LEGIBILITY NOTICE

A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

LA-UR -88-3927

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405 ENG 36

LA-UR--88-3927

DE89 005447

TITLE STATISTICAL PROPERTIES OF CELLULAR AUTOMATA IN THE CONTEXT OF LEARNING AND RECOGNITION PART II: INVERTING LOCAL STRUCTURE THEORY EQUATIONS TO FIND CELLULAR AUTOMATA WITH SPECIFIED PROPERTIES

AUTHOR(S) Howard A. Gutowitz

SUBMITTED TO Learning and Recognition - A Modern Approach
Sept. 1988, Beijing, CHINA

DISCLAIMER

Los Alamos Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Statistical Properties of Cellular Automata In the Context of Learning and Recognition Part II: Inverting Local Structure Theory Equations To Find Cellular Automata With Specified Properties

Howard A. Gutowitz

Center for Nonlinear Studies and Theoretical Division 13
Los Alamos National Laboratory, MS-B258
Los Alamos, New Mexico 87545

ABSTRACT

This is the second of two lectures. In the first lecture the map from a cellular automaton to a sequence of analytical approximations called the local structure theory was described. In this lecture the inverse map from approximation to the class of cellular automata approximated is constructed. The key matter is formatting the local structure theory equations in terms of block probability estimates weighted by coefficients. The inverse mapping relies on this format. Each possible assignment of values to the coefficients defines a class of automata with related statistical properties. It is suggested that these coefficients serve to smoothly parameterize the space of cellular automata. By varying the values of the parameters a cellular automaton network may be designed so that it has a specified invariant measure. If an invariant measure is considered a "memory" of the network, then this variation of parameters to specify the invariant measure must be considered "learning". It is important to note that in this view learning is not the storage of patterns in a network, but rather the tailoring of the dynamics of a network.

November 18, 1988

Statistical Properties of Cellular Automata In the Context of Learning and Recognition Part II: Inverting Local Structure Theory Equations To Find Cellular Automata With Specified Properties

Howard A. Gutowitz

Center for Nonlinear Studies and Theoretical Division 13
Los Alamos National Laboratory, MS-B258
Los Alamos, New Mexico 87545

Laboratory of Mathematical Physics
Rockefeller University
1230 York Ave, Bx 75
New York, New York 10021-6399

Introduction

In the last talk I gave a rather general introduction to cellular automata. I showed you that, for some cellular automata at least, there is a probability measure which remains invariant as the rule is applied to it. Any other measure tends toward this fixed measure as the rule is iteratively applied to it. I suggested that this unique stable invariant measure might be called the "memory" maintained by the dynamics of the cellular automaton network. I then went on to develop some methods for analyzing the statistical behavior of cellular automata. I showed that a sequence of models, called the local structure theory, can be associated with an automaton. As the order of theory is increased, so does the accuracy with which statistical properties of the rule are predicted by the theory.

Now I want to attack the inverse problem, "Given a system of local structure theory equations, find the set of rules which are approximated by this system of equations." There are several reasons why this is an important problem to consider.

1) When thinking of modelling physical phenomena with cellular automata, it is not clear that there should always be *an* automaton which describes the observed behavior. Rather, it should be that in most cases there is a *class* of automata each of which is an equally valid description. If this is true, one would like to have a way of characterizing the features shared by the automata in this class. These are the features

which are most likely to be the pertinent ones. If all one has is the rule table of a single rule which does something of interest, one is likely to focus on irrelevant details of this rule table while searching to explain how the rule does what it does.

2) Most physical systems, and all biological systems, operate in the presence of significant noise. Both noise in the structure of the system and noise in the inputs to the system. In this situation, it should always be the case that if a single cellular automaton is a good model of a system, then many different cellular automata are good models of the system. The noise will tend to wash out formal, irrelevant differences between rules. Only the most robust features of their construction and action will survive the noise. The local structure theory can be considered as a model of the action of cellular automata in the presence of noise. The higher the order of theory, the less the noise.

3) In the course of constructing the map from local structure theory approximation to classes of cellular automata, the local structure theory equations will be formated so that certain sets of coefficients appear in the equations. To each set of coefficients values corresponds a class of cellular automata. With these coefficients on hand, one can begin to see what happens as the values of the coefficients change. That is, the coefficients can now be treated as parameters and one can ask how the properties of cellular automata change as the parameters are smoothly varied.

This third point has a number of important consequences. The first consequence is that one of the stumbling blocks in front of applying techniques developed for the study of smooth dynamical systems to the study of cellular automata has been removed. That is, in the study of smooth dynamical systems, one likes to have some parameter in the map which can be continuously varied. The changes in the map's properties with change in the parameter value are then examined. A deterministic cellular automaton has no such parameters. The cellular automaton behaves according to the specification of its rule table. If the rule table is changed then one has a different cellular automaton. In the local structure theory, one has a set of parameters whose variation changes properties not of a single rule, but the properties of classes of rules

arranged in a nice way in the space of all automata.

