DYNAMIC SYSTEM CONTROL USING RULE LEARNING AND GENETIC ALGORITHMS

David E. Goldberg

The University of Alabama, Department of Engineering Mechanics, P. 0. Box 2908, University, Alabama 35486

ABSTRACT

In this paper, recent research results are presented which demonstrate the effectiveness of a rule learning system in two dynamic system control tasks. This system, called a learning classifier system (LCS), learns rules to control a simple Inertlal object and a simulated natural gas pipeline.

Starting from a randomly generated state of mind, the learning classifier system learns string-rules called classifiers which match strings called messages. Messages are sent by environmental sensors or by previously activated classifiers. Each classifier's effectiveness is evaluated by an internal service economy complete with bidding and auction. Furthermore, new rules are created by an innovative search mechanism called a genetic algorithm. Genetic algorithms are search algorithms based on the mechanics of natural genetics.

Results from computational experiments in both tasks are presented. In the inertial object task, the LCS learns an effective set of rules to center the object repeatedly. In the pipeline task, the LCS learns to control the pipeline under normal summer and winter conditions. It also learns to alarm correctly for the presence or absence of a leak. These results demonstrate the effectiveness of the learning classifier system approach and suggest further refinements which are currently under Investigation.

I BACKGROUND

Many industrial tasks and machines that once required human intervention have been all but completely automated. Where once a person tooled a part, a machine tools, senses, and tools again. Where once a person controlled a machine, a computer controls, senses, and continues its task. Repetitive tasks requiring a high degree of precision have been most susceptible to these extreme forms of automated control. Yet despite these successes, there are still many tasks and mechanisms that require the attention of a human operator. Piloting an airplane, controlling a pipeline, driving a car, and fixing a machine are just a few examples of ordinary tasks which have resisted a high degree of automation. What is it about these tasks that has prevented more autonomous, automated control? Primarily, each of the example tasks requires, not just a single capability, but a broad range of skills for successful performance. Furthermore, each task requires performance under circumstances which have never been encountered before. For example, a pilot must take off, navigate, control speed and direction, operate auxiliary equipment, communicate with tower control, and land the aircraft. He may be required to do any or all of these tasks under extreme weather conditions or with equipment malfunctions which he has never faced before. Clearly, the breadth and perpetual novelty of the piloting task (and similarly complex task environments) prevents the ordinary algorithmic solution used in more repetitive chores. In other words, difficult environments are difficult because not every possible outcome can be anticipated in advance, nor can every possible response be predefined. This truth places a premium on adaptation.

In this paper, we present research results from the application of a learning classifier system (LCS) to the control of two dynamic systems [1], an Inertlal object and a natural gas pipeline. A learning classifier system is a rule learning system which combines a computationally complete rule and message system, an apportionment of credit system based on a service economy analogue, and a genetic algorithm to form a system with sufficiently broad adaptability and efficiency to learn how to control each dynamic system starting from a random state of mind.

In the remainder of the paper we first examine the origins and structure of learning classifier systems. We examine Its application to the control of an inertial object. Finally, we observe its adaptation to normal summer and winter conditions, as well as abnormal leak events, in the control of a simulated natural gas pipeline.

II ORIGINS AND STRUCTURE OF LEARNING CLASSIFIER SYSTEMS

Learning classifier systems are the latest outgrowth of Holland's continuing work on adaptive systems. In 1962, when Holland outlined his

This paper describes work done at the University of Michigan. It was partially supported by U. S. Department of Energy Contract DE-FG02-80ER10125, assistance from the MW Goodrich Fund, the H. W. King Fund, the Rackham School of Graduate Studies, and the U. M. Office of Energy Research.

theory of adaptive systems [2] he developed a general theory encompassing many adaptive systems, but ultimately he was addressing himself toward programmable machines that could reprogram themselves.

