Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems

ANDREW G. BARTO, MEMBER, IEEE, RICHARD S. SUTTON, AND CHARLES W. ANDERSON

Abstract—It is shown how a system consisting of two neuronlike adaptive elements can solve a difficult leaning control problem. The task is to balance a pole that is hinged to a movable cart by applying forces to the cart's base. It is assumed that the equations of motion of the cart–pole system are not known and that the only feedback evaluating performance is a failure signal that occurs when the pole falls past a certain angle from the vertical, or the cart reaches an end of a track. This evaluative feedback is of much lower quality than is required by standard adaptive control techniques. It is argued that the learning problems faced by adaptive elements that are components of adaptive networks are at least as difficult as this version of the pole-balancing problem. The learning system consists of a single *associative search element* (ASE) and a single *adaptive critic element* (ACE). In the course of learning to balance the pole, the ASE constructs associations between input and output by searching under the influence of reinforcement feedback, and the ACE constructs a more informative evaluation function than reinforcement feedback alone can provide. The differences between this approach and other attempts to solve problems using neuronlike elements are discussed, as is the relation of this work to classical and instrumental conditioning in animal learning studies and its possible implications for research in the neurosciences.

Manuscript received August 1, 1982; revised April 20, 1983. This work was supported by AFOSR and the Air Force Wright Aeronautical Laboratory under Contract F33615-80-C-1088.

The authors are with the Department of Computer and Information Science, University of Massachusetts, Amherst, MA 01003.

0018-9472/83/0900-0834$01.00

I. Introduction

MATHEMATICALLY formulated networks of neuronlike elements have been studied both as models of specific neural circuits and as abstract, though biologically inspired, computational architectures. As models of specific neural circuits, network models can provide theories to explain anatomical and physiological data. As computational architectures, they represent attempts to explore possible substrates for intelligent behavior, both natural and artificial. Networks of this second category are relevant to brain and behavioral science to the extent that their behavior can be related to phenomena of animal behavior for which no plausible mechanisms are known, thereby suggesting novel lines of empirical research. They are relevant to artificial intelligence to the extent that they exhibit forms of problem solving, knowledge acquisition, or data storage that are difficult to achieve by more conventional means.

In this article we illustrate an abstract neural network approach that we believe can have relevance for both neuroscience and computer science. Advances in our appreciation of the complexity of biological cells make it clear that the 35-year old metaphor that places the neuron at the level of the computer logic gate is inadequate. Neurons and synapses have information processing capabilities that make use of both short- and long-term information storage, locally implemented by complex biochemical mechanisms. Biochemical networks within cells are known to perform functions that had previously been attributed to networks of interacting cells. These facts call for new neural metaphors. Moreover, advances in computer science suggest the possibility of achieving sophisticated problem-solving capacity through networks of interacting components that are themselves powerful problem-solving systems (e.g., [1] and [2]). In our approach, network components are neuronlike in their basic structure and behavior and communicate by means of excitatory and inhibitory signals rather than by symbolic messages, but they are much more complex than neuronlike adaptive elements studied in the past. Rather than asking how very primitive components can be interconnected in order to solve problems, we are pursuing questions about how components that are themselves capable of solving relatively difficult problems can interact in order to solve problems that are even more difficult.

This article is devoted to the justification of the design of two types of neuronlike adaptive elements and an illustration of the problem-solving capacities of a system consisting of a single element of each type. We call one element an *associative search element* (ASE) and the other an *adaptive critic element* (ACE). As a vehicle for introducing our adaptive elements, we describe an earlier adaptive problem-solving system, called "boxes," developed by Michie and Chambers [3], [4]. We show that a learning strategy similar to theirs can be implemented by a *single* ASE, and we show how its learning performance can be improved by the addition of a *single* ACE. To illustrate the problem-solving capabilities of these elements, we use the pole-balancing control problem posed by Michie and Chambers to illustrate their boxes algorithm, and we compare the performance of their system with that of our own. We conclude with a brief discussion of behavioral interpretations of our adaptive elements and their possible implications for neuroscience. A strong analogy exists between the behavior of the ACE and animal behavior in classical conditioning experiments, and parallels can be seen between the behavior of the ASE/ACE system and animal behavior in instrumental learning experiments. The adaptive elements we describe are refinements of those we have discussed previously [5]–[10] and were suggested by the work of Klopf [11], [12]. Our approach also has similarities with the work of Widrow and colleagues [13], [14] on what they called "bootstrap adaptation."

The significance of endowing single adaptive elements with this level of problem-solving capability is twofold. First, we wish to suggest neural metaphors, constrained by the computational demands of problem-solving, that postulate functions for the complex cellular mechanisms that are rapidly being elucidated as the study of the cellular basis of learning progresses. Second, we wish to suggest that if adaptive elements are to learn effectively *as network components*, then they must possess adaptive capabilities at least as robust as those of the elements discussed here. As we argue in the following, the learning problem faced by an adaptive element that is deeply embedded in the interior of a network is characterized by some of the same types of complexities that are present in the pole-balancing task considered here.

Thus, although the algorithms that we implement by means of single adaptive elements can obviously be implemented by networks of many simpler elements, we are attempting to delineate those properties required of components if they are to learn how to function as interconnected, cooperating components of networks. The extensive history of attempts to construct powerful adaptive networks and the generally acknowledged failure of these attempts suggest that network components as simple as those usually considered are not adequate. This lesson from previous theoretical studies, together with our contention that the view of neural function that constrained these studies was too limited, leads us to study elements as complex as the ASE and ACE. Despite our ultimate interest in networks, we do not present results in this paper that show that the elements discussed here are able to learn as components of powerful adaptive networks. However, previous simulation experiments with networks of similar elements have provided preliminary support for our approach to adaptive networks [5], [6], [8], and the research discussed here represents an initial attempt to move toward more difficult learning problems.

Although we intend to raise questions about the level in the functional hierarchy of the nervous system at which neurons can be said to act, we are not claiming that there is necessarily a strict correspondence between single neurons and ACE's and ASE's. Some of the features of these

elements clearly are not neuronlike but can be implemented in standard ways by elements more faithful to neural limitations. For example, the ASE can "fire" with both negative and positive output values, but it can be implemented by a pair of reciprocally inhibiting elements, each capable only of positive "spikes." Consequently, by the term "neuronlike element" we do not mean a literal neuron model, and we purposefully exclude well-known neuron properties which would have no clear functional role in the present problem.

Our interest in the pole-balancing problem arises from its convenience as a test bed for exploring a variety of algorithms that may enable elements to learn effectively when embedded in networks. We are not interested in pole balancing *per se*, and our formulation of the problem, following that of Michie and Chambers [3], [4], makes it much more difficult than it would need to be if one were simply interested in controlling this type of dynamical system. We assume that the controller's design must be based upon very little knowledge of the controlled system's dynamics and that the evaluative feedback provided to the controller is of much lower quality than is required by standard adaptive control methods. These constraints produce a difficult learning control problem and reflect some of the conditions that we believe characterize the tasks faced by network components. While a variety of well-developed adaptive control methods can be (and have been) successfully applied to pole balancing, we know of none that are directly applicable to the problem subject to the constraints we impose. Additionally, the algorithm we describe can be applied to nonnumerical problems as well as to problems requiring the control of dynamical systems.

II. Learning within Networks

Many of the previous studies of adaptive networks of neuronlike elements focused on adaptive elements that are capable of solving certain types of pattern classification problems. Elements such as the ADALINE (adaptive linear element [16]) and those employed in the Perceptron [15] perform supervised learning pattern classification (see, for example, [17]). These elements form linear discrimination rules by adjusting a set of "synaptic" weights in an attempt to match their response to each training input pattern with a desired response, or correct classification, that is provided by a "teacher." The resulting discrimination rule can be used to classify new pattern instances (perhaps incorrectly), thereby providing a form of generalization. The algorithms implemented by these adaptive elements are closely related to iterative regression methods used in adaptive control for the identification of unknown system parameters [17].

