LA-UR 89-3053

CONF-8905201--3

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--89-3053

DE90 000600

TITLE EMERGENT BEHAVIORS OF CLASSIFIER SYSTEMS

AUTHOR(S) STEPHANIE FORREST
JOHN H. MILLER

SUBMITTED TO PROCEEDINGS OF THE " EMERGENT COMPUTATION..., "
CONFERENCE, HELD IN LOS ALAMOS, NM, MAY 22-26, 1989.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

By acceptance of this article the publisher recognizes that the U S Government retains a nonexclusive royalty-free license to publish or reproduce the published form of this contribution or to allow others to do so for U S Government purposes

The Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U S Department of Energy

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

FORM NO 836 R4
ST NO 2629 5/81

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Emergent Behaviors of Classifier Systems

Stephanie Forrest
Center for Nonlinear Studies, MS-B258
Los Alamos National Laboratory
Los Alamos, NM 87545
sxf@lanl.gov

John H. Miller
Santa Fe Institute and Carnegie-Mellon University
SFI, 1120 Canyon Road
Santa Fe, NM 87501
miller@sfi.santafe.edu

August 30, 1989

ABSTRACT GOES HERE

1 Introduction

Classifier systems are computational systems that model cognitive behavior. Like connectionist networks, classifier systems consist of a parallel machine (most often implemented in software) and various learning algorithms that adjust the configuration of the underlying machine over time. Classifier systems differ from connectionist networks in the details of both the parallel machine and the learning algorithms. Specifically, the classifier system machine computes with patterns called messages instead of real-valued weights, and it controls its state with IF/THEN rules that specify patterns of messages. Classifier systems also incorporate a mechanism called the genetic algorithm [11] that synthesizes connectivity patterns (from inputs through intermediate states to outputs) from initially random configurations.

In the context of classifier systems, emergent computation arises when co-adapted sets of rules evolve and function as one entity, either in the case of an entire set of classifiers interacting with an external environment, or within one classifier set when groups of rules interact among themselves. A set of classifier rules forms an ecology, in which each individual rule evolves in the context of both the external environment and the other rules in the classifier system. Competition forces individuals into uncrowded and productive niches, and over time, the various rules in the system learn to act cooperatively.

Classifier systems are interesting for several reasons. They provide a computational theory of cognition based on learning, intermittent feedback from the environment, and hierarchies of internal models that represent the environment [12]. As a theory of cognitive activity, classifier systems can be used to model other "intelligent" processes, such as how people behave in economic and social situations (trading goods in a simple market, playing the stock market, obeying social norms, *etc.*). The classifier system model also provides an example of a programming language in which correct programs can be either learned or programmed [2]. Any particular configuration of a classifier system can be viewed as a program in which the individual rules (called classifiers) correspond instructions. Previous attempts to apply machine learning techniques to the problem of generating or debugging computer programs have been largely unsuccessful, due to the brittle nature of most programming languages, in which a program's behavior can be changed dramatically by one misplaced character. In a classifier system, however, the restricted syntax of each instruction allows almost any combination of instructions to form a legal program. Additionally, the relative position of a single instruction does not determine its effect on the program. These two properties of classifier systems support the notion of a program as an "ecology" of individual instructions, each instruction filling some useful niche in the overall program and evolving in the context of the other instructions.

Classifier systems have various global properties that arise from the interactions among their components, both interactions among individual rules (as mentioned above) and interactions at a coarser level among the learning algorithms (bucket brigade and genetic algorithm), the model of computation, and the interface to the environment. These properties are poorly understood even though they have a profound effect on the overall performance of the system. In the absence of analytical results that characterize the aggregate behavior of classifier systems, most of the current understanding of classifier system behavior has been obtained through trial and error, or through careful experimentation on small sets of classifiers in artificial environments that can be controlled [16]. Recently, however, there have been several attempts to study the emergent properties of classifier systems using techniques from nonlinear dynamical systems [18,1,19,15].

