A Theoretical Model of Behavioral Shaping

Manu Chhabra[1] **(mchhabra@cs.rochester.edu)**
Daniel Stefankovic[1] **(stefanko@cs.rochester.edu)**
Robert A. Jacobs[2] **(robbie@bcs.rochester.edu)**
Departments of Computer Science[1] and Brain & Cognitive Sciences[2]
University of Rochester
Rochester, NY 14627 USA

Abstract

Behavioral shaping is an incremental training procedure commonly used to teach complex behaviors. Using this procedure, a learner is initially rewarded for producing coarse approximations of the target behavior. Over time, only more refined approximations are rewarded until, finally, the learner receives reward only when the target behavior is produced. In this paper, we mathematically formalize the notion of behavioral shaping in the context of search problems. In a search problem, an agent uses the membership oracle of a target concept to find a positive example of the concept. When the concepts are intervals on the real line, we show that the use of a shaping sequence—a sequence of increasingly restrictive concepts leading to the target concept—exponentially decreases the number of queries required to solve the search problem. We also show that there does not exist an algorithm which can solve the search problem using a smaller number of queries. Lastly, we conjecture that convexity may be an important requirement for a shaping procedure to be helpful.

Keywords: learning; training; shaping; computational learning theory

Introduction

Behavioral shaping is a training procedure commonly used to teach complex behaviors. Using this procedure, a complex task is taught to a learner in an incremental manner. The learner is initially rewarded for performing an easy task that coarsely resembles the target task that the teacher wants the learner to perform. Over time, the learner is rewarded for performing more difficult tasks that monotonically provide better approximations to the target task. At the end of the training sequence, the learner is rewarded only for performing the target task. Shaping was first proposed by B. F. Skinner in the 1930s (Skinner, 1938). In one experiment, Skinner demonstrated that pigeons could be trained to move in a circle. Initially, any movement to the left was rewarded. When the pigeon acquired this behavior, only larger movements were rewarded, and so on. Eventually, the pigeon learned to move in a full circle. In recent decades, shaping has been used to train animals (including people) to perform tasks that they will not learn to perform through direct reinforcement (i.e., by only rewarding the target behavior). Shaping has also been used in the field of artificial intelligence, especially in the area of machine learning known as reinforcement learning (Sutton & Barto, 1998), to train agents to perform complex tasks (Dorgio & Colombetti, 1994; Ng, Harada, & Russell, 1999; Randlov, 2000).

This paper has two goals. The primary goal is to mathematically formalize the notion of shaping, and to show that shaping makes certain tasks easier to learn. We specifically concentrate on tasks requiring search in which the objective of an agent (either biological or artificial) is to find states in a search space which are rewarded. Intuitively, we ask whether searching for a reward state is easier when a teacher is available to guide the search. The secondary goal is to make a methodological contribution to the field of Cognitive Science by illustrating how concepts and techniques from the field of Computational Learning Theory (Anthony & Biggs, 1992; Kearns & Vazirani, 1994) can be used to formalize and study cognitive phenomena. A more detailed description of this work, and several additional results, can be found in Chhabra, Jacobs, and Stefankovic (2007).

Consider the following task in which an agent has to find a reward state in a one-dimensional array of states. There is an array of size n whose elements are filled with the values of a reward function; i.e., the elements are filled with zeros, except at some random, fixed location with index T (for target), which has a one. The agent's goal is to find this reward location. The agent can query any element of the array. Clearly, to find the reward location, $O(n)$ queries are needed in the worst case. Consider, now, the availability of a teacher to guide the agent's search. The learning process proceeds in iterations. At iteration 1, the teacher fills a contiguous region of size $n/2$ in the array with 1s (with the constraint that the actual target reward location T is within this region). All other elements in the array are assigned a zero. When the agent queries an element containing a 1, the current iteration ends and the learning process moves to the next iteration. At iteration 2, the teacher reduces the region containing 1s to a contiguous subregion of size $n/4$ (the subregion at iteration 2 is a subset of the subregion at iteration 1 with the constraint that it also contains location T). Eventually, after $\lceil \log_2 n \rceil$ iterations, the teacher assigns a 1 only to element T, the actual target reward location. Assuming that the agent knows that the teacher will shrink the "reward area" by 1/2 at each iteration, it is easy to show that the reward location can be found with just $O(\log_2 n)$ queries. Importantly, this is an exponential improvement in performance relative to the case when there is no teacher available to guide the agent's search.