The important consequence of the parameterization of the space of automata for learning is that by varying parameters, classes of rules with specified properties can be found. If one accepts that the "memories" of a cellular automaton are its stable invariant probability measures, then one component of "learning" becomes the process of, given a probability measure, that is, given something to remember, find cellular automata which fix that probability measure. When one says that an organism or a machine "learns" one usually means not only that the organism or machine fixes memories, but that somehow it fixes these memories "by itself", without programming from the outside. What I will be describing here are the knobs that must be twisted to fix a memory. I will not describe the "ghost in the machine" that twists the knobs. I *will* mention one way that the knobs can be twisted "automatically", that is, as part of some explicit optimization scheme. I will make no claim, however, that this is how it is actually done in the brain.

There is a rough correspondence between the "twisting of the knobs" in local structure theory equations to find rules with specified invariant measures and the back-propagation algorithm in standard neural nets. In each case one attempts to find interactions between elements in a network so that the network as a whole behaves in a desired way.

It turns out that in many instances smooth variation in parameters results in smooth variation in the properties of the rules described. This encourages belief that one has found a good and useful way to parameterize the space of rules. Recall from the last talk that the "raw" description of a cellular automaton, that is, the rule table, is not the proper format for making smooth changes in the properties of rules. Rule 22, which forms patterns with a chaotic peppering of triangles, is only one bit different from rule 54, which forms patterns with large patches of periodic background pattern punctuated by chaotic discontinuities. I will try to convince you that to make small changes in rules, one should make small changes in the local structure theory approximations to the rules.

1. LST ORDER -1 : PURE MONTE CARLO

I want you to appreciate that it is very difficult to find a particular cellular automaton with some specified statistical properties if you don't have any idea how the space of cellular automata is structured. First of all, the space of cellular automata is very big. The number of cellular automata with two states per cell goes like $2^{2^{2r+1}}$ where r is the radius of the rule. So there are 2^{32} or about 4 billion radius 2 rules, 2^{128} radius 3 rules etc. So if you want to find a single automaton with some good property, you won't ever find it by picking automata at random and seeing how they behave.

It's nonetheless interesting to pick automata at random and see how they behave. Here is an experiment (figure 1) in which I took 10,000 cellular automata at random from each of the sets of radius 2, radius 3, and radius 4 rules. I then ran these rules on a very long configuration, again generated at random. The density of cells in state 1 was determined by sampling. Iteration of each rule continued until the value of the density seemed to stabilize. This figure shows the distribution over the set of rules of the large-time density determined in this way.

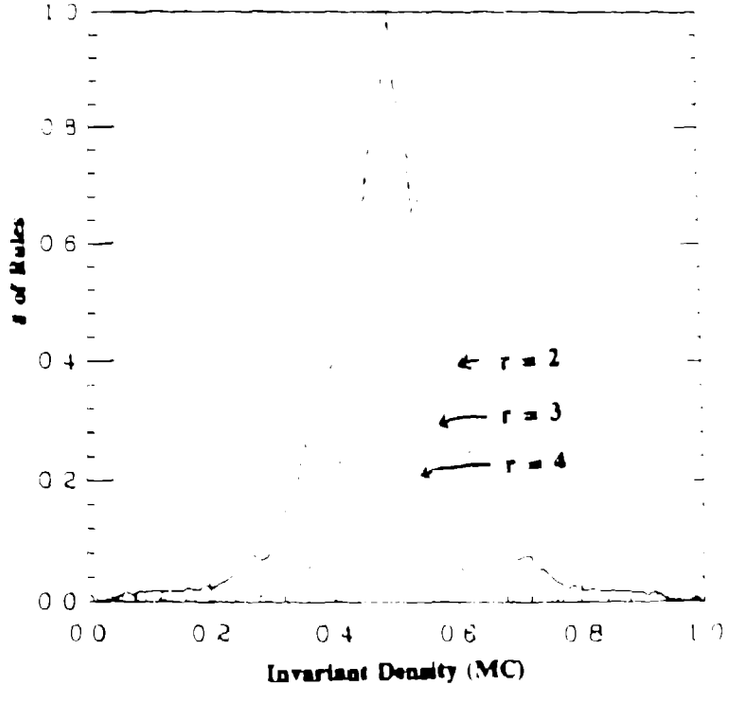

Figure 1 The distribution of invariant densities of 10,000 rules of each radius 2-4.

The widest distribution is for radius 2 rules, and the more narrow distributions for radius 3 and radius 4 rules. It appears that as the radius of rules increases, the expectation that the final density of a randomly-chosen rule will be 1/2 rapidly approaches 1. If one were to look at some other statistical property, say the large-time probability of a 11, one would see the same thing. That is, the large-time probability of a 11 will be the probability of a 11 in a completely random configuration. This is to say that the typical cellular automaton has a very boring invariant measure--it is the measure that gives all blocks of a given length the same probability. If we want to find cellular automata which fix more interesting measures, we are going to have to find a more interesting way to pick rules out of the space of automata.

