Operant Conditioning in Skinnerbots

David S. Touretzky
Computer Science Department &
Center for the Neural Basis of Cognition
Carnegie Mellon University
Pittsburgh, PA 15213-3891
dst@cs.cmu.edu

Lisa M. Saksida
Robotics Institute &
Center for the Neural Basis of Cognition
Carnegie Mellon University
Pittsburgh, PA 15213-3891
saksida@ri.cmu.edu

To appear in Adaptive Behavior **5**(3/4), 1997.

Abstract

Instrumental (or operant) conditioning, a form of animal learning, is similar to reinforcement learning (Watkins, 1989) in that it allows an agent to adapt its actions to gain maximally from the environment while only being rewarded for correct performance. But animals learn much more complicated behaviors through instrumental conditioning than robots presently acquire through reinforcement learning. We describe a new computational model of the conditioning process that attempts to capture some of the aspects that are missing from simple reinforcement learning: conditioned reinforcers, shifting reinforcement contingencies, explicit action sequencing, and state space refinement. We apply our model to a task commonly used to study working memory in rats and monkeys: the DMTS (Delayed Match to Sample) task. Animals learn this task in stages. In simulation, our model also acquires the task in stages, in a similar manner. We have used the model to train an RWI B21 robot.

Keywords: operant conditioning, instrumental learning, shaping, chaining, learning robots

Running Head: Operant Conditioning in Skinnerbots

1. Introduction

A service dog trained to assist a handicapped person with the tasks of daily living can respond to over 60 verbal commands to turn on lights, open the refrigerator door, retrieve dropped objects, etc. (CCI, 1995). Many other animals, including rodents, pigeons, and dolphins, have also been shown capable of learning complicated behavioral routines. (See (Breland and Breland, 1961; Pryor, 1975) for striking accounts of behaviors taught to a variety of species.) Animals learn new behaviors quickly when they are trained using instrumental (or operant) conditioning techniques, which are based on principles of cognitive psychology.

Mobile robots trained by methods such as Q-learning (Watkins, 1989) have not come close to matching the sophistication and versatility of animal learners. While this disparity is no doubt partly due to the superior perceptual and motor capabilities of animals, techniques for animal training have also been studied for considerably longer, and by many more investigators, than those for robot training. We suggest that closer attention paid to the animal training literature and a serious attempt to model the effects described there may yield benefits of immediate value to robot learning researchers, and also provide a new, computationally-oriented perspective on animal learning.

Due to the pioneering work of B. F. Skinner on operant conditioning (Catania and Harnad, 1988), we have coined the term "skinnerbot" to describe autonomous learning robots that employ strategies and exhibit behavioral effects characteristic of instrumental learning (Touretzky and Saksida, 1996). The present paper describes the investigation of a particular conditioning technique called *chaining* — in which behavioral routines are built up from smaller action segments — and how it can be applied to mobile robot learning. We developed a learning algorithm that incorporates aspects of chaining which reinforcement learning techniques do not address, such as shifting reinforcement contingencies and learning of conditioned reinforcers. We chose a classic cognitive assessment task that involves behavioral sequences, the Delayed Match to Sample (DMTS) task (Blough, 1959), as the first test case for our learning model. We have also implemented the model on an RWI B21 mobile robot.

1.1. Operant Conditioning

In operant conditioning, the acquisition and further performance of an action depends on the consequences experienced upon its completion. This type of learning is called "operant" because the behavior operates on (has an effect on) the environment; it is "instrumental" because the behavior is instrumental in producing reward. It is this type of learning that affords the animal some degree of control over its environment in that it has the ability to produce changes in its situation by performing an appropriate action. For example, it may learn that food can be produced by pressing a lever. It follows that instrumental learning enables animals to cope with a dynamic environment in which the consequences of their actions may vary.

This contrasts with classical, or Pavlovian conditioning (Pavlov, 1927), in which learning is limited to associating a possibly arbitrary conditioned stimulus with a reinforcing (unconditioned) stimulus that elicits some type of innate behavioral response. For example, food as the unconditioned stimulus naturally produces appetitive responses such as salivation; electric shocks produce fear and avoidance responses; and a noxious stimulus such as a puff of air delivered to the eyeball produces defensive responses. In a classical conditioning procedure, an initially neutral stimulus such as a tone or light is repeatedly followed by an unconditioned stimulus. After learning, the conditioned stimulus comes to elicit a similar behavioral response, even in the absence of the unconditioned stimulus. Thus, when the bell rings, the dog salivates even if no food is delivered. Because the responses that occur during classical conditioning are innate, an animal that could only learn Pavlovian contingencies would be wholly dependent on evolutionary processes to construct responses appropriate to the stimuli received.

The phenomena of classical and operant conditioning are not as easily distinguished as the above description suggests; each shares some properties of the other, suggesting that they involve the same underlying mechanism. Pavlovian responses are known to occur as part of instrumentally conditioned behaviors, e.g., a rat that has learned to press a bar to get food will begin to salivate when the bar is pressed (Holland, 1993). Conversely, in a process known as autoshaping, classical contingencies can produce the sort of voluntary behavior normally associated with instrumental conditioning, rather than purely autonomic or reflexive responses (Brown and Jenkins, 1968). In the original autoshaping experiments, done with pigeons, a standard operant chamber with a lighted response key and food hopper was used. The key was illuminated for a fixed, brief period of time, and food was presented at the offset of each illumination. The pigeon learned to peck the key even though reinforcement was not contingent on that action.

Classical conditioning has a well-developed computational theory, the Rescorla-Wagner theory and its descendants (Rescorla and Wagner, 1972; Sutton and Barto, 1981; Klopf, 1988; Barto and Sutton, 1990), that predicts the strength of a stimulus-reward association based on factors such as stimulus saliency, background stimulus rate, and training history. In addition, some simple models of classical conditioning have been implemented on robots (Verschure et al., 1995). Classical conditioning is of some value to robots: it is useful for a robot to be able to learn predictive values of stimuli and possibly follow them with innate anticipatory responses. But instrumental learning, which involves associations between actions and their outcomes, allows for the modification of responses in an unstable environment; it confers an ability that is probably more critical to the robustness and practicality of a mobile robot. At present, there are no theories of instrumental conditioning comparable in scope and explicitness to the Rescorla-Wagner model of classical conditioning, much less a unified theory of both phenomena. The goal of our work is to provide such a theory, instantiated as a computational model. This paper describes an initial step in that direction.

1.2. Chaining

Complex behaviors can often be broken down into components and analyzed as a sequence of operants.[1] For example, a chick trained to "play the piano" pecks a sequence of keys to obtain a food reinforcement at the end of the tune (Breland and Breland, 1961). A pig taught to "grocery shop" pushes a cart and selects specific items to place in it, one after the other (Breland and Breland, 1961).

A behavioral chain can be analyzed as a sequence of stimuli and responses. The core unit of a chain is called a *link*; it consists of a discriminative stimulus, a response, and a reinforcer. The chain begins with the presentation of the first discriminative stimulus. When the animal makes the appropriate response in the presence of this stimulus, a conditioned reinforcer[2] is presented as a reward for the response. The reinforcer also functions as a discriminative stimulus for the next link in the chain, setting the occasion for the next desired response. This process continues for a number of links until reaching the final stimulus in the chain, which is a primary (innate) reinforcer such as food. The links are "overlapped" in that the discriminative stimulus for the production of one response is the reinforcer for the previous response; this holds the chain together. The concept of chaining differs from other examples of response sequences, such as fixed action patterns, in that chains of behavior can be modified through reinforcement. Fixed action patterns, as found in many animals, are hardwired: once the sequence is initiated it goes to completion independent of the consequences of the behavior. An example of this type of behavior occurs in the Greylag Goose. When an egg rolls out of its nest it will stand up, put its bill on the egg, pull back toward its chin, and roll the egg into its nest. While engaged in this fixed action pattern, the goose always performs the same behaviors in the same order. It will continue the pattern even if it loses its grip on the egg, and the pattern is triggered by any round stimulus outside its nest, including beach balls. Thus this type of behavior is much less flexible

[1] The animal learning literature defines at least two classes of behavioral responses (Schwartz, 1989): (i) respondents, which originate with the stimuli that elicit them (e.g., a reflex), and (ii) operants, which are determined by their effects on the environment since they do not require eliciting stimuli.

