

Finally, we briefly mention the vicinity convergence properties. The fraction of vicinity patterns of Λ^0, say, that have not been attracted to Λ^0 after t on an N site lattice is given by

$$f(N, t) = \frac{\left\| \tilde{V}_t^0(N) \right\|}{\left\| V^0(N) \right\|}$$

where $\tilde{V}_t^0 = \Phi_t\left(V^0\right) - \Lambda^0$. We estimate this statistic via a Monte Carlo sample of V^0. There are two results of immediate use. First, on even-size lattices up to 100 sites we found an upper bound $f_{\max}(N, \infty) \propto N^{-\gamma}$ with $\gamma = 0.42 \pm 0.05$ using 20,000 initial vicinity patterns. The largest nonattracted fractions were found at lattice sizes N_i^* obeying the recursive rule $N_{i+1}^* = 2N_i^* + 2$, $N_1^* = 4$. Thus, although $f(N, t)$ has erratic, number-theoretic fluctuations, the fraction of nonattracted vicinity patterns decreases rapidly. We also found that for fully one-third of the lattice sizes the nonattracted patterns were a vanishing fraction. The second result is that, on a lattice of 1,000 sites using 10,000 initial vicinity patterns, we found near-diffusive decay of $f\left(10^3, t\right) \propto t^{-\alpha}$ with $\alpha = 0.47 \pm 0.04$. Thus, when restricted to the vicinity, decay to the attractor also appears to be diffusive, with the fraction of nonattracted patterns decreasing monotonically in time and in lattice size.

The figures and the estimated spreading, decay, and finite-size scaling rates give strong evidence that the dislocation trajectories are diffusive. Via the above results on domains, attractors, and vicinities, the diffusive behavior further supports the proposition that the attractors Λ^0 and $\Lambda^{0,0}$ are the only statistically important attracting invariant sets of the system. As noted above, assuming that the dislocations are actually diffusive, the previous results on their attracting properties can be established in the statistical sense. That is, the fraction of initial configurations leading to diffusive motion of two dislocations that do not annihilate is vanishingly small. The appendix gives an estimate for this fraction.

Computing with Dislocations

The domain isomorphism $f_{\mathrm{T}_{\mathrm{decimate}}(\Lambda^0)}$ defined above is a generalization of an encoding procedure elsewhere called the "blocking transformation".[1] Recall that in the latter scheme every other cell in space and time is decimated, for example, and the resulting pattern is treated as a new space-time pattern. This new pattern represents the evolution of the encoded initial condition under a different CA rule, generally one with a larger neighborhood size. The spatial period 2 nature of Λ^0 makes this transformation particularly striking for rule 18. Considering the blocking transformation as the restricted type of transducer that it implements suggests a new interpretation of its effect. When a domain is in phase with the blocking scheme's internal state, arbitrary spatial patterns will be observed. When it is out of phase, however, all the nonzero sites will be removed. The resulting blocked pattern is a homogeneous patch of 0's. Blocked space-time patterns of arbitrary states evolving under rule 18 therefore show the characteristic signature used to identify Wolfram's class 4 behavior: spatially isolated transients of many shapes evolving for arbitrarily long times.[1] One might therefore argue that rule 18 should be placed in his class 4. However, we have shown that these structures are caused only by the wall dynamics. They are misinterpreted with blocking transformations since the domains alternate between being in-phase and out-of-phase with the blocking. All of the important, potentially useful computation is performed by the domain boundaries. The evolution of the boundaries of an in-phase domain governs the evolution of the blocked patterns. When the boundaries meet, they annihilate and the "spatially isolated transient" disappears.

The use of finite transducers suggests a re-evaluation of CA classification, especially class 4. CA in this class have been conjectured to be capable of universal computation.[1] But the classification ultimately was based on direct visual observations of spatially isolated propagating blobs. The conjecture was that using these propagating blobs a serial computational model could be emulated. Rule 18, though, can be easily recoded, i.e. misleadingly block coded, to show such structures. Its computational capability appears quite limited, being dominated by diffusive behavior of the dislocations. It is not clear whether that behavior can be harnessed to perform arbitrary computations. We suggest that a principled definition of class 4 would be based on estimating, (say) via machine reconstruction,[26,37] a CA's intrinsic computational capability. See [26] for a related discussion of the computational classification of conventional dynamical systems.

Having identified the source of significant computation, we can make a quantitative estimate of the computational capability of rule 18 dislocations in the following way. A single trajectory can be coded as a symbol stream, e.g., by assigning the symbol L to a move to the left and R to a move right. We can then apply the machine reconstruction technique to this R-L data stream. The resulting machine, displayed in Figure 21, shows an asymmetric structure: every R must be followed by an L. Noting that the topological entropy of this machine is $\log_2 \phi$, where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden mean, we call it the golden mean machine M_{gm}. The asymmetry in M_{gm} is due to the asymmetry of the dislocation-recognizing transducer $\mathrm{T}_{\mathrm{domain}}(\Lambda^0)$ which identifies dislocations by a misplaced 1 in the spatial data set. As we noted above, other dislocation-recognizing algorithms can be used for which this constraint would not be present. Apart from this, however, we see that all trajectories consistent with this single constraint are allowed. Numerical estimates indicate a uniform branching probability from the start state. Thus, one type of computation of which rule 18 is capable is the map from $\Lambda^{0,0}$ to $L(\mathrm{M}_{\mathrm{gm}})$.

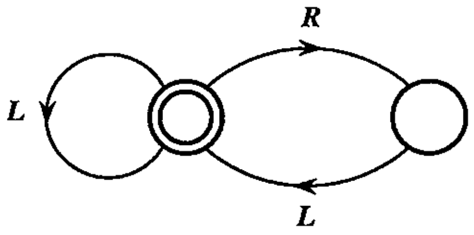


Figure 21 The golden mean machine M_{gm} reconstructed from the space-time trajectory of a single dislocation. The symbol L denotes a dislocation move to the left; R a move right. Its entropy is $\log_2 \phi$, where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden mean.