2. THE ACTION OF A CELLULAR AUTOMATON ON A MEASURE

In the last lecture I explained how the local structure theory worked in a pictorial fashion. To understand how the inverse map is constructed, however, we will definitely need some equations. The most concise equation which describes how a cellular automaton acts on measures is

$$\tau\mu(E) = \mu(\tau^{-1}(E)). \qquad (1)$$

This says the following: Say you have a probability measure μ which describes the probabilities of all sets of configurations at some given time in the evolution of a cellular automaton τ. Now you want to find the measure $\tau\,\mu$ at the next time. The probability of a set E under $\tau\,\mu$ is the measure under μ of the *preimage* of E. The preimage of a set E of configurations is the set of configurations which map to E under the rule. This is denoted $\tau^{-1}(E)$.

Some basic facts of measure theory allow us to expand equation (1) out so that it is both more understandable and more concrete. All sets of configurations can be constructed out of fundamental sets called *cylinder sets* or *blocks*. Blocks are sets of

configurations which share a specified contiguous sequence of cell states at a specified position in the configuration. A block can be specified by giving the states $s_1 s_2 \cdots$ which defined the block, and the starting position for these states. So we would like to write equation (1) in terms of its action on blocks.

Cellular automata are *shift-invariant*. That is, applying a rule to a configuration and then shifting the result (say to the left) is the same as shifting the configuration and then applying the rule. This means that we can confine ourselves to probability measures which are shift-invariant. Under a shift-invariant measure the probability of a block depends only the sequence of cell states which define the block, not on *where* this sequence starts.

With all this in mind, we can rewrite equation (1) as an infinite system of equations of the form

$$P^{t+1}(b) = \sum_{|B| = |b|+2r} \delta(\tau(B),b) P^t(B). \qquad (2)$$

This says that the probability of a block b at time t+1 denoted $P^{t+1}(b)$ is the sum over the probabilities at time t of blocks B whose size is equal to the size of b plus 2 times the radius of the rule and which lead to b under the rule. The delta function serves to pick out just those blocks B which lead to b under the rule. It has the value 1 if $\tau(B)$ is b, and 0 otherwise. $P^t(B)$ is the probability of block B at time t. We can think of the system of equations (2) as being arranged hierarchically. First there are two equations for the probability of a 0 and a 1, then there are 4 equations for the probabilities of 2-blocks 00,01,10,11 and so on.

This system of equations is entirely impractical to use as it stands. Say we want to know what the probability of a 1 is after one application of a cellular automaton of radius 1. To determine this using equation (2) we need to know the probabilities at the previous time of all 3-blocks which lead to a 1. This may not be too bad. But now say we want to continue the iteration for two time steps. For this we need to know initially the probabilities of 5-blocks. In general, to continue the iteration for t time steps, we need initial information about the probability of blocks of length 1+2t. Since the

number of blocks grows exponentially with the length of the blocks, this gets rapidly impractical.

The local structure theory gets around this problem by truncating the system of equations at some finite block size.

3. 0th-ORDER LST

The most radical truncation is called the 0th-order theory. In 0th-order theory one assumes that the probability of a block does not depend on what the block is made of, only on its size. All blocks of the same size are assumed to have the same probability. So each block of length n gets probability $\frac{1}{2^n}$. Substitution of that estimate into the equation for the evolution of the probability of a 1, one has

$$P_1 = \sum_{|B| = 1+2r} \frac{\delta(\tau(B),1)}{2^{1+2r}}. \qquad (3)$$

In this equation the probability of a 1 does not depend on time. All the time dependence was removed when we assumed that all blocks of the same size always have the same probability. Now all the δ function does is count the number of neighborhoods which lead to a 1 under the rule. If we call that number λ, then equation (3) is just:

$$P_1 = \frac{\lambda}{2^d}, \qquad (4)$$

where d is 1 plus twice the radius of the rule, otherwise known as the diameter of the rule. What the 0th-order theory says is that the density (fraction of 1's) of a configuration at any time is just the density of the rule table itself. In particular, the prediction of 0th-order theory for the invariant density of a rule is the density of the rule table.

Every rule yields a particular value of λ. Conversely, to each value of λ is associated many rules. I will say that rules of a given radius are in the same *(0-th order) class* if they yield the same value for λ.

To invert the 0th-order equation, that is, to find all rules in a 0th-order class defined by some value of λ is rather trivial. Given a value for λ, just find all ways of filling a rule table so that exactly λ neighborhoods lead to a 1, and all the other neighborhoods lead to 0. Here (figure 2) are the rule tables for all nearest-neighbor rules in the class $\lambda = 3$.