With this foundation more concrete suggestions emerged for classes of schemata processors [3] which in some limited respects resemble the present day LCS. This work has evolved into the intricately interesting, but as yet unimplemented broadcast language [A]. The first practical implementation of a learning system based on these theories appeared in 1978. Holland and Reitman [5] describe this first Classifier System which learns a simple maze running task. Though the task is simple, the achievement Is remarkable because of its successful marriage of a rule-based knowledge system and a genetic algorithm for discovery of new rules. Others have continued and extended this work in a variety of areas ranging from visual pattern recognition to draw poker [6-10].

A learning classifier system (LCS) is an artificial system that learns rules, called classifiers, to guide its interaction in an arbitrary environment. It consists of three main elements:

1. Rule and Message System
2. Apportionment of Credit System
3. Genetic Algorithm

A schematic of an LCS is shown in Figure 1. In this schematic, we see that the rule and message system receives environmental information through its sensors, called <u>detectors</u>, which decode to some standard <u>message</u> format. This environmental message is placed on a <u>message list</u> along with a finite number of other internal messages generated from the previous cycle. Messages on the message list may activate <u>classifiers</u>, rules in the <u>classifier store</u>. If activated a classifier may then be chosen to send a message to the message list for the next cycle. Additionally, certain messages may call for external action through a number of action triggers called <u>effectors</u>. In this way, the rule and message system combines both external and Internal data to guide behavior and the state of mind in the next state cycle.

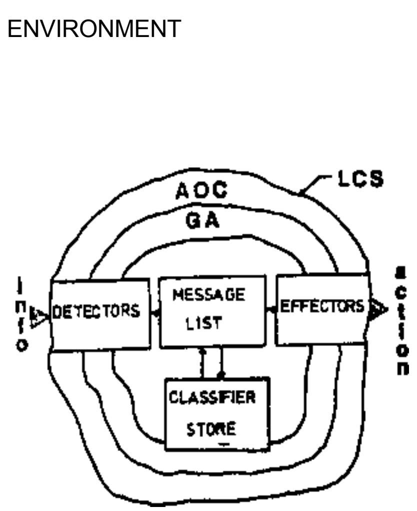


Figure 1. Schematic - Learning Classifier System

In an LCS, it is important to maintain simple syntax in the primary units of information, messages and classifiers. In the current study messages are 1-bit (binary) strings and classifiers are 3l-positlon strings over the alphabet {0,1,#}. In this alphabet the # is a wild card, matching a 0 or a 1 in a given message. Thus, we maintain powerful pattern recognition capability with simple structures.

In traditional rule-based expert systems, the value or rating of a rule relative to other rules is fixed by the programmer in conjunction with the expert or group of experts being emulated. In a rule learning system, we don't have this luxury. The relative value of different rules is one of the key pieces of information which must be learned. To facilitate this type of learning, Holland [6] has suggested that rules coexist in a competitive service economy. A competition is held among classifiers where the right to answer relevant messages goes to the highest bidders with this payment serving as s source of income to previously successful message senders. In this way, a chain of middlemen is formed from manufacturer (source message) to message consumer (environmental action and payoff). The competitive nature of the economy insures that the good rules survive and that bad rules die off.

In addition to rating existing rules, we must also have a way of searching for new, possibly better, rules. The primary mechanism for this type of creative learning within an LCS is the genetic algorithm. A genetic algorithm (GA) is a search algorithm based upon the mechanics of natural genetics. It combines a Darwinian survival of the fittest among a population of artificial chromosomes (string rules) and a structured, yet randomized, Information exchange among randomly mated pairs of rules. GA simplicity of operation and power of effect have been demonstrated in function optimization, optimal control, as well as LCS domains.

Taken together, the learning classifier system with its computationally complete and convenient rule and message system, an apportionment of credit system modeled after a competitive service economy, and the innovative search of a genetic algorithm, provides a unified framework for investigating the learning control of dynamic systems. In the remainder, we examine the application of the LCS to inertial object and gas pipeline control.

III <u>INERTIAL OBJECT CONTROL</u>

We test the LCS and its ability to control an inertial object in the 1-D space depicted in Figure 2. The object is frictionless and is governed by Newton's second law. As shown, the domain is bounded by inelastic walls, and the LCS receives perfect, yet crude and discrete, knowledge of the object's position, velocity, force, and reward in an eight bit environmental message. The LCS has a simple behavioral repertoire: it can apply a force of given magnitude to the right or to the left.