[2] At least two types of reinforcers can be distinguished (Reynolds, 1968): (i) primary reinforcers can reinforce behavior without the animal having had any prior experience with them (e.g., food, water). (ii) conditioned reinforcers acquire the power to reinforce behavior during the lifetime of the animal via a Pavlovian mechanism in which the stimulus that becomes the conditioned reinforcer is repeatedly paired with a primary reinforcer.

than that involved in instrumental chaining. See (Barnett, 1981) for additional examples of fixed action patterns.

The idea that patterns of responding can be reduced to a succession of stimulus-response units has been controversial: Skinner (Skinner, 1938) claimed that all behavior, including language, could be represented this way, while others, such as Chomsky (Chomsky, 1959) and Lashley (Lashley, 1951) held that sequential behavior could not be adequately accounted for in these terms. There is now considerable evidence, however, that many, though probably not all types of behavior sequences are held together this way (Gollub, 1977).

The concept of constructing behavioral sequences for mobile robots from small elements is appealing in that the programmer's responsibility would be limited to the construction of just these behavioral primitives, plus the learning algorithm for putting them together. Many different behaviors could be assembled from a well-designed set of primitives, and learning could potentially be made faster because knowledge could be shared among tasks with similar sub-tasks.

1.3. Previous models

Only a few previous computational models of operant conditioning phenomena have been described. Models of conditioning in *Aplysia* (Baxter et al., 1991; Raymond et al., 1992) have focused on learned suppression of a motor action. Mixed classical/operant models (known as "two process models") of escape and avoidance behavior in vertebrates (Grossberg, 1972; Schmajuk and Urry, 1995) address only simple responses to conditioned stimuli. None of these models approach the full richness of vertebrate learning, involving, for example, acquisition of secondary reinforcers and construction of behavior chains.

Graham, Alloway and Krames (Graham et al., 1994) describe a "virtual rat" designed to let undergraduates try their hand at operant conditioning. Their program is hard-wired to acquire a particular conditioned reinforcer (the sound of a food dispenser) and to respond to a specific shaping strategy to teach the simulated rat to bar press for food (Krames et al., 1995). Other primitive actions, such as grooming, can be encouraged by linking them to food rewards, but the program is not flexible enough to permit shaping anything complex except for bar pressing, nor is it possible to teach the rat to respond to external signals such as a tone or light. There are also presently no provisions for chaining behaviors, for modifying the qualities of a particular motor response, or for refining the simulated animal's perceptual abilities.

Maki and Abunawass (Maki and Abunawass, 1991) model learning of a match-to-sample task (no delay) using a backpropagation network. In animals, this task requires learning a complex sequence of actions (see section 3). Maki *et al.*'s network, however, takes a sample stimulus and two potential match stimuli as input, and learns to compute two exclusive or functions for its output. It produces no overt behavior, just a "match left" or "match right" signal. Thus, the model does not emulate operant conditioning, but it does offer some suggestions about the learned internal representations of stimuli that might result from such conditioning.

Reinforcement learning bears some similarity to operant conditioning, since reinforcement learners do not need to be shown correct responses as training stimuli, as is required by supervised learners such as backpropagation. Like operant conditioning, reinforcement learning is appealing because it theoretically allows an agent to autonomously adapt its actions to get the most from its environment as it gains information over time. And reinforcement learning does capture some aspects of animal behavior, such as adaptive foraging in bees (Montague et al., 1995). Furthermore, a neural representation of the kind of reward signal required for reinforcement learning has been found in both bees (the VUMmx1 neuron, (Hammer, 1993)) and primates (dopamine neurons in the substantia nigra pars compacta and ventral tegmental area, (Schultz et al., 1995).) But as we argue in the following section, operant conditioning in mammals is a richer phenomenon than can be addressed by current reinforcement learning theories.

Some work has been done on reinforcement learning of sequential tasks. Singh (Singh, 1992) describes a sequential task learner in which separate "Q-modules" learn different elemental and composite tasks, and

Mahadevan and Connell (Mahadevan and Connell, 1992) use Q-learning to acquire multiple behaviors that can then be controlled by a hardwired switching scheme to designate which should be active at a given time. Asada et al. constructed a robot soccer player that learned to push a ball into a goal while avoiding an opponent (Asada et al., 1994). They examine techniques for combining ball shooting and collision avoidance behaviors, each acquired separately using Q-learning, into a policy function that accomplishes both tasks. Although these papers look at sequential task learning, their approaches have been demonstrated only in simple environments, since they are subject to the usual combinatorial limitations of Q-learning.

Composing behaviors from primitives, a more general problem than chaining, has also been investigated for mobile robots. Matarić showed how foraging behavior could be constructed from routines for wandering, homing, aggregation, and dispersion, plus some innate reflexes such as grasping an object whenever one is encountered (Matarić, 1995). In later work, she used reinforcement learning to implement a complex behavior by deriving a policy for selecting the appropriate primitive behavior based on the robot's current state (Matarić, 1996).

Other examples of reinforcement-based robot learning include (Dorigo and Columbetti, 1994; Colombetti et al., 1996) and (Millán, 1996). However, these authors rely on immediate reinforcement strategies in which the robot's action at every time step is positively or negatively reinforced by an automated trainer. (A human trainer would be unable to keep up.) This does not make for a realistic model of animal learning.

2. Issues Critical to the Success of Instrumental Conditioning

2.1. A Model of Instrumental Conditioning

A close examination of the steps involved in the chaining of animal behavior reveals several important issues that are critical to the success of the procedure, and which have not been considered in previous computational models of conditioning:

2.1.1. Conditioned Reinforcers (Bridging Stimuli).

Contiguity of action and outcome are critical to instrumental learning: an action must be closely followed by a reinforcer in order for the animal to learn an association between the two. In a training situation, however, it is often difficult to reward an animal with food immediately after the occurrence of the desired response. Conditioned reinforcers are stimuli that become associated with food or water (or some other innate reward – even exercise), and serve as a signal that "the reward is coming," thereby eliminating the gap between the desired action and the reinforcement signal. For example, in a Skinner box, every time the animal is about to receive a food pellet it will hear the click of the food dispenser operating. The sound quickly becomes a conditioned reinforcer; the animal learns the click means food will be available soon. The close temporal contiguity of the action and the sound of imminent reward is sufficient to produce a much increased likelihood of performing that action again.

Another important consideration is that the sound of the dispenser can be heard throughout the Skinner box. This makes it possible to reward the animal when it is not at the food hopper.

In typical RL techniques, credit assignment is a major problem. After completing a sequence of actions leading the agent to the goal, in order to learn which of those actions should be credited with contributing to the final success, the agent needs to evaluate the goodness of each action that was performed. When the only reward obtained occurs at the end of the sequence, only knowledge of the cumulative effect of the actions can be derived. Our model deals with this issue by learning conditioned reinforcers of the sort described above. It discovers primitive subgoals, such as hearing the sound of the dispenser activating, and seeks ways to achieve them.

2.1.2. Shifting Reinforcement Contingencies.

In operant conditioning there is often a nonstationary reward function. When constructing a complex behavioral chain, a new reward contingency is introduced each time a new phase of training is begun.

A more dramatic example of response to a change in reward is the phenomenon known as "extinction." When reinforcement of a behavior is discontinued, the animal will eventually stop producing that behavior. But in the short term its activity level actually rises in response to discontinued reinforcement, and furthermore, the variability of its responses increases. In this way, the animal broadens its exploration of the action space and may discover a variant of the learned action that will once again produce the expected reward. Animal trainers exploit this phenomenon to shape complex behaviors. See (Millán, 1994) for a related idea: increasing the variability of actions of a robot learner as a response to failure to improve performance.

Reinforcement learning algorithms such as Q-learning (Watkins, 1989) can track nonstationary environments, but do not explicitly detect changes in reinforcement contingencies and respond to them as animals do. Our model detects nonstationarity and uses this information to trigger changes in its representation of the environment. For example, in the early stages of learning the Delayed Match to Sample task (described later), a switch press produces a water reward. When more of the task has been acquired, a switch press can have two outcomes: either the water pump runs, or a light comes on, depending on context. The initial predictor for the water pump, which used to be quite successful, is now suddenly wrong half the time. Our model responds by resetting the reward tables for the "hear pump" goal and the success counts of its predictors. This causes the model to temporarily become more plastic, readily entertaining new predictors that exhibit greater accuracy, and willingly dropping old ones – but not until they can be replaced by better versions.

