An Introspection Approach to Querying a Trainer

Jeffery A. Clouse

Technical Report 96-13
January 22, 1996

Department of Computer Science
Lederle Graduate Research Center
University of Massachusetts
Amherst, MA 01003-4610
clouse@cs.umass.edu

Abstract

This paper introduces the Introspection Approach, a method by which a learning agent employing reinforcement learning can decide when to ask a training agent for instruction. When using our approach, we find that the same number of trainer's responses produced significantly faster learners than by having the learner ask for aid randomly. Guidance received via our approach is more informative than random guidance. Thus, we can reduce the interaction that the training agent has with the learning agent without reducing the speed with which the learner develops its policy. In fact, by being intelligent about when the learner asks for help, we can even increase the learning speed for the same level of trainer interaction.

Contents

1 Introduction

In learning to perform a multiple-step task (*e.g.* problem-solving and control tasks), an automated agent develops a decision policy—a mapping from states to actions—that specifies for each state which action to perform. Instead of utilizing a set of labeled training examples, necessary for supervised learning, an agent employing reinforcement learning methods (Sutton, 1984; Watkins, 1989; Barto, Bradtke & Singh, 1995) adapts based on evaluative feedback that indicates how well the agent is performing the task. Unfortunately this feedback, which is in the form of sparse scalar payoff, is only weakly informative; certainly not as instructive as supervised training.

In recognition of this deficiency of reinforcement learning, several researchers have recently studied systems in which a training agent is added to the learning scenario (Clouse & Utgoff, 1992; Lin, 1992; Maclin & Shavlik, 1994; Gordon & Subramanian, 1994; Clouse, 1995). In such systems, rather than relying solely on the simple signals provided, the learning agent also has access to supervised instruction.

While it is not surprising that a learning agent employing a reinforcement learning method will be aided by additional information about its task, it is not well understood how the agent can acquire and use the information. Among the many problems associated with adding a training agent to the learning scenario, such as the form of the instruction and the manner in which the learner adapts based on the instruction, we focus on when the trainer offers instruction. We introduce the *Introspection Approach* (IA), a method by which the learning agent determines when it requires aid from the training agent. We show that guidance received via IA is more informative than random guidance, thus making better use of the training agent.

2 Learning with a trainer

Consider the training agent's role in the learning scenario. The unshaded portions of Figure 1 depict the standard components of reinforcement learning—a *task*, a *learning agent*, and a *critic*—and the interactions between the components. The shaded portion of the figure shows how the *training agent* is incorporated into the learning scenario, observing the task state and providing instruction to the learning agent.

Several researchers have studied systems based on this model. In Lin's work (1992, 1993), the learner receives entire sequences of human problem-solving performance, in which the learner is led from the start state to the goal. The learner then updates its decision policy

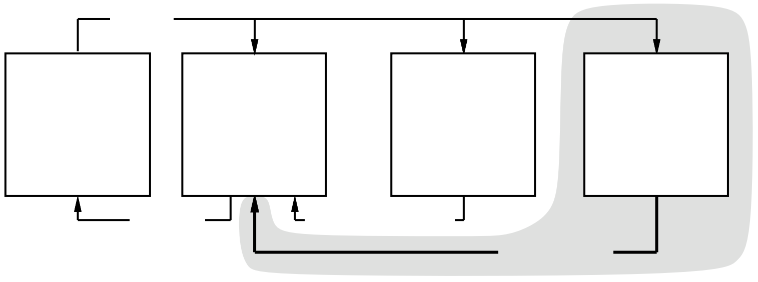

Figure 1. Trainer augmented reinforcement learning

as if it itself had performed the training sequences. In another system (Clouse & Utgoff, 1992), a human training agent interacts with the learner, occasionally providing actions that the learner then performs. Maclin and Shavlik (1994) employ IF-THEN rules, compiling the trainer's instruction directly into the learner's policy via knowledge-based neural network techniques. Similarly, Gordon and Subramanian (1994) rely on IF-THEN rules, which are converted into operational rules that become part of a population that is refined via genetic algorithms. Finally, Clouse (1995) employs an automated trainer that is based on heuristic rules. Although taking different approaches to adding the trainer's knowledge to the system, each of these methods exhibits good performance.