For intrinsic computation to be useful, there is the additional requirement that it be robust. That is, at the very least, small perturbations must not change the computational class of the temporal evolution. This in turn requires an understanding of the global organization the CA induces in regular language space. Concretely, we can associate a CA's computational capability with the either evolution toward attractors with higher than regular language complexity or with the existence of propagating wall structures. The latter are useful only if their movement can be controlled via a compiler. A dislocation compiler, for example, would be a mapping of a computational task specification T to an initial condition $s_o \in \Lambda^{0,0}$ such that the evolution s_t executes the task via the $R - L$ movement of wall and domain structures. The output would be a string in $L(\mathrm{M}_{\mathrm{gm}})$. Many alternatives suggest themselves, but we must leave the question of the existence of such compilers for the future.

Although formulated in the present context of spatially-extended systems, the general questions raised concerning the design and existence of compilers for CA are germane to parallel programming and distributed processing. Should such compilers exist, then analyses along the lines developed here would indicate where useful computation can lie in high-dimensional state space. They also would give a global view of the geometry underlying the computation. The practical use of this appears to be in the analysis of error and stability for a given CA implementation.

Concluding Remarks

Domains and dislocations are basic dynamical paradigms. In physics dislocations are observed as crystal defects and elementary particles, in chemistry they are seen in polymer chains and chemical waves, and in biology they are found in cellular membrane development and they have been posited to be the information processing foundation of prebiotic evolution.[40]* These structures often result from lateral inhibition coupling between processing elements, to use the neurophysiological term. This same coupling is responsible for the ability of artificial neural networks to learn, although in these high-dimensional systems their nonspatial architecture make the domain and dislocation structures substantially more difficult to visualize. We have avoided this important difficulty by focusing on spatially-extended systems. The attempt in the preceding was to set up a constructive framework to study structures supporting complex information processing in high dimensional state spaces.

Let us recapitulate our results. First, we developed a structural analysis of CA based on elementary notions from computation theory. Second, this allowed us to describe the attractor-basin portrait and subbasin-portal structure of a specific CA. Third, along the way, we introduced specific tools for observing and manipulating CA structures: the various transducers or, more formally, endomorphisms of Sofic systems. Finally, as an application these were applied to the statistical analysis of dislocation trajectories within specific subspaces of the state space. The results, along with the attractor-subbasin-portal portrait, strongly suggest that there are only two statistically significant attracting invariant sets. A rigorous proof of this remains an open problem.

Within a larger arena, we have considered one aspect of the general problem of recognizing structures in space-time. Consider, as another example, a soliton wave in a shallow water channel. Typically, the recognition of a propagating structure is done intuitively via visual inspection. In some cases, such as with solitons, this identification eventually develops a formal expression. The propagating entity has a mathematically defined shape expressed in closed form. With this one can search for them by convolving the space-time pattern with the soliton kernel. In a literal sense, the shapes form a basis for the space of recognizable patterns. The method of transducer machine recognition introduced above gives a different, but complementary, computational approach to space-time pattern recognition, as we have just described it, that is both general and reconstructive. It, along with machine reconstruction, suggests an automated method of identifying significant structures and tracking them in space-time. Although the present approach starts with discretized data, we note that rather than a liability this is also the domain of many important problems in time series and image analysis and processing. The authors are well aware of the need to extend the present techniques to continuum-state spatially-extended systems.

During the attractor-basin analysis, we alluded to another general problem, concerning transients in high dimensions.[39,42,43] Rule 18 is one nontrivial, i.e. nonlinear, space-time

*An additional brief list is given in [41].

system that apparently can be analyzed completely in this way. The key elements in this are the global linearization techniques of the intercalation operator,[36] the dynamical homogeneity of regular domains that we have introduced here, and the organization of subbasins and portals.[39] The analysis of domain-wall dominated CA hints at similar analyses for a wider class of spatially-extended systems, such as map lattices and soliton-bearing PDEs.

Implicit in our investigation of subspaces of initial configurations, represented by taking Ω_0 to be various regular process languages other than $\mathcal{A}^*$, is a set of techniques to study the geometry of high-dimensional spatially-extended systems. This was motivated, in fact, by techniques used in earlier work on continuous-state lattice dynamical systems[44] and their extremely long transients.[39,42] And these too seem applicable to a wide range of similar problems posed by high-dimensional dynamical systems.

In order to illustrate the basic ideas we have used the example of rule 18. The question naturally arises of whether the analysis generalizes to other CA. The answer is yes. The additional rules can be seen by simply examining the space-time diagrams of other CA. Due to the reconstructive approach we have taken and the computation theoretic formulation of the dynamical systems concepts, the interesting possibility arises that the analysis for these other CA can be automated. We have alluded how an automated machine reconstruction "expert system" could be used as an aid to analysis. The goal is for it to help with automated proofs of language invariance and attraction, with the derivation of basins and vicinities, and in the investigation of the computational structure of basins and separatrices.

The analysis we have given is not complete, though our and others' results strongly suggest the possibility that it is nearly so. Rather it outlines a range of problems and a constructive computational approach that integrates qualitative dynamics. We are left, therefore, with several future questions. We close by mentioning a few of the most pressing ones. Of what complexity are the basins of the two regular attracting domains? How about the separatrix? The near completeness of the analysis suggests that there should be a rigorous proof that the deterministic rule 18 is diffusive. This will require an exact analysis of the short time behavior and an analogous development of approximation of stochastic process languages.[38] The relationship between the intercalation linearization and the subbasin-portal picture is important to finally understand the global structure of the state space. Finally, how can we compute with dislocations? Does a dislocation compiler exist? If not, why is this structure not usable to perform computations?

Acknowledgements

The authors thank Peter Grassberger and Erica Jen for helpful discussions. Visits to the Center for Nonlinear Studies and the Theoretical Division at the Los Alamos National Laboratory are warmly acknowledged. They are also grateful to Professor Carson Jeffries for his continuing support. This work was funded in part by ONR contract N00014-90-J-1774, NASA-Ames University Interchange NCA2–488, and a Cray Research, Inc., University Research and Development grant. It was completed at the Santa Fe Institute where JPC was supported by a Robert Maxwell Foundation Visiting Professorship.