0	0	1	0	1	0	1	0
0	0	0	1	1	0	1	0
1	0	0	0	0	1	1	0
0	1	0	0	0	1	1	0
0	0	1	0	0	1	1	0
0	0	0	1	0	1	1	0
0	0	0	0	1	1	1	0
1	1	0	0	0	0	0	1
1	0	1	0	0	0	0	1
0	1	1	0	0	0	0	1
1	0	0	1	0	0	0	1
0	1	0	1	0	0	0	1
0	0	1	1	0	0	0	1
1	0	0	0	1	0	0	1
0	1	0	0	1	0	0	1
0	0	1	0	1	0	0	1
0	0	0	1	1	0	0	1
1	0	0	0	0	1	0	1
0	1	0	0	0	1	0	1
0	0	1	0	0	1	0	1
0	0	0	1	0	1	0	1
0	0	0	0	1	1	0	1
1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1
0	0	0	1	0	0	1	1
0	0	0	0	1	0	1	1
0	0	0	0	0	1	1	1

1	1	1	0	0	0	0	0
1	1	0	1	0	0	0	0
1	0	1	1	0	0	0	0
0	1	1	1	0	0	0	0
1	1	0	0	1	0	0	0
1	0	1	0	1	0	0	0
0	1	1	0	1	0	0	0
1	0	0	1	1	0	0	0
0	1	0	1	1	0	0	0
0	0	1	1	1	0	0	0
1	1	0	0	0	1	0	0
1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	0
1	0	0	1	0	1	0	0
0	1	0	1	0	1	0	0
0	0	1	1	0	1	0	0
1	0	0	0	1	1	0	0
0	1	0	0	1	1	0	0
0	0	1	0	1	1	0	0
0	0	0	1	1	1	0	0
1	1	0	0	0	0	1	0
1	0	1	0	0	0	1	0
0	1	1	0	0	0	1	0
1	0	0	1	0	0	1	0
0	1	0	1	0	0	1	0
0	0	1	1	0	0	1	0
1	0	0	0	1	0	1	0
0	1	0	0	1	0	1	0

Figure 2 The r=1 rules in the class λ=3.

There are 56 rules in this class out of the total number of 256 nearest-neighbor rules. The 0th-order theory predicts that the invariant density of each of these rules should be 3/8 = 0.375. Actually, (figure 3), the invariant densities of rules in this class form a distribution whose center is near 0.375, but none of the rules actually have an invariant density of 0.375.

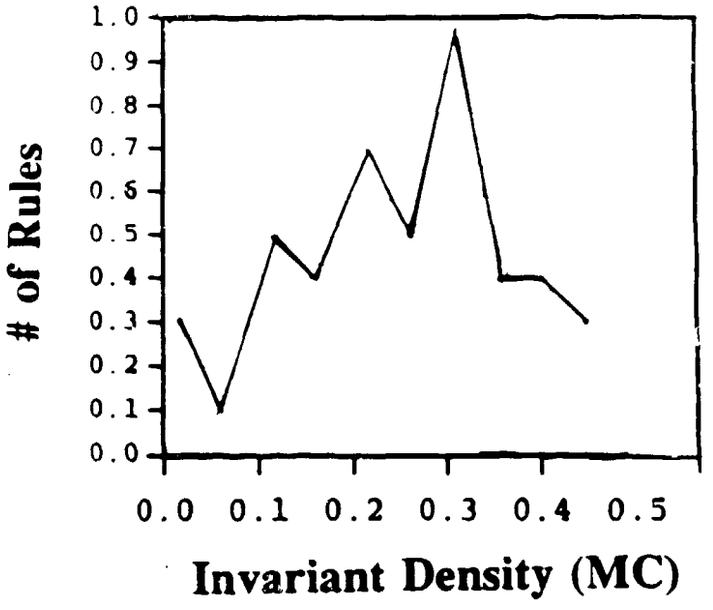

Figure 3 The distribution of invariant densities of r=1 rules in the class $\lambda = 3$.

This distribution is quite broad, and there is no control over the probabilities of longer-length blocks. The 0th-order theory has to be improved, and that is what I want to turn to next.

4. FIRST ORDER THEORY (MEAN FIELD THEORY)

The next order of local structure theory is also known as the mean field theory. The mean field theory, like the 0th-order theory, truncates the infinite system of equations (2) which describe how a cellular automaton acts on a measure. In both cases the system is truncated at the level where neighborhood blocks map to states of single cells. In the 0th-order theory all blocks of the same size were assumed to have the same probability. In the mean field theory the probability of a block is estimated in terms of the probability of the states of cells the block contains. In the mean field theory, the probability of a block B is given by

$$P(B) = P_1^{\#1(B)} P_0^{\#0(B)}. \tag{5}$$

where #0(B) and #1(B) are the number of 0's and 1's respectively in the block B. This equation is exact in the case in which the states of different cells are completely uncorrelated. It is important to observe that two blocks which have the same number of cells in states 0 and 1 will be assigned the same probability.