Unfortunately, a network composed of these types of adaptive elements can only learn if its environment contains a teacher that can supply *each* component adaptive element with its individual desired response for each pattern in a training sequence. This is the Achilles' heel of supervised learning pattern classifiers as network components. In many problem-solving tasks, the network's environment may be able to provide assessments of certain *consequences* of the *collective activity* of all of the network components but the environment cannot know the desired responses of individual elements or even evaluate the behavior of individual elements. To use terms encountered in the artificial intelligence literature (e.g., [18]), the network's internal mechanism is not very "transparent" to the "critic."

Other approaches to the problem of learning within adaptive networks rely on adaptive elements that require neither teachers nor critics. These elements employ some form of unsupervised learning, or clustering, algorithm, often based on Hebb's [19] hypothesis that repeated pairing of pre- and postsynaptic activity strengthens synaptic efficacy. While clustering is likely to play an important role in sophisticated problem-solving systems, it does not by itself provide the necessary means for a system to improve performance in tasks determined by factors external to the system, such as, for example, the task of controlling an environment having initially unknown dynamics. For these types of tasks, a learning system must not just cluster information but must form those clusters that are useful in terms of the system's interaction with its environment. Thus it seems necessary to consider networks that learn under the influence of some sort of evaluative feedback, but this feedback cannot be so informative as to provide individualized instruction to each adaptive element.

These considerations have led us to study adaptive elements that are capable of learning to improve performance with respect to an evaluation function that assesses the consequences, which may be quite indirect, of element actions but does not directly specify these actions. Further, these elements are capable of improving performance under conditions of considerable uncertainty. Since evaluative feedback, or reinforcement feedback, will generally assess the performance of the entire network rather than the performance of individual elements, a high degree of uncertainty is necessarily present in the optimization problem faced by any individual component. Additional uncertainty arises from any delay that might exist between the time of an element's action and the time it receives the resulting reinforcement. The reinforcement feedback received by a network component at any time will generally depend upon factors other than its own action taken some fixed time earlier; it will additionally depend upon the actions of a large number of components taken at a variety of earlier times.

The ASE implements one part of our approach to these problems. Since we assume its environment is unable to provide desired responses, the ASE must *discover* what responses lead to improvements in performance. It employs a trial-and-error, or generate-and-test, search process. In the presence of input signals, it generates actions by a random process. Based on feedback that evaluates the problem-solving consequences of the actions, the ASE "tunes in" input signals to bias the action generation process, conditionally on the input, so that it will more

likely generate the actions leading to improved performance. Different actions can be optimal when taken in the presence of different input signals. Actions that lead to improved performance when taken in the presence of certain input signals become associated with those signals in a developing input–output mapping. This type of stochastic search allows the ASE to improve performance under conditions of uncertainty. We have called this general process *associative search* [8] to emphasize both its association formation and generate-and-test search aspects.

In providing elements with these capabilities, we have been guided by the hypothesis of Klopf [11], [12] that neurons implement a strategy for attempting to maximize the frequency of occurrence of one type of input signal and minimize the frequency of occurrence of another type. According to this hypothesis, in other words, neurons can be conditioned in an operant or instrumental manner, where certain types of inputs act as rewarding stimuli and others act as punishing stimuli. A neuron learns how to attain certain types of inputs and avoid others by adjusting the transmission efficacy of its synapses according to the consequences of its discharges as fed back through pathways both internal to the nervous system and external to the animal. The ASE departs in several ways from Klopf's hypothesis, but his underlying idea remains the same.

III. Error Correction Versus Reinforcement Learning

Considerable misunderstanding is evident in the literature about how this type of "reinforcement learning" differs from supervised learning pattern classification as performed, for example, by Perceptrons and ADALINE's. It is important to emphasize these differences before we describe our adaptive elements. Supervised learning pattern classification elements are sometimes formulated in such a manner that the training process occurs as follows. A training pattern is presented to the element which responds as directed by its current set of weights; based on knowledge of the correct response, the element's environment feeds back an error signal giving the difference between the actual and correct resonses; the element uses this error signal to update its weight values. This sequence is repeated for all of the training patterns until the error signals become zero. These error signals are response-contingent feedback to the adaptive element, but it is misleading to view this process as a general form of reinforcement learning.

One important difference between the error-correction process just described and reinforcement learning as implemented by the ASE is that the latter does not rely exclusively on its weight values to determine its actions. Instead, it generates actions by a random process that is merely *biased* by the combination of its weight values and the input patterns. Actions are thus not appropriately viewed strictly as *responses* to input patterns. The random component of the generation process introduces the variety that is necessary to serve as the basis for subsequent selection by evaluative feedback. The ASE therefore searches in its action space in a manner that supervised learning pattern classification machines do not.

Additionally, significant differences exist between general performance evaluation signals and the signed error signals required by supervised learning pattern classification elements. To supply a signed error signal, the environment must know both what the actual action was and what it should have been.[1] Evaluation of performance, on the other hand, may be based on a relative assessment of certain consequences of the element's actions rather than on knowledge of both the correct and actual actions. Widrow *et al.* [13] used the phrase "learning with a critic" to distinguish this type of process from learning with a teacher, as supervised learning pattern classification is sometimes called.

Very few studies have been made of neuronlike elements capable of learning under reinforcement feedback that is less informative than are signed error signals (Farley and Clark [20]; Minsky [21]; and Widrow *et al.* [13]). Indeed, considerable confusion arises from an unfortunate inconsistency in the usage of the term "error." What psychologists mean by trial-and-error learning is not the same as the error-correction process used by supervised learning pattern classification machines. Like the process employed by our ASE, trial-and-error learning is a "selectional" rather than an "instructional" process (cf. the usage of these terms by Edelman [22], although the selectional mechanism of the ASE is quite different from the one he proposes). Much more could be said about these issues, but we shall let the following example further clarify them. It will be apparent that elements such as Perceptrons and ADALINE's cannot by themselves solve the control problem we will consider.

IV. The Credit Assignment Problem

One can view the uncertainty discussed in the foregoing as a result of a fundamental problem that faces any learning system, whether it is natural or artificial, that has been called the "credit-assignment" problem by artificial intelligence researchers [18], [23]. This is the problem of determining what parts of a complex interacting or interlocking set of mechanisms, decisions, or actions deserve credit (blame) for improvements (decrements) in the overall performance of the system. The credit-assignment problem is especially acute when evaluative feedback to the learning system occurs infrequently, for example, upon the completion of a long series of decisions or actions.

Given the widely acknowledged importance of the credit-assignment problem for adaptive problem-solving systems, it is surprising that techniques for its solution have not been more intensely studied. The most successful,

[1] It is thus possible to formulate this training paradigm as one in which the learning machine's environment provides training patterns together with their desired responses (as we have done in Section II), and the system itself determines its error. This formulation does not involve feedback that passes through the machine's environment and more clearly reveals the limited nature of this type of process.

and perhaps the most extensible, solution to date was used in the checkers-playing program written by Samuel [24] more than two decade ago. A few isolated studies using similar techniques have been undertaken (Doran [25]; Holland [26]; Minsky [21], [23]; and Witten [27]), but the current approaches to the credit-assignment problem in artificial intelligence largely rely on providing the critic with domain-specific knowledge [18], [27]. Samuel's method, on the other hand, is one by which the system improves its own internal critic by a learning process.

The ACE implements a strategy most closely related to the methods of Samuel [24] and Witten [27] for reducing the severity of the credit-assignment problem. It adaptively develops an evaluation function that is more informative than the one directly available from the learning system's environment. This reduces the uncertainty under which the ASE must learn. The ACE was developed primarily by Sutton as a refinement of the adaptive element model of classical conditioning introduced by Sutton and Barto [9].

