Proceedings of the 4th IEEE International
Symposia of Intelligent Control,
October 1989.

# Collective Learning Systems: a Model for Automatic Control

Stephen A. Osella

National Institute of Standards and Technology
Gaithersburg, MD 20899

## ABSTRACT

Adaptive control is usually required in situations where conditions are widely varying and/or unpredictable. Adaptive controllers have been implemented using Artificial Intelligence (AI) techniques. Knowledge acquisition, in the classical AI approach, consists of programming a knowledge-base encoded in the form of rules, frames, or object-oriented data structures. However, except for the most trivial of problems, the time and resources required to accumulate enough knowledge is intractably high. Collective Learning Systems (CLS) is a paradigm for the acquisition and application of knowledge through learning. In the CLS approach, Collective Learning Automata acquire knowledge through a learning process consisting of trial-and-error interactions with the environment. This paper investigates how CLS theory may be used to model and implement adaptive control systems. The pole-balancing problem is posed as the experimental system paradigm.

## 1.0 Introduction

Researchers in control theory have known for some time the power, and possibly the necessity, of using adaptive control in situations where conditions are widely varying and/or generally unpredictable. In broad terms, an adaptive controller is one which adjusts its actions in accordance to changing conditions [1]. This paper deals with the branch of adaptive control theory called Learning Control which is "a process of forcing the system to have a particular response to a specific input signal by repeating the input signals and then correcting the system externally" [2]. A distinction is sometimes made in that a learning controller bases its decisions on experience (memory) gained from interactions with the environment.

Control theorists are not the only ones interested in issues relating to automatic control. For instance, Norbert Weiner made contributions to control theory and, as the founder of Cybernetics, inspired the study of Artificial Intelligence (AI). Having begun with the common goal of designing machines capable of satisfactory performance in uncertain conditions, today, control theorists and AI researchers have diverged into interests with seemingly disparate goals. Recently, the desire for greater versatility has reunited the two fields in an attempt to achieve robust "intelligent" control (behavior).

Knowledge acquisition, in the classical AI approach, consists of programming a knowledge-base, the content of which is usually encoded in the form of rules, frames, or object-oriented data structures [3]. This form of AI has been used for a variety of well-defined specific tasks such as medical diagnosis, computer system configuration, and experiment design. However, except for the most trivial of problems, the time and resources required to accumulate enough knowledge is excessively high.

Collective Learning Systems (CLS) theory is a paradigm for the acquisition and application of knowledge through learning [4]. CLS theory is derived mainly from work on Stochastic Automata performed by Tsetlin in the 1960's [5]. Learning in CLS theory involves an informal method of induction in which the transformation from the current state to the next state is determined by successive approximation.

A Collective Learning System is a group of Automata participating in the accomplishment of a specific task [6]. The Automata, which in CLS terminology are called Collective Learning Automata (CLA), are interconnected in a heterarchical and hierarchical fashion. Each level of the hierarchy constitutes a decomposition (sub-goal) of the task objective. In the CLS approach, the individual CLA acquire knowledge through a learning process consisting of trial-and-error interactions with their environment.

Research on the use of learning automata in control applications was very apparent in

the 1960's and early 70's with researchers such as Gibson, Tsypkin, Thathachar, Narendra, Fu, and Shapiro [7]. However, by the mid-70's, the research all but vanished, giving ground to the field of parametric adaptive control.

Unlike CLS theory, in these early experiments the environment did not collect the responses; instead, the environment evaluated the performance of the automaton after each response. This, of course, quickly leads to a combinatorial explosion of the search-space which is the reason generally credited for the loss of interest in learning automata.

CLS attempt to conquer this problem by evaluating a series of responses, aggregating the stimulus domain, and distributing the information through the use of parallelism and goal decomposition. CLS theory has been applied to complex problems, exhibiting robust learning and dynamic adaptivity. The objective of this paper is to investigate how CLS theory may be used to model adaptive control systems.