Bibliography

[1] S. Wolfram. *Theory and Applications of Cellular Automata.* World Scientific Publishers, Singapore, 1986.

[2] A. R. Smith. Simple computation-universal cellular spaces. *J. ACM*, 18:339, 1971.

[3] T. Toffoli. *Cellular Automata Mechanics.* PhD thesis, University of Michigan, Michigan, 1977. Tech. Rep. 208, Comp. Comm. Sci. Dept.

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

[5] L. Hurd. Formal language characterizations of cellular automaton limit sets. *Complex Systems*, 1:69, 1987.

[6] E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning Ways for your Mathematical Plays*, volume 2. Academic Press, New York, 1984.

[7] T. Toffoli and N. Margolis. *Cellular Automata Machines: A New Environment for Modeling.* MIT Press, Cambridge, Massachusetts, 1987.

[8] O. Martin, A. Odlyzko, and S. Wolfram. Algebraic properties of cellular automata. *Commun. Math. Phys.*, 93:219, 1984.

[9] K. Kaneko. Attractors, basin structures and information processing in cellular automata. In S. Wolfram, editor, *Theory and Applications of Cellular Automata.* World Scientific, Singapore, 1986.

[10]H. Ito. Intriguing properties of global structure in some classes of finite cellular automata. *Physica*, 31D:318, 1988.

[11]E. Jen. Cylindrical cellular automata. *Comm. Math. Phys.*, 118:569, 1990.

[12]S. Wolfram. Statistical mechanics of cellular automata. *Rev. Mod. Phys.*, 55:601, 1983.

[13]H. A. Gutowitz, J. D. Victor, and B. W. Knight. Local structure theory for cellular automata. *Physica*, 28D:18, 1987.

[14]W. Li, N. H. Packard, and C. G. Langton. Transition phenomena in cellular automata rule space. preprint, 1990.

[15]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. preprint, 1990.

[16]J. Park, K. Steiglitz, and W. Thurston. Soliton-like behavior in automata. *Physica*, 19D:423, 1986.

[17]P. Grassberger. New mechanism for deterministic diffusion. *Phys. Rev. A*, 28:3666, 1983.

[18]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.

[19]K. Lindgren. Correlations and random information in cellular automata. *Complex Systems*, 1:529, 1987.

[20]D. R. J. Chillingworth. *Differential Topology with a View to Applications*. Pitman, London, 1976.

[21]J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer-Verlag, New York, 1983.

[22]J. P. Crutchfield and J. E. Hanson. Computational mechanics of cellular automata: Space and time machines. in preparation, 1991.

[23]P. Grassberger. Chaos and diffusion in deterministic cellular automata. *Physica D*, 10:52, 1984.

[24]D. Griffeath. *Additive and Cancellative Interacting Particle Systems*, volume 724 of *Springer Lecture Notes in Mathematics*. Springer, Berlin, 1979.

[25]J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA, 1979.

[26]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*, page 223. Addison-Wesley, 1990.

[27]J. P. Crutchfield. Reconstructing language hierarchies. In H. A. Atmanspracher, editor, *Information Dynamics*, New York, 1990. Plenum.

[28]R. E. Blahut. *Principles and Practice of Information Theory*. Addision-Wesley, 1987.

[29]D. A. Lind. Applications of ergodic theory and sofic systems to cellular automata. *Physica*, 10D:36, 1984.

[30]M. G. Nordahl. Formal languages and finite cellular automata. *Complex Systems*, 3:63, 1989.

[31]L. P. Hurd. Appendix. table 10. regular language complexities. In S. Wolfram, editor, *Theory and Applications of Cellular Automata*. World Scientific, Singapore, 1986.

[32]N. H. Packard. Complexity in growing patterns in cellular automata. In J. Demongeot, E. Goles, and M. Tchuente, editors, *Dynamical Behavior of Automata: Theory and Applications*. Academic Press, 1984.

[33]S. Wolfram. Universality and complexity in cellular automata. *Physica*, 10D:1, 1984.

[34]F. Harary. *Graph Theory*. Addison-Wesley, 1969.

[35]J. P. Crutchfield and J. E. Hanson. Computational mechanics of cellular automata: Spacetime machines. in preparation, 1991.

[36]E. Jen. Aperiodicity in one-dimensional cellular automata. *Physica*, 44D, August 1990.

[37]J. P. Crutchfield and K. Young. Inferring statistical complexity. *Phys. Rev. Let.*, 63:105, 1989.

[38]K. Lindgren and M. G. Nordahl. Complexity measures and cellular automata. *Complex Systems*, 2:409, 1988.

[39]J. P. Crutchfield. Subbasins, portals, and mazes: Transients in high dimensions. *J. Nucl. Phys. B*, 5A:287, 1988.

[40]A. G. Cairns-Smith. *Genetic takeover and the mineral origins of life*. Cambridge University Press, New York, 1982.

[41]J. P. Crutchfield. Spatio-temporal complexity in nonlinear image processing. *IEEE Trans. Circ. Sys.*, 37:770, 1988.

[42]J. P. Crutchfield and K. Kaneko. Are attractors relevant to turbulence? *Phys. Rev. Let.*, 60:2715, 1988.

[43]J. P. Crutchfield and K. Kaneko. Transients in high dimensions. in preparation, 1991.

[44]J. P. Crutchfield and K. Kaneko. Phenomenology of spatio-temporal chaos. In Hao Bai-lin, editor, *Directions in Chaos*, page 272. World Scientific Publishers, Singapore, 1987.

Appendix: Dislocation Decay Statistics

We estimate for an infinite lattice the probability that randomly diffusing dislocations have not annihilated by some time t. This also gives an estimate of the fraction, or basin measure, of those initial patterns which have decayed by t.