V. A Learning Control Problem: Pole Balancing

Fig. 1 shows a schematic representation of a cart to which a rigid pole is hinged. The cart is free to move within the bounds of a one-dimensional track. The pole is free to move only in the vertical plane of the cart and track. The controller can apply an impulsive "left" or "right" force F of fixed magnitude to the cart at discrete time intervals. The cart–pole system was simulated by digital computer using a very detailed model that includes all of the nonlinearities and reactive forces of the physical system (the Appendix provides details of the cart–pole model and simulations). The cart–pole model has four state variables:

- x position of the cart on the track,
- θ angle of the pole with the vertical,
- $\dot{x}$ cart velocity, and
- $\dot{\theta}$ rate of change of the angle.

Parameters specify the pole length and mass, cart mass, coefficients of friction between the cart and the track and at the hinge between the pole and the cart, the impulsive control force magnitude, the force due to gravity, and the simulation time step size.

The control problem we pose is identical to the one studied by Michie and Chambers. We assume that the equations of motion of the cart–pole system are not known and that there is no preexisting controller that can be imitated. At each time step, the controller receives a vector giving the cart–pole system's state at that instant. If the pole falls or the cart hits the track boundary, the controller receives a failure signal, the cart–pole system (but not the controller's memory) is reset to its initial state, and another learning trial begins. The controller must attempt to generate controlling forces in order to avoid the failure signal for as long as possible. No evaluative feedback other than the failure signal is available.

Learning to avoid the failure signal under these constraints is a very different problem than learning to balance

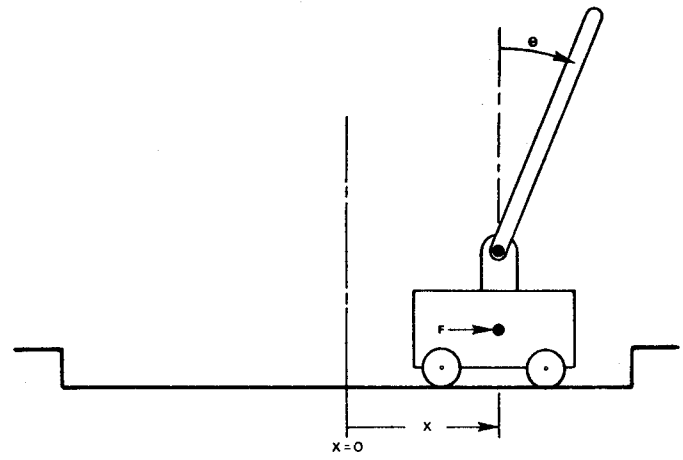

Fig. 1. Cart–pole system to be controlled. Solution of system's equations of motion approximated numerically (see Appendix).

the pole under the conditions usually assumed by control theorists. Since the failure signal will occur only after a long sequence of individual control decisions, a difficult credit-assignment problem arises in the attempt to determine which decisions were responsible for the failure. Neither a continuously available error signal nor a continuously available performance evaluation signal exists, as is the case in more conventional formulations of pole balancing. For example, Widrow and Smith [14] used a linear regression method, implemented by an ADALINE, to approximate the bang-bang control law required for balancing the pole. In order to use this method, however, they had to supply the controller with a signed error signal at each time step whose determination required external knowledge of the correct control decision for that time step. The present formulation of the problem, on the other hand, requires the learning system to discover for itself which control decisions are correct, and in so doing, solve a difficult credit-assignment problem that is completely absent in the usual versions of this problem.

VI. The Boxes System

By first describing Michie and Chambers' [3], [4] boxes system, we can provide much of the justification for the design of our adaptive elements. The strategy of these authors was to decompose the pole-balancing problem into a number of independent subproblems and to use an identical generate-and-test rule for learning to solve each subproblem. They divided the four-dimensional cart–pole state space into disjoint regions (or boxes) by quantizing the four state variables. They distinguished three grades of cart position, six of the pole angle, three of cart velocity, and three of pole angular velocity [4]. We use a similar partition of the state space based on the following quantization thresholds:

1) x: ± 0.8, ± 2.4 m,
2) θ: 0, ± 1, ± 6, $\pm 12°$,
3) $\dot{x}$: ± 0.5, $\pm \infty$ m/s,
4) $\dot{\theta}$: ± 50, $\pm \infty°/s$.

This yields $3 \times 3 \times 6 \times 3 = 162$ regions corresponding to all of the combinations of the intervals. The physical units of these thresholds differ from those used in [3] and [4]. We chose these values and units to produce what seemed like a physically realistic control problem, given our parameterization of the cart–pole simulation (Michie and Chambers did not publish the parameters of their cart–pole simulation. See the Appendix for our parameter values). At present we assume, as Michie and Chambers did, that this quantization is provided from the start (see Section X).

Each box is imagined to contain a *local demon* whose job is to choose a control action (left or right) whenever the system state enters its box. The local demon must learn to choose the action that will tend to be correlated with long system lifetime, that is, a long time until the occurrence of the failure signal. A *global demon* inspects the incoming state vector at each time step and alerts the local demon whose box contains that system state. When a failure signal is received, the global demon distributes it to all local demons. Each local demon maintains estimates of the expected lifetimes of the system following a left decision and following a right decision. A local demon's estimate of the expected lifetime for left is a weighted average of actual system lifetimes over all past occasions that the system state entered the demon's box and the decision left was made. The expected lifetime for the decision right is determined in the same way for occasions in which a right decision was made.

More specifically, upon being signaled by the global demon that the system state has entered its box, a local demon does the following.

1) It chooses the control action left or right according to which has the longest lifetime estimate. The control system emits the control action as soon as the decision is made.

2) It remembers which action was just taken and begins to count time steps.

3) When a failure signal is received, it uses its current count to update the left or right lifetime estimate, depending on which action it chose when its box was entered.

Michie and Chambers' actual algorithm is somewhat more complicated than this, but this description is sufficient for our present purposes. Details are provided in [3] where it is shown that the system is capable of learning to balance the pole for extended periods of time (in one reported run, the pole was balanced for a time approximately corresponding to one hour of real time). Notice that since the effect of a demon's decision will depend on the decisions made by other demons whose boxes are visited during a trial (where a trial is the time period from reset to failure), the environment of a local demon, consisting of the other demons as well as the cart–pole system, does not consistently evaluate the demon's actions.

VII. The Associative Search Element (ASE)

Obviously, many possibilities exist for implementing a system like boxes using neuronlike elements. We know, for example, that any algorithm can be implemented by a network of McCulloch–Pitts abstract neurons acting as logic gates and delay units. Such an implementation would illustrate the neural metaphor resulting from the very earliest contact between neuroscience and digital technology [29]. More recent neural metaphors suggest that each local demon might be implemented by a network of adaptive neurons that would be set into reverberatory activity under conditions corresponding to the demon's box being entered by the state vector. Upon receipt of the failure signal, the magnitude of this reverberatory activity would somehow alter synapses used for triggering control actions. The global demon might be implemented by a neural network responsible for quantizing the system state vectors, conjunctively combining the results, and activating appropriate local demon networks (a neural decoder—see Section X). Finally, an element or network of elements would be required for channeling the action of each local demon network to a common efferent pathway.

In the neuronlike implementation we are pursuing, however, a local demon corresponds to the mechanism of a single synapse (to use the language of neural metaphor), and the output pathway of the postsynaptic element (the ASE) provides the common efferent pathway for control signals. At each synapse of the ASE are both a long-term memory trace that determines control actions and a short-term memory trace that is required to update the long-term trace, a role similar to that of a local demon's counter in the boxes algorithm. To accomplish the global demon's job of activating the appropriate local demon, we assume the existence of a decoder that has four real-valued input pathways (for the system state vector) and 162 binary valued output pathways corresponding to the boxes of Michie and Chambers' system (Fig. 2). The decoder transforms each state vector into a 162-component binary vector whose components are all zeros except for a single one in the position corresponding to the box containing the state vector. This vector is provided as input to the ASE and effectively selects the synapse corresponding to the appropriate box. For the other job of the global demon, that of distributing a failure signal to all of the local demons, we just let the adaptive element receive the failure signal via its reinforcement pathway and distribute the information to all of of its afferent synapses. In this way the entire boxes algorithm can be implemented by a single neuronlike ASE and an appropriate decoder.