## 2.0 Problem Statement

The intelligence manifested by a state-of-the-art robot is almost exclusively implemented using programmed Artificial Intelligence techniques such as rule-based systems or state-tables. This approach requires that the strategies be devised a priori and programmed into the computer's knowledge-base. Furthermore, the nature of these techniques is that the information processing is heuristic by design.

In general, however, a controller must operate in an environment with unknown characteristics, thus making the programming task extremely difficult. Moreover, the coordination of sensory feedback with motor control actions cannot, in general, be solved analytically. What is required is a method of associating the sensory feedback with the control actions so that in time the appropriate control law is obtained. As will be demonstrated, this can be achieved using CLS theory where the sensory feedback is the stimulus and the control action is the response.

A simple, yet clearly representative, example of a dynamical system requiring rudimentary control is the pole-balancing problem. This system has been used in a similar experiment [8]. The system, illustrated in Figure 1, consists of a cart fitted with a pole attached to a hinge. The cart is allowed to move laterally in one dimension in order to keep the pole balanced. The acceleration of the cart imposes a force at the base of the pole such that the moment created has the effect of balancing the pole.

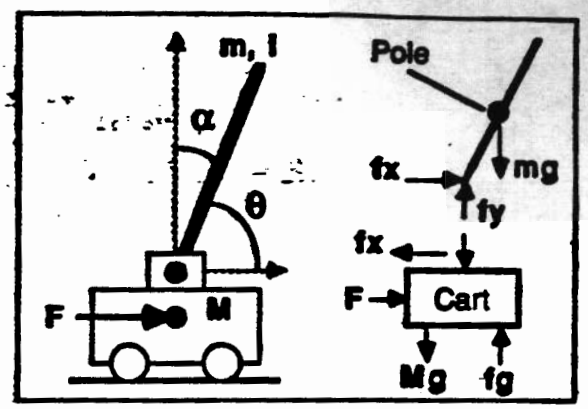



Figure 1: The Cart-Pole System

The objective of the controller is to move the cart back and forth, by applying a horizontal force at the cart's center of gravity, such that the pole remains vertical. The sensory feedback consists of the angular position, angular velocity, and angular acceleration of the pole. The learning process establishes a relationship (function) between the input feedback and output control action.

## 3.0 Proposed Solution

The behavior of any dynamical system can be described by an n-th order differential equation. This type of notation is exact in representation but is difficult to determine and manipulate analytically. To facilitate derivations, a notation called the state-variable form was developed [9].

In state-variable form, the system is represented by n first-order differential equation, each describing one system variable. The n equations can be represented in matrix form. As a result, an n-th order multiple input-multiple output system has the following form:

$$X(t+1) = A(t) * X(t) + B(t) * U(t) \quad (1)$$

$$Y(t) = C(t) * X(t) \quad (2)$$

where X(t) is the state of the system, X(t+1) is the next state, U(t) is the input, Y(t) is the output, and A(t), B(t), and C(t) are the control parameters.

Equation (1) is the general form of a state-transition function more commonly encountered in Automata Theory [10]. A state-transition function defines the next state of the system given the current state and the input stimuli. A state-transition function has the following form:

$$X(t+1) = E(X(t), U(t)) \quad (3)$$

where E is the state-transition function.

If $X(t)$ and $U(t)$ in equation (3) are combined to form a "total" system state, then a system can be represented by:

$$S(t+1) = T(t) * S(t) \qquad (4)$$

where $S(t)$ is the current total state of the system, $S(t+1)$ is the next state, and $T(t)$ is the state-transition transform.

Equation (4) can be recognized as the general Automaton equation. The state-transition transform is equivalent to the state-transition function of equation (3) in matrix form. Consequently, any control system can be represented as an automaton describable by equation (4).

### 3.1 Learning Automata

The vast majority of research and development in AI has emphasized the deductive acquisition of knowledge [11]. This approach involves specifying the state-transition transform and successively applying it until the goal-state is reached. Programming languages such as PROLOG, LISP, or expert systems have been used because they provide mechanisms for symbolic representation and manipulation.