Consider the effect of flipping a single cell in each state of an ensemble consisting of a uniform distribution on Λ^0. We assume that the perturbation occurs in the region of compact support, i.e., there are nonzero cells on both sides of it. We count dislocations using the machine metric $\|\cdot\|_{M(\Lambda^0)}$ defined above. With probability $\frac{1}{2}$, the cell flipped will be a wild-card, so the perturbation will not generate any dislocations. Otherwise, it will produce two dislocations a distance d apart, with $d-1$ equal to the number of adjacent 0's to the right of the perturbed cell. The uniform distribution of initial conditions implies that the wild-card cells are independently, identically distributed with $p(\sigma = 0) = \frac{1}{2}$. This in turn implies that the probability that the dislocations are exactly d cells apart is given by

$$p(d) = \begin{cases} 2^{-\frac{d+1}{2}} & d \text{ odd} \\ 0 & d \text{ even} \end{cases}$$

Thus, the probability that two dislocations are d_t or fewer cells apart is

$$p(d \le d_t) = \sum_{d=1}^{d_t} p(d) = 1 - 2^{-\frac{d_t+1}{2}}$$

for d_t odd.

Two dislocations will typically collide when their plumes overlap; that is, when $\sigma \approx d$. As noted in the main text, the standard deviation of the plume gives a time-dependence of $\sigma \approx 8\pi D t^{-\frac{1}{2}}$ with $D \approx \frac{1}{2}$.

The probability that two dislocations have collided after t steps is the probability that their initial separation was σ or less, which is given by

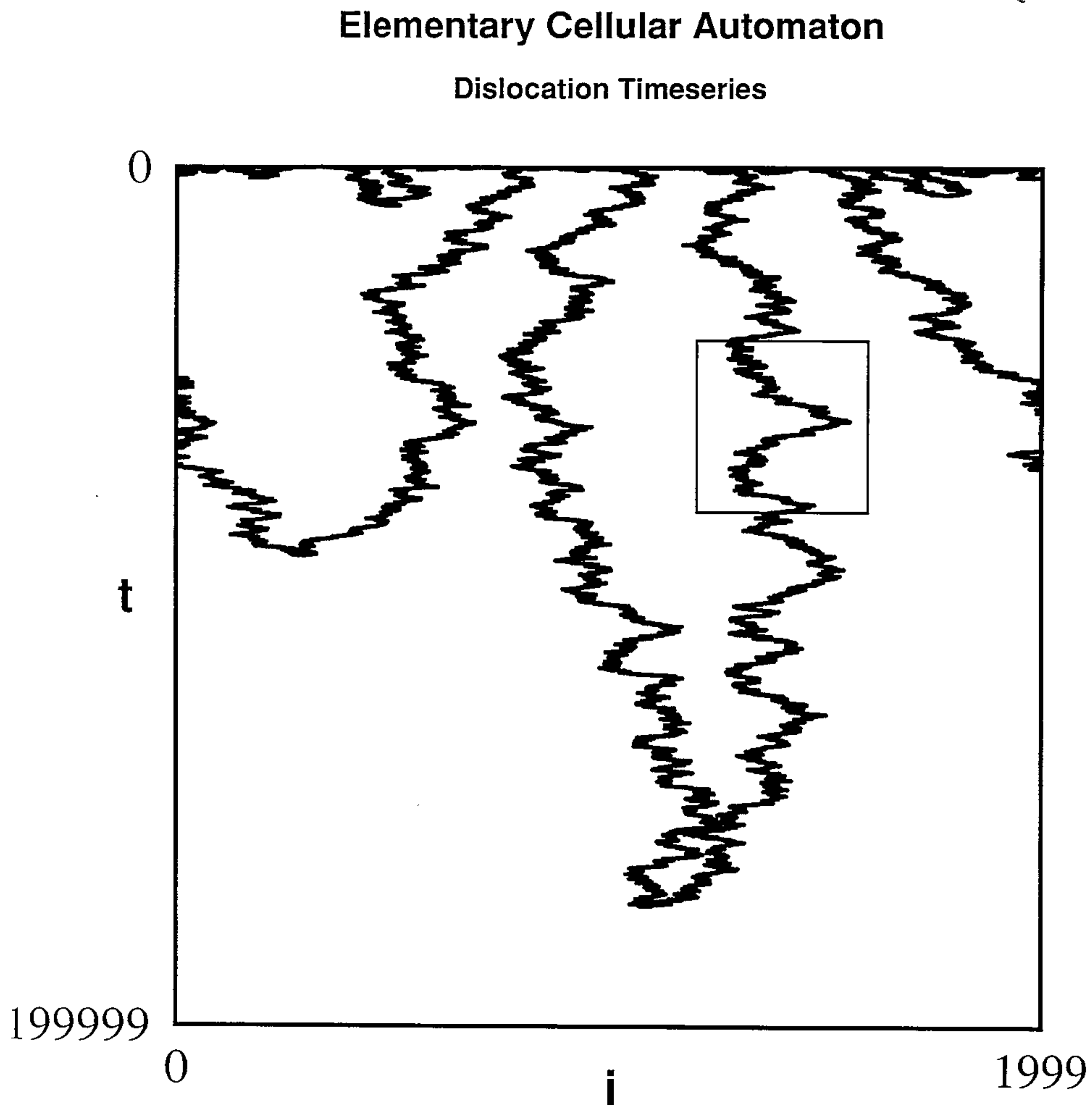
$$p(d \le \sigma) = 1 - 2^{-\frac{8\pi D\sqrt{t}+1}{2}}$$

For long times, the probability that the dislocations have not yet collided is therefore

$$p_{\text{survive}}(t) \approx 2^{-2\pi\sqrt{t}}$$

This gives an estimate of the decay in the number of initial conditions that have not yet fallen onto the attractor, assuming diffusive motion. It clearly vanishes in the limit $t \to \infty$.