Although this paper analyzes shaping with respect to its benefits on search problems, the reader should recognize that shaping is often intimately related to reinforcement learning. The objective in reinforcement learning is to find a policy (i.e., a mapping from states to actions) that maximizes the reward obtained. In general, searching the policy space for an optimal policy (or even a good policy) is computationally intractable. However, it is possible that an agent can learn to perform a task faster if a teacher is available that adaptively modifies the reward function in the manner suggested by behavioral shaping. For example, consider a sequential decision problem in which an agent attempts to reach the reward state by choosing one of two actions at each moment in time. The agent chooses sequences of actions of length $\lceil \log_2 n \rceil$, and then is told whether the resulting state is the reward state. This situation can be characterized by a binary tree in which nodes correspond to states and edges correspond to actions. The tree has n leaf nodes, one of which corresponds to the reward state, and the tree's depth is $\lceil \log_2 n \rceil$. Consequently, a policy is a sequence of $\lceil \log_2 n \rceil$ actions. The optimal policy is the sequence leading to the leaf corresponding to the reward state. This scenario is analogous to the scenario described above; instead of searching for a reward element in an array, the agent now searches for a sequence of actions leading to a reward leaf of a tree. Using the actual target reward function, the agent accumulates $O(n)$ regret in the worst case where regret is defined as the total number of action sequences that the agent tries before discovering the optimal sequence leading to the reward state. What if a teacher is available that modifies the reward function in the manner suggested by behavioral shaping? Assuming that the teacher marks 1/2 of the leaves with reward at iteration 1 and successively halves the reward area at each iteration (as described above), then the agent can learn the optimal policy with only $O(\log_2 n)$ regret. Again, this is an exponential improvement in performance relative to the case where there is no teacher available to guide the agent's search.

Our work is related to the "reward shaping" framework of Ng, Harada, & Russell (1999). These authors sought to speed-up reinforcement learning by transforming the original reward function to a new reward function that provides additional training information to guide an agent's search. They gave mathematical conditions under which a transformation is policy invariant meaning that an optimal policy for the original reward function is also an optimal policy for the new reward function. Our approach is different from theirs in at least two ways. First, we use a temporal sequence of reward functions as compared to their fixed reward transform. Consequently, we believe that our approach is more consistent with the practice of behavioral shaping as reported in the psychological literature (Skinner, 1938). Second, our reward functions are binary, whereas those of Ng et al. allow real-valued rewards.

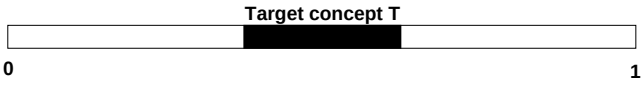

Figure 1: The target concept T is a subset of the concept $S = [0, 1]$. The search agent needs to search S for a point in T.

In the next section, we describe a formal model of behavioral shaping. This model can be used to prove the benefits of shaping in several cases involving convex reward regions lying in multi-dimensional spaces. For pedagogical reasons, the following section focuses on a simple one-dimensional case where we prove that shaping can be helpful in finding intervals on a straight line. We show a lower bound on regret when shaping is used, and a learning algorithm which hits that bound. Lastly, we show that the convexity of reward regions is an important requirement of our framework.

Formal Model

Let $(X, \mathcal{B})$ be a measurable space with a measure μ (see Doob, 1994, for an introduction to mathematical measure theory). Let $\mathcal{C} \subseteq \mathcal{B}$. The set $\mathcal{C}$ is called a *concept class* and its members are called *concepts*. Examples of concept classes that we study include intervals in $\mathbb{R}$, axis-parallel rectangles in $\mathbb{R}^d$, balls in $\mathbb{R}^d$, and convex bodies in $\mathbb{R}^d$. We will assume that there is a representation scheme for the concept class $\mathcal{C}$ (Kearns & Vazirani, 1994).