As stated above, it is generally very difficult, if not impossible, to predetermine the correct transform to apply. Furthermore, the transform cannot be easily modified or checked for consistency. For these and other reasons, doubts remain concerning the usefulness of this approach for complex problems and in particular for automatic control.

In the event that the transform cannot be determined, induction can be used to find it. To do so, however, both the current state and the next state must be known. This poses a problem since knowledge of the next state is impossible. The only recourse is to choose a feasible transform, utilize it, evaluate the quality of the response, and adjust the transform appropriately. A machine using this procedure is called a Learning Automaton (LA) [12]. If the state-transitions are probabilistic, the LA is called a Learning Stochastic Automaton (LSA) [13]. This latter type of automaton forms the basis for the controller investigated in this paper.

The state-transition transform comprises the memory of an LA. The LA's memory consists of a set of stimulants which is discretized into N stimulus states. Each stimulus state has an associated respondent vector each of which contains one or more response components. An example of an LSA's memory is shown in Figure 2. Each response represents a state-transition for which a state-transition probability is assigned.

| Stimulants: State Id | Respondents | | | |
|---|---|---|---|---|
| 0 | 0.16 | 0.29 | 0.28 | 0.27 |
| 1 | 0.00 | 0.00 | 1.00 | 0.00 |
| 2 | 0.25 | 0.25 | 0.25 | 0.25 |
| ... | ... | ... | ... | ... |
| N - 1 | 0.11 | 0.23 | 0.12 | 0.54 |

Figure 2: An Example of an LSA's Memory

As mentioned above, an LA acquires knowledge through a learning process, illustrated in Figure 3, consisting of trial-and-error interactions with the environment. The environment submits an initial **stimulus** to the LA. Based on the stimulus and the LA's **memory** the LA **selects** a response and issues it to the environment.

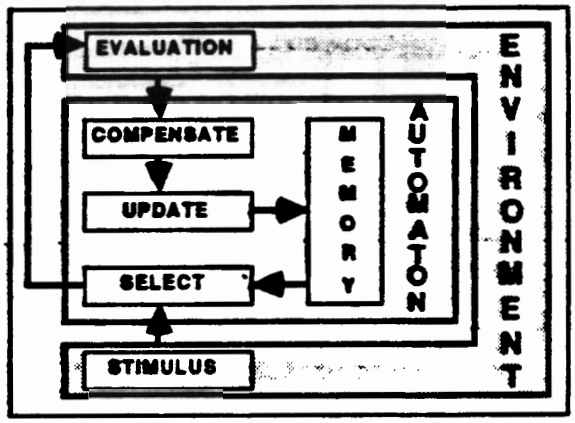



Figure 3: An LA's Learning Process

The environment **evaluates** the response with respect to the effectiveness of the response in achieving the LA's objective. The LA can **compensate** the evaluation based on an inherent or learned knowledge of the "quality" of the appraisal. The environment is considered to be consistent, infallible, and truthful which in learning systems theory is said to be stationary, deterministic, and correct. Lastly, the learning automaton **updates** its **memory** resulting in the acquisition of knowledge. In time, the LA will consistently correlate the most appropriate response for a given stimulus and thus demonstrate learning.

### 3.2 Collective Learning

If the LA is given an evaluation for each response, the process described above is nothing more than an exhaustive search of the state-space until the correct response

is found. - In general, this procedure is infeasible because for most real-life problems it is not possible to evaluate each response in real-time. Furthermore, the computational requirements for such a process are prohibitively high.

The learning process discussed above can be modified such that the automaton receives an evaluation only after a number of responses have been issued. An automaton undergoing such a process is called a Collective Learning Automaton (CLA). The theory of the interaction of the CLA with its environment is called Collective Learning Systems (CLS) Theory.

## 4.0 Technical Approach

The architecture of the control system used in the simulation experiments is illustrated in Figure 4. In the next section, the individual components of the CLA and environment are discussed in detail.