2.1.3. Action Sequencing.

One difficulty with sequential task decomposition is that a mechanism must be in place for directing the construction of the behavior chain: it is highly unlikely that an agent will be able to achieve a complicated goal state simply by composing action sequences randomly. Our model uses an explicit temporal predicate representation that allows it to distinguish the order in which events occur in order to learn behavioral sequences. The primitive subgoals mentioned above also function as discriminative stimuli and thus set the occasion for an action to take place.

2.1.4. Perceptual (State Space) Refinement.

A further problem with RL techniques is that they are usually restricted to a very small state space to avoid combinatorial explosion. As in other approaches based on explicit symbolic representations (including much of classical AI), we accomodate a large state space by factoring it into a set of predicates. We can then construct logical expressions to refer to collections of states in economical ways. Another advantage of this approach is that it permits incremental refinement of the state space by adding new predicates. For example, under either classical or operant conditioning, an animal will learn to distinguish tones at two different frequencies if they are associated with different rates of reward. Thus, a predicate like `HEAR(tone)` might eventually be supplanted by `HEAR(high-tone)` and `HEAR(low-tone)`. We have not included shaping of stimulus discriminations in our current learning algorithm, but intend to address it in the future.

2.1.5. Action Refinement.

Combinatorial considerations also limit the number of actions available in most reinforcement learning simulations. But animals are capable of an infinite variety of actions. They can be taught to produce an arbitrary gesture by a shaping process in which, initially, an innate behavior close to the desired gesture is rewarded to increase its repetition rate; then the criterion for reward is smoothly altered to shift the animal's performance in the desired direction. For example, a dog is shaped to turn in a circle by initially rewarding it for any slight movement to the left, then raising the requirements until eventually a full 360 degree turn is achieved. We will not address shaping in this article, but we will note that when actions are represented as parameterized templates, a simple version of shaping can be had by hill climbing. The reward signal then not only increases the likelihood that the action will be repeated, it also shifts the description of the action itself.

2.2. Memory Representation

==

Table 1 about here.

==

To introduce our model, we consider the simple case of a rat pushing down on a bar to get water. The working memory module holds a collection of time-labeled predicates describing the rat's current perceptions and actions and those of the recent past. At the instant when the rat receives a water reward, having previously heard the pump run, the contents of working memory might appear as in Table 1. In this notation, time 0 is the present, time 1 is the previous instant, time 2 is the instant before that, and so on.

==

Table 2 about here.

==

The algorithm for inferring reinforcement contingencies operates on a collection of slightly more abstract items called *temporal predicates*. These are derived from working memory elements by replacing numeric time tags with the symbolic labels shown in Table 2. For conciseness, the `:now` tag will usually be left implicit in the rest of this paper. The `:fut` tag is used to refer to predicates that became true at $t+1$ during retrospective analysis of the results of action at time t.

Working memory elements persist for only a small number of time steps, depending on the predicate involved. In the present simulation, `AT` and `GOTO` predicates last for two time steps, `SEE` and `HEAR` for six, and `PRESS` for eight. This is because the animal is always at some location and nearly always moving, so location-related items accumulate quickly. Conditioned stimuli and specalized actions (such as bar pressing) occur less frequently, and so are more memorable.

In the next level of representation in the program, we form conjunctions of predicates. Temporal tagging of these items allows us to infer cause and effect relationships between actions and stimuli, and construct temporal sequences. For example, the crucial match relationship in the DMTS task is described by the conjunction `PRESS(switch1):past & PRESS(switch1):now`, plus a similar conjunction for the second switch.

2.3. Learning Reinforcement Contingencies

The conjunctions our program constructs describe sequences of stimuli and actions that can occur in the world. Some of these occur frequently; others might never be encountered. Furthermore, some sequences are often followed by a reinforcement signal whereas others never are. In order to extract this information from its experience of the world, the program maintains two tables for each reinforcer. One counts the number of times each conjunction has been satisfied since that reinforcer was acquired; the other table counts the number of times a conjunction's occurrence has been followed on the next time step by the reinforcer. The *reward rate* of a conjunction is the second quantity divided by the first. The program attempts to find conjunctions with maximum reward rates. A conjunction that predicts rewards with no false alarms would have a reward rate of 1.0.

In complex domains we cannot afford to test all possible conjunctions of predicates, so heuristic search is used. Conjunctions are constructed incrementally by combining a pool of currently "best" conjunctions (starting with the null conjunction) with a pool of "best" predicates. A "best" conjunction is one whose reward rate is at least one standard deviation above the mean rate, or whose reward count is at least one standard deviation above the mean count. Both tests are necessary. Items with high reward *counts* constitute important features of the environment that need to be incorporated into conjunctions even if their reward rates in isolation are low. For example, since going to the water dispenser does not make the pump run, `GOTO(dispenser)` has a low reward rate, but since water is available nowhere else, it has a high reward count. Items with high reward *rates* are accurate predictors and should be retained for further exploration, even if their reward *counts* are relatively low (meaning they each account for only a limited number of the rewards that have been received.) Drescher makes a similar observation in his Piagetian learning model, where he distinguishes between *relevance* and *reliability* measures (Drescher, 1991).

During learning, conjunctions that are sufficiently well correlated with rewards generate "predictors," i.e., rules for predicting reward. These may displace earlier predictors that have not performed as well. To allow for the effects of noise, predictors are not replaced until they have a reasonably high application count (so their success rate can be accurately estimated) and their replacement has a significantly higher success rate.

During initial *magazine training* (learning to go to a food or water dispenser when the mechanism is heard to activate), a typical sequence of learned predictors and their reward rates is shown below.

1: RECEIVE(water) ← GOTO(dispenser) [0.154]
2: RECEIVE(water) ← HEAR(pump) [0.3]
3: RECEIVE(water) ← HEAR(pump) & GOTO(dispenser) [1.0]

Predictor # 3 says that 100% of the time, when the simulated rat heard the pump and immediately went to the water dispenser, it received a water reward. So this conjunction predicts water with perfect accuracy.

To generate behavior, we look for predictors that can be satisfied by the rat's taking some action currently available to it. Predictor # 1 suggests going to the water dispenser, so initially the rat spends a lot of time there. This causes the reward rate of `GOTO(dispenser)` to drop, since on most occasions there will be no water available. (But the reward count for the predictor remains high relative to all other predicates, since the dispenser is still the only place where water can be obtained.) Because predictor # 1 gives many false expectations of reward, it is soon dropped. Predictor # 2 is somewhat more successful, but it cannot be satisfied by the rat's own actions, because at this point in training the rat has no way to cause the pump sound to occur. But predictor # 3, which most accurately describes the current reward contingencies in this environment, *can* be used by the rat to generate behavior whenever the pump is heard to run.

Predictors in our model can be divided into two classes: those that contain an action term, and those that do not. Since the former can be satisfied by executing the action, they are instrumental rules. The latter, such as predictor # 2 above, may be viewed as Pavlovian rules: they represent an association between stimuli. The response generated by this association would have to be innate and hardwired.

2.4. Acquiring Conditioned Reinforcers

The second type of learning in our model is the acquisition of new conditioned reinforcers. If the skinnerbot could find a way to make `HEAR(pump)` be true, then predictor #3 suggests it could get water whenever it wanted. So `HEAR(pump)` becomes a secondary reinforcer, and the skinnerbot begins trying out theories of what causes the pump to run. At about this point in the training process, the trainer stops randomly triggering the pump for magazine training and only rewards bar presses. By exploration, the skinnerbot eventually discovers that pressing the bar will make the pump run, and nothing else will. Thus, the following predictor is acquired:

HEAR(pump) $\leftarrow$ PRESS(bar) [1.0]

Now suppose the skinnerbot is at the water dispenser along the north wall of the Skinner box, and the switch is in the northeast corner. The predictor above cannot apply because the `PRESS(bar)` operator is not available at the dispenser. This impasse generates a new secondary reinforcer, `CAN(PRESS(bar))`, which the skinnerbot also seeks to control, leading it to discover another predictor:

CAN(PRESSS(bar)) $\leftarrow$ AT(ne-corner):fut [1.0]

The `:fut` tag indicates that if the skinnerbot is at the northeast corner at time $t+1$ it will be able to press the switch at $t+1$. It can make the predicate true by a `GOTO` action, or if it is already at the south wall, it need only refrain from going elsewhere.