In more detail, an ASE is defined as follows. The element has a reinforcement input pathway, n pathways for nonreinforcement input, and a single output pathway (Fig. 2). Let $x_i(t), 1 \leqslant i \leqslant n$, denote the real-valued signal on the ith nonreinforcement input pathway at time t, and let $y(t)$ denote the output at time t. Associated with each nonreinforcement input pathway i is a real-valued weight with value at time t denoted by $w_i(t)$.

The element's output $y(t)$ is determined from the input vector $X(t) = (x_1(t), \cdots, x_n(t))$ as follows:

$$y(t) = f\left[\sum_{i=1}^{n} w_i(t)x_i(t) + \text{noise}\,(t)\right] \qquad (1)$$

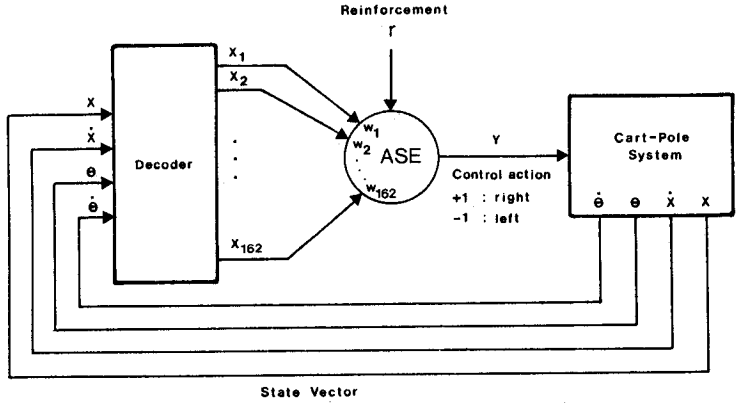

Fig. 2. ASE controller for cart–pole system. ASE's input is determined from current cart–pole state vector by decoder that produces output vector consisting of zeros with single one indicating which of 162 boxes contains state vector. ASE's output determines force applied to cart. Reinforcement is constant throughout trial and becomes -1 to signal failure.

where noise (t) is a real random variable with probability density function d and f is either a threshold, sigmoid, or identity function. For the pole-balancing illustration, d is the mean zero Gaussian distribution with variance σ^2, and f is the following threshold function:

$$f(x) = \begin{cases} +1, & \text{if } x \geqslant 0 \quad \text{(control action right)} \\ -1, & \text{if } x < 0 \quad \text{(control action left)}. \end{cases}$$

This follows the usual linear threshold convention common in adaptive network studies, but our approach does not depend strongly on the specifics of the input/output function of the element.

According to (1), actions are emitted even in the absence of nonzero input signals. The element's output is determined by chance, with a probability biased by the weighted sum of the input signals. If that sum is zero, the left and right control actions are equally probable. Assuming the decoder input shown in Fig. 2, a positive weight w_i, for example, would make the decision right more probable than left when box i is entered by the system state vector. The value of a weight, therefore, plays a role corresponding to the difference between the expected lifetimes for the left and right actions stored by a local demon in the boxes system. However, unlike the boxes system, the weight only determines the probability of an action rather than the action itself. The learning process updates the action probabilities. Also note that an input vector need not be of the restricted form produced by the decoder in order for (1) and the equations that follow to the meaningful.

The weights $w_i, 1 \leqslant i \leqslant n$, change over (discrete) time as follows:

$$w_i(t+1) = w_i(t) + \alpha r(t) e_i(t) \qquad (2)$$

where

- α positive constant determining the rate of change of w_i,
- $r(t)$ real-valued *reinforcement* at time t, and
- $e_i(t)$ *eligibility* at time t of input pathway i.

The basic idea expressed by (2) is that whenever certain conditions (to be discussed later) hold for input pathway i, then that pathway becomes eligible to have its weight modified, and it remains eligible for some period of time after the conditions cease to hold. How w_i changes depends on the reinforcement received during periods of eligibility. If the reinforcement indicates improved performance, then the weights of the eligible pathways are changed so as to make the element more likely to do whatever it did that made those pathways eligible. If reinforcement indicates decreased performance, then the weights of the eligible pathways are changed to make the element more likely to do something else. The term "eligibility" and this weight update scheme are derived from the theory of Klopf [11], [12] and have precursors in the work of Farley and Clark [20], Minsky [21], and others. This general approach to reinforcement learning is related to the theory of stochastic learning automata [30], [31], which has its roots in the work of Bush and Mosteller [32] and Tsetlin [33].

Reinforcement: Positive r indicates the occurrence of a rewarding event and negative r indicates the occurrence of a punishing event.[2] It can be regarded as a measure of the *change* in the value of a performance criterion as commonly used in control theory. For the pole-balancing problem, r remains zero throughout a trial and becomes -1 when failure occurs.

Eligibility: Klopf [11] proposed that a pathway should reach maximum eligibility a short time after the occurrence of a pairing of a nonzero input signal on that pathway with the "firing" of the element. Eligibility should decay thereafter toward zero. Thus, when the consequences of the element's firing are fed back to the element, credit or blame can be assigned to the weights that will alter the firing probability when a similar input pattern occurs in the future. More generally, the eligibility of a pathway reflects the extent to which input activity on that pathway was paired in the past with element output activity. The eligibility of pathway i at time t is therefore a *trace* of the product $y(\tau)x_i(\tau)$ for times τ preceding t. If either or both of the quantities $y(\tau)$ and $x_i(\tau)$ are negative (as they can be for the ASE defined earlier), then credit is assigned

[2]A negative value of r is not the same as a psychologists's "negative reinforcement." In psychology, negative reinforcement is reinforcement due to the cessation of an aversive stimulus.

appropriately via (2) if eligibility is a trace of the signed product $y(\tau)x_i(\tau)$.

For computational simplicity, we generate exponentially decaying eligibility traces e_i using the following linear difference equation:

$$e_i(t+1) = \delta e_i(t) + (1-\delta)y(t)x_i(t), \qquad (3)$$

where $\delta, 0 \leqslant \delta < 1$, determines the trace decay rate. Note that each synapse has its own local eligibility trace.

Eligibility plays a role analogous to the part of the boxes local-demon algorithm that, when the demon's box is entered and an action has been chosen, remembers what action was chosen and begins to count. The factor $x_i(t)$ in (3) triggers the eligibility trace, a kind of count, or contributes to an ongoing trace, whenever box i is entered ($x_i(t) = 1$). Instead of explicitly remembering what action was chosen, our system contributes a different amount to the eligibility trace depending on what action was chosen (via the term $y(t)$ in (3)). Thus the trace contains information not only about how long ago a box was entered but also about what decision was made when it was entered.

Unlike the count initiated by a local demon in the boxes system, however, the eligibility trace effectively counts down rather than up (more precisely, its magnitude decays toward zero). Recall that reinforcement r remains zero until a failure occurs, at which time it becomes -1. Thus whatever control decision was made when a box was visited will always be made *less* likely when the failure occurs, but the longer the time interval between the decision and the occurrence of the failure signal, the less this decrease in probability will be. From one perspective, this process seems appropriate. Since the failure signal always eventually occurs, the action that was taken may deserve some of the blame for the failure. However, this view misses the point that even though both actions inevitably lead to failure, one action is probably *better* than the other. The learning process defined by (1)–(3) needs to be more subtle to ensure convergence to the actions that yield the least punishment in cases in which only punishment is available. In the present article, we build this subtlety into the ACE rather than into the ASE. Among its other functions, the ACE constructs predictions of reinforcement so that if punishment is less than its expected level, it acts as reward. For the pole-balancing task, the ASE as defined here must operate in conjunction with the ACE.