Definition 1 (Search problem)**.** *Let $\mathcal{C}$ be a concept class. A search problem is a pair of concepts (S, T) such that $T \subseteq S$, and $S, T \in \mathcal{C}$.*

A search agent is given a representation of S and has to find a point in T using the membership oracle of T. This scenario is illustrated in Figure 1 where the target concept T is a subset of the concept $S = [0, 1]$. The number of oracle queries until a point is found in T is defined as the regret. Throughout this paper, without loss of generality, we assume that $\mu(S) = 1$, and $\mu(T) = 1/R$, where $R \geq 1$. Note that for any concept class there is a natural randomized algorithm to solve the search problem: query independent uniform random points from S until you find a point in T. The expected regret of the algorithm is R. For sufficiently complicated concept classes (e.g., finite unions of intervals), the use of randomness might be inevitable because a deterministic algorithm with bounded regret need not exist.

In a *shaped search problem* the agent's search task will be aided by a *shaping sequence* which is a sequence of nested sets between S and T. The sets in the shaping sequence will be gradually shrinking concepts from the underlying concept class $\mathcal{C}$. The rate of shrinking will

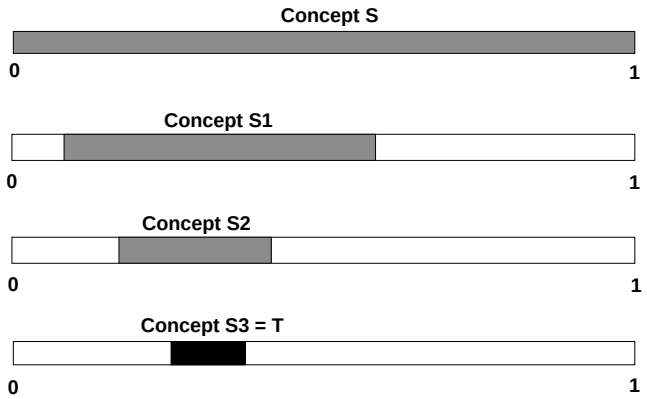

Figure 2: This figure illustrates the use of a sequence of nested concepts. Intuitively, the agent initially searches concept S for a point that lies in concept S_1. Next, the agent searches in the neighborhood of the discovered point in S_1 for a point that lies in S_2, and then searches in the neighborhood of the discovered point in S_2 for a point that lies in S_3 which is the target concept T.

be a parameter denoted γ. This scenario is illustrated in Figure 2 which shows a sequence of nested concepts. Intuitively, the agent initially searches concept S for a point that lies in concept S_1. Next, the agent searches in the neighborhood of the discovered point in S_1 for a point that lies in S_2, and then searches in the neighborhood of the discovered point in S_2 for a point that lies in S_3 which is the target concept T.

Definition 2 (Shaped search problem)**.** *Let $\mathcal{C}$ be a concept class. A shaped search problem is defined by the tuple $(S, T, \gamma, S_1, \ldots, S_k)$ such that $S, T, S_1, \ldots, S_k \in \mathcal{C}$, $\frac{1}{R} \leq \gamma < 1$, $T = S_k \subseteq S_{k-1} \subseteq \ldots S_1 \subseteq S$, $\mu(S_1) = \gamma\mu(S)$, $\mu(S_{i+1}) = \gamma\mu(S_i)$ for all $i = 1, \ldots, k-2$. k is such that $\mu(S_{k-1}) \geq 1/R \geq \gamma\mu(S_{k-1})$; i.e., $k = \lceil \log_{\frac{1}{\gamma}} R \rceil$. The sequence $(S_1, \ldots, S_{k-1})$ is called a γ shaping sequence.*

An agent in a shaped search problem setting is given a representation of S and has access to the membership oracles $O_1, \ldots, O_k$ of $S_1, \ldots, S_k$, respectively. However, if the agent makes a query to O_i, it can no longer make a query to any O_j such that $j < i$. In other words, the oracles O_i are presented to the agent in k iterations, with the agent making (zero or more) queries only to oracle O_i at iteration i. The agent has to find a point in T by making queries to the oracles. In this context, the regret is defined as the total number of queries made to all oracles until a point in T is found.