This paper discusses some examples of emergent behavior in classifier systems, describes some recently developed methods for studying them based on dynamical systems theory, and presents some initial results produced by the methodology. The goal of this work is to find techniques for noticing when interesting emergent behaviors of classifier systems emerge, to study how such behaviors might emerge over time, and make suggestions for designing classifier systems that exhibit preferred behaviors.

2 Emergent Behaviors in Classifier Systems

Classifier systems have various emergent global properties that determine their behavior and contribute to the overall computational properties of the system. This section discusses three different aspects of emergent behavior and how they relate to emergent computation: (1) emergent symbolic reasoning systems, (2) the role of learning, and (3) global dynamical behaviors.

2.1 Symbolic Reasoning

When co-adapted sets of rules act together as a unit, that unit may be viewed as a higher-level structure. Of central importance is understanding how high-level structures can emerge in learning classifier systems and come to have a selective advantage over direct input/output maps. For example, if a group of rules were to evolve that implemented the system's understanding of the concept "cooperation" and that group functioned as a unit with respect to the rest of the system, we would say that the concept "cooperation" was operating at a higher level than that of the system description (individual rules). If other high-level concepts evolved and the system were able to use these concepts in its reasoning process, the classifier system would be exhibiting emergent computation. This form of emergent computation in classifier systems can be contrasted with direct 1-1 mappings from inputs to outputs. We refer to a direct mapping which does not use any intermediate reasoning steps or representations as a stimulus/response system.

The question of how high-level concepts can come to have a life of their own in a classifier system is an example of the more general problem of how symbolic computations can arise in a low-level learning system that is closely tied to its input/output interface [2]. By symbolic computation we mean both the formation of representations that are somewhat removed (abstracted) from the sensory interface to the system, which we call symbols, and procedures for manipulating symbols. High-level symbolic representations have several advantages over lower-level direct mappings. They can express sophisticated reasoning strategies, for example, allocating appropriate amounts of processing time to various aspects of a problem. Further, such representations are comprehensible to an outside observer trying to understand what the system has learned or to an external agent trying to communicate with the system. More importantly, in any realistic problem space, the number of explicit input/output mappings will exceed the capacity of the system, so higher-level abstractions of the input/output map are advantageous for reasons of efficiency. Thus, a classifier system exhibiting emergent computation in the form of high-level symbolic knowledge is desirable. However, there must be some competitive advantage or utility for these higher-level constructs if they are to be maintained by learning. Additionally, the concepts need to evolve in such a way that they will be stable under variable inputs from the environment and stochastic learning algorithms.

A crucial aspect of emergent concepts is that their representation is "distributed" over many different classifiers. In a distributed representation, any one concept is comprised of several classifiers, and each individual classifier may be part of many different concepts simultaneously. For example, in a default hierarchy of concepts built from classifiers the complete meaning of a concept would be distributed across all of the higher-level classifiers in the hierarchy. Likewise, any one classifier in the hierarchy could be participating in many different hierarchies simultaneously. While there are some theories of distributed and associative memories [13,20,10], distributed representations are difficult to recognize, especially in classifier systems.

Thus, an important question is how distributed, high-level representations can emerge in learning classifier systems and come to have a selective advantage over direct input/output maps. Although classifier systems were designed with this goal in mind and there are many proposals for how classifier systems can exhibit this kind of behavior, we don't know of any learning classifier systems that demonstrably exhibit high-level symbolic manipulations in any but the most artificial situations. Riolo [16] has studied this question in the most detail using an environment built from finite Markov processes in which some states are assigned non-zero payoffs.

2.2 Learning

Learning systems are important from the perspective of emergent computation for several reasons. Firstly, most of our current examples of emergent computation, including classifier systems, rely on some sort of adaptive mechanism for their control Because it is generally quite difficult to predict the exact behavior of systems that exhibit emergent computation, it will likely be quite difficult to program in desired behaviors directly, so adaptive control is a natural approach. Secondly, learning systems themselves have interesting emergent properties (*e.g.*, fixed points, attractors, state cycles, *etc.*) that are poorly understood. Nonetheless, these emergent properties affect the learning system's behavior. Because learning systems can be complex and *ad hoc*, they typically have many different components that can interact with one another in unpredictable ways. An extreme example is provided by classifier systems in which there are at least three major components, each of which is itself quite complex.