Three of the systems described above allow instruction to be provided at the whim of a human trainer (Clouse & Utgoff, 1992; Lin, 1993; Maclin & Shavlik, 1994). Gordon and Subramanian (1994) provided the trainer's domain information before training begins, and not during the on-line learning. The final system described above (Clouse, 1995) takes a controlled stochastic approach to providing training information, allowing the training agent to offer training direction a fixed percentage of the time. None of these systems directly address the issue of when instruction should be provided, either allowing the trainer to guide the system in an unprincipled manner, or interacting randomly with the learner. In the next section , we consider when the trainer should give instruction.

3 An Introspection Approach

In the approach described here, we use the basic model used by three of the above systems: The automated learner relies on instruction in the form of on-line, trainer-suggested actions for given situations. Upon receipt of an action from the trainer, the learning agent executes the action just as if it had chosen the action with its own policy. Thus, the underlying reinforcement learning algorithm does not need to be modified to handle the trainer's actions: the learner performs the action, changing the task state, and then receives evaluative feedback and updates its decision policy. The crux of the new approach is the mechanism for deciding when the learner should ask the trainer for aid. One of our goals is to maximize the impact of the trainer's instruction, so that the learner develops its decision policy quickly, with very little training. In some sense, we are searching for an ideal training strategy: minimize the trainer's usage while simultaneously minimizing the training time.

When should the trainer give the learner instruction? To answer this question, we rely on our informal perception of when human learners require instruction. Certainly, humans benefit from help when in novel situations. Indeed, humans seek help when they are confused or otherwise unable to decide upon a course of action, even if they have experienced the situation in the past. It is difficult, however, to specify exactly when humans are unsure. Fortunately, this is not the case for an automated learner: One has access to the decision policy and, in particular, the evaluations on which the decision is based. Our Introspection Approach takes into consideration the uncertainty of the learning agent based on the values of the agent's choices as determined by its current decision policy.

To implement IA, one must develop a test that determines whether the learner is unsure of its choices. The test should indicate the need for help in novel situations, when all of the action choices have similar values. One method for doing this is to examine the two extreme values: if they are close to each other, then the intervening values must also be close, which means that the learner has not experienced this state often enough to produce a clear choice. Examining the interval between the extreme values identifies only situations in which the learner is clearly uncertain. Another form of uncertainty may arise when, say, the top two choices have similar values. Our approach will not have the learner ask for help in this situation because it may be the case that the two choices are truly similar.

In the Introspection Approach, if the two extreme values are sufficiently close together, the test succeeds and the learner asks for aid from the trainer. Sufficiency is here determined simply by examining the difference between the minimum and maximum values. The test succeeds when the difference between the extreme values is smaller than a width parameter. This parameter controls how conservative the learner is. With a small width, the learner is infrequently uncertain, but with a large width, the learner asks for aid quite frequently. In the experiments described below, we set this width parameter at different values to analyze different learners.

4 Empirical Study

The experiments described below test the Introspection Approach against an approach in which the learning agent request help randomly (similar to Clouse 1995). In the following section we lay out the details of the experimental study, describing the problem domain, the training agents, and the learning algorithm. Finally, the section ends with a description of the experiments.

4.1 Problem Domain

The problem domain considered is graph-traversal in the form of two-dimensional mazes (see Figure 2), a class of Markovian decision tasks. The objective of the learner is to traverse the maze optimally from the top-left cell to the bottom-right cell. In each state, the learner can choose one of four actions: up, down, left, or right. When performing an action that is blocked by a wall, the agent does not move. Such actions are penalized simply by the virtue of the agent's taking a step and not progressing. The experiments were run with both deterministic problems, in which each action is performed as specified, and with stochastic problems, in which each action has a 25% chance of being changed to an action at a right-angle to itself, half the time resulting in a right turn and the other half, in a left turn. When the agent reaches the goal cell, it receives a 1.0 unit reward, and is placed back at the start cell. In our study, we employed a series of five mazes, Whose sizes ranged from a 10×10 maze with only 42 states, up to a 80×80 maze with 4268 states.