Although the boxes system and the version of the pole-balancing problem described earlier serve well to make an ASE's design understandable, the ASE does not represent an attempt to duplicate the boxes algorithm in neuronlike form. We are interested in tasks more general than the pole-balancing problem and in learning systems that are more general than the boxes system. An ASE is less restricted than the boxes system in several ways. First, the boxes system is based on the subdivision of the problem space into a finite number of nonoverlapping regions, and no generalization is attempted between regions. It develops a control rule that is effectively specified by means of a lookup table. Although a form of generalization can be easily added to the boxes algorithm by using an averaging process over neighboring boxes (see Section X) it is not immediately obvious how to extend the algorithm to take advantage of the other forms of generalization that would be possible if the controlled system's states could be represented by arbitrary vectors rather than only by the standard unit basis vectors which are produced by a suitable decoder. The ASE can accept arbitrary input vectors and, although we do not illustrate it in this article, can be regarded as a step toward extending the type of generalization produced by error-correction supervised learning pattern classification methods to the less restricted reinforcement learning paradigm (see Section III).

The boxes system is also restricted in that its design was based on the *a priori* knowledge that the time until failure was to serve as the evaluation criterion and that the learning process would be divided into distinct trials that would always end with a failure signal. This knowledge permitted Michie and Chambers to reduce the uncertainty in the problem by restricting each local demon to choosing the same action each time its box was entered during any given trial. The ASE, on the other hand, is capable of working to achieve rewarding events and to avoid punishing events which might occur at any time. It is not exclusively failure-driven, and its operation is specified without reference to the notion of a trial.

VIII. The Adaptive Critic Element (ACE)

Fig. 3 shows an ASE together with an ACE configured for the pole-balancing task. The ACE receives the externally supplied reinforcement signal which it uses to determine how to compute, on the basis of the current cart–pole state vector, an improved reinforcement signal that it sends to the ASE. Expressed in terms of the boxes system, the job of the ACE is to store in each box a prediction or expectation of the reinforcement that can eventually be obtained from the environment by choosing an action for that box. The ACE uses this prediction to determine a reinforcement signal that it delivers to the ASE whenever the box is entered by the cart–pole state, thus permitting learning to occur throughout the pole-balancing trials rather than solely upon failure. This greatly decreases the uncertainty faced by the ASE. The central idea behind the ACE algorithm is that predictions are formed that predict not just reinforcement but also future predictions of reinforcement.

Like the ASE, the ACE has a reinforcement input pathway, n pathways for nonreinforcement input, and a single output pathway (Fig. 3). Let $r(t)$ denote the real-valued reinforcement at time t; let $x_i(t), 1 \leqslant i \leqslant n$, denote the real-valued signal on the ith nonreinforcement input pathway at time t; and let $\hat{r}(t)$ denote the real-valued output signal at time t. Each nonreinforcement input pathway i has a weight with real value $v_i(t)$ at time t. The output $\hat{r}$ is the improved reinforcement signal that is used by the ASE in place of r in (2).

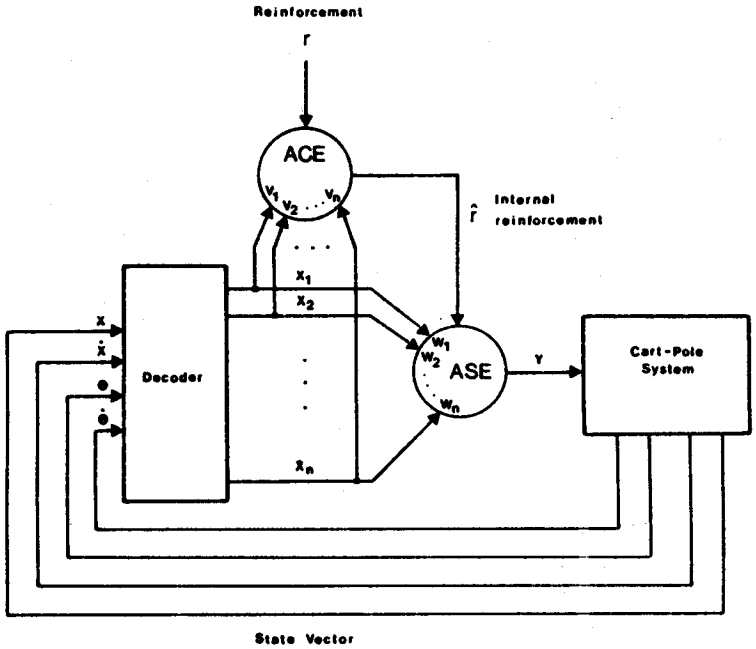

Fig. 3. ASE and ACE configured for pole-balancing task. ACE receives same nonreinforcing input as ASE and uses it to compute an improved or internal reinforcement signal to be used by ASE.

In order to produce $\hat{r}(t)$, the ACE must determine a prediction $p(t)$ of eventual reinforcement that is a function of the input vector $X(t)$ (which in the boxes paradigm, simply selects a box). We let

$$p(t) = \sum_{i=1}^{n} v_i(t)x_i(t) \tag{4}$$

and seek a means of updating the weights v_i so that $p(t)$ converges to an accurate prediction. The updating rule we use is

$$v_i(t+1) = v_i(t) + \beta[r(t) + \gamma p(t) - p(t-1)]\,\bar{x}_i(t), \tag{5}$$

where β is a positive constant determining the rate of change of v_i; $\gamma, 0 < \gamma \leqslant 1$, is a constant to be explained below; $r(t)$ is the reinforcement signal supplied by the environment at time t; and $\bar{x}_i(t)$ is the value at time t of a *trace* of the input variable x_i.

It is beyond the scope of the present paper to explain the derivation of this learning rule fully (see [7] and [9]). Very briefly, the trace $\bar{x}_i$ acts much like the eligibility trace e_i defined by (3). Here, however, an input pathway gains positive eligibility whenever a nonzero signal is present on that pathway, irrespective of what the element's action is. We compute $\bar{x}_i$ using the following linear difference equation (cf. (3)):

$$\bar{x}_i(t+1) = \lambda\bar{x}_i(t) + (1-\lambda)x_i(t), \tag{6}$$

where $\lambda, 0 \leqslant \lambda < 1$ determines the trace decay rate.

According to (6), an eligible pathways's weight changes whenever the actual reinforcement $r(t)$ plus the current prediction $p(t)$ differs from the value $p(t-1)$ that was predicted for this sum. Closely related to the ADALINE learning rule and related regression techniques, this rule provides a means of finding weight values such that $p(t-1)$ approximates $r(t) + \gamma p(t)$, or, equivalently, such that $p(t)$ approximates $r(t+1) + \gamma p(t+1)$. By attempting to predict its own prediction, the learning rule produces predictions that tend to be the earliest possible indictions of eventual reinforcement. The constant γ, related to Witten's [27] "discount factor," provides for eventual extinction of predictions in the absence of external reinforcement. If $\gamma = 1$, predictions will be self-sustaining in the absence of external reinforcement; whereas if $0 < \gamma < 1$, predictions will decay in the absence of external reinforcement. In our simulations, $\gamma = 0.95$.

The ACE's output, the improved or internal reinforcement signal, is computed from these predictions as follows:

$$\hat{r}(t) = r(t) + \gamma p(t) - p(t-1). \tag{7}$$

This is the same expression appearing in (5). The reader should note that with $\hat{r}$ substituted for r in (2), the weight updating rules for the ASE and ACE ((2) and (5), respectively) differ only in their forms of eligibility traces. The ASE's traces are conditional on its output, whereas the ACE's are not.

Although this process works for arbitrary input vectors, it is easiest to justify (7) by again specializing to the boxes input representation. According to (7), as the cart–pole state moves between boxes without failure occurring (i.e., $r(t) = 0$), the reinforcement $\hat{r}(t)$ sent to the ASE is the difference between the prediction of reinforcement of the current box (discounted by γ) and the prediction of reinforcement of the previous box. Increases in reinforcement prediction therefore become rewarding events (assuming $\gamma = 1$), and decreases become penalizing events.