Note that an agent solving the shaped search problem cannot have a regret larger than what its regret would be on the original search problem; the agent can always choose to ignore the shaping sequence and only use the last membership oracle O_k for $S_k = T$. Because a shaping sequence provides extra information about the location of T (through the membership oracles $O_1, \ldots, O_{k-1}$), an agent with access to a shaping sequence can potentially achieve a smaller regret than would otherwise be the case. For certain classes of search problems, for example, it might be possible to approximately determine S_i using oracle O_i. If S_i is approximately located, then S_{i+1} could be approximately located (via queries to O_{i+1}) by searching only inside the approximation of S_i rather than inside S. As S_i has a smaller measure than S, this search may require a relatively small number of queries. This process can be used iteratively from $i = 1$ to $i = k - 1$ to find a point inside $S_k = T$. In the next section we show that it is indeed possible to reduce an agent's regret using this method when S and T are intervals on a straight line.

The regret accumulated by the agent during the search also depends on the value of γ. Very small (close to 0) γ values are detrimental because the agent will need to search a relatively large region for a point lying in a much smaller region [e.g., $\mu(S_1)$ will be a small fraction of $\mu(S)$], meaning that the agent will need to search for a "needle in a haystack". Large values of γ (close to 1) may seem detrimental (because the teacher will take a long time to converge to T), but are actually not as we discuss below.

Note that our approach is related to a technique from the mathematical optimization literature known as quasi-convex optimization (Boyd & Vandenberghe, 2004). A weaker version of the shaped search problem in which the concepts are convex and all the oracles are available to the agent simultaneously can be viewed as an instance of a quasi-convex optimization problem. However, in our approach, the oracles are not available simultaneously but, rather, are available as a temporal sequence. This is because behavioral shaping almost always proceeds in a temporal fashion (typically, rewards are initially provided to coarse approximations to the target behavior, and then only finer approximations are rewarded at later stages of training).

Finding a Point in an Interval

To keep our discussion relatively simple, this section focuses on the case where concepts are intervals on the real line. Before analyzing the effectiveness of using a shaping sequence in this circumstance, we consider learning in the absence of such a sequence.

Consider a search problem (S, T) such that $S = [0, 1] \subset \mathbb{R}$ is a closed interval of length 1, and $T \subset S$ is an interval of size $1/R$. The search agent has access to O, which is the membership oracle of T. Consider the following simple deterministic algorithm to find a point inside T. The algorithm starts with the leftmost point in S, and query points $0, 1/R, 2/R, 3/R, \ldots, 1$ (as a matter of terminology, we say that the algorithm "makes jumps of size $1/R$ in S"). As T is a contiguous, closed interval of

size $1/R$, the algorithm is guaranteed to hit one point inside T. As T has size $1/R$, there will be at most $O(R)$ queries. Therefore the regret will be at most $O(R)$.

The previous algorithm assumed that R is known. Next, we show that even without this information, it is possible to solve the problem with $O(R)$ regret. To do this, the agent first sets $\hat{\mu}(T)$, its estimate of the length of T, to 1 and makes queries at points 0 and 1. If it hits a point in T, it stops; otherwise it halves the value of $\hat{\mu}(T)$ to 1/2, and queries points 0, 1/2, and 1. The algorithm keeps halving $\hat{\mu}(T)$, and keeps making jumps of size $\hat{\mu}(T)$ in S, until it hits a point inside T. In general, if the agent estimates $\hat{\mu}(T) = \frac{1}{2^k}$, it queries less than 2^{k+1} points. Importantly, there is a nonnegative integer k_0 such that $\frac{1}{2^{k_0+1}} \leq 1/R \leq \frac{1}{2^{k_0}}$. When the agent guesses $\hat{\mu}(T) = \frac{1}{2^{k_0+1}}$, it is guaranteed to hit a point in T. The total number of queries to O will be $2 + \cdots + 2^{k_0+2} = 2^{k_0+3} - 2$ which is less than $8R$. Therefore, there is an algorithm to solve the search problem with regret $O(R)$ even if the agent does not know the value of R and, thus, does not know the size of T. Throughout the rest of the paper, we assume that the agent knows the value of R and knows that the size of T is $1/R$.