Now the skinnerbot has a hierarchy of reinforcers. The primary (innate) reinforcer is water. The most important secondary reinforcer is the pump sound. A more remote secondary reinforcer is the ability to press the switch.

2.5. Action Selection

At each time step, the skinnerbot seeks a predictor it can satisfy. Predictors are prioritized by the nature of the reinforcement they promise, so that given a choice, the skinnerbot will always act to secure a more basic reward (water) over a more abstract one (the ability to press the bar.) If it finds a predictor where all but one of the predicates is currently true (i.e., matches an item in working memory), and the last one can be made true by taking some action that is presently available, then it will select that action with high probability. There is also some randomness in the action selection mechanism, to facilitate exploration.

`CAN` goals must be handled specially. They are only looked at when the program has some other subgoal that could be satisfied if the `CAN` goal's action were available. For example, if the predictor for bar pressing had as antecedent the conjunction `SEE(light)` & `PRESS(bar)`, the skinnerbot only needs to be able to press the bar if it sees the light; the rest of the time it doesn't matter whether it can press the bar or not, as the predictor cannot be satisfied without the light. When `SEE(light)` is true, the predictor can be satisfied if `PRESS(bar)` is an available action. Hence `CAN(PRESS(bar))` becomes a goal worth satisfying at that point, and this in turn allows the predictor for the `CAN` goal to lure the skinnerbot to the location of the bar.

With the previously discussed predictors for the bar pressing task and the ordering imposed by the reinforcer priorities, the skinnerbot will shuttle back and forth between the northeast corner and the center of the north wall, alternately pressing the bar and collecting its water reward.

==
Figure 1 about here.
==

Figure 1 shows the overall flow of information in the learning program.

2.6. Predictor Creation and Deletion

We use a variety of heuristics to constrain the search through the space of possible predictors. New predictors for a given reinforcer are created only when that reinforcer has just been received and the reward counts updated. At that point, the program can check candidate predictors against its working memory, so that it only constructs predictors that would have predicted the reward it just got. Furthermore, in order for new predictors to be created the reward must either have been unexpected, meaning the current set of predictors is incomplete, or there must have been at least one false prediction since the last reward was encountered, meaning there is an erroneous predictor, one that is not specific enough to accurately express the reward contingencies.

New predictors are created from the best-scoring conjunctions currently maintained for that reinforcer. If several conjunctions are tied for top score, the ones with the fewest number of terms are selected. If there are still several candidates, two are chosen at random to become new predictors. If these turn out not to be the best choices, further experience will increase the scores of the conjunctions not chosen, because their instance and reward counts are still being updated in working memory at every time step. In addition, occasional selection of random actions may give rise to unexpected rewards that cause new conjunctions to be created. As the best performing conjunctions see their scores increase, they become predictors on some subsequent trial and eventually replace the predictors they outperform.

Two numerical measures are used to assign scores to conjunctions and predictors: *merit* and *demerit*. They estimate the lower and upper bounds, respectively, on the true reward rate based on the number of examples seen so far. Let n be the number of times a conjunction has been observed to be true, and r the number of times the reinforcer was received on the subsequent time step. Merit and demerit are defined as:

$$M(r,n) = \frac{r}{n} \cdot \max(0.2, 1 - \frac{1.175}{n})$$

$$D(r,n) = \min(1, \frac{r}{n} + \frac{n-r}{0.7n^2})$$

For $n = 0$ the merit is defined to be 0 and the demerit 1. The difference between the merit and demerit values can be taken as an uncertainty measure for our estimate of reward rate (Gallistel, 1990, chap. 13). See Figure 2. As n approaches ∞, merit and demerit both converge to r/n, the true reward rate. The constants in the equations for merit and demerit were chosen empirically to give an appropriate balance between reward estimates r/n and confidence values when n is small. For example, a conjunction that has been true 6 times and followed by reward 4 times has an estimated reward rate of 0.667, and a merit of 0.536. A conjunction that has only been true twice, and followed by reward each time, has an estimated reward rate of 1.0 but a merit of only 0.413, due to low confidence. The constants in the merit equation were chosen so that $M(4,6) > M(2,2) > M(2,3)$.

==
Figure 2 about here.
==

When creating new predictors, candidate conjunctions are sorted by merit rather than raw reward rate to give greater weight to conjunctions that have been sampled more heavily. When deleting predictors, demerit is used, so the program is conservative in its judgements and does not delete too quickly.

Predictors are deleted in three circumstances. First, if the predictor has just given a false alarm, it may be deleted if its demerit is below a certain minimum value. This is necessary so that poor predictors do not trap the program in unsuccessful behaviors, hindering its discovery of better options. Second, if a reinforcer has just been correctly predicted, the predictor may be deleted if its *demerit* is less than the highest *merit* of any other successful predictor for that reinforcer. In other words, when the program is confident that a better-performing predictor exists, the poorer one is displaced. Both predictors must have predicted the reinforcer just received. Finally, a predictor will be deleted if there is another predictor whose antecedent uses a strict subset of the terms in this predictor's conjunction, whose merit is nearly as good, and whose number of trials is sufficiently high that there is reasonable confidence that the two predictors are equivalent. This helps the program find minimal predictors for the domain, replacing those with extraneous terms. However, once an adequate set of predictors has been learned so that there are no unexpected occurences of a reinforcer and no false alarms, the current version of the program stops creating new predictors for that reinforcer, so a strictly minimal set of predictors is not always attained.

3. The DMTS Task

The Delayed Match to Sample task is widely used in cognitive neuroscience to measure properties of working memory. The basic idea is to present the subject with a stimulus (the sample), impose a delay, and then present a pair of stimuli, one of which matches the sample. The subject must select the matching stimulus in order to receive a reward. The delay period can be varied to control the length of time the sample must be maintained in working memory. There are both spatial and nonspatial versions of this task. In the spatial version, all stimuli are identical; they are distinguished on the basis of the location at which they appear (Hampson et al., 1993). In the nonspatial version, the sample appears in one location and the two probe stimuli appear in other locations; it is the visual characteristics of the stimuli that matter (Bussey et al., 1994). There is some debate as to whether the spatial version of the task actually involves working memory; animals might be using some other type of mediating strategy, such as aligning their body toward the site of the last stimulus, to bridge the delay (Gutnikov et al., 1994). Our learning algorithm can be applied to either the spatial or non-spatial version of the task.

==

Figure 3 about here.

==

Hampson, Heyser, and Deadwyler (Hampson et al., 1993) describe a spatial version of DMTS for rats that uses as stimuli two retractable switches mounted on the wall of a Skinner box, as shown in Figure 3. A water dispenser is located between the two switches, and a light and nosepoke port are mounted on the opposite wall. (A nosepoke port is a hole with an infrared LED and sensor positioned so that every time the rat pokes his nose into the hole, it breaks the beam, allowing the event to be detected.) At the start of a trial, one of the switches extends, and the rat must go over and press that switch. This causes the switch to retract, while at the same time a light goes on over the nosepoke port. The rat must now go to the opposite wall and make repeated nosepokes for a variable delay period averaging one minute. (The nosepoke requirement is intended to prevent the rat from parking itself in front of the switch it just pressed until the switches extend again. That sort of mediating strategy would eliminate the need for working memory.) At the conclusion of the delay period, the next nosepoke causes the light to go out and both switches to extend. Now the rat must return to the switch it pressed previously and press it again. If it chooses the correct switch, it receives a water reward.

Rats are taught this task in stages. Hampson et al. report a training time of two to three months (Hampson et al., 1993).

3.1. Shifting Reinforcement Contingencies in the DMTS Task

Both rats and our model require multiple training stages to learn the DMTS task. In our simulation there are four stages. In the first stage, the pump is triggered at random times and the task is to learn that water is available at the dispenser whenever the pump sound is heard.