When failure occurs, the situation is slightly different. Given the way the control problem is represented, when failure occurs the cart–pole state is not in any box. Thus all $x_i(t)$ are equal to zero at failure, and according to (4), so is $p(t)$. Upon failure, then, the reinforcement sent to the ASE is the externally supplied reinforcement $r(t) = -1$, minus the previous prediction $p(t-1)$. Consequently, an unpredicted failure results in $\hat{r}(t)$ being negative. This both

punishes the actions made preceding the failure and, via (5), increments the predictions of failure (i.e., decrements the p's) of the boxes entered before the failure. A fully predicted failure generates no punishment. However, when a box with such a high prediction of failure is entered from a box with a lower prediction of failure, the recently made actions are punished and the recent predictions of failure are incremented, just as they were initially upon failure. Similarly, if the cart–pole state moves from a box with a higher prediction of failure to a box with a lower prediction, the recent actions are rewarded and recent predictions of failure are decremented (i.e., the p's are incremented). The system thus learns which boxes are "safe" and which are "dangerous." It punishes itself for moving from any box to a more dangerous box and rewards itself for moving from any box to a safer box. In the following we discuss the relation between the behavior of the ACE and that of animals in classical conditioning experiments.

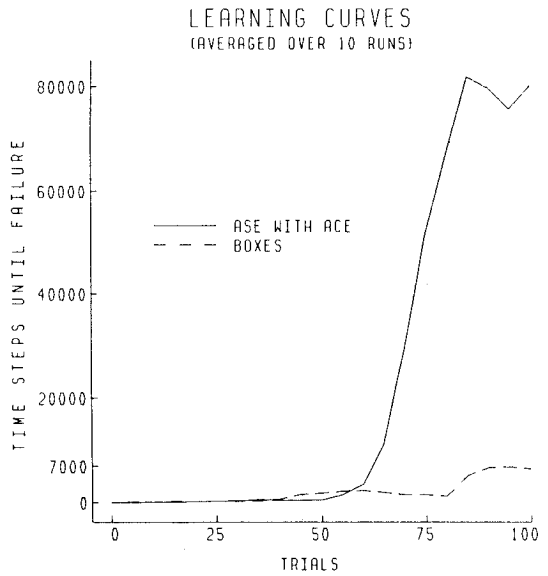

Fig. 4. Simulation results. Performance of boxes system and ASE/ACE system averaged over ten runs. See text for complete explanation.

IX. Simulation Results

We implemented the boxes system as described in [3] and [4] as well as our systems shown in Fig. 2 (ASE alone) and Fig. 3 (ASE with ACE). We wanted to determine what kinds of neuronlike elements could attain or exceed the performance of the boxes system. Our results suggest that a system using an ASE with internal reinforcement supplied by an ACE is easily able to outperform the boxes system. We must emphasize at the outset, however, that it is not our intention to criticize Michie and Chambers' program: the boxes system they described was in an initial state of development and clearly could be extended to include a mechanism analogous to our ACE. We make comparisons with the performance of the boxes system because it provides a convenient reference point.

We simulated a series of runs of each learning system attempting to control the same cart–pole simulation (see the Appendix). Each run consisted of a sequence of trials, where each trial began with the cart–pole state $x = 0$, $\dot{x} = 0$, $\theta = 0$, $\dot{\theta} = 0$, and ended with a failure signal indicating that θ left the interval $[-12°, 12°]$ or x left the interval $[-2.4$ m, 2.4 m$]$. We also set all the trace variables e_i to zero at the start of each trial. The learning systems were "naive" at the start of each run (i.e., all the weights w_i and v_i were set to zero). At the start of each boxes run, we supplied a different seed value to the pseudorandom number generator that we used to initialize the state of the learning system and to break ties in comparing expected lifetimes in order to choose control actions. We did not reset the cart–pole state to a randomly chosen state at the start of each trial as was done in the experiments reported in [3] and [4]. At the start of each run of an ASE system, we supplied a different seed to the pseudorandom number generator that we used to generate the noise used in (1). We approximated this Gaussian random variable by the usual procedure of summing uniformly distributed random variables (we used an eightfold sum). Since the ASE runs began with weight vectors equal to zero, initial actions for each box were equiprobable, and initial ACE predictions were zero. Except for the random number generator seeds, identical parameter values were used for all runs. Runs consisted of 100 trials unless the run's duration exceeded 500 000 time steps (approximately 2.8 h of simulated real time), in which case the run was terminated. For our implementation of the boxes system, we used the parameter values published in [3]. We experimented with other parameter values without obtaining consistently better performance. We did not attempt to optimize the performance of the systems using the ASE. We picked values that seemed reasonable based on our previous experience with similar adaptive elements.

Figs. 4 and 5 show the results of our simulations of boxes and the ASE/ACE system. The graphs of Fig. 4 are averages of performance over the ten runs that produced the individual graphs shown in Fig. 5. In both figures, a single point is plotted for each bin of five trials giving the number of time steps until failure averaged over those five trials. Almost all runs of the ASE/ACE system, and one run of the boxes system, were terminated after 500 000 time steps before all 100 trials took place (those whose graphs terminate short of 100 trials in Fig. 5). We stopped the simulation before failure on the last trials of these runs. To produce the averages for all 100 trials shown in Fig. 4, we needed to make special provision for the interrupted runs. If the duration of the trial that was underway when the run was interrupted was less than the duration of the immediately preceding (and therefore complete) trial, then we assigned to fictitious remaining trials the duration of that preceding trial. Otherwise, we assigned to fictitious remaining trials the duration of the last trial when it was interrupted. We did this to prevent any short interrupted trials from producing deceptively low averages.

The ASE/ACE system achieved much longer runs than did the boxes system. Fig. 5 shows that the ACE/ASE system tended to solve the problem before it had experienced 100 failures, whereas the boxes system tended not to. Obviously, we cannot make definitive statements about the relative performance of these systems, or about the general utility of the ASE/ACE system solely on the basis of these

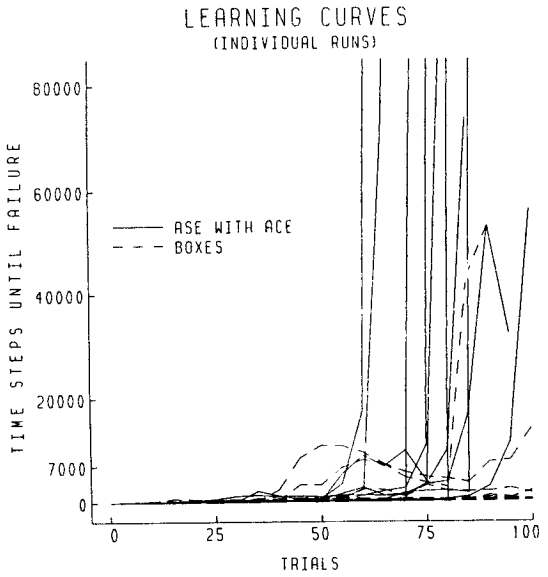

Fig. 5. Simulation results. Performance of boxes system and ASE/ACE system in individual runs that were averaged to produce Fig. 4. See text for complete explanation.

experiments. However, these results encourage us to continue developing the principles upon which the ASE/ACE system is based.

The parameter values used in producing these results were $\alpha = 1000$, $\beta = 0.5$, $\delta = 0.9$, $\gamma = 0.95$, $\lambda = 0.8$, and $\sigma = 0.01$. Except for one set of extreme values, these were the only values we tried. The large value of α was chosen so that large changes in the weights w_i occurred upon reinforcement. This caused the probability of a rewarded action to become nearly one and the probability of a penalized action to become nearly zero. We did this in an attempt to implement in our system the feature of the boxes system that causes each local demon to choose the same action each time its box is entered in any given trial. This greatly reduces the uncertainty in the problem but would be inappropriate, we think, for problems in which other reinforcing events could occur during trials. The parameters δ and λ determine the durations of the eligibility traces. Their values, 0.9 and 0.8, respectively, cause long, slowly decaying traces to form, as seemed appropriate given the nature of the problem.