Substituting this new probability estimate into the equation for the evolution of the probability of a 1, we have

$$P_1^{t+1} = \sum_{(B \mid |B| = d)} \delta(\tau(B),1)(P_1^t)^{\#1(B)} (P_0^t)^{\#0(B)}. \tag{6}$$

Observe that any two blocks which both lead to a 1 under the rule, and have the same number of cells in states 0 and 1 in them will contribute the same amout of probability to the sum. This means that the equation can be rewritten as

$$P_1^{t+1} = \sum_{i=0}^{d} a_i (P_1^t)^i (1 - P_1^t)^{d-i}. \tag{7}$$

where the coefficients a_i count the number of neighborhood blocks which lead to a 1 under a rule and also contain i 1's . This polynomial equation is a model of the evolution of any cellular automaton which yields the coefficient values **a**. A fixed point of the equation, if it has one, is an estimate of the invariant density of any cellular automata which yields the coefficient values **a**.

Observe that many different rules of a given radius may have the same values for the **a** coefficients. Such rules are indistinguishable at the level of mean field theory. So, just as we had before 0th-order classes, we now have mean field theory classes of cellular automata.

It is not difficult to find all rules in a mean field theory class. The best way to explain how to do this is by giving you an example. This example concerns nearest-neighbor rules. There are four coefficients in the mean field theory, which we can label a_0 through a_3. Each of these coefficients "controls" a certain number of neighborhood blocks, as hown here. In this example each of the coefficients is set to one of its allowed values.

coefficient	blocks controlled
$a_0 = 0$	000
$a_1 = 2$	001 010 100
$a_2 = 1$	011 101 110
$a_3 = 0$	111

Consider first a_0, which has the value 0. a_0 controls the block 000. Since it has the value 0, 000 must lead to 0 under any rule in this class. Now consider a_1, which has the value 2. This value can be achieved in several ways, for instance if 001 and 100 lead to 1, and 010 leads to 0 then a_2 will have the value 2. Ignoring the other coefficients for a moment, all possible ways of chosing two blocks to lead to a 1 out of the list of blocks controlled by a_1 will lead to a rule in this mean field class. In the same way there are several ways to achieve the value 1 for a_2, and just one way the achieve the value for a_3. So to find all rules in the class, find all ways of achieving each coefficient value, and then take these in all possible combinations.

This leads to the following set of rule tables:

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

Hence, there are nine rules in this class. Notice that every one of these rules have 3 neighborhoods which lead to a 1. Every one of these rules belong to the 0th-order class defined by $\lambda = 3$. It will always be the case that the rules in the same mean field class also belong to the same 0th-order class.

The mean field equation which defines this class has a fixed point density of 0.38. The average invariant density of rules in this class is 0.42. So in this case the mean field theory does a good job of predicting the behavior of rules in this class. Still, even here, there is room for improvement, and so we turn to 2nd-order theory.

5. SECOND ORDER LST

The mean field theory was derived from the assumption that correlations between the states of different cells are not generated as the cellular automaton operates. Under this assumption, the probability of a large block is estimated as the product of the probability of the states of cells it contains. As we saw in the first talk, the mean field theory fails to accurately model a cellular automaton if correlations are generated as the rule is iterated. I showed you last time that there is a process, called Bayesian extension, by which the correlations represented by the probabilities of blocks of some size can be used to estimate correlations in blocks of a larger size. This lead to a systematic generalization of the mean field theory. Here I will only talk about the first step of this generalization. From there the general case will become clear. In the first step of the generalization, called the 2nd-order theory, correlations are represented in terms of the probabilities of contiguous pairs of cells.

Let $s_i \in \{0,1\}$ be the possible states of a cell in position i in a block. Let $(s_1 s_2 \cdots s_n)$ be an n-block, and $P(s_1 s_2 \cdots s_n)$ be the probability of an n-block. If the probabilities of all 2-blocks are known, the probability of an n-block, $n > 2$, may be estimated by

$$P(s_1 s_2 \cdots s_n) = \frac{\prod_{i=1}^{n-1} P(s_i s_{i+1})}{\prod_{i=2}^{n-1} P(s_i)} \qquad (8)$$

where the 1-block probabilities are found by appropriate summation of the 2-block probabilities.

Blocks which always have the same probability according to equation (8) are said to be of the same 2nd-order type. In the mean field theory the type of a block was determined by how many cells in state 1 it had. 2nd-order types are determined by the number of the various 2-blocks they contain. 2nd-order types can be coded by a triple $(x,y,z)_n$ where x is the total number of 10 and 01 sub-blocks counting overlaps, y is the number of 11 sub-blocks again counting overlaps, and z is the number of cells in

state 1 in the central n-2 region of the n-block. The number of other 1- and 2-blocks in the n-block can be found by appealing to the Kolmogorov consistency conditions. As an example, 10010 and 10100 are both expressed as $(3,0,1)_5$ and are hence of the same 2nd-order type. Here are the 2nd-order types of 3- and 4-blocks. These are used in the 2nd-order theory for radius 1 rules.