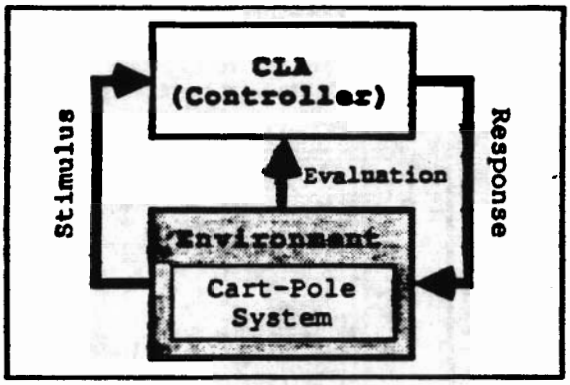



Figure 4: The Control System Architecture

### 4.1 CLS Definition

**Stimulus**: The stimulus issued by the environment to the CLA consisted of the state of the pole (plant) $S_t = \{\theta, \dot{\theta}, \ddot{\theta}\}$. For this experiment, in determining the control function, the state of the cart was ignored. Each state-variable was partitioned into 32 divisions, resulting in a stimulus domain having 32768 states. The partition's class marks were nonlinearly distributed, with a greater number positioned around the vertical pole position and zero velocity and acceleration.

**Selection**: The response range for each stimulus state was -1.0 to 1.0 representing the fraction of the maximum force to be applied to the cart's center of gravity. The response range was partitioned into 11 divisions. To permit fine-tuned control, the response range class marks were distributed with more classes around zero gain. Given a stimulus, the response, and therefore the force to apply, was chosen using *maximal thresholded deterministic selection with random arbitration*. The probability threshold was set to one percent.

**Evaluation**: The objective policy of the controller was to keep the pole balanced (vertical) as long as possible. This policy can be restated as an objective function requiring the minimization of the average angle $\alpha$ between the pole position $\theta$ and $\Pi/2$ (See Figure 1). Therefore, the evaluation, which in this experiment was issued every 10 CLA responses, is given by:

$$\xi = (\alpha_{previous} - \alpha_{current}) \; / \; \Delta\alpha_{total} \qquad (5)$$

where $\alpha_{current}$ is the value of $\alpha$ at the time of the evaluation, $\alpha_{previous}$ is the value of $\alpha$ at the time of the previous evaluation, and $\Delta\alpha_{total}$ is running sum of the absolute value of the difference between the current and previous values of $\alpha$.

**Compensation**: For this experiment, the CLA did not modify the evaluation.

**Update**: The update procedure used in this experiment to modify the state-transition probabilities is the following:

1. Update the probability of the selected response.

$$P_s = P_s + \xi * (1 - P_s), \; \xi > 0 \qquad (6a)$$

$$P_s = P_s + \xi * P_s, \; \xi <= 0 \qquad (6b)$$

2. Sum the respondent vector's probabilities.

3. Adjust each probability proportional to the sum.

$$P_i = P_i \; / \; P_{sum} \qquad (7)$$

### 4.2 Experimental Procedures

The learning control of the cart-pole system was tested using a discrete-time simulation. The system's equations of motion were derived using the forces and masses shown in Figure 1. A fourth-order Runge-Kutta numerical method was used to approximate the system's differential equations. A time interval of 1 millisecond was used. The CLA controller sampled and responded every 10 time intervals (1 centisecond). As mentioned above, the environment evaluated the CLA's performance every 10 responses (1 decisecond).

The learning period consisted of 50,000 runs. The system variables were initialized at the start of each run. The pole position was randomly set between 60 and 120 degrees and its velocity and acceleration were set to 0. The cart's position, velocity, and acceleration were also set to 0. A run lasted for 10,000 time intervals (10 seconds) or until the pole fell outside the range [0, Π].

## 5.0 Simulation Results

Figure 5 illustrates the percentage of runs over the length of the learning period which lasted for at least 10,000 time intervals. As can be seen, after 1000 runs the CLA controller was already able to keep the pole balanced in 663 of the runs. Naturally, because the initial pole position was set randomly, many of the runs started with an angle close to vertical leading to many of these early successes. Most of the failures resulted from initial angles farther away from Π/2. The runs starting close to vertical had the effect of habituating the CLA to keeping the pole upright once balance was achieved.