In the second stage, one or the other switch is presented at the start of a trial. The switch immediately retracts when pressed, and the pump runs. This leads to the following predictor for switch1 (a similar rule is learned for switch2):

HEAR(pump) ← PRESS(switch1)

Since the switches can only be pressed when they are in their extended position, and hence visible, the following predictor is also learned:

CAN(PRESS(switch1)) ← SEE(switch1) & GOTO(sw1-loc)

Notice that the predictor uses `GOTO(sw1-loc)` rather than `AT(sw1-loc):fut`. If the rat is at some other location, either predicate would suffice to bring it to the switch. The reason `GOTO` is preferred is that once the rat goes to the switch, it will press the switch at time t, and at time $t+1$ the switch will be retracted. The predictor that used `AT` would have its antecedent satisfied at t and thus generate a false expectation that `CAN(PRESS(switch1))` would still be true at $t+1$. The preferred predictor avoids this error because it will not be satisfied if the action at time t is `PRESS` rather than `GOTO`. Note that if the rat is already at the switch but executes a `GOTO(sw1-loc)` anyway, the action has no effect, but in this case the antecedent is satisfied and we have a true prediction that the switch will be pressable at $t+1$.

In the third training stage, a trial starts with a light going on over the nosepoke port. The rat learns that poking the port when the light is on causes the switches to extend:

SEE(switch1) ← SEE(light) & POKE(poke-port)

(There is a similar rule for switch2.)

In the fourth and final stage, a single switch appears at the start of the trial, and pressing it causes the light to go on. Before this stage, switch pressing always ran the pump. Now it can produce either the pump or a light, depending on context. The previously-learned predictors for `HEAR(pump)` and `SEE(light)` are no longer 100% accurate. The program observes the changed contingencies and responds by resetting the reward tables that govern its behavior for those reinforcers, which has the effect of making the model more plastic – eager to acquire new predictors and more willing to replace old ones. Predictors that earlier did an adequate job of characterizing the environment's reinforcement contingencies are then replaced with more selective versions:

SEE(light) ← NOT(reinforced):past & PRESS(switch1)

HEAR(pump) ← PRESSS(switch1):past & PRESS(switch1)

The `NOT(reinforced)` predicate is true at time t if the model has not received any reinforcers recently (no record in working memory.) It is a way of marking the start of a trial. Since seeing a switch is itself reinforcing, when it comes time to nose poke the model has already begun the new trial, so the predictor above actually checks for `NOT(reinforced):past`. By the time the switch reappears near the end of the trial, this item will have faded from working memory. As a marker for the second half of the trial the model uses `SEE(light):past`. In the second half, a switch press produces the pump sound rather than the light.

3.2. Superstitious Behavior

Because the model is always trying to formulate explanations for reinforcers, it will construct them even when no explanation is possible. For example, at the start of training, the pump is triggered at random times. The model will generate predictors for `HEAR(pump)` and act on whichever ones have the highest apparent reward rate. Thus, if on several occasions it happened to be moving from the southeast corner to the northeast corner when the pump ran, it might decide that this action was *causing* the pump to run, and begin repeating it deliberately. If water is actually being dispensed randomly, but at sufficiently frequent intervals, the predictor will be successful often enough to be retained and perhaps even strengthened. This sort of "superstitious" behavior has been observed in real animals (Skinner, 1948) although the underlying mechanisms are at this point unclear (Staddon and Simmelhag, 1971).

In the second training stage the pump sound is reliably predicted by switch pressing, so the model replaces the superstitious predictors with accurate ones. It then begins forming superstitious predictors for the appearance of a switch that occurs at the start of each trial.

4. Results

4.1. DMTS Simulation

A minimum of ten predictors are required for the DMTS task, as shown in Figure 4. Our program learns a set of predictors equivalent to this optimal set, but with some extraneous terms and redundant predictors remaining. For example, in our formulation of the DMTS there are two equivalent predictors for receiving water when the pump runs, which are learned at nearly the same time:

RECEIVE(water) ← HEAR(pump) & GOTO(dispenser)

RECEIVE(water) ← HEAR(pump) & AT(dispenser):fut

Some examples of correct predictors the program learned with harmless extra terms are:

HEAR(pump) ← PRESS(switch1):past & SEE(switch2) & PRESS(switch1)

SEE(switch1) ← AT(sw1-loc):prev & SEE(light):prev & POKE(poke-port)

SEE(switch2) ← NOT(reinforced):past & SEE(light):past & POKE(poke-port)

At some points during the learning process, an animal may develop a bias in favor of one of the two switches, so that it almost never presses the other switch. Our program sometimes does this as well. If an animal develops a bias, the trainer introduces a few "correction trials" in which the animal is forced to choose the correct switch. We use the same approach. Our training mechanism alternates between the two switches as samples (correct responses) and keeps track of how many times each was recently chosen in the match phase of a trial. If the ratio between the two switches exceeds 2:1, a correction trial is given where only the correct switch is presented in the match phase, so the program is forced to pay attention to that switch. This leads it to propose new predictors involving the switch, which compete with the earlier ones until an overall correct set of predictors is found.

In a typical run, the program experienced 170 trials: 10 for stage 1, 20 for stage 2, 20 for stage 3, and 120 for stage 4. It created a total of 300 predictors, nearly 200 of which were for the `SEE(switch1)` and

SEE(switch2) goals, since when these occur at the start of a trial they are not predictable from preceding stimuli. (As long as its predictions for a goal are less than perfect, the program keeps trying new predictors.)

==
Figure 5 about here.
==

The program's performance on trial 170 is shown in Figure 5, and the actual predictors that produced this behavior are shown in Figure 6. At time steps 2981-2982 the program is shuttling between the nose poke port and the location of switch1, waiting for a new trial to begin. The shuttling is caused by a recently created supserstitious predictor, number 293, that will soon be deleted. At time 2983 switch2 appears, and at 2984 the program presses it. (Note that it did not use predictor 44 to get to switch2. Instead it happened to use predictor 148, which predicts SEE(light) rather than CAN(PRESS(switch2)). This predictor is a successful predictor because other predictors, such as 107, will cause switch2 to be pressed once the program has arrived at that location.) At time step 2985 the light turns on in response to the switch press, and at 2986 the program nose pokes. This causes both switches to extend at 2987, and predictor 211 generates a CAN goal that predictor 44 satisfies by taking the program to switch2. At 2988 the program presses the matching switch, causing the pump to run at 2989. The trial concludes at 2990 with the receipt of the water reward.

==
Figure 6 about here.
==

4.2. Robot Implementation

Amelia, shown in Figure 7, is an RWI B21 mobile robot with a color camera on a pan-tilt head and a three degree-of-freedom arm with a gripper. Computing power is provided by three on-board Pentium processors. A 1 Mbps radio modem links Amelia to a network of Sparc stations that contribute additional processing cycles. For our experiments, we ran the learning program in Allegro Common Lisp on a Sparc 5 and used TCA (Task Control Architecture) (Simmons, 1994) to communicate with the robot.

==
Figure 7 about here.
==

To provide reinforcement stimuli to the robot, we added a Logitech three-button radio trackball. The human trainer can stand anywhere in the vicinity of the robot and press a button to send a reward signal when a desired response occurs. The button press is picked up by an on-board receiver plugged into a serial port on the laptop and relayed to the learning program on the Sparc. The robot acknowledges button presses with a brief audio response.

As a preliminary experiment in robot training, Amelia was taught to sort objects into bins based on color. We used pink and green plastic dog toys as the manipulanda and two blue recycling bins as receptacles. In the first training stage we provided a single recycling bin to the left of the platform where dog toys were presented to the robot. We presented Amelia with pink toys one at a time and rewarded it for dropping them in the bin. In the second stage we placed the recycling bin to the right of the platform and provided green toys, again rewarding Amelia for dropping them in the bin. At this point Amelia had learned that dropping objects in bins produced rewards, but had not learned to make discriminations based on color or location. In the third stage we placed bins on either side of the central platform and alternately presented pink and green toys, rewarding the robot only for dropping a pink toy in the left bin or a green toy in the right bin. Correct behavior was quickly obtained by constructing these predictors:

SENSE(reward) ← HOLDING(pink-toy) & DROP-AT(loc1)

SENSE(reward) ← HOLDING(green-toy) & DROP-AT(loc3)

Training a robot in realtime raises a number of issues not addressed in computer simulation, such as the proper delivery of reward signals. Since the current learning program works in discrete time steps, it is important that an action performed at time t be rewarded at $t+1$, not at t or $t+2$. Our solution is to have the clock "tick" at the instant an action is begun. If the robot decides at time t to lower its gripper and drop a pink toy in the bin to its left, then `DROP-AT(loc1)` becomes the action for time t and the clock is advanced to $t+1$. Thus, as soon as the trainer sees the robot moving to drop the toy, he or she can press the reward button, and the reward will be recorded as occurring at $t+1$. If the trainer waits too long, however, the action will complete, and since it takes essentially no time for the robot to decide on its next action, the clock may move on to $t+2$ before the reward arrives. In future work we expect to switch to a continuous time model to avoid this sort of brittleness.