A classifier system consists of three layers, with the rule and message passing system forming the lowest level. The rule system is the fundamental computational component of the system; the remaining layers are algorithms for modifying its structures. At the second level is the bucket brigade learning algorithm which manages credit assignment among competing classifiers. It plays a role similar to that of back-propagation in neural networks. Finally, at the highest level are genetic operators that create new classifiers. For a detailed description of classifier systems and genetic algorithms the reader is referred to [11,8,12].

Associated with each classifier is a parameter called its strength. This measure reflects the utility of that rule, based on the system's past experience with its use. The bucket brigade algorithm is the mechanism for altering each rule's strength. The algorithm is based on the metaphor of an economy, with the environment acting both as the producer of raw materials and the ultimate consumer of finished goods, and each classifier acting as a middleman in an economic chain of production. It can be shown that the strength of each classifier will stabilize at a level that reflects its "profit": how successful it is at producing messages that are useful to other classifiers (that are in turn useful in making the classifier system get rewards from the environment).

Using the bucket brigade, a classifier system is able to identify and use the subset of its rule-base that has proven useful in the past. However, a classifier system's initial rule-base usually will not contain the appropriate classifiers. The third layer, the genetic algorithm [11], is used to discover new rules. The genetic algorithm relies on the metaphor of population genetics. In this model each classifier acts as an individual, with its condition and action specifications acting as its genetic code, competing to fill an ecological niche. After the performance system and bucket brigade have operated long enough for each classifier's strength to stabilize at an accurate measure of its utility, the genetic algorithm uses this same parameter as a measure of the individual's "fitness." The genetic algorithm produces a new population of classifiers through simulated sexual reproduction, using the more fit existing classifiers as parents.

Both the genetic algorithm and the bucket brigade have interesting emergent properties. The genetic algorithm raises questions about how rule ecologies develop, how stable they are, and the effects of competition and randomness on the rule base. Bucket brigades raise questions about how the strengths evolve through time: do they converge on a stable set of strengths, oscillate, or vary indeterminately? Because these systems are nonlinear the standard analysis tools from theoretical computer science are difficult to apply to these questions. (see Introduction to these proceedings).

It is the interactions between the environment and a learning system that are at the heart of learning. This makes learning systems difficult to analyze because both the learning system and its external environment (reward function, test cases, *etc.*) must be taken into account. In many cases the environment is so complex that no reasonable analytical solutions exist. In fact, problem domains that are not understood analytically are often considered to be good candidates for learning, since if we knew enough about the environment we could simply program a solution to it. A common approach is to study the behavior of a learning system in the context of a "toy" problem that is well understood, and then make predictions about how the behavior will scale as the size and complexity of the problem is increased. These "scaling" arguments are problematic, especially because interactions between a learning system and its environment often result in nonlinear behaviors, making it difficult to predict performance.

2.3 Emergent Dynamical Properties of Classifier Systems

Any system which changes in time is a dynamical system. Thus, any classifier system can be viewed as a dynamical system and studied from that perspective. In this subsection, we discuss why we believe that the emergent dynamical properties of classifier systems are interesting and relevant to the question of emergent computation.

The computational properties of the individual components of classifier systems are quite well understood. In addition to outlining the classifier system architecture [12], Holland [11] described both the mechanics and underlying theory of genetic algorithms. Forrest [7] proved one form of computational completeness for the performance element, and in related work demonstrated that the classifier system architecture is suitable for problems requiring deep reasoning and sophisticated data structures. Riolo [16] investigated the ability of the bucket brigade to maintain hierarchies and sequences of rules, and several other researchers have recently explored the mathematics of bucket brigades.