Fig 14

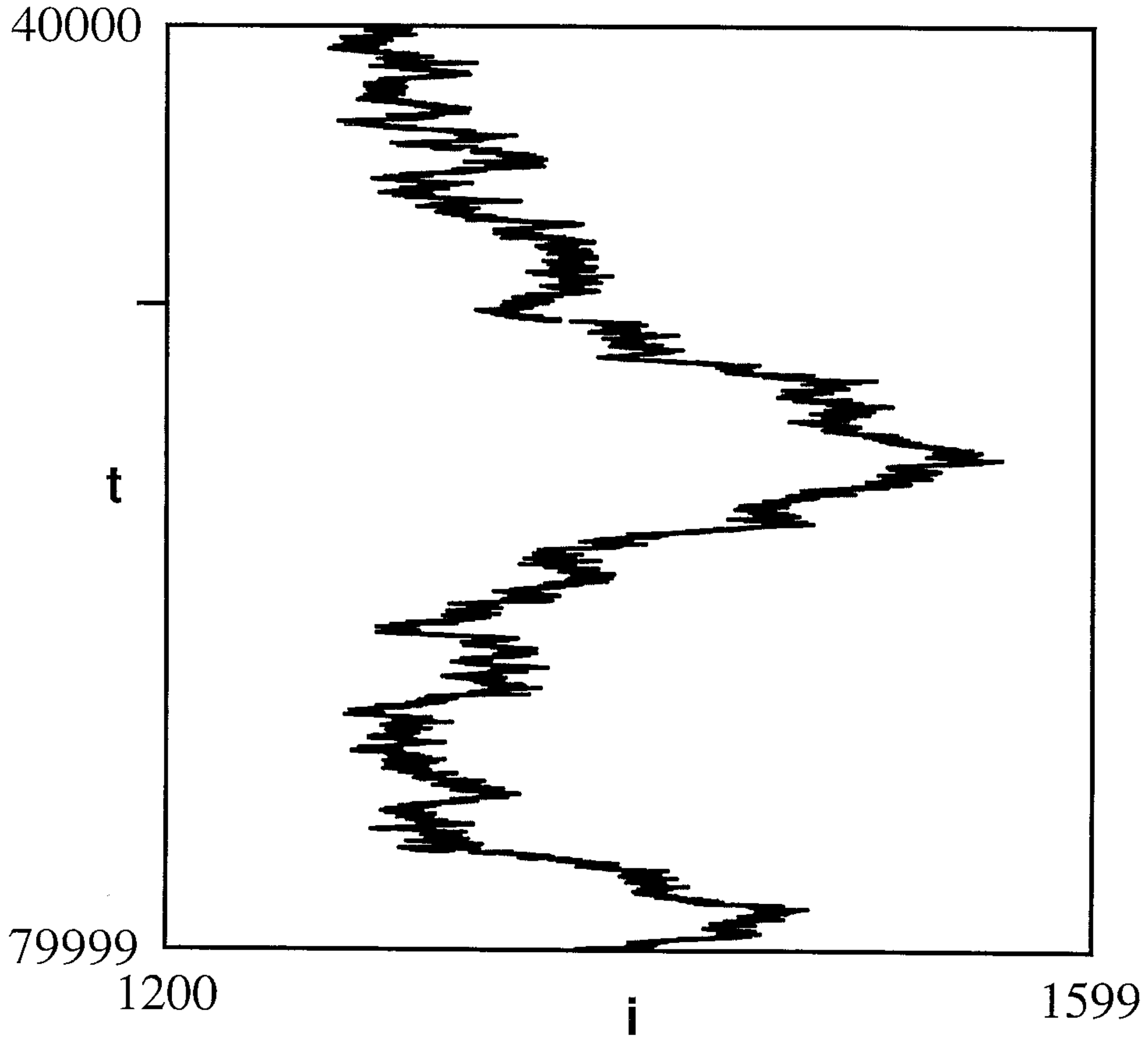


Rule 18, Periodic BC
Random initial condition
2000 cells in spatial array
Dislocations shown in black
1 in 5 iterations displayed