3-block 2nd-order types (**b** coefficients)

type	blocks of this type
(0,0,0)	000
(0,2,1)	111
(1,0,0)	001,100
(1,1,1)	011,110
(2,0,0)	101
(2,0,1)	010

4-block 2nd-order types (**c** coefficients)

type	blocks of this type
(0,0,0)	0000
(0,3,2)	1111
(1,0,0)	0001,1000
(1,1,1)	0011,1100
(1,2,2)	0111,1110
(2,0,0)	1001
(2,0,1)	0100,0010
(2,1,1)	1101,1011
(2,1,2)	0110
(3,0,1)	0101,1010

The second order LST preserves the combinatorial information contained in both the cellular automaton map from neighborhood blocks to single cells and the map from (d+1)-length blocks onto 2-blocks. We choose to parameterize the probabilities of 1- and 2- blocks by P_1 and P_{11}. Any other pair of linearly independent 1- and/or 2-block probabilities could also serve as parameters. The other 2-block probabilities can be found from the parameters chosen using the Kolmogorov consistency conditions, e.g. $p_{01} = p_{10} = p_1 - p_{11}$.

The 2nd-order theory is constructed by substitution of the probability estimate given by equation (8) into equations of the form (2) for the evolution of P_1 and P_{11}. Then, as was done in the derivation of the mean field equation (7), the sum is rearranged so that blocks of the same type are collected together. A coefficient $b_{(x,y,z)_d}$ is associated to each type of d-block, and a coefficient $c_{(x,y,z)_{d+1}}$ is associated to each type of (d+1)-block. The **b** coefficients count the number of d-blocks of the given type which lead to a 1 under the cellular automaton, and the **c** coefficients count the number of (d+1)-blocks which lead to 11. Let $P^t((x,y,z)_n)$ be the probability at time t of a block of type $(x,y,z)_n$ according to equation (8). The second order equations are then

$$p_1^{t+1} = \sum_{(x,y,z)_n} b_{(x,y,z)_n} P^t((x,y,z)_d) \qquad (9)$$

$$p_{11}^{t+1} = \sum_{(x,y,z)_n} c_{(x,y,z)_n} P^t((x,y,z)_{d+1}),$$

where sums run over the 2nd-order types of d- and (d+1)- blocks respectively.

As was the case for 0th- and 1st-order theories, many rules may give rise to the same 2nd-order coefficient values. Thus each allowed set of 2nd-order coefficients defines a 2nd-order class of cellular automata.

At second order the construction of a LST class becomes slightly involved. It may be difficult to directly infer a rule table from a specification of theoretical coefficient values because each neighborhood block of length d may be part of several d+1 blocks each controlled by a different c coefficient. This means that the values of the c coefficients may interact in a complicated way to determine which transitions in

the rule table are consistent with a specification of coefficient values. Below a two step process which handles these complications is outlined.

The first step of the construction of a second order class relies on the observation that both the **a** coefficients of the mean field theory and the **b** coefficients of the second order LST for d-diameter rules control blocks of the neighborhood size d. By employing exactly the method described above for the construction of a mean field class, we can find a set of rules with potential membership in a second order class. Such rules have the desired **b** coefficient values, but their **c** coefficients values have yet to be determined.

The second step of the construction determines the **c** coefficient values. The forward map from a rule table to a set of LST coefficients is easily computed. In the second step of construction, we use the forward map to determine the **c** coefficient values of all cellular automata isolated in the first step, and then check these values against the **c** coefficient values which define the class in question.

The 2nd-order theory almost completely splits the first order class I showed earlier into individual cellular automata. The estimates for the invariant densities for the rules in this class as compared with the 0th- and 1st-order estimates are shown here (figure 4). Clearly, for the most part, increase in order of theory produces better estimates of the statistical properties of these rules.

Num	0th	1st	2nd	MC
84	0.375	0.382	0.500	0.500
52	0.375	0.382	0.010	0.281
28	0.375	0.382	0.500	0.500
82	0.375	0.382	0.383	0.386
50	0.375	0.382	0.500	0.500
26	0.375	0.382	0.383	0.386
70	0.375	0.382	0.500	0.500
38	0.375	0.382	0.010	0.281
14	0.375	0.382	0.394	0.500
Ave	0.375	0.382	0.350	0.42

Figure 4 The mean field class defined by a= (0,2,1,0). Estimates of the invariant densities of these rules by Monte Carlo and LST orders 0-2.

6. CLASSIFICATION of r=2 RULES

The point of all of this discussion, that the local structure theory supplies a good way to explore the space of cellular automata, is somewhat lost if one only considers radius 1 rules. There are only 256 such rules, less if one takes into account trivial automorphisms between rules, so all of these rules can be examined on a case by case basis. Life becomes more interesting, however, when one goes on to consider radius 2 rules. As I said earlier, there are about 4 billion radius 2 rules. So it is impossible to look at them all individually. In this situation, the ability to examine rules in terms of classes becomes a very powerful tool.

A lot of work has been done on the classification of radius 2 rules. Here I just want to talk about some of the highlights.

The two most important questions to ask about the classification concern its accuracy and its homogeneity. The accuracy question is, "how well are the properties of rules in a class predicted by the local structure theory equations which define the

class?" The homogeneity question is, "how similar to each other are the rules in a class?" These questions are important for applications, in particular, the application to learning.