None of this work adequately accounts for the aggregate behavior of a classifier system that combines the production rule system, the bucket brigade, and a genetic algorithm. Empirical results indicate that classifier systems are significantly more complex (and difficult to get working correctly) than any of the individual components. It has proved difficult to design learning classifier systems with the ability to follow long chains of reasoning (long sequences of computation), and there is little solid understanding of why some classifier system designs are successful and others are not.

Some of the basic concepts from nonlinear dynamical systems theory provide a natural interpretation of the aggregate behaviors of classifier systems. By viewing a classifier system as a dynamical system, different regimes of behavior (periodic, chaotic, *etc.*) can be associated with various aspects of classifier system behavior. For example, the concept of basins of attraction and state cycles can be used to discuss how robust a classifier system is to noisy data from the environment or random perturbations of its rule base. A robust system would stay in the same basin of attraction under small perturbations and similar initial conditions. Likewise, a responsive system is one in which significantly different inputs will cause it to converge on different attractors. A second example is provided by classifier systems that suddenly change their behavior as a result of learning, adding more rules, or changing various parameters. These systems can be said to have gone through a phase transition. The nonlinear dynamics perspective suggests that there are likely to be narrow ranges in which classifier system performance is particularly good or bad. A third area in which the dynamical systems perspective is useful is in the formation of chains of classifiers (see Section 3.4 for details).

Thus, the following kinds of questions, which are highly relevant to understanding classifier systems, have natural interpretations as emergent dynamical behaviors:

- How many classifiers are required in the system before interesting behavior can occur?
- Under what conditions will chains of classifiers form?
- How dense will these chains be?
- What is the effect of various classifier parameter settings on these properties?
- What is the impact of learning and different representations on aggregate behavior?
- How stable are these systems to random perturbations? (such as those introduced by learning algorithms or by noisy data from the environment)

3 Classifier Systems as Dynamical Systems

Dynamical systems theory provides a number of tools for studying complex phenomena that change in time, such as learning systems. There are several problems, however, with trying to apply the standard techniques directly.

First, most dynamical systems analysis considers only asymptotic behavior. Thus, a standard approach is to run the system under study for a long time before observing its behavior. After a sufficiently long time, most systems settle into some sort of steady state behavior, possibly converging on one fixed point, oscillating among a small set of states, or wandering around a strange attractor. Analysis of complex systems focuses on this steady state behavior that arises "in the limit" and ignores the transient behavior of the system as it approaches the steady state. Any learning system that must interact with a dynamic environment is highly unlikely to reach a meaningful asymptote in any reasonably small dimensional space. Further, even if it did we would be at least as interested in understanding how it approached asymptotic behavior as in understanding what particular asymptote it reached.

A second problem with applying dynamical systems techniques to learning systems is that most of the mathematics has been designed to describe continuous physical systems. Most learning systems have at least some aspects that are discrete. Farmer [6] has pointed out that dynamical systems may be discrete or continuous along several dimensions, including time, state spaces, and internal variable values. If we are to apply the techniques of dynamical systems theory to the analysis of learning systems then it is crucial that this distinction between discrete and continuous models be accounted for.

In the case of classifier systems, most of the relevant dimensions are discrete. The system runs synchronously and time is discrete. The messages on the message list are discrete and classifiers are either activated or not activated with no intermediate states. Continuous values do appear, however, both for the strengths associated with classifiers that are adjusted by the bucket brigade and the intensities associated with messages[1]

In finite systems, some of the standard definitions change, since there can be no true notion of infinity in a finite system. For example, chaos cannot exist in a finite system since all finite systems repeat their states eventually. However, if the length of the shortest cycle is roughly equivalent to the number of possible states, and if the number of possible states is large enough, the system may be viewed as effectively chaotic.

A number of different researchers are currently investigating classifier sys-

---

[1] In some classifier systems a continuous quantity called "intensity" is associated with each message. A high intensity message increases the corresponding classifier's bid.

tems from the dynamical systems perspective. Here, we discuss four different approaches.