Fig. 15

Elementary Cellular Automaton

Dislocation Timeseries

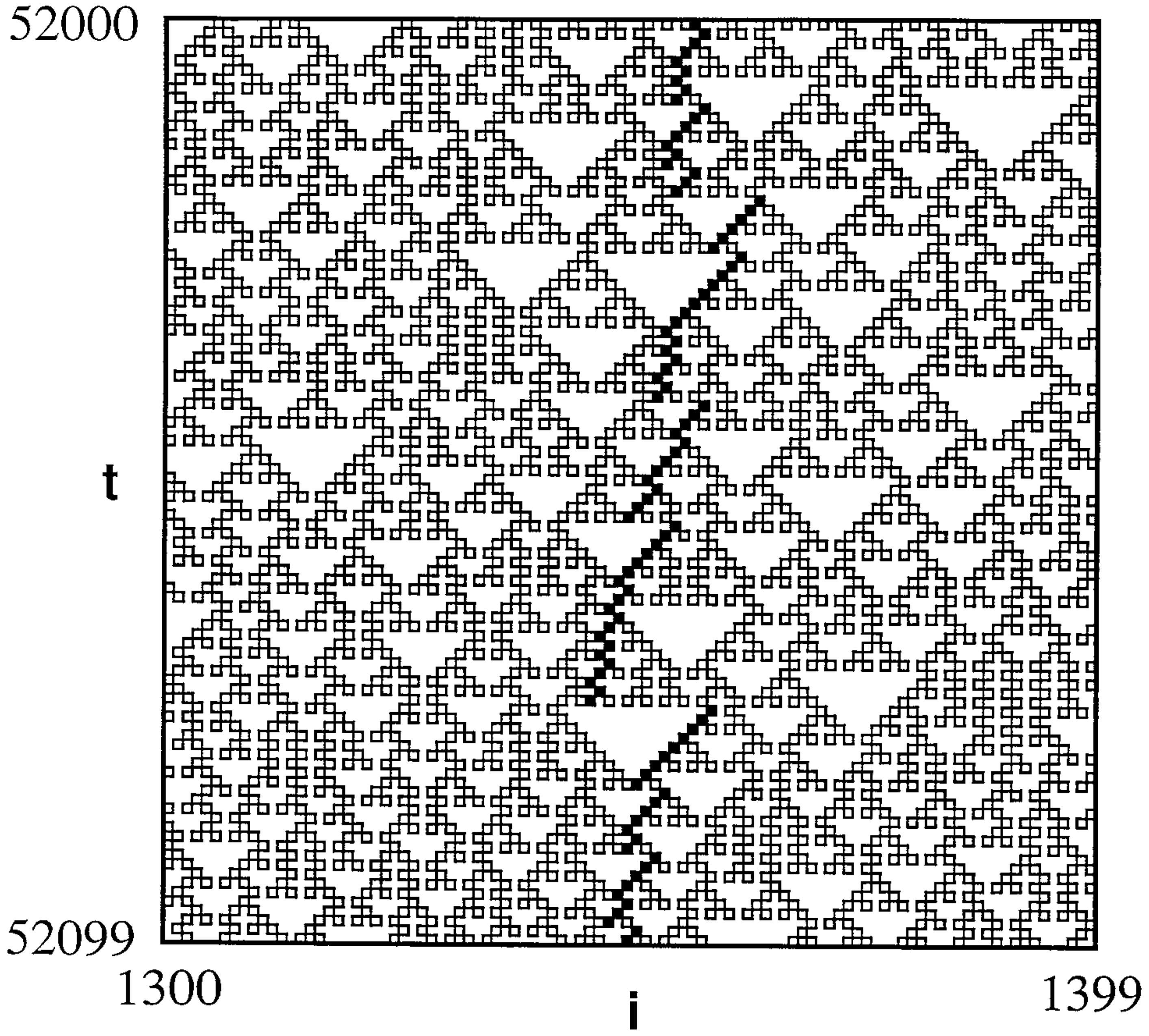


Rule 18, Periodic BC
Random initial condition
2000 cells in spatial array
Dislocations shown in black
All iterations displayed

Fig. 16

Elementary Cellular Automaton

Dislocation Timeseries

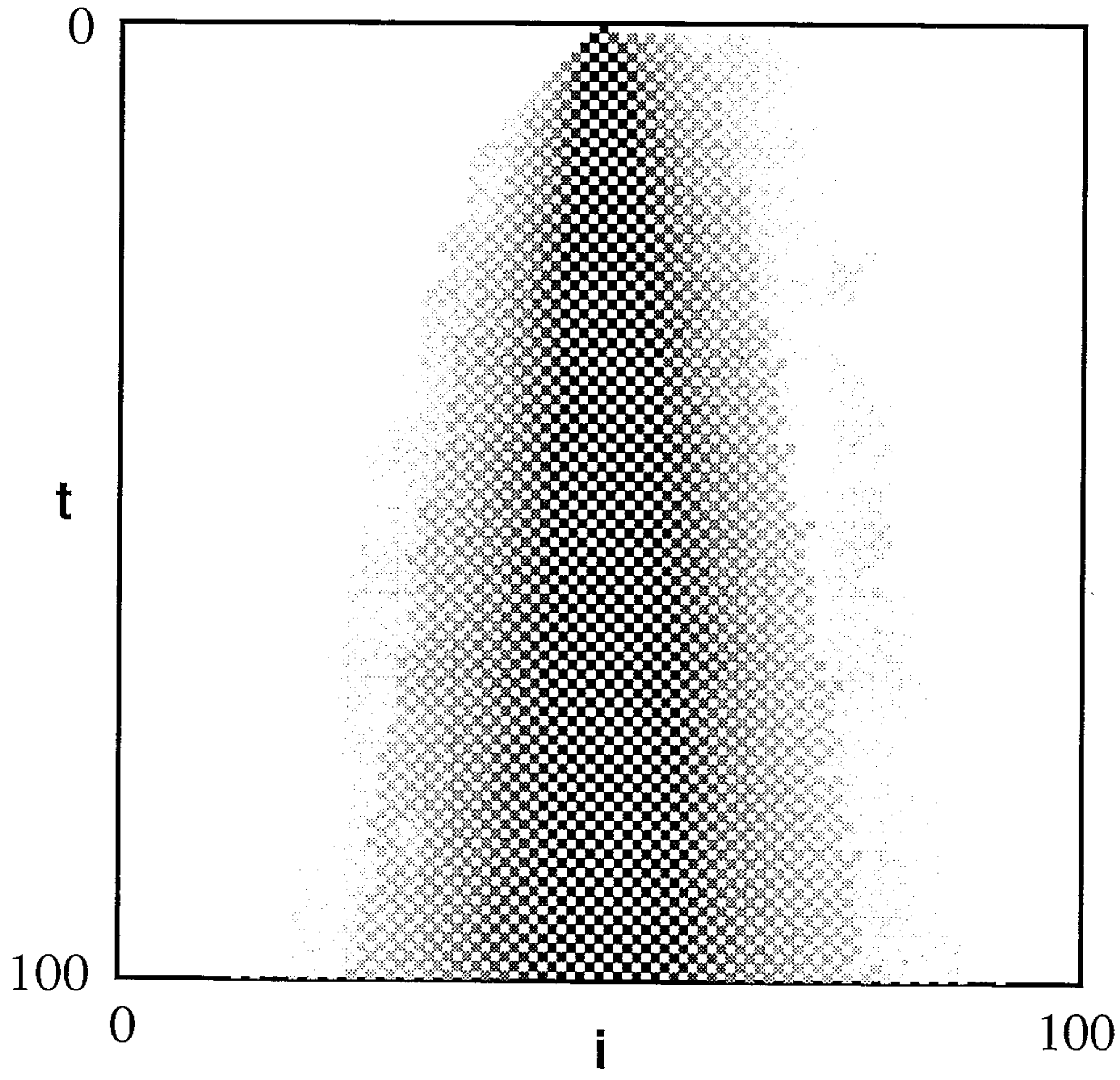


Rule 18, Periodic BC
Random initial condition
2000 cells in spatial array
Dislocations shown in black
Nonzero cells shown in outline
All iterations displayed