5. Discussion

We have described a model of operant conditioning that successfully learns a task requiring a complex behavioral chain: the DMTS task. Our work highlights several aspects of operant conditioning not addressed in most earlier models:

1. Reinforcement contingencies change over time. The incorporation of this into our model allows a trainer to add new elements to the animal's (or robot's) behavior without losing previously learned information.

2. Conditioned reinforcers hold the behavioral chain together. These learned reinforcers, such as the sound of a pump or the appearance of a light, help the animal deal with the credit assignment problem when learning a complex task.

3. Discriminative stimuli "set the occasion" for performing a particular behavior. In other words, they signal to the animal that execution of an action will now produce a reward. Our model represents these stimuli as terms in a conjunction. In chained behaviors, discriminative stimuli help the animal keep track of the order of actions. An example is the light in the DMTS task, whose presence indicates it is time to nose poke.

   Discriminative stimuli may change over time. Animal trainers use a technique called "stimulus fading," where the signal to perform an action is gradually made more subtle, or perhaps replaced altogether with a less salient stimulus. We do not yet model stimulus fading.

4. Animals have large, continuous state and action spaces. In our model, we factor the state space into conjunctions of predicates that refer to collections of states in an economical way. In this way we avoid the limitations of table-driven reinforcement learning algorithms, which require a small state space to avoid combinatorial explosion. Many other AI programs use this same approach. What is novel here is the development of heuristics based on reward rate and total reward to guide the construction of new conjunctions and predictors based on the program's recent experience.

Much work remains to be done to complete a computational-level model of operant conditioning. In order to expand our model of chaining to incorporate more results from the animal learning literature, one idea that we would like to explore is operator shaping. We would like to equip our robot with an initial set of innate behaviors (operators) which may not necessarily be able to fully satisfy the requirements of the trainer or the environment. What would then be needed is a means for refining operators with experience, similar to the animal training technique called "shaping". Evidence for the existence of innate behavioral elements (Berridge et al., 1987) combined with the success of shaping in animal learning paradigms suggests that this would be a very powerful mechanism for improving the performance of our learning algorithm.

We also need to add facilities for refining perceptual predicates with experience, so that the model can acquire finer-grain discriminations if the reinforcement contingencies require this. Animals can learn to make very fine distinctions in pitch, intensity, and color, but they can also generalize on these properties, depending on the demands of the task. This makes the state space dynamically refinable with experience.

We have coined the term "skinnerbot" to refer to a class of agents designed for operant conditioning, but unlike Skinner, we do not eschew representations in our theory. Once we have laid more of the computational-level groundwork for our model, we will be able to move on to a model that addresses some of the presently unsettled psychological issues in instrumental learning (Dickinson, 1995).

Acknowledgments

This work was supported by National Science Foundation grant IRI-9530975. We thank Joseph O'Sullivan and Scott Raymond for technical assistance with the robot, and Randy Gallistel, Rob Hampson, and Bruce Blumberg for helpful discussions.

Figure Captions

Figure 1: Flow of information in the learning program.

Figure 2: Merit and demerit curves converge toward the true reward rate as the number of samples increases.

Figure 3: Skinner box configured for DMTS task.

Figure 4: Optimal predictors for the DMTS task.

Figure 5: Performance on trial 170 of the DMTS task. The number embedded in each arrow is the predictor being applied; a "C" indicates no predictor is satisfied in the present situation (clueless).

Figure 6: The actual predictors responsible for the program's behavior on trial 170, shown in the order in which they are applied.

Figure 7: Amelia performing the toy sorting task. The human trainer issues rewards using the radio trackball in her left hand.

Table Captions

Table 1: State of working memory at the moment water is received.

Table 2: Tags used to construct temporal predicates.

References

Asada, M., Uchibe, E., Noda, S., Tawaratsumida, S., and Hosoda, K. (1994). Coordination of multiple behaviors acquired by a vision-based reinforcement learning. In *International Conference on Intelligent Robots and Systems*. IEEE.

Barnett, S. (1981). *Modern Ethology*. Oxford University Press.

Barto, A. G. and Sutton, R. S. (1990). Time-derivative models of Pavlovian conditioning. In Gabriel, M. and Moore, J., editors, *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, pages 497–537. MIT Press, Cambdridge, MA.

Baxter, D. A., Buonomano, D. V., Raymond, J. L., Cook, D. G., Kuenzi, F. M., Carew, T. J., and Byrne, J. H. (1991). Empirically derived adaptive elements and networks simulate associative learning. In Commons, M. L., Grossberg, S., and Staddon, J. E. R., editors, *Neural Network Models of Conditioning and Action*, pages 13–52. Lawrence Erlbaum Associates, Hillsdale, NJ.

Berridge, K., J.C.Fentress, and Parr, H. (1987). Natural syntax rules control action sequence of rats. *Behavioural Brain Research*, 23:59–68.

Blough, D. (1959). Delayed matching in the pigeon. *Journal of the Experimental Analysis of Behavior*, 2:151–160.

Breland, K. and Breland, M. (1961). The misbehavior of organisms. *American Psychologist*, 16:681–684.

Brown, P. and Jenkins, H. (1968). Auto-shaping of the pigeon's keypeck. *Journal of the Experimental Analysis of Behavior*, 11:1–8.

Bussey, T. J., Muir, J. L., and Robbins, T. W. (1994). A novel automated touchscreen procedure for assessing learning in the rat using computer graphic stimuli. *Neuroscience Research Communications*, 15(2):103–109.

Catania, A. C. and Harnad, S., editors (1988). *The Selection of Behavior*. Cambridge University Press.

CCI (1995). *The CCI Program*. Canine Companions for Independence, Santa Rosa, CA. Informational page available at http://grunt.berkeley.edu/cci/cci.html.

Chomsky, N. (1959). Review of Skinner's Verbal Behavior. *Language*, 35(26-58).

Colombetti, M., Dorigo, M., and Borghi, G. (1996). Behavior analysis and training: A methodology for behavior engineering. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26(3):365–380.

Dickinson, A. (1995). Instrumental conditioning. In Mackintosh, N. J., editor, *Handbook of Perception and Cognition. Volume 9*. Academic Press, Orlando, FL.

Dorigo, M. and Columbetti, M. (1994). Robot shaping: Developing autonomous agents through learning. *Artificial Intelligence*, 70(2):321–370.

Drescher, G. L. (1991). *Made-Up Minds*. The MIT Press, Cambridge, MA.

Gallistel, C. R. (1990). *The Organization of Learning*. MIT Press, Cambridge, MA.

Gollub, L. (1977). Conditioned reinforcement: Schedule effects. In Honig, W. and Staddon, J., editors, *Handbook of operant behavior*. Prentice-Hall.

Graham, J., Alloway, T., and Krames, L. (1994). Sniffy, the virtual rat: Simulated operant conditioning. *Behavio Research Methods, Instruments, & Computers*, 26(2):134–141.

Grossberg, S. (1972). A neural theory of punishment and avoidance, ii: Quantitative theory. *Mathematical Biosciences*, 15:253–285.

Gutnikov, S., Barnes, J., and Rawlins, J. (1994). Working memory tasks in five-choice operant chambers: use of relative and absolute spatial memories. *Behavioral Neuroscience*, 108(5):899–910.

Hammer, M. (1993). An identified neuron mediates the unconditioned stimulus in associative olfactory learning in honeybees. *Nature*, 366:59–63.

Hampson, R. E., Heyser, C. J., and Deadwyler, S. A. (1993). Hippocampal cell firing correlates of delayed-match-to-sample performance in the rat. *Behavioral Neuroscience*, 107(5):715–739.

Holland, P. C. (1993). Cognitive aspects of classical conditioning. *Current Opinion in Neurobiology*, 3:230–236.

Klopf, A. H. (1988). A neuronal model of clasical conditioning. *Psychobiology*, 16:85–125.