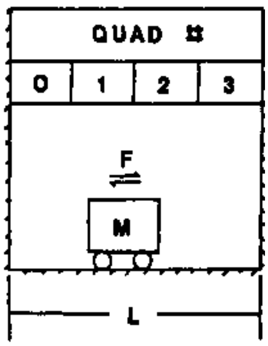


Figure 2. Inertial Object Domain - Schematic

In these tests the LCS is rewarded if the action it has taken is consistent with the goal of centering the object (maximum point score = 6). Furthermore, a criterion count is incremented If the object is centered for 10 consecutive time steps. After this, the object is randomly disturbed by a large force to make the system try again.

Starting from a randomly generated set of 30 rules we compare the performance of the LCS with genetic algorithm and without genetic algorithm to a random walk on the basis of time-averaged point count (TOTALEVAL/T) in Figure 3. We note that both the LCS runs are much better than random performance. Furthermore, case I0LCS.2 (with GA) eventually overtakes and outperforms run I0LCS.1 (without GA). In fact, while the differences appear small on this basis, the difference In physical control is much better in the case with genetic algorithm.

To see this we shift the basis of comparison to the more sensitive measure, time-averaged number of criterion achievements displayed as Figure 4. Again, LCS performance is far better than random. Performance with the genetic algorithm is that much better than without. In fact, the run with genetic algorithm has found restoration and braking rules similar to those that might be programmed by a knowledgeable control engineer. The run without genetic algorithm has ineffective braking rules thereby limiting its capability*

IV GAS PIPELINE CONTROL

A pipeline model, load schedule, and upset conditions are programmed and Interfaced to the LCS. We briefly discuss this environmental model and present results of normal operations and upset tests.

A model of a pipeline has been developed which accounts for linepack accumulation and

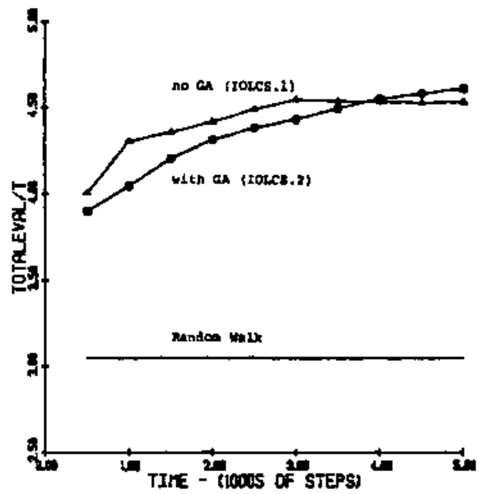


Figure 3. Time-averaged TOTALEVAL vs. Time Random Rule Set - Runs I0LCS.1 and I0LCS.2

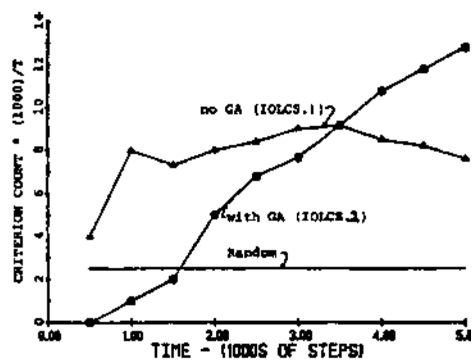
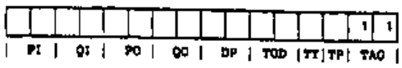


Figure 4. Time-averaged Goal Count vs. Time Random Rule Set - Runs I0LCS.1 and I0LCS.2

frlctlonal resistance. User demand varies on a dally basis and depends upon the weather. Different patterns may be used for winter and summer operation. In addition to normal summer and winter conditions, the pipeline may be subjected to a leak upset. During any given time step, a leak may occur with a specified leak probability. If a leak occurs, the leak flow, a specified value, is extracted from the upstream junction and persists for a specified number of time steps.