We now show that no deterministic algorithm can find a point in T with less than $\lfloor R-1 \rfloor$ regret. Suppose a deterministic algorithm has a regret $r < \lfloor R-1 \rfloor$. Without loss of generality, assume that all the r queries to the oracle are distinct. These r distinct points will induce $r+1$ intervals in S. The average size of the interval will be $\frac{1}{r+1}$. Therefore, at least one interval will be of size at least $\frac{1}{r+1}$. As $r < \lfloor R-1 \rfloor$, at least one interval will be of size at least $1/R$ which means that there is at least one way in which T can be placed such that the agent will not be able to query any point inside T if it makes less than $\lfloor R-1 \rfloor$ queries. Using a similar argument, it can be shown that any randomized algorithm will accumulate $\Omega(R)$ regret. This gives us the following proposition:

Proposition 1. *If the concept class $\mathcal{C}$ is the set of closed intervals then there is an algorithm that can solve the search problem (S, T) with $O(R)$ regret where $|S| = 1$ and $|T| = 1/R$. Further, any (randomized) algorithm that solves the search problem will have $\Omega(R)$ regret.*

We now turn our attention to the case where a shaping sequence is available. In summary, we show that when any γ shaping sequence $S_1, \ldots, S_{k-1}$ is available such that the S_i are closed intervals, it is possible to find a point in T with a regret which is logarithmic in R. Suppose a shaping sequence $S_1, \ldots, S_{k-1}$ of nested intervals is available. We present two algorithms which make use of this sequence to find a point in T. Algorithm 1 solves the problem with $O(\frac{1}{\gamma} \log_{\frac{1}{\gamma}} R)$ regret for $\frac{1}{R} \leq \gamma < 1$. Although this regret is logarithmic in R for a fixed γ, it goes beyond the $O(R)$ regret when γ approaches 1. Algorithm 2 fixes this problem and achieves a regret of $O(\log_2 R)$ for $\gamma > 1/2$.

We start with Algorithm 1. At the first iteration, the algorithm's goal is to find a point inside S_1. The algorithm does not know where S_1 lies, but it knows that S_1 is of size γ, so it makes jumps of size γ in S and is guaranteed to hit a point inside S_1. At the first iteration, the algorithm makes at most $\lceil \frac{1}{\gamma} \rceil$ queries to oracle O_1. At the i^{th} iteration (where $i > 1$), the algorithm's goal is to find a point in S_i. The algorithm has a point p which lies in S_{i-1} (from the previous iteration). The interval S_i of size γ^i can contain points lying on either side of p but, because $S_i \subset S_{i-1}$, its left (right) edge can lie at most γ^{i-1} to the left (right) of p. The algorithm makes $\lceil \frac{1}{\gamma} \rceil$ jumps of size γ^i to the left and right of p, and hits a point in S_i. Therefore, at each iteration, at most $2\lceil \frac{1}{\gamma} \rceil$ queries are made. As there are $\lceil \log_{\frac{1}{\gamma}} R \rceil$ iterations, the total regret is $2\lceil \frac{1}{\gamma} \rceil \lceil \log_{\frac{1}{\gamma}} R \rceil = O(\frac{1}{\gamma} \log_{\frac{1}{\gamma}} R)$. For a fixed value of γ, this is an exponential improvement over the case when a shaping sequence is not available. However, when γ is close to 1, this regret exceeds $O(R)$ (as discussed below, Algorithm 2 solves the problem for large γ by skipping oracles). This gives us the following theorem:

Theorem 1. *If the concept class $\mathcal{C}$ is the set of closed intervals then there is an algorithm that can solve the shaped search problem $(S, T, \gamma, S_1, \ldots, S_k)$ for any γ shaping sequence $S_1, \ldots, S_k$ with $O(\frac{1}{\gamma} \log_{\frac{1}{\gamma}} R)$ regret where $|S| = 1$ and $|T| = 1/R$.*

Algorithm 1 An algorithm to solve the shaped search problem $(S, T, \gamma, S_1, \ldots, S_k)$ such that $T, S, S_1, \ldots, S_k \subset \mathbb{R}$ using the sequence of oracles $O_1, \ldots, O_k$.

Require: $1/R \leq \gamma < 1$, $S = [0, 1]$, membership oracles $O_1, \ldots, O_k$ of the sets $S_1, \ldots, S_k$.

```
p ⇐ 0
for i = 1 to k do
    if i=1 then
        query O_1 with points 0, γ, 2γ ...
        p ← first point at which O_1 outputs 1.
    else
        Query oracle O_i with points p − γ^{i−1}, ..., p − 2γ^i, p − γ^i, p, p + γ^i, p + 2γ^i, ..., p + γ^{i−1}
        p ← first point at which O_i outputs 1.
    end if
end for
return p
```

An interesting aspect of our analysis is that it reveals a number of issues regarding the shrinking rate parameter γ. First, the above algorithm assumes that γ is known. Importantly, it can be shown that even if γ is unknown, it is still possible to solve the shaped search

problem with $O(\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R)$ regret. The method to do so is very similar to the one described above which solved the search problem (S,T) when R was unknown (and, thus, will not be described here).

Second, there exists an optimal value for γ that minimizes $\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R$. This value is $\gamma = 1/e$. This result is obtained by differentiating $\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R$ with respect to γ, setting the derivative to zero, and solving for γ.

Lastly, there is a trade-off involving the value of γ. If $\gamma = \frac{1}{R}$, then $k = 1$ and the shaping sequence is non-existent (i.e., $S_1 = T$) meaning that the regret becomes $O(R)$. If γ is close to one, then k is large and the shaping sequence is very long. In this case, the regret increases rapidly to infinity as γ approaches 1. Consequently, Algorithm 1 is not optimal because, when γ is large, the agent could simply ignore the shaping sequence and make queries only to the final oracle O_k which will lead to $O(R)$ regret. Fortunately, it is possible to do better than this. When γ is large, the agent can choose to make queries only to oracles $O_s, O_{2s}, \ldots, O_k$ where s is the first integer such that $\gamma^s < 1/2$ (for simplicity, we assume that k is a multiple of s). In other words, the strategy of a new algorithm, denoted Algorithm 2, is to use sets of the shaping sequence that are successively shrinking by a factor of about 1/2, and ignore other elements of the shaping sequence. In this way, γ is effectively reduced to 1/2 for the agent, and the regret reduces to $O(\log_2 R)$. We next prove a lower bound on regret, and show that Algorithm 2 achieves this lower bound. That is, we show that no algorithm can do better than Algorithm 2 (up to a constant factor).

Theorem 2. *If the concept class $\mathcal{C}$ is the set of closed intervals then there is a deterministic algorithm that can solve the shaped search problem $(S,T,\gamma,S_1,\ldots,S_k)$ with $O(\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R)$ regret for $\gamma < 1/2$ and with $O(\log_2 R)$ regret for $\gamma \geq 1/2$ for any shaping sequence $S_1,\ldots,S_k$ where $|S| = 1$ and $|T| = 1/R$. Further, there is no algorithm, deterministic or otherwise, that can solve the shaped search problem (with S_i being closed intervals) with smaller regret (up to a constant factor).*

Proof. First we show an information-theoretic lower bound on regret which is independent of γ. For a given R, we can place R intervals of size $1/R$ next to each other inside an interval of size 1. To encode the identity of a particular interval, we need $\log_2 R$ bits. As the oracles are providing one bit of information at each query, any algorithm that successfully solves the problem must make $\Omega(\log_2 R)$ total queries to the oracles.