3.1 Bucket Brigades

As mentioned in Section 2.2, the bucket brigade addresses credit allocation problem [17]. A classifier system has a set of rules each of which can be fired by the presence of appropriate messages posted on a centralised message list. Once the appropriate messages are on the message list, the probability that a rule will be fired is proportional to a measure of the rule's own strength.

A rule's strength is determined by the impact of the message that the rule posts. If the rule's message is posted when external payoffs are received from the environment then strength is increased. Furthermore, if the message is instrumental in allowing other rules to fire, then its strength will be increased over time by an amount proportional to the firing rule's strength. Reward propagates back through the chain of contributing classifiers indirectly. on every iteration of the system, each classifier$_i$ pays out a portion of its strength to the classifier$_j$ immediately "upstream" of it, that is, the classifier whose output message made it possible for classifier$_i$ to become active. Over time, classifiers that help "set up" other classifiers to be rewarded will have their strength increased through this indirect payoff method. Since the probability that any rule will fire is closely tied to its strength (as well as the set of currently posted messages matching the rule's conditions), an understanding of the dynamics of strength allocation provides insight to the set of rules that will emerge over time.

Arthur (1989) and Simon (1989) have both begun to analyse the dynamics of strengths by mapping variations of the standard credit allocation algorithms for simple classifier systems into an equivalent set of stochastic differential equations. They have focused on N armed bandit problems, studying what kinds of strategies evolve under different algorithms. This work has shown that in these classifier system the choice of an allocation algorithm can have a large impact on the set of emergent strengths. Moreover, design criteria can be derived which can tune the dynamics so that different properties can be emphasized. Specifically, they have found that one version of the standard classifier system bidding algorithm leads to a probability matching strategy in which the classifier system selects the various arms with probabilities that match their payoffs, while a bidding mechanism in which individual bids are not subtracted from a classifier's strength leads to an optimizing strategy in which the classifier system learns to always select the arm with the highest payoff. These criteria clearly show the trade off between maintaining high levels of adaptability verses the ability to exploit the current environment and are similar to results obtained by Dave Goldberg [9].

3.2 Karnaugh Maps

Another area for analysis is how recombination of old rules affects the ecosystem of rules. Smith and Valenzuela-Rendón [19] have looked at the dynamics of rules from an ecological perspective. Through the use of a simplified environment and classifier system (2-bit, single-condition classifiers), they have explicitly modeled the dynamics of the recombination process. Their analysis indicates that the coevolution of rule sets may follow complicated paths, and may be modified by the introduction of various operators that encourage speciation and the formation of niches. Rule sets may have long meta-stable states, which eventually converge on small sets of inadequate rules dominating the population. NO STRENGTHS IN THEIR CLASSIFIERS?

3.3 Asymptotic Dynamics

Compiani *et al* [3] have used dynamical systems techniques to study the behavior of classifier systems on a letter prediction task. They view the genetic algorithm and bucket brigade as operating on two completely different time scales, so they can treat them independently. They are concerned with questions such as the stability of a learned solution under the operations of the genetic algorithm, and the influence of various system parameters on performance. They have found that the two major influences on classifier system performance are the size of the message list and the maximum number of copies that are allowed of any one classifier. For more details, please see CompianiEtAl90]] in this volume.

3.4 Boolean Network Models of Classifier Systems

One dynamical system that is well suited for studying classifier systems is Boolean networks. They are discrete along the same dimensions as classifier systems, although a slight modification to the conventional model is required to accomodate strengths. The connection between structural and dynamical properties of random Boolean networks has been studied extensively [14,4,5], and general techniques have been developed for determining the dynamical properties of specific networks. These studies show that the dominant dynamical behaviors of Boolean networks can be characterized by a small set of emergent properties.