Fig 17

Elementary Cellular Automaton

Dislocation Plume Histogram

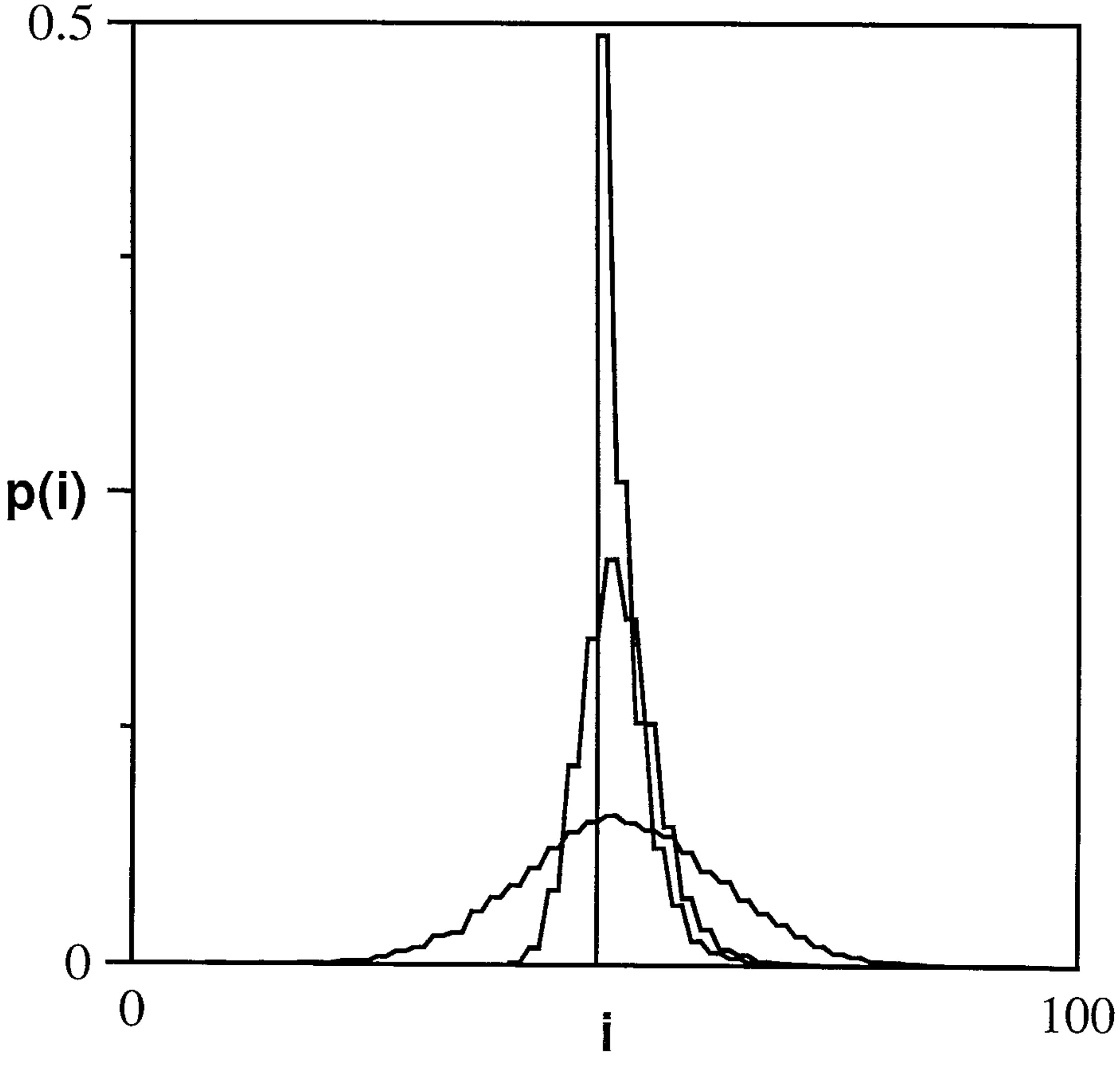


Rule 18, Periodic BC
10000 States, 101 Cells
Initial Dislocation at cell 50
Time increment = 1
Spatial increment = 1

Fig. 18

Elementary CA Rule 18

Dislocation Plume Histograms



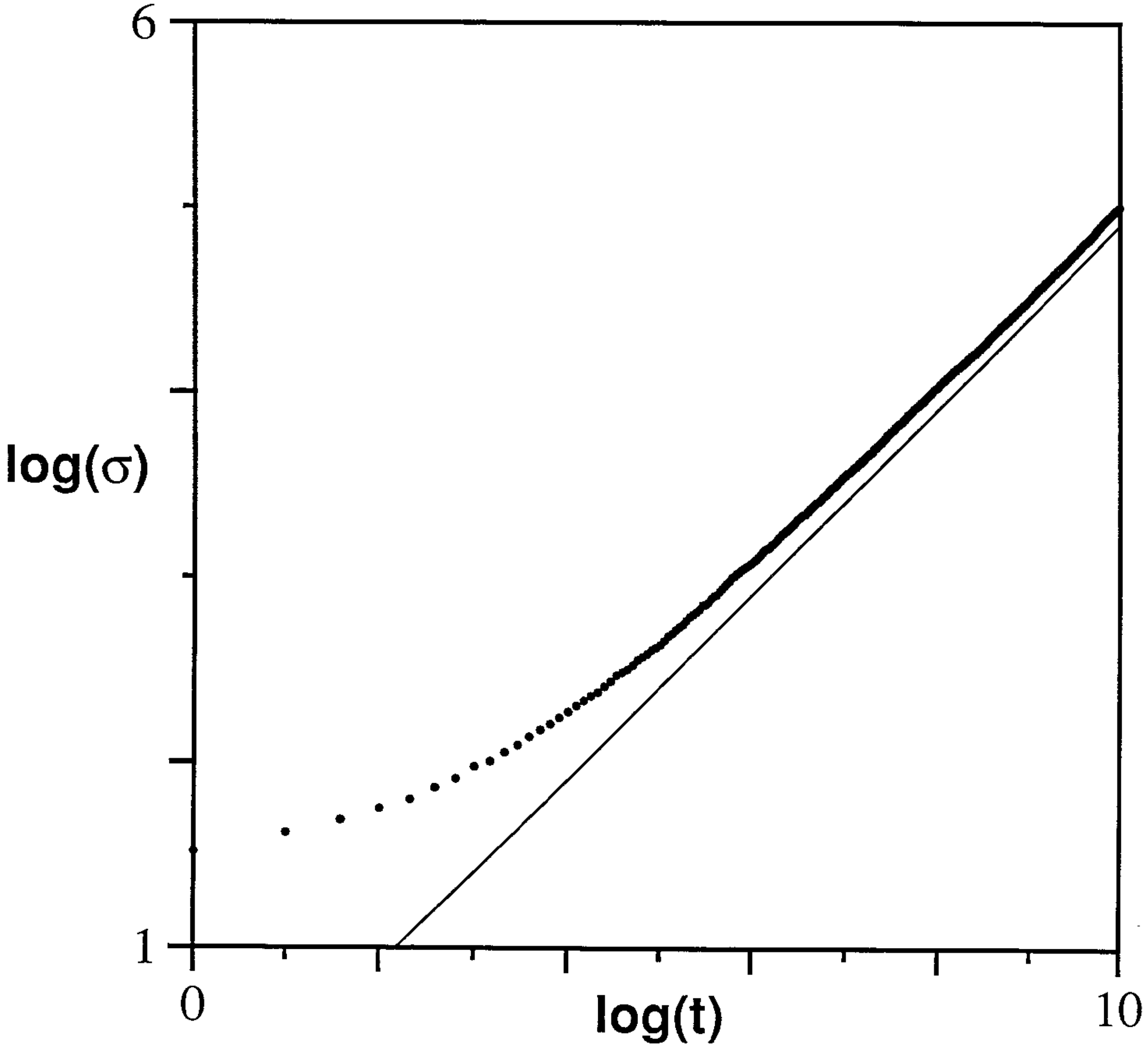
Periodic Boundary Conditions
10000 states, 101 cells
IC: Single disloc at cell 50
Normalized to total area = 1.