Imagine that we want a network to "remember" some probability measure. This memory will be stored in the dynamics of a cellular automaton acting on itself. Learning involves the selection of the "right" cellular automaton to store the probability measure in question. I want to suggest that the way to select such an automaton is to 1) vary the coefficients in some order of local structure approximation until values are found such that the equation fixes the desired measure, then 2) use the inverse map to find an automaton, or a set of automata, which are in the class described by that set of coefficients. This automaton will do the job only if the local structure theory equation accurately describes the behavior of the rules in its class. That is, it should be that the performance of a rule in a class should match fairly well with the performance of the equations which describe the class. It is also desirable for all the rules in the class to be similar to each other. It could be, for instance, that a particular class has millions of elements, and the equations which define the class could accurately describe the *average* over the whole class of some property, but any individual rule in the class could be very different in behavior from that average behavior. This would make the program for learning that I just outlined unworkable.

Lets first take up the question of accuracy. To discuss how far one measure is from another, we need a notion of distance in the space of measures. The distance I will use is

$$d(\mu,\nu) = \frac{1}{2} \sum_{(B\,|\,|B|=2)} |\mu(B)-\nu(B)|. \qquad (10)$$

That is, given two measures, μ and ν, the distance between them is one-half of the sum over 2-blocks of the absolute value of the difference of the probability of the 2-blocks under the two measures. The maximum distance between any two measures is 1. The way this will be used is this: Many rules will be selected out of an LST class. The invariant 2-block probabilities of all of these rules will be determined by applying them many times to a random initial configuration. The average over all these rules of

the invariant 2-block probabilities will be found. This average I will call the empirical invariant measure of the class. On the other hand, the theoretical invariant measure can be found by solving for the fixed point of the equations which define the class. Finally, the distance between the empirical and theoretical invariant measures can be found using the metric I just defined.

As you might expect, for some classes there will be a small distance between the empirical and theoretical invariant measures and for other classes this distance could be quite large. In general, there will be some distribution over classes of this distance. Here (figure 5) is what this distribution looks like for classes of radius 2 rules defined at orders 0 through 2.

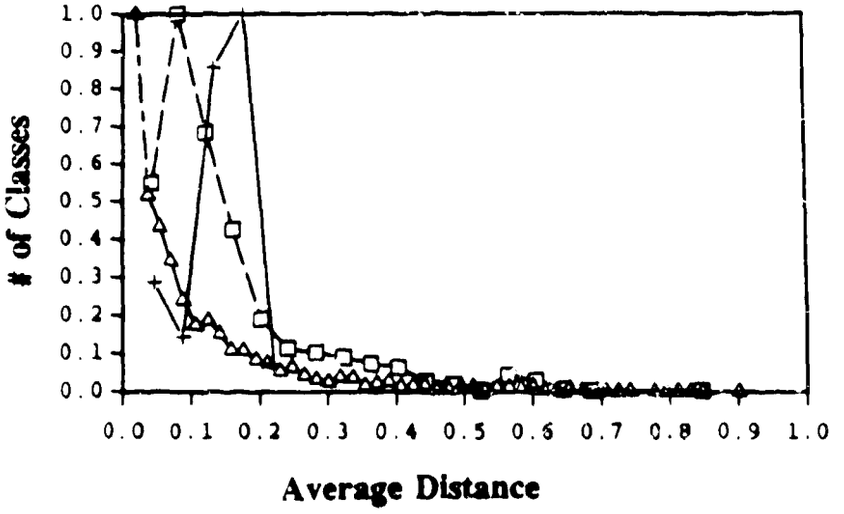

Figure 5 The L_1 distance between theoretical and empirical estimates of the invariant measures. These distributions are over classes of r=2 rules defined at orders 0-2. +) 0th-order, □) 1st-order, and Δ) 2nd-order.

These curves were gotten by selecting thousands of LST classes, then selecting tens to hundreds of rules out of each class, empirically determining the invariant measures of these rules, and finding the distance from the empirical measures, averaged over a class, to the fixed-point measure of the equations which define the class.

As you can see, as the order of theory increases, the typical distance between the empirical and theoretical invariant measures decreases. Even the 0th-order theory is

fairly accurate, on average. Recall that the maximum distance between measures is 1. The peak of the 0th-order curve is much less than that, approximately 0.17, while the peak of the 2nd-order curve is at 0.05. Presumably, if the order of theory were increased still further, the typical accuracy would continue to improve.

Now let me take up the question of the homogeneity of LST classes. The homogeneity of a class is assessed by determining how much some property of rules varies over the class. Consider measuring the invariant density of all the rules in a class. These invariant densities will form some sort of distribution, and the standard deviation of this distribution is a well-defined object, which we can call the class standard deviation of the invariant density. Again, some classes will have a small standard deviation of the density, and others will have a large one. Again, there is a distribution over *classes* of the class standard deviation of the density, and again the distribution will depend on the order of theory considered. The results here are more striking than the results concerning accuracy of LST predictions (figure 6).