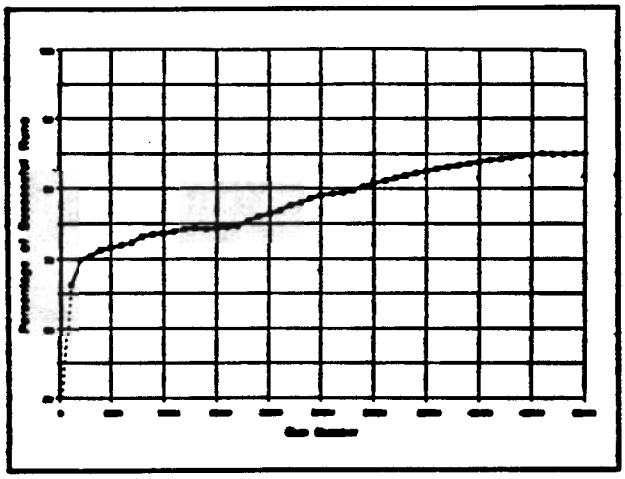
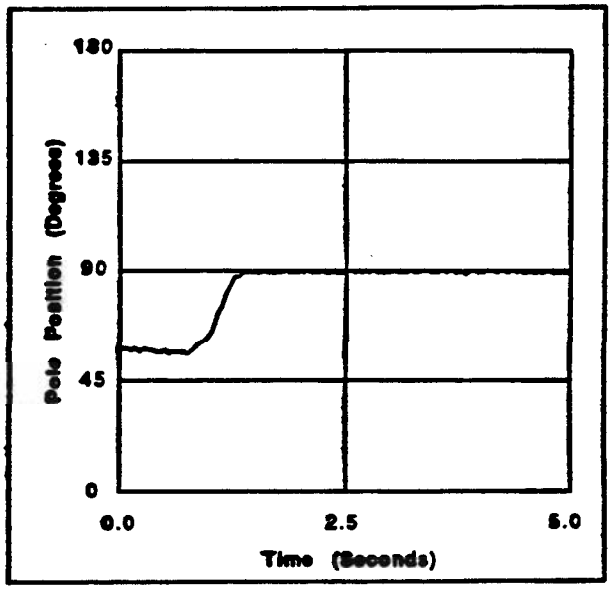


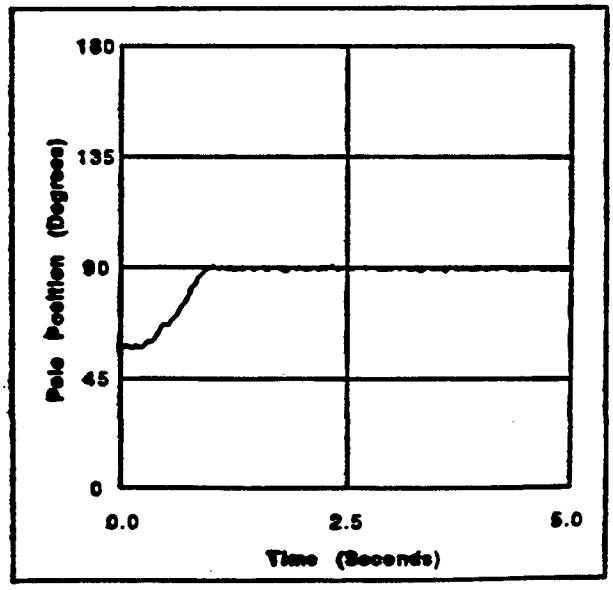
Figure 5: Percentage of runs lasting at least 10 seconds.

Except for a small perturbation in the learning at around 15,000 runs, the level of learning (as measured by the number of successful runs) can be observed to be gradually and steadily increasing throughout the learning period. The CLA was continuing to learn even at the end of the learning period even though the rate of learning had decreased noticeably. The progress of a CLA's learning is a function of the size of the CLA's memory, the environment's evaluation policy, the CLA's selection, compensation, and update functions, the response collection length, and the system's dynamic characteristics.