Briefly, a Boolean network consists of a set of nodes, each of which has two possible states, 0 or 1. The state of each node at time (t + 1) is determined by its own Boolean function which takes as input the states of other nodes in the network (indicated by a directed arc between the nodes) at time t. The variables of a node's Boolean function correspond to the states of the connected nodes. The Boolean functions can vary for different nodes, as can the number and location of the input nodes. See [refs] for details. Maybe a sentence or two here about how they are hooked up randomly, iterated, and interesting properties noted.

Miller and Forrest (1989) have taken a different approach to understanding the emergent properties of classifier system by focusing on the dynamics of the message passing system. By watching the patterns of messages that appear on the message list, all of the different components of classifier system behavior are combined together. The message list reflects all of the interacting components in that the rule set determines which messages could be posted, the bucket brigade and bidding mechanism control the strengths (and hence the probability that an activated rule will actually be allowed to post its message), and the genetic algorithm controls which rules are in the current rule set.

By defining a mapping between classifier systems and Boolean networks [15], classifier system behavior can be modeled by the Boolean network, and the techniques that have already been developed for studying Boolean networks can be applied to classifier systems. If messages are considered to be nodes in a network, then the set of classifier rules can be mapped onto a set of Boolean functions activating each of these nodes based on the state of other nodes in the system (the mapping takes the node associated with the given message(s) a rule may post, and connects it to those nodes that can actually fire the rule with the appropriate Boolean function). The mapping between classifier systems and Boolean networks preserves the functional behavior of classifier systems. For each possible state of the message list at time t (corresponding to a set of nodes in the Boolean network that are in State 1), the message list produced by the classifier system at time t+1 will be equivalent to the set of active boolean net nodes (those in State 1) at time t+1. By mapping a particular classifier system to its implied network, one can easily follow the dynamics of the message system. Moreover, by considering generic classes of classifier systems a link between such systems and specific classes of randomly connected Boolean networks can be formed. Randomly connected Boolean networks exhibit a large array of emergent properties. Thus, the above mapping provides insight about those properties which classifier systems must either exhibit or actively surmount.

The types of properties which can be explained by the Boolean network mapping of classifier system include the number and sizes of internally connected rules, the distribution of messages on the message list, the potential for message cycles and freezing[2], the impact of noise on the classifier system, the effect of adding new rules to the system, *etc.* These properties can be linked to different design choices of the classifier system (e.g., the number of rules in the system), strength assignment rules, and learning mechanisms.

The behavior of Boolean nets has been shown to be dominated by a small set of emergent properties, such as state cycles (size and number), frozen components, and stability to perturbation. Further, these emergent properties have been shown to depend directly on various structural properties of the Boolean network, such as the number of input arcs on each node and form of the Boolean function stored at each node. These properties have important implications for classifier systems.

An example of the relevance of these dynamic properties to classifier systems is the concept of self-sustaining activity. The classifier system's architecture derives much of its power through the formation of chains of rules. Such chains support internal reasoning processes that allow a classifier system to go beyond simple stimulus/response behaviors. In particular, cyclic chains (loops) are one of the most powerful computational constructs. State and sub-state cycles that emerge in Boolean networks are closely related to chains in a classifier system. It is known that the emergence of state cycles in Boolean networks depends critically on the underlying structure of the net-

---

[2] Boolean networks are subject to a condition called "freezing" in which a region of the network becomes locked into one state (either 1 or 0) and is impervious to fluctuating states in the rest of the network).

work. Based on this connection, we can predict that the likelihood of chain formation in classifier systems is closely tied to the classifier system's configuration (number of classifiers, percentage of don't cares in the population, *etc.*). Critical values probably exist that catalyze the formation and survival of chains. Earlier results for Boolean networks suggest that the number of cyclic chains that form for typical classifier system configurations is relatively small and that such chains have few members.

Cyclic chains allow classifier systems to exhibit self-sustaining activity, that is, to generate internal activity in the absence of sustained external input. Some level of internal activity is required if a classifier system is to form *large internal representations and operate with intermittent* environmental input. Without any internal activity, these important characteristics of classifier systems would be lost. Too much activity, however, would be likely to hurt performance. We conjecture that "interesting" classifier system behavior occurs at or near the boundary between these two extremes.