Krames, L., Graham, J., and Alloway, T. (1995). *Sniffy, the Virtual Rat.* Brooks/Cole, Pacific Grove, CA. Includes software diskette.

Lashley, K. (1951). The problem of serial order in behavior. In Jeffries, L., editor, *Cerebral mechanisms in behavior.* John Wiley and Sons.

Mahadevan, S. and Connell, J. (1992). Automatic programing of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55:311–365.

Maki, W. S. and Abunawass, A. M. (1991). A connectionist approach to conditional discriminations: Learning, short-term memory, and attention. In Commons, M. L., Grossberg, S., and Staddon, J. E. R., editors, *Neural Network Models of Conditioning and Action*, pages 241–278. Lawrence Erlbaum Associates, Hillsdale, NJ.

Matarić, M. J. (1995). Designing and understanding adaptive group behavior. *Adaptive Behavior*, 4(1):50–81.

Matarić, M. J. (1996). Reinforcement learning in multi-robot domains. *Autonomous Robots*, 4(1):73–83.

Millán, J. d. R. (1994). Learning efficient reactive behavioral sequences from basic reflexes in a goal-directed autonomous robot. In *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, pages 266–274, Cambridge, MA. MIT Press.

Millán, J. d. R. (1996). Rapid, safe, and incremental learning of navigation strategies. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26(3):408–420.

Montague, P. R., Dayan, P., Person, C., and Sejnowski, T. J. (1995). Bee foraging in uncertain environments using predictive Hebbian learning. *Nature*, 377:725–728.

Pavlov, I. P. (1927). *Conditioned Reflexes.* Oxford University Press.

Pryor, K. (1975). *Lads Before the Wind.* Harper and Row, New York.

Raymond, J. L., Baxter, D. A., Buonomano, D. V., and Byrne, J. H. (1992). A learning rule based on empirically derived activity-dependent neuromodulation supports operant conditning in a small network. *Neural Networks*, 5(5):789–803.

Rescorla, R. A. and Wagner, A. R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In Black, A. H. and Prokasy, W. F., editors, *Classical Conditioning II: Theory and Research.* Appleton-Century-Crofts, New York.

Reynolds, G. S. (1968). *A Primer of Operant Conditioning.* Scott, Foresman.

Schmajuk, N. A. and Urry, D. W. (1995). The frigtening complexity of avoidance: An adaptive neural network. In *Models of Action.* Lawrence Erlbaum Associates, Hillsdale, NJ.

Schultz, W., Romo, R., Ljungberg, T., Mirenowicz, J., Hollerman, J. R., and Dickinson, A. (1995). Reward-related signals carried by dopamine neurons. In Houk, J. C., Davis, J. L., and Beiser, D. G., editors, *Models of Information Processing in the Basal Ganglia*, chapter 12, pages 233–248. MIT Press.

Schwartz, B. (1989). *Psychology of Learning and Behavior.* W.W. Norton.

Simmons, R. (1994). Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43.

Singh, S. (1992). Transfer of learning across sequential tasks. *Machine Learning*, 8:323–339.

Skinner, B. (1938). *Behavior of Organisms.* Appleton-Century-Crofts.

Skinner, B. (1948). "Superstition" in the pigeon. *Journal of Experimental Psychology*, 38:168–172.

Staddon, J. and Simmelhag, V. (1971). The "superstition" experiment: A reexamination of its implications for the principle of adaptive behavior. *Psychological Review*, 78:3–43.

Sutton, R. S. and Barto, A. G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review*, 88:135–170.

Touretzky, D. S. and Saksida, L. (1996). Skinnerbots. In Maes, P., Mataric, M., Meyer, J.-A., Pollack, J., and Wilson, S. W., editors, *From animals to animats 4: Proceedings of the fourth international conference on simulation of adaptive behavior*, pages 285–294. MIT Press.

Verschure, P. F. M. J., Wray, J., Sporns, O., Tononi, G., and Edelman, G. M. (1995). Multilevel analysis of classical conditioning in a real world artifact. *Robotics and Autonomous Systems*, 16(2-4):247–265.

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards.* PhD thesis, Cambridge University, Cambridge, England.

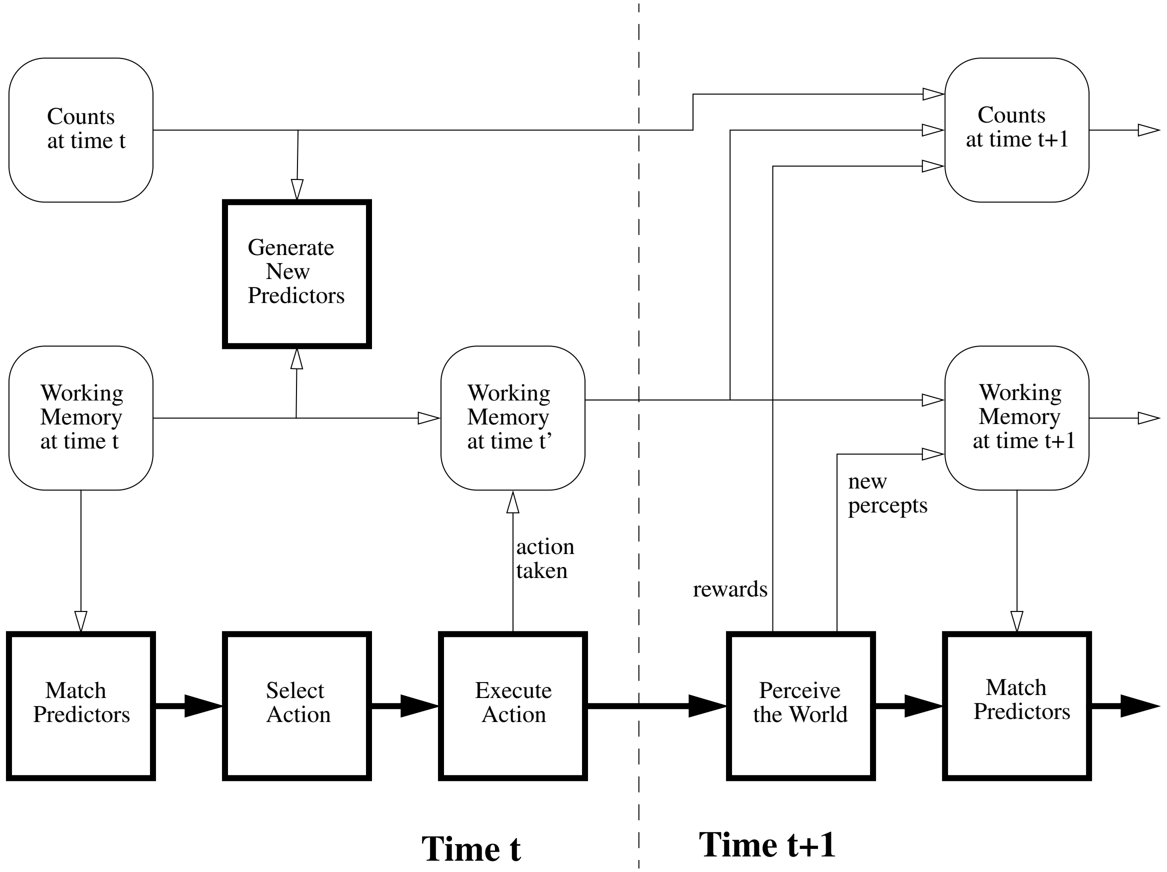

Figure 1: Flow of information in the learning program.