Histogram times:
t = {1, 10, 100}

Fig 19

Elementary CA Rule 18

Dislocation Plume Width



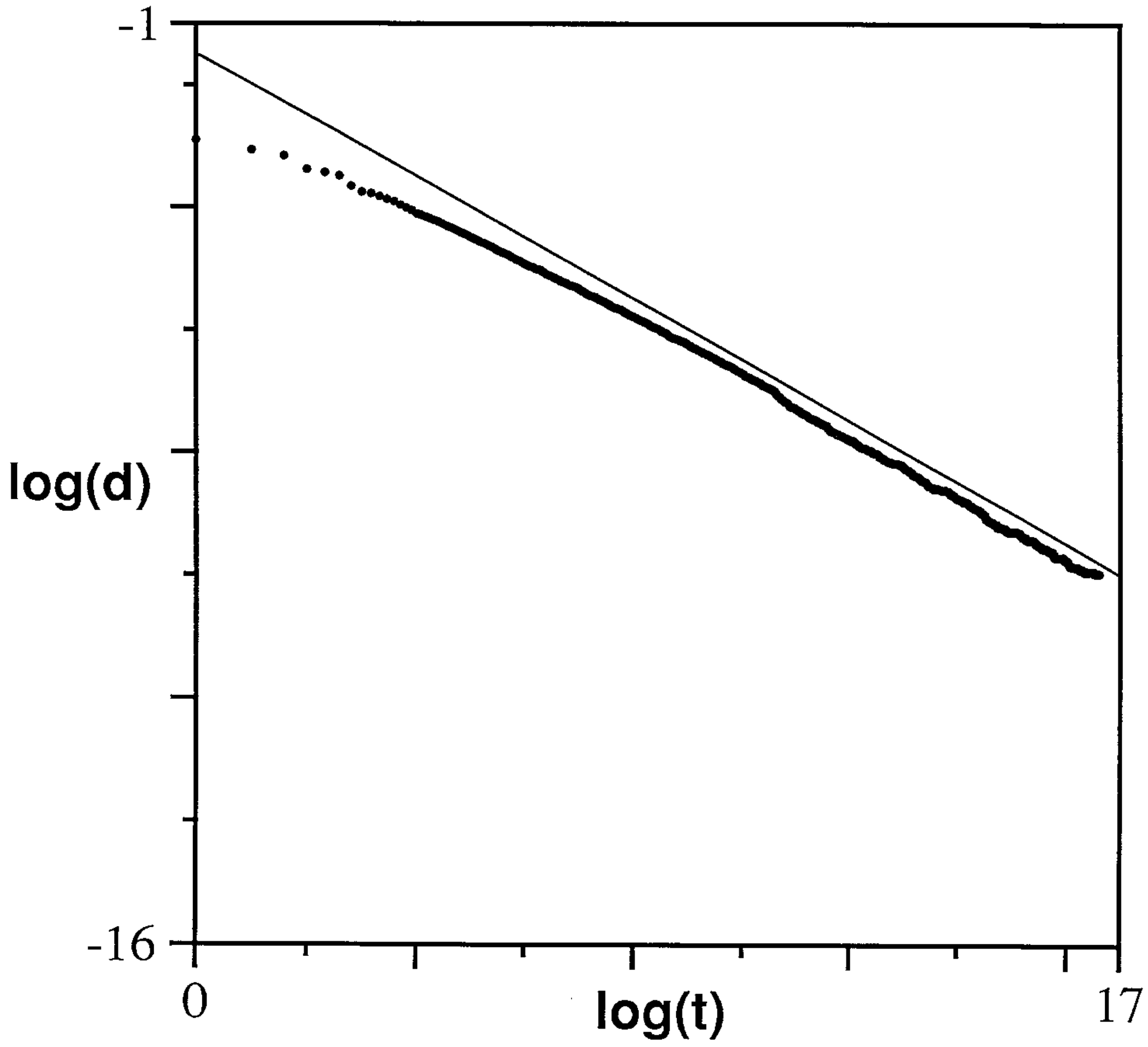
Periodic Boundary Conditions
10000 states, 501 cells
IC: Single disloc at cell 250
Time increment = 1

Least-squares fit of data:
Fit to points in [512, 1023]
Slope = 0.501811
Intercept = -0.0085978
Coeff. of det. = 0.999352
Chi-squared = 1.32713e-05

Fig 20

Elementary CA Rule 18

Dislocation Decay Rate



Periodic Boundary Conditions
100000 cells, Random IC
Time increment = 1

Least-squares fit of data:
Fit to points in [1, 99999]
Slope = -0.492291
Intercept = 14.7305
Coeff. of det. = 0.997485
Chi-squared = 0.00127083