One way to test the property of self-sustaining activity is to intialize all of the nodes of the Boolean network to State 1 and iterate the network until it reaches a state cycle. Of course, one potential state cycle is the one where all message activity dies out. In this case, the length of the state cycle is 0. Figure 1 shows the average length of self-sustaining cycles (that is, the number of different nodes in the self-sustaining cycle) for classifier systems with two different percentages of #s.[3] The third line shows the theoretical maximum since it plots the total number of nodes in the network for variously-sized classifier systems. In some configurations, small changes in the number of classifiers causes a rapid transition from no self sustaining activity to a large amount.

An important question is how the structural properties of networks differ for networks corresponding to randomly generated classifier systems and for classifier systems that have evolved under learning. We have obtained several pairs of classifier sets (before and after learning) and are in the process of comparing their structural properties. We compared internal connectivity, self sustaining activity, and how long activity takes to die out (a weaker measure) for classifier sets before and after learning. Riolo kindly provided us

---

[3] These results are for 8 bit classifier systems, and the number of classifiers needed to generate self sustaining activity is substantially lower than what we expect for the 16 bit case.

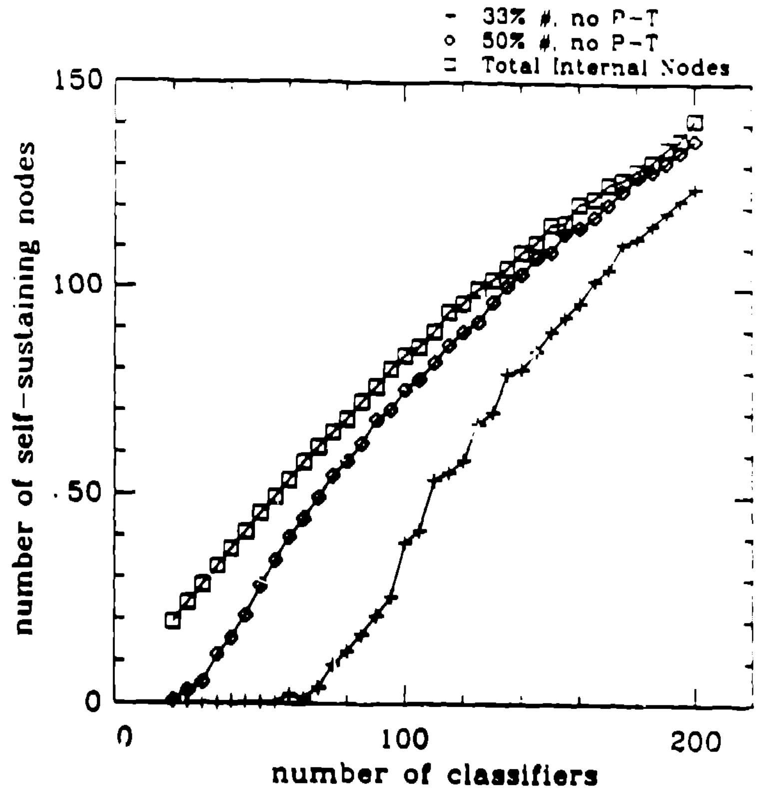

Figure 1: Self-Sustaining Activity in Classifier Systems

with classifier sets that learn the letter prediction task. Preliminary analysis indicates that the global structural properties do not change significantly, even after a significant amount of learning. In particular, the classifier sets we have examined do not show any significant increase in internal structure. This suggests that classifier systems may not be using internal chains of reasoning when solving problems, at least for the letter prediction problem.

Another aspect of learning is how stable a classifier system is to the operations of the genetic algorithm. It is important, for example, that a random mutation or cross-over be capable of having a measurable effect on the overall system, but that it not completely disrupt all ongoing activity. In Boolean network terminology, it would be interesting to know whether one genetic algorithm operation were capable of unfreezing a set of frozen components (or freezing a set of unfrozen components). More generally, we are interested in the expected amount of perturbation caused by the application of the learning operators. This can be measured by comparing the dynamics of networks before and after the application of genetic operators.