The good performance of the ASE/ACE system was almost entirely due to the ACE's supplying reinforcement throughout trials. For the boxes system and for an ASE without an ACE, learning occurs only upon failure, an event that becomes less frequent as learning proceeds. With the ACE in place, an ASE can receive feedback on every time step. The learning produced by this feedback causes the system to attempt to enter particular parts of the state space and to avoid others. We simulated the control problem using an ASE without an ACE, using the same parameter settings that worked well for the ASE/ACE experiments. The ASE was not able to attain the level of performance shown by the boxes system. These shortcomings of the ASE are due to difficulties in the convergence process in tasks involving only penalizing feedback, as discussed in Section VII. The use of reinforcement computed by the ACE markedly changes this property of the pole-balancing problem. At present, we have little experience with ASE-like elements operating without ACE supplied reinforcement in the pole-balancing problem.

X. The Decoder

We have assumed the existence of a decoder that effectively divides the cart–pole state space into a number of disjoint regions by transforming each state vector into a vector having 162 components, all but one of which is zero. We call this a decoder after a similar device used in computer memory circuits to transform each memory address into a signal on the wire connected to the physical location having that address. With this decoder providing its input, the ASE essentially fills in a lookup table of control actions. Similarly, the ACE fills in a table of reinforcement predictions. Each item of information is stored by the setting of the value of a single synaptic weight at a given location.

As a consequence of this localized storage scheme, no generalization occurs beyond the confines of a given box. Given the relative smoothness of the cart–pole dynamics, learning would be faster if information stored in a box could be extrapolated to neighboring boxes (using the Euclidean metric). This can be accomplished by using a kind of decoder that produces activity on overlapping sets of output pathways. It is interesting to note that in several theories of sensorimotor learning, it is postulated that the granular layer of the cerebellum implements just this kind of decoder and that Purkinje cells are adaptive elements [34], [35].[3]

Localized extrapolation is not the only type of generalization that can be useful. There has lately been increasing interest in "associative memory networks" that use distributed representations in which dispersed rather than localized patterns of activity encode information [36], [37]. Rather than implementing table-lookup storage, associative memories use weighted summations to compute output vectors from input vectors. This style of information storage provides generalization among patterns according to where they lie with respect to a set of linear discriminant functions. Since the ASE and ACE use weighted summations that are defined for arbitrary input vectors, they implement linear discriminant functions and are capable of forming information storage networks having all of the properties that have generated interest in associative memory networks. Unlike the associative memory networks discussed in the literature, however, networks of ASE-like components are capable of discovering via reinforcement learning what information is useful to store. These aspects

[3]In these theories, the adaptive elements perform supervised learning pattern classification, with climbing fiber input providing the desired responses, and not the type of reinforcement learning with which the present article is concerned. If the adaptive capacity of a Purkinje cell were limited to that postulated in these theories, then a Purkinje cell would not be able to solve the type of problem illustrated by the pole-balancing task described in this article.

of ASE's are emphasized by Barto *et al.* [8] where *associative search* networks are discussed.

Whether environmental states are represented using localized or distributed patterns, the problem remains of how to choose the specifics of the representations in order to facilitate learning. In this article we followed Michie and Chambers in choosing a state–space partition based on special knowledge of the control task. As they point out, it is easy to choose a partition that makes the task impossible [4]. For the next stage of development of the boxes system, Michie and Chambers planned to give the system the ability to change the boundaries of the boxes by the processes of "splitting" and "lumping" [4]. We are not aware of any results they published on these processes, but we were motivated in part by their comments to experiment with layered networks of ASE-like adaptive elements in order to examine the feasibility of implementing a kind of adaptive decoder. Some preliminary results, reported in [5], were encouraging, and we are continuing our investigation in this direction.

XI. Animal Learning

Minsky has pointed out [23] that methods for reducing the severity of the credit-assignment problem like the one used in Samuel's checkers player are suggestive of secondary or conditioned reinforcement phenomena in animal learning studies. A stimulus acquires reinforcing qualities (i.e., becomes a secondary reinforcer) if it predicts either primary reinforcement (e.g., food or shock) or some other secondary reinforcer. It is generally held that higher order classical conditioning, whereby previously conditioned conditioned stimuli (CS's) can act as unconditioned stimuli (US's) for earlier potential CS's, is the basis for the development of secondary reinforcement [38].

The ACE is a refinement of the model of classical conditioning that was presented in [9]. That model's behavior is consistent with the Rescorla–Wagner model of classical conditioning [39]. While not without certain problems, the Rescorla–Wagner model has been the most influential model of classical conditioning for the last ten years [40]. One interpretation of the basic premise of the Rescorla–Wagner model is that the degree to which an event is "unexpected" or "surprising" determines the degree to which it enters into associations with earlier events. Stimuli lose their reinforcing qualities to the extent that they are expected on the basis of the occurrence of earlier stimuli. The model upon which the ACE is based extends the basic mechanism of the Rescorla–Wagner model to provide for some of the features of higher order conditioning, the influence of relative event timing within trials, and the occurrence of conditioned responses (CR's) that anticipate the US. In these terms, the failure signal r corresponds to the US, the signals x_i from the decoder correspond to potential CS's, and the prediction p corresponds to a component of the CR. We have not yet thoroughly investigated the extent to which the ACE/ASE system is a valid model of animal behavior in instrumental conditioning experiments.

XII. Conclusion

It should be clear that our approach differs from that of the pioneering adaptive neural-network theorists of the 1950's and 1960's. We have built into single neuronlike adaptive elements a problem-solving capacity that in many respects exceeds that achieved in the past by entire simulated neural networks. The metaphor for neural function suggested by this approach provides, at least to us, the first convincing inkling of how nervous tissue could possibly be capable of its exquisite feats of problem solving and control.

We argued that components of powerful adaptive networks must be at least as sophisticated as the components described in this article. If this were true for biological networks as well as for artificial networks, then it would suggest that parallels might exist between neurons and the adaptive elements described here. It would suggest, for example, that 1) there are single neuron analogs of instrumental conditioning and chemically specialized reinforcing neurons that may themselves be adaptive (see [41]); 2) the random component of an instrumental neuron's behavior is necessary for generating variety to serve as the basis for subsequent selection; and 3) mechanisms exist for maintaining relatively long-lasting synaptically local traces of activity that modulate changes in synaptic efficacy. Although some of these implications are supported in varying degrees by existing data, there are no data that provide direct support for the existence of the specific mechanisms used in our adaptive elements. By showing how neuronlike elements can solve genuinely difficult problems that are solved routinely by many animals, we hope to stimulate interest in the relevant experimental research.

Appendix
Details of the Cart–Pole Simulation

The cart–pole system is modeled by the following nonlinear differential equations (see [42]):

$$\ddot{\theta}_t = \frac{g \sin\theta_t + \cos\theta_t \left[\dfrac{-F_t - ml\dot{\theta}_t^2 \sin\theta_t + \mu_c \operatorname{sgn}(\dot{x}_t)}{m_c + m} \right] - \dfrac{\mu_p \dot{\theta}_t}{ml}}{l\left[\dfrac{4}{3} - \dfrac{m\cos^2\theta_t}{m_c + m} \right]}$$

$$\ddot{x}_t = \frac{F_t + ml\left[\dot{\theta}_t^2 \sin\theta_t - \ddot{\theta}_t \cos\theta_t\right] - \mu_c \operatorname{sgn}(\dot{x}_t)}{m_c + m}$$

where

$g = -9.8$ m/s^2, acceleration due to gravity,
$m_c = 1.0$ kg, mass of cart
$m = 0.1$ kg, mass of pole,
$l = 0.5$ m, half-pole length,
$\mu_c = 0.0005$, coefficient of friction of cart on track,
$\mu_p = 0.000002$, coefficient of friction of pole on cart,
$F_t = \pm 10.0$ newtons, force applied to cart's center of mass at time t.