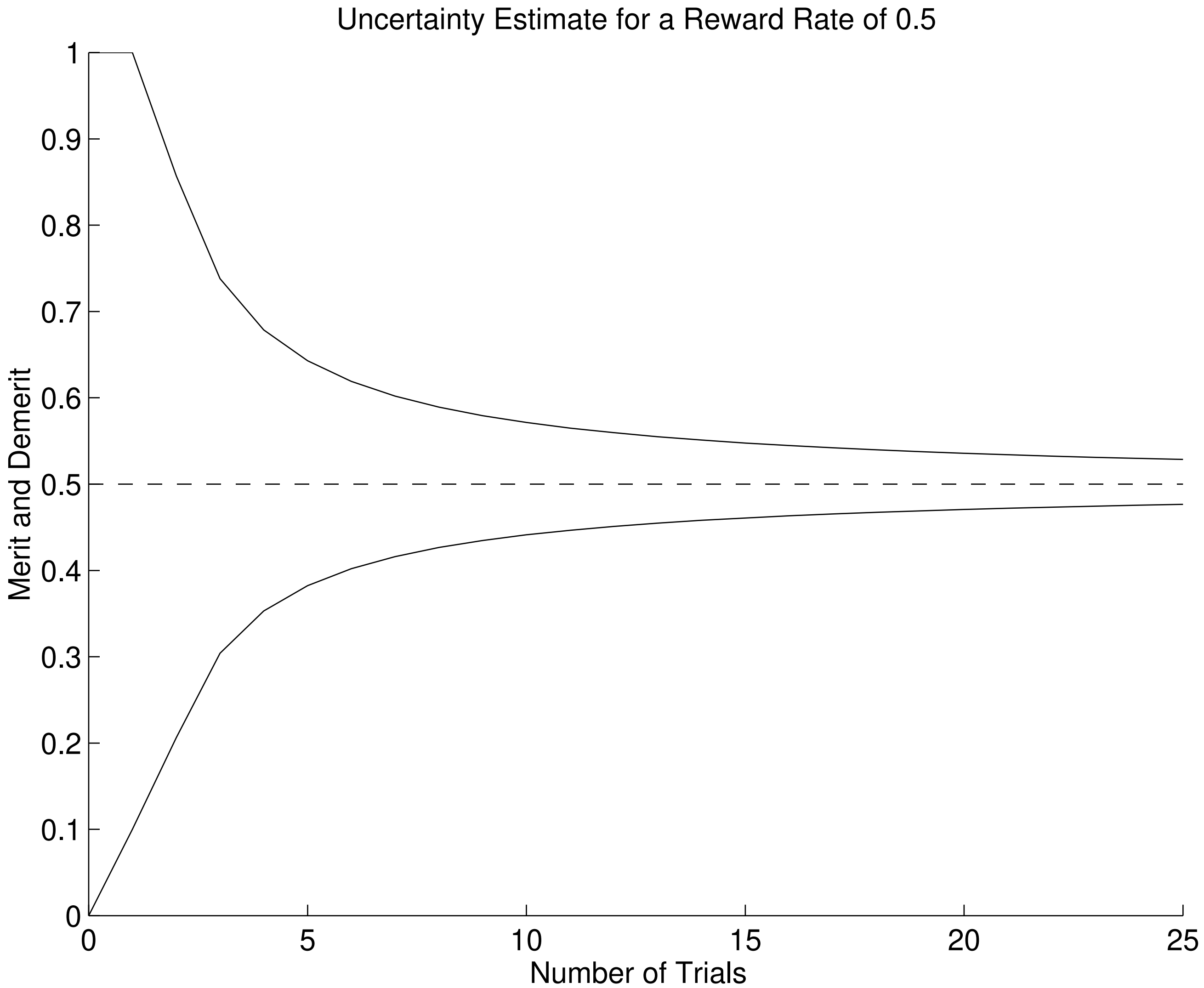

Figure 2: Merit and demerit curves converge toward the true reward rate as the number of samples increases.

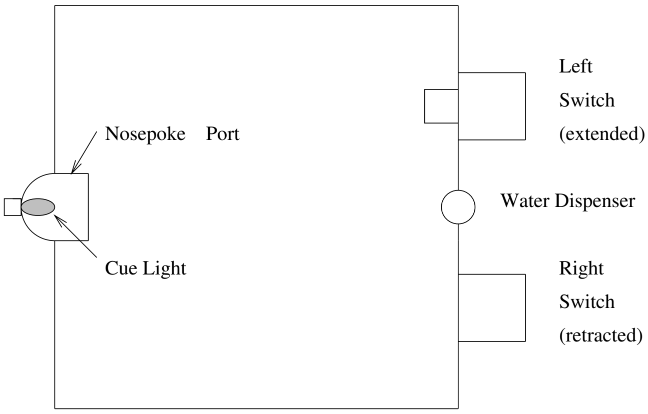

Figure 3: Skinner box configured for DMTS task.

$\text{RECEIVE(water)} \leftarrow \text{HEAR(pump) \& GOTO(dispenser)}$

$\text{HEAR(pump)} \leftarrow \text{PRESS(switch1):past \& PRESS(switch1)}$
$\text{HEAR(pump)} \leftarrow \text{PRESS(switch2):past \& PRESS(switch2)}$

$\text{SEE(light)} \leftarrow \text{NOT(reinforced):past \& PRESS(switch1)}$
$\text{SEE(light)} \leftarrow \text{NOT(reinforced):past \& PRESS(switch2)}$

$\text{SEE(switch1)} \leftarrow \text{SEE(light) \& POKE(poke-port)}$
$\text{SEE(switch2)} \leftarrow \text{SEE(light) \& POKE(poke-port)}$

$\text{CAN(PRESS(switch1))} \leftarrow \text{SEE(switch1) \& GOTO(sw1-loc)}$
$\text{CAN(PRESS(switch2))} \leftarrow \text{SEE(switch2) \& GOTO(sw2-loc)}$

$\text{CAN(POKE(poke-port))} \leftarrow \text{AT(port-loc):fut}$

Figure 4: Optimal predictors for the DMTS task.

```
2981: NOT(reinforced) AT(sw1-loc) -293-> GOTO(port-loc)

2982: CAN(POKE(poke-port)) NOT(reinforced) AT(port-loc) -293-> GOTO(sw1-loc)

2983: SEE(switch2) AT(sw1-loc) -148-> GOTO(sw2-loc)
      switch2: extended

2984: CAN(PRESS(switch2)) SEE(switch2) AT(sw2-loc) -107-> PRESS(switch2)
      switch2: extended

2985: SEE(light) AT(sw2-loc) -230-> GOTO(port-loc)
      light: on

2986: CAN(POKE(poke-port)) SEE(light) AT(port-loc) -299-> POKE(poke-port)
      light: on

2987: CAN(POKE(poke-port)) SEE(switch1) SEE(switch2) AT(port-loc) -44-> GOTO(sw2-loc)  [req by 211]
      switch1: extended   switch2: extended

2988: CAN(PRESS(switch2)) SEE(switch1) SEE(switch2) AT(sw2-loc) -211-> PRESS(switch2)
      switch1: extended   switch2: extended

2989: HEAR(pump) AT(sw2-loc) -25-> GOTO(dispenser)
      pump: running

2990: AT(dispenser) RECEIVE(water) -C-> GOTO(sw1-loc)
```

Figure 5: Performance on trial 170 of the DMTS task. The number embedded in each arrow is the predictor being applied; a "C" indicates no predictor is satisfied in the present situation (clueless).

293: SEE(switch1) $\leftarrow$ AT(port-loc):past & NOT(reinforced):prev & GOTO(sw1-loc):prev

148: SEE(light) $\leftarrow$ NOT(reinforced):past & SEE(switch2):prev & GOTO(sw2-loc):prev

107: SEE(light) $\leftarrow$ NOT(reinforced):past & PRESS(switch2)

230: SEE(switch1) $\leftarrow$ AT(sw1-loc):past & SEE(light):prev & GOTO(port-loc):prev

299: SEE(switch2) $\leftarrow$ SEE(light):prev & POKE(poke-port)

44: CAN(PRESS(switch2)) $\leftarrow$ SEE(switch2) & GOTO(sw2-loc)

211: HEAR(pump) $\leftarrow$ PRESS(switch2):past & SEE(switch1) & PRESS(switch2)

25: RECEIVE(water) $\leftarrow$ HEAR(PUMP) & AT(A2):fut

Figure 6: The actual predictors responsible for the program's behavior on trial 170, shown in the order in which they are applied.

(black and white photograph to be supplied separately)

Figure 7: Amelia performing the toy sorting task. The human trainer issues rewards using the radio trackball in her left hand.

Age	Sensory Predicates	Action Predicate
2	AT(ne-corner)	GOTO(se-corner)
1	AT(se-corner)	
	HEAR(pump)	GOTO(dispenser)
0	AT(dispenser)	
	RECEIVE(water)	

Table 1: State of working memory at the moment water is received.

Tag	Meaning
:fut	True at $t+1$
:now	True at t
:prev	True at $t-1$
:past	True at $t' < t$

Table 2: Tags used to construct temporal predicates.