We now show that Algorithm 2 hits this bound when $\gamma > \frac{1}{2}$, meaning that the bound is tight for $\gamma > \frac{1}{2}$. When $\gamma > \frac{1}{2}$, the algorithm sets $s = \lceil \frac{1}{\log_2 \frac{1}{\gamma}} \rceil$ (without loss of generality, assume that $\gamma^s = 1/2$) and calls Algorithm 1 with the oracles $O_s, O_{2s}, \ldots, O_k$. Therefore the algorithm is solving the shaped search problem $(S,T,1/2,S_s,S_{2s},\ldots,S_k)$ using Algorithm 1, which solves the search problem with $O(\log_2 R)$ regret.

When $\frac{1}{R} \leq \gamma < \frac{1}{2}$, the bound on regret depends on γ. Note that Algorithm 2 calls Algorithm 1 when $\gamma < \frac{1}{2}$, meaning that the total regret is $O(\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R)$. We show that this is also a lower bound when $\gamma < \frac{1}{2}$. In Algorithm 1, suppose that at each iteration i, the location of the interval S_i is fully revealed if the algorithm makes more than $\lfloor \frac{1}{\gamma} - 1 \rfloor$ queries. We show that even with this extra information, the total regret to find a point in the interval T is still $O(\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R)$.

At iteration i, the interval S_i of size γ^i must lie within the interval S_{i-1} of size γ^{i-1}. For each i, we are supposing that the algorithm knows S_{i-1} exactly, as it was revealed at the end of the previous iteration. Without loss of generality, we assume the agent queries only inside S_{i-1} at iteration i. If the algorithm queries t points, it induces $t+1$ intervals on S_{i-1}. The average interval size of these intervals will be $\frac{\gamma^{i-1}}{t+1}$. This means that there will be at least one interval which will be at least the average size. If $t < \lfloor \frac{1}{\gamma} - 1 \rfloor < \frac{1}{\gamma} - 1$, then $\frac{\gamma^{i-1}}{t+1} > \gamma^i$. In other words, there is at least one interval which will be of size at least γ^i. As S_i might lie anywhere in S_{i-1}, the alleged algorithm is not guaranteed to be able to find S_i if S_i lies in this interval of size at least γ^i. Consequently, any correct search algorithm will have to make at least $\lfloor \frac{1}{\gamma} - 1 \rfloor = O(\frac{1}{\gamma})$ queries at each iteration and, thus, the total regret is bounded from below by $\Omega(\frac{1}{\gamma}\log_{\frac{1}{\gamma}} R)$. A similar argument holds for randomized algorithms. □

Algorithm 2 An optimal algorithm to solve the shaped search problem $(S,T,\gamma,S_1,\ldots,S_k)$ such that $T,S,S_1,\ldots,S_k \subset \mathbb{R}$ using the sequence of oracles $O_1,\ldots,O_k$.

Require: $1/R \leq \gamma < 1$, $S = [0,1]$, membership oracles $O_1,\ldots,O_k$ of the sets $S_1,\ldots,S_k$.

1. if $\gamma \leq \frac{1}{2}$, call Algorithm 1 and return p, otherwise continue.
2. $s = \lceil \frac{1}{\log_2 \frac{1}{\gamma}} \rceil$
3. call Algorithm 1 with $\gamma = 1/2$ and membership oracles $O_s, O_{2s}, \ldots, O_k$ and return p.

Convexity is Important

In Theorem 2 we showed that if the concept class is the set of closed intervals, it is possible to solve the shaped search problem with smaller regret than required to solve the original search problem. In this section, we show that if the members of the concept class are a union of two closed intervals, then there are shaped search problems such that, for a fixed γ, it is not possible to find a

point in T with less than $O(R)$ regret for certain shaping sequences.

Consider the following example with $\gamma = 1/2$. Suppose each set S_i in the shaping sequence consists of two segments. The first segment is the target interval T, and it remains fixed throughout all sets in the shaping sequence. The second segment shrinks at a rate such that successive sets in the sequence maintain a size ratio of $1/2$. In the last iteration, this second segment vanishes.