We initially used the Adams–Moulton predictor–corrector method to approximate numerically the solution of these equations, but the results reported in this article were produced using Euler's method with a time step of 0.02 s for the sake of computational speed. Comparisons of solutions generated by the Adams–Moulton methods and the less accurate Euler method did not reveal discrepencies that we deemed significant for the purposes of this article.

Acknowledgment

We thank A. H. Klopf for bringing to us a set of ideas filled with possibilities. We are grateful also to D. N. Spinelli, M. A. Arbib, and S. Epstein for their valuable comments and criticisms; to D. Lawton for first making us aware of Michie and Chambers' boxes system; to D. Politis and W. Licata for pointing out an important error in our original cart–pole simulations; and to S. Parker for essential help in preparing the manuscript.

References

[1] B. Chandrasekaran, "Natural and social system metaphors for distributed problem solving: Introduction to the issue," *IEEE Trans. Syst., Man., Cybern.*, vol. SMC-11, pp. 1–5, 1981.

[2] V. R. Lesser and D. D. Corkill, "Functionally-accurate, cooperative distributed systems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, pp. 81–96, 1981.

[3] D. Michie and R. A. Chambers, "BOXES: An experiment in adaptive control," in *Machine Intelligence 2*, E. Dale and D. Michie, Eds. Edinburgh: Oliver and Boyd, 1968, pp. 137–152.

[4] D. Michie and R. A. Chambers, "'Boxes' as a model of pattern-formation," in *Towards a Theoretical Biology*, vol. 1, *Prolegomena*, C. H. Waddington, Ed. Edinburgh: Edinburgh Univ. Press, 1968, pp. 206–215.

[5] A. G. Barto, C. W. Anderson, and R. S. Sutton, "Synthesis of nonlinear control surfaces by a layered associative search network," *Biol. Cybern.*, vol. 43, pp. 175–185, 1982.

[6] A. G. Barto and R. S. Sutton, "Landmark learning: An illustration of associative search," *Biol. Cybern.*, vol. 42, pp. 1–8, 1981.

[7] ———, "Simulation of anticipatory responses in classical conditioning by a neuron-like adaptive element," *Behavioral Brain Res.*, vol. 4, pp. 221–235, 1982.

[8] A. G. Barto, R. S. Sutton, and P. S. Brouwer, "Associative search network: A reinforcement learning associative memory," *Biol. Cybern.*, vol. 40, pp. 201–211, 1981.

[9] R. S. Sutton and A. G. Barto, "Toward a modern theory of adaptive networks: Expectation and prediction," *Psychol. Rev.*, vol. 88, pp. 135–171, 1981.

[10] ———, "An adaptive network that constructs and uses an internal model of its world," *Cognition and Brain Theory*, vol. 4, pp. 213–246, 1981.

[11] A. H. Klopf, "Brain function and adaptive systems—A heterostatic theory," Air Force Cambridge Res. Lab. Res. Rep., AFCRL-72-0164, Bedford, MA, 1972. (A summary appears in *Proc. Int. Conf. Syst., Man, Cybern.*, 1974).

[12] A. H. Klopf, *The Hedonistic Neuron: A Theory of Memory, Learning, and Intelligence.* Washington, DC: Hemisphere, 1982.

[13] B. Widrow, N. K. Gupta, and S. Maitra, "Punish/reward: learning with a critic in adaptive threshold systems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 455–465, 1973.

[14] B. Widrow and F. W. Smith, "Pattern-recognizing control systems," in *Computer and Information Sciences*, J. T. Tow and R. H. Wilcox, Eds. Clever Hume Press, 1964, pp. 288–317.

[15] F. Rosenblatt, *Principles of Neurodynamics.* New York: Spartan, 1962.

[16] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *1960 WESCON Conv. Record*, part IV, 1960, pp. 96–104.

[17] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley, 1973.

[18] P. R. Cohen and E. A. Feigenbaum, *The Handbook of Artificial Intelligence*, vol. 3. Los Altos, CA: Kauffman, 1982.

[19] D. O. Hebb, *Organization of Behavior.* New York: Wiley, 1949.

[20] B. G. Farley and W. A. Clark, "Simulation of self-organizing systems by digital computer," *IRE Trans. Inform. Theory*, vol. PGIT-4, pp. 76–84, 1954

[21] M. L. Minsky, "Theory of neural-analog reinforcement systems and its application to the brain-model problem," Ph.D. dissertation, Princeton Univ., Princeton, NJ, 1954.

[22] G. M. Edelman, "Group selection and phasic reentrant signaling: A theory of higher brain function," in *The Mindful Brain: Cortical Organization and the Group-Selective Theory of Higher Brain Function*, G. M. Edelman and V. B. Mountcastle, Eds. Cambridge, MA: MIT Press, 1978.

[23] M. L. Minsky, "Steps toward artificial intelligence," *Proc. IRE*, vol. 49, pp. 8–30, 1961.

[24] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM J. Res. Develop.*, vol. 3, pp. 210–229, 1959.

[25] J. Doran, "An approach to automatic problem solving," in *Machine Intelligence*, vol. 1, E. L. Collins and D. Michie, Eds. Edinburgh: Oliver and Boyd, 1967, pp. 105–123.

[26] J. H. Holland, "Adaptive algorithms for discovering and using general patterns in growing knowledge-bases," *Int. J. Policy Anal. Inform. Syst.*, vol. 4, pp. 217–240, 1980.

[27] I. H. Witten, "An adaptive optimal controller for discrete-time Markov environments," *Inform. Contr.*, vol. 34, pp. 286–295, 1977.

[28] T. D. Dietterich and B. G. Buchanan, "The role of the critic in learning systems," Stanford Univ. Tech. Rep., STAN-CS-81-891, 1981.

[29] W. S. McCulloch and W. H. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, pp. 115–133, 1943.

[30] K. S. Narendra and M. A. L. Thatachar, "Learning automata—A survey," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-4, pp. 323–334, 1974.

[31] S. Lakshmivarahan, *Learning Algorithms Theory and Applications.* New York: Springer-Verlag, 1981.

[32] R. R. Bush and F. Mosteller, *Stochastic Models for Learning.* New York: Wiley, 1958.

[33] M. L. Tsetlin, *Automaton Theory and Modelling of Biological Systems.* New York: Academic, 1973.

[34] J. A. Albus, *Brains, Behavior, and Robotics.* Peterborough, NH: BYTE Books, 1981.

[35] D. Marr, "A theory of cerebellar cortex," *J. Physiol.*, vol. 202, pp. 437–470, 1969.

[36] G. E. Hinton and J. A. Anderson, Eds., *Parallel Models of Associative Memory.* Hillsdale, NJ: Erlbaum, 1981.

[37] T. Kohonen, *Associative Memory: A System Theoretic Approach.* Berlin, Germany: Springer, 1977.

[38] R. A. Rescorla, *Pavlovian Second-Order Conditioning: Studies in Associative Learning.* Hillsdale, NJ: Erlbaum, 1980.

[39] R. A. Rescorla and A. R. Wagner, "A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement," in *Classical Conditioning II: Current Research and Theory*, A. H. Black and W. R. Prokasy, Eds. New York: Appleton-Century-Crofts, 1972.

[40] A. Dickinson, *Contemporary Animal Learning Theory.* Cambridge: Cambridge Univ. Press, 1980.

[41] L. Stein and J. D. Belluzzi, "Beyond the reflex arc: A neuronal model of operant conditioning," in *Changing Concepts of the Nervous System.* New York: Academic, 1982.

[42] R. H. Cannon, *Dynamics of Physical Systems.* New York: McGraw-Hill, 1967.