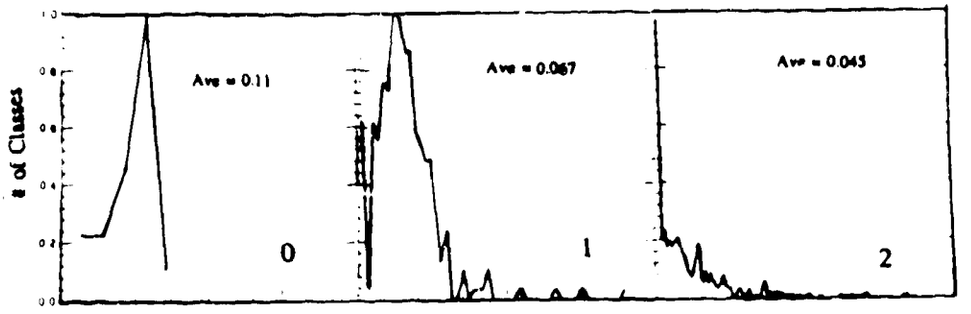

Figure 6 The class standard deviation of the invariant density. Distribution over classes or r=2 rules defined at orders 0-2.

The typical size of the class standard deviation of the density drops dramatically as the

order of theory is increased. By order 2, the properties of rules in a class are so tightly controlled that rather careful empirical work is needed to see the variation over a class in statistical properties of rules.

In summary, it seems that we are in the best of all possible worlds! Even at very low order, the LST serves to break the space of cellular automata into classes of rules with similar properties. These properties may be accurately determined by solving for the fixed-points of a system of equations. This means that we can now begin to explore the space of cellular automata, not by picking rules at random and simulating them, but by varying the coefficients in small systems of equations. The difference in the amount of computer time required by these two methods is enormous. The cost of picking rules at random and running them to see how they perform is so great that it is clear that learning, or anything else for that matter, is not done in that way, at least not on a routine basis.

7. VARIATION OF LST COEFFICIENTS

So far we've been mostly thinking of the coefficients in the LST equations as having some fixed set of values, and then asking how well fixing the coefficient values serves to fix the properties of the rules in the class defined by the equations. Having determined that the coupling between LST equations and the rules in the classes defined is fairly tight, we can forget about the rules themselves for a moment, and just consider the behavior of the equations. Now, at a given order of theory, we have a family of equations parameterized by the coefficients. We can ask what happens to the properties of these equations as the parameter values are smoothly varied. For most applications that one can think of, it would be best that small changes in parameter values should result in small changes in the properties of the equations. I'm going to close this talk by showing you some results which suggest that this is in fact the case. Here (figure 7) you see what happens when two parameters in the mean field theory for radius 2 rules are varied.

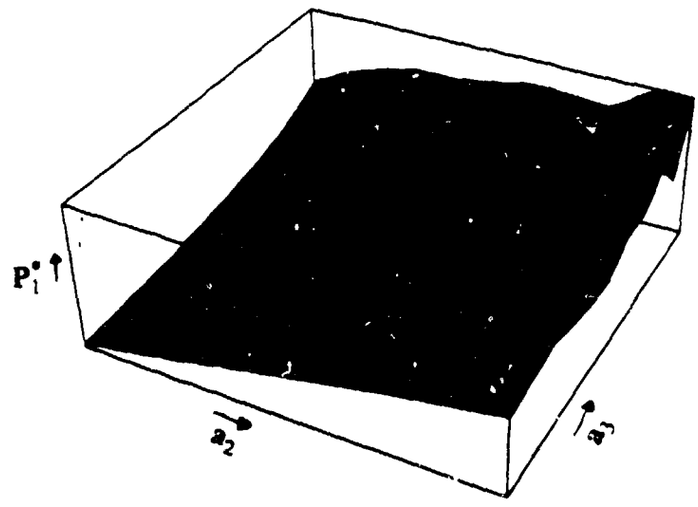

Figure 7 Variation of two parameters in the mean field theory for r=2 rules (a_2 and a_3) vs. the fixed-point density of these equations.

The values of the parameters are shown as x and y, and the fixed point density of the equations determined by x and y (with the other parameters held fixed) is plotted in the z direction. As either x or y or both are increased, the fixed-point density of the equations also increases, smoothly. This behavior seems rather typical.

Now let's put together the pieces. Given a system of local structure theory equations, all the rules which are approximated by these equations can be found. The properties of these rules match well with the properties of the equations which describe them. As the parameters in the equations are smoothly varied, the properties of the equations vary smoothly. This implies that by smoothly varying parameters in some system of equations we can smoothly move around in the space of cellular automata. It is now easy to imagine that given a measure which we want to be invariant under a cellular automaton, we can find such an automaton by varying parameters in LST equations until we have the desired result. This variation could be controlled by any sort of optimization scheme which is set up to minimize the distance between the

measure we wish held fixed and the measure fixed by the system of equations at each setting of the parameter values. In this interpretation learning is not the storage of patterns in a network, but rather the tailoring of the dynamics of a network.