Figure 6 shows the change in pole position during two runs for which the initial pole position was set to 60 degrees. The figures demonstrate the controller's performance at two stages of the learning period. As can be observed, the control function is critically damped. The CLA learned that to minimize negative evaluations (punishments) it had to prevent overshooting.




(a)




(b)

Figure 6: Performance of the CLA (a) after 10,000, (b) after 50,000 runs

The nature of the evaluation policy used in this experiment is such that the CLA would attempt to achieve balance in the shortest amount of time possible. This behavior is in conflict with the previously mentioned performance requirement in that this would lead to overshoot. To meet both requirements, the CLA learned to apply maximum force to get the pole upright and then, as the pole approached the vertical position, to apply the maximum force in the opposite direction to decrease the pole's velocity. This behavior is demonstrated by the jaggedness of the plots after the pole reached $\Pi/2$. A performance requirement such as fuel conservation or minimization of stress could be used to alleviate this phenomenon.

## 6.0 Conclusions

It was demonstrated that using very simple evaluation, selection, and update functions the CLA was able to learn satisfactory control of the cart-pole system. The success rate of 85% at the end of the learning period was mainly a result of a relatively low incidence on the states corresponding to starting angles close to 60 and 120 degrees. The performance can be improved by presenting the CLA with a comprehensive range of operation conditions.

The evaluation policy was purposefully defined in terms of an objective function in order to be able to generalize the concept to a wide variety of problems. For instance, if the cart position were of importance, the objective function would require the minimization of both the pole inclination as well as the difference between the desired and actual cart position. Notice that these two goals are contradictory; the CLA would have to learn to relax the constraints in order to satisfy the objective.

Further areas of investigation include methods of testing the CLA's performance given more realistic operation conditions. It is impractical to assume that sensors are free of errors and noise or that data acquisition is instantaneous. The sensors may not even accurately describe the process under control (the identification problem) or worse, provide erroneous information. Similarly, a controller's actuators are in general inefficient and have inertia. The CLA would have to receive feedback describing the level of force being effected. Lastly, the controlled system's dynamics are usually nonlinear resulting in a more difficult control problem.

## REFERENCES


[1] Harris, C.J., and Billings, S.A., Self-Tuning and Adaptive Control: Theory and Applications, Peter Peregrinus Ltd., 1981.

[2] Tsypkin, Ya. Z., Adaptation and Learning in Automatic Systems, Academic Press, 1971.

[3] Bock, P., "The Emergence of Artificial Intelligence: Learning to Learn", The AI Magazine, Fall, 1985.

[4] Bock, P., "Observations of the Properties of a Collective Learning Stochastic Automaton", Proc. International Information Sciences Symposia, Patras, Greece, Aug. 1976.

[5] Tsetlin, M. L. "On the behavior of finite automata in random media", Avtomat. Telemekh., vol. 22, pp. 1345-1354, Oct. 1961.

[6] Bock, P., Weingard, F., White, J., "A Universal Model for the Structure and Function of Collective Learning Systems", IEEE Conference on New Directions in Computing, August, 1985.

[7] Shapiro, I.J., Narendra, K.S., "The use of Stochastic Automata in Adaptive Control," Yale University, New Haven, Conn., Tech. Report CT-23, Dec. 1968.

[8] Barto, A.G., Sutton, R.S., and Anderson, C.W., "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 13, No. 5, September/October, 1983.

[9] Schultz, D.G., and Melsa, J.L., State Functions and Linear Control Systems, Magraw-Hill, 1967.

[10] Veelenturf, L.P.J., "An Automata-Theoretical Approach to developing Learning Neural Networks", Cybernetics and Systems Journal, 1981.

[11] Bock, P., "Learning to Learn", Lecture Notes, George Washington University, 1986.

[12] Simha, R., and Kurose, J.F., "Relative Reward Strength Algorithms for Learning Automata", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, No. 2, March/April 1989.

[13] Narendra, K.S., and Thathacher, M. "Learning Automata -- A Survey", IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-4, no. 4, pp. 323-334, July, 1974.