The LCS receives a message about the pipeline condition every time step. A template for that message Is shown in Figure 5. The system has complete, albeit Imperfect and discrete, knowledge of its state Including inflow, outflow, inlet pressure, outlet pressure, pressure rate change, season, time of day, time of year, and current temperature reading.

In the pipeline task, the LCS has a larger range of alternatives for actions it may take compared with the Inertial object task. It may send out a flow rate chosen from one of four



Variable	Description	min	max	# of positions
PI	inlet pressure	0	2000	2
QI	inlet flow	0	80	2
PO	outlet pressure	0	2000	2
QO	outlet flow	0	80	2
DP	u. s. pressure rate	-200	200	2
TOD	time of day	0	24	2
TT	time of year	0	1	1
TP	temperature	0	1	1

Figure 5. Pipeline LCS Environmental Message Template

values and it may send a message indicating whether a leak is suspected or not.

The LCS receives reward from its trainer depending upon the quality of its action in relation to the current state of the pipeline. To make the trainer ever-vigilant, a computer subroutine has been written which administers the reward consistently. This is not a necessary step, and reward can come from a human trainer.

Under normal operating conditions we examine the performance of the learning classifier system with and without the genetic algorithm enabled. Without the genetic algorithm, the system is forced to make do with its original set of rules. The results of a normal operating test are presented in Figure 6. Both runs with the LCS outperform a random walk (through the operating alternatives). Furthermore, the run with genetic algorithm enabled is superior to the run without GA. In this figure, we show time-averaged total evaluation versus time of simulation (maximum reward per tlmestep - 6).

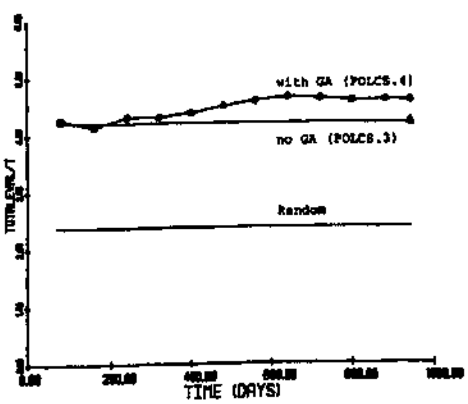


Figure 6. Time-averaged TOTALEVAL vs. Time Normal Operations - Runs P0LCS.3 & P0LCS.4

More dramatic performance differences are noted when ve have the possibility of leaks on the system. Figure 7 shows the time-averaged total evaluation versus time for several runs with leak upsets. Once again the LCS is initialized with random rules and permitted to learn from external reward. Both LCS runs outperform the random walk and the run with GA clearly beats the run with no new rule learning. To understand this, ve take a look at some auxiliary performance measures. In Figure 8 ve see the percentage of leaks alarmed correctly versus time. Strangely, the run without GA alarms a higher percentage of leaks than the run vlth GA. This may seem counterintuitive until ve examine the false alarm statistics in Figure 9. The run without GA is only able to alarm a high percentage of leaks correctly because it has so many false alarms. The run with GA decreases its false alarm percentage, while increasing its leaks correct percentage.