4 Conclusions

All of the above work has attempted to understand various aspects of the emergent properties of classifier system by focusing on various aspects of the dynamics inherent in such systems. By focusing on the dynamics of classifier strengths or on the effects of genetic operations it is possible to study the performance of the different learning algorithms in the context of classifier systems. By viewing the patterns of messages on the message list as a dynamical system it is possible to study how all of the various classifier system components interact with one another.

A paragraph here on emergent properties / emergent computation.

Other speculations: Start with a dynamical system and ask how you can get it to learn. Questions like how to shape basins of attraction, etc. Under this view, a "classical" dynamical system - e.g., differential equations that prescribe how a system changes over time would become "the learning rule." Under this approach, it is much easier to point to predict behaviors of different learning rules by referring to past work in dynamical systems theory.] The point here is that dynamical systems specify the way in which a system changes. This "principle of change" could be called learning under some circumstances.

References

[1] Brian Arthur. On classifier systems and models of learning. 1984. Unpublished Manuscript.

[2] Richard K. Belew and Stephanie Forrest. Learning and programming in classifier systems. *Machine Learning*, 3:192–223, 1988.

[3] M. Compiani, D. Montanari, and R. Serra. Asymptotic dynamics of classifier systems. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 298–303, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1989.

[4] B. Derrida and D. Stauffer. Phase-transitions in two-dimensional kauffman cellular automata. *Europhysics Letters*, 2:739–45, 1986.

[5] B. Derrida and G. Weisbuch. Evolution of overlaps between configurations in random boolean networks. *Journal Physique*, 47:1297–1303, 1986.

[6] J. Doyne Farmer. 1989. Personal Communication.

[7] Stephanie Forrest. *A Study of Parallelism in the Classifier System and Its Application to Classification in KL-ONE Semantic Networks.* PhD thesis, The University of Michigan, Ann Arbor, MI, 1985.

[8] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Publishing Company, Inc., 1989.

[9] David E. Goldberg. *Probability Matching, the Magnitude of Reinforcement, and Classifier System Bidding.* TCGA Report 88002, The University of Alabama, Dept. of Engineering Mechanics, Tuscaloosa AL, 1988.

[10] Geoffrey E. Hinton and James A. Anderson, editors. *Parallel Models of Associative Memory.* Lawrence Erlbaum Associates, Inc., Hillsdale, N.J., 1981.

[11] John H. Holland. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, MI, 1975.

[12] John H. Holland, Keith J. Holyoak, Richard E. Nisbett, and Paul R. Thagard. *Induction: Processes of Inference, Learning, and Discovery.* The MIT Press, Cambridge, MA, 1986.

[13] Pentti Kanerva. *Self Propagating Search: A Unified Theory of Memory.* PhD thesis, Stanford University, 1984.

[14] Stuart A. Kauffman. Emergent properties in randomly complex automata. *Physica D*, 10:145–56, 1984.

[15] John H. Miller and Stephanie Forrest. The dynamical behavior of classifier systems. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 304–310, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1989.

[16] Rick L. Riolo. *Empirical Studies of Default Hierarchies and Sequences of Rules in Learning Classifier Systems.* PhD thesis, The University of Michigan. Ann Arbor, MI, 1988.

[17] A. L. Samuel. Some studies in machine learning using the game of checkers. In *Computers and Thought*, pages 71-108, McGraw-Hill, Inc., 1963.

[18] Carl Simon. 1989. Personal Communication.

[19] Robert E. Smith and ManuelValenzuela-Rendón. A study of rule set development in a learning classifier system. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 340–346, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1989.

[20] David Touretzky and Geoffrey E. Hinton. Symbols among the neurons: details of a connectionist inference architecture. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, AAAI, 1985.