More formally, let $S = [0, 1]$ and $\gamma = 1/2$. The shaping sequence $S_1, \ldots, S_{k-1}$ is the sequence of sets $S_i = T \cup Q_i, i = 1, \ldots, k-1$, where $Q_i = [0, 1/2^i - 1/R]$, $T = [l, l + 1/R]$, and $1/2 \leq l \leq 1 - 1/R$. In this case, even if the teacher fully reveals the location of the segment Q_i to the agent at each iteration, the agent still needs to locate the set T, which can lie anywhere in a region of size $1/2$. Hence, to find a point in T, $O(R)$ queries are needed.

We have just shown that there are shaping sequences which do not exponentially reduce an agent's regret if the concept class is the set of unions of two closed intervals. Although this example assumes a fixed value of γ, similar results can be shown for arbitrary values of γ when the concept class consists of more complicated non-convex concepts. For example, if the concept class is the set of unions of three closed intervals, an exponential reduction in regret is not possible for any value of γ.

We conjecture that convexity is an important requirement for shaped search. That is, we conjecture that when an agent seeks to discover a concept through a shaped search procedure, it will only be able to achieve a small regret when the concept is convex. In fact, all our additional results regarding shaped search in multi-dimensions are for concept classes consisting of convex bodies (Chhabra, Jacobs, & Stefankovic, 2007).

Summary

Shaping is a commonly used procedure for teaching complicated tasks to people, animals, and robots. Both behavioral experiments and computer simulations have demonstrated that learners trained via shaping achieve significant improvements in learning speeds.

In this paper, we mathematically formalized a model of shaping, and studied it in the context of search problems. To keep our discussion simple, we focused on the one–dimensional case where concepts are intervals on the real line. When a shaping sequence is available, the search problem can be solved with exponentially less regret than would otherwise be possible. We also showed that there do not exist algorithms which can solve the search problem using a smaller number of queries. Our analysis revealed a number of interesting issues regarding the shrinking rate parameter γ. Lastly, we conjectured that it is important that concepts are convex for shaping sequences to be useful. We showed that when the concept class is the set of unions of two intervals, there are shaping sequences which do not reduce regret.

The results presented here form the foundation for additional results reported in a longer article (Chhabra, Jacobs, and Stefankovic, 2007). In this article, we study the cases where concepts are rectangles, ellipsoids, or general convex bodies in high dimensions. In multi-dimensions, new methods are required to create efficient search algorithms.

Acknowledgments

This work was supported by AFOSR research grant FA9550-06-1-0492.

References

Anthony, M. & Biggs, N. (1992). *Computational Learning Theory.* Cambridge, UK: Cambridge University Press.

Boyd, S. & Vandenberghe, L. (2004). *Convex Optimization.* Cambridge, UK: Cambridge University Press.

Chhabra, M., Jacobs, R. A., & Stefankovic, D. (2007). Behavioral shaping for geometric concepts. *Journal of Machine Learning Research*, in press.

Doob, J. L. (1994). *Measure Theory.* New York: Springer-Verlag.

Dorigo, M. & Colombetti, M. (1994). Robot shaping: Developing autonomous agents through learning. *Artificial Intelligence Archive*, **71**, 321-370.

Kearns, M. J. & Vazirani U. V. (1994). *An Introduction to Computational Learning Theory .* Cambridge, MA: MIT Press.

Ng, A., Harada, D., & Russell, S.(1999). Policy invariance under reward transformations: Theory and application to reward shaping. In I. Bratko (Ed.), *Proceedings of the 16th International Conference on Machine Learning.* San Francisco: Morgan Kaufmann.

Randlov J. (2000). Shaping in reinforcement learning by changing the physics of the problem. In P. Langley (Ed.), *Proceedings of the 17th International Conference on Machine Learning.* San Francisco, CA: Morgan Kaufmann.

Skinner, B. F. (1938). *The Behavior of Organisms.* New York: Appleton-Century-Crofts.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction.* Cambridge, MA: MIT Press.