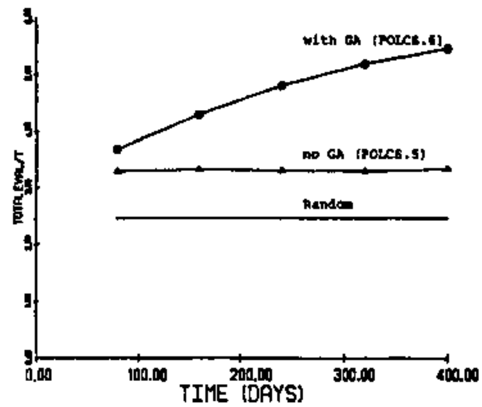


Figure 7. Time-averaged TOTALEVAL vs. Time - Leak Runs - POLCS.5 & POLCS.6

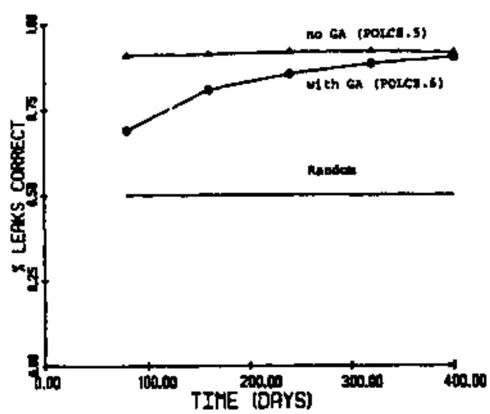


Figure 8. Percentage of Leaks Correct vs. Time Runs POLCS.5 & POLCS.6

V CONCLUSIONS

In this paper, we have applied a learning classifier system to the control of two different dynamic systems, an inertlal object and a natural gas pipeline.

In the tvo applications the LCS learns effective rules in normal and abnormal operating

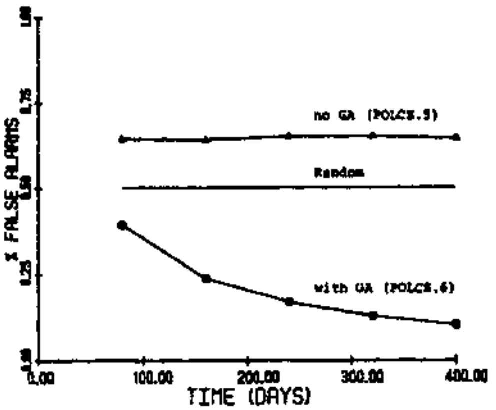


Figure 9, Percentage of False Alarms vs. Time Runs POLCS.5 & POLCS.6

conditions alike. In the inertial object task, the LCS learns to center the object consistently. In the pipeline task, the LCS learns to operate the pipeline under normal summer and winter conditions. It also learns to alarm correctly with increasing accuracy for the presence or absence of a leak.

While the applications of the LCS in this paper have been necessarily specific, this work's implications for other research in artificial Intelligence are more far-reaching. The use of ruthlessly spartan syntax and simple, yet powerful learning heuristics drawn from nature may elsewhere prove promising in our quest for programs which effectively reprogram themselves with better Instructions.

REFERENCES

1. Goldberg, D. E., "Computer-Aided Gas Pipeline Operation using Genetic Algorithms and Rule Learning," Ph.D. dissertation, University of Michigan, Ann Arbor, 1983.

2. Holland, J. H., "Outline for a Logical Theory of Adaptive Systems,"J.ACM, 3(1962), 297-314.

3. Holland, J. H., "Processing and Processors for Schemata," in Associative Information Techniques, Jacks, E.L. (ed.) pp. 127-146, American Elsevier, New York, 1971.

4. Holland, J. H., Adaptation In Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.

5. Holland, J. H. and J. S. Reltman, "Cognitive Systems Based on Adaptive Algorithms," in Pattern-Directed Inference Systems, Waterman, D. A. and F. Hayes-Roth (eds.), pp. 313-329, Academic Press, New York, 1978.

6. Holland, J. H., "Adaptive Algorithms for Discovering and Using General Patterns in Growing Knowledge-Bases," Int. J. Policy Analysis and Information Systems, 4(1980).

7. Holland, J. H., "Genetic Algorithms and Adaptation," Tech. Report #34, Cognitive Science Series, Univ. of Michigan and Univ. of Chicago, December, 1981.

8. Smith, S. F., "A Learning System Based on Genetic Adaptive Algorithms," Ph.D. dissertation, University of Pittsburgh, Pittsburgh, 1980.

9. Booker, L. B., "Intelligent Behavior as an Adaptation to the Task Environment," Ph.D. dissertation, University of Michigan, Ann Arbor, 1982.

10. Wilson, S., "Adaptive 'Cortical' Patter Recognition," unpublished manuscript, Rowland Institute of Science, Cambridge, MA, 1983.