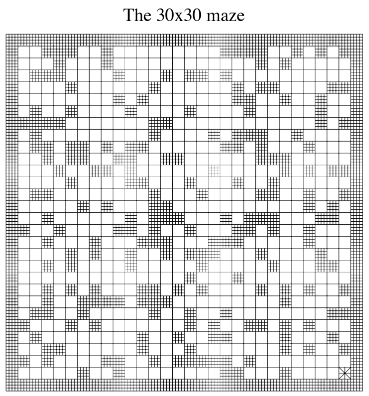

Figure 2. A maze, with 562 states

An advantage of maze problems, which we exploit in the construction of the trainer and in the definition of the stopping criterion, is that the optimal policy can be found. Another advantage is that the maze problems are discrete: we can employ simple tables to store the developing decision policies rather than rely on not well-understood generalization mechanisms. Although the issue of generalization is interesting, we avoid these mechanisms in this study because their use would introduce a confounding factor, clouding the issue of when the trainer should provide actions.

4.2 Trainers

In the experiments, different trainers instruct the learning agents. The trainers are automated and can perform the maze problems at varying levels of proficiency. The baseline trainer performs the task optimally; other trainers are degradations of the optimal. By adjusting the frequency by which the trainer responds with a suboptimal action, we have a wide range of problem-solving expertise.

For any given state, the trainers can provide the learner with an action to perform. The trainer simply examines the current location of the learner in the maze and provides the action that it considers best. In the case of the optimal trainer, this action is optimal. The other trainers may provide an optimal action, but they may also provide an action that is not optimal.

4.3 Learning Algorithm

In all of the experiments, the learner employs the reinforcement learning technique Q-learning (Watkins, 1989) to develop its policy. The Q-functions for each of the four actions are stored in separate tables. The Q-value for a particular state and action, $Q(s, a)$, is just the value in the sth location of the table for action a.

The policy defined by the Q-functions is based on the Q-values and on a random factor to facilitate exploration. A large portion of the time (95%), the learner performs the action whose Q-function has the highest value for the current state. During the other 5% of the time, the learner chooses an action uniformly from among the four.

We modify the Q-learning method only slightly to take into account the trainer's instruction. In addition to using its policy to choose an action, the learning agent can also ask the trainer for instruction. When the trainer receives such a query, it always responds with an action, which the learner then performs.

The learning parameters are: $\alpha = 0.15$ (learning rate), and $\gamma = 0.99$ (discount factor).

4.4 Experiments

The first set of experiments define the control set, where the learning agent requested actions from the trainer a fixed percentage of the time. We varied this rate from 0% up to 100%. In the second set of experiments, the learner decides to ask for instruction based on the IA. We control the amount of trainer-provided actions indirectly by varying the IA width parameter.

In both sets of experiments, we performed individual experiments with trainers that differed in expertise, having them return suboptimal actions from 0% to 50% of the time. We also varied the size of the mazes, from 44 states up to 4268 states.

Each individual experiment consisted of ten runs, each of which began with the entries in the Q-function tables set to zeroes. A run ended when the learner could meet the stopping criterion: while traversing the maze ten times from start to goal, 95% of the learner's actions were optimal. Remember that the learner cannot perform 100% optimal actions because it chooses a random action 5% percent of the time.

Each run consisted of many trials, each of which began with the learner in the start cell of the maze (top-left corner) and ended when the learner reached the goal cell (bottom-right corner). The stopping criterion was tested at the end of each trial. If the criterion was met, the run ended; otherwise, the run continued with the beginning of a new trial.

For each run we recorded two variables: the total number of actions performed by the learning agent throughout the run, and the total number of trainer responses.

5 Results and Discussion

In this section, we present the results of the experiments, and discuss their implications. We only present results for the maze 30×30 maze; The results for the other mazes, both stochastic and deterministic, are similar. First, consider Figure 3, which presents the results for the control experiments, in which the training agent provides actions a fixed percentage of the time. The trainer's instruction rate is presented along the dependent axis, and the average number of actions necessary to achieve the stopping criterion is presented on the dependent axis. To facilitate comparison with a learning agent that does not have access to the trainer (0% interaction rate), the dashed line labelled "Q" was extended out from the axis, with two parallel lines that represent the 95% confidence interval for that value. The other lines on the graph represent the different levels of trainer expertise, ranging from an optimal trainer (at 0% error) to a trainer that provides a suboptimal action half of the time.

As expected, Figure 3 indicates that the trainer's instruction reduces the amount of training necessary to achieve the stopping criterion. For all levels of trainer expertise and response rate, the difference between having a trainer and not having the trainer is significant. Furthermore, the better trainers speed the learner's acquisition of the optimal policy more than the degraded trainers. And, providing more instruction seems, in general, to lead to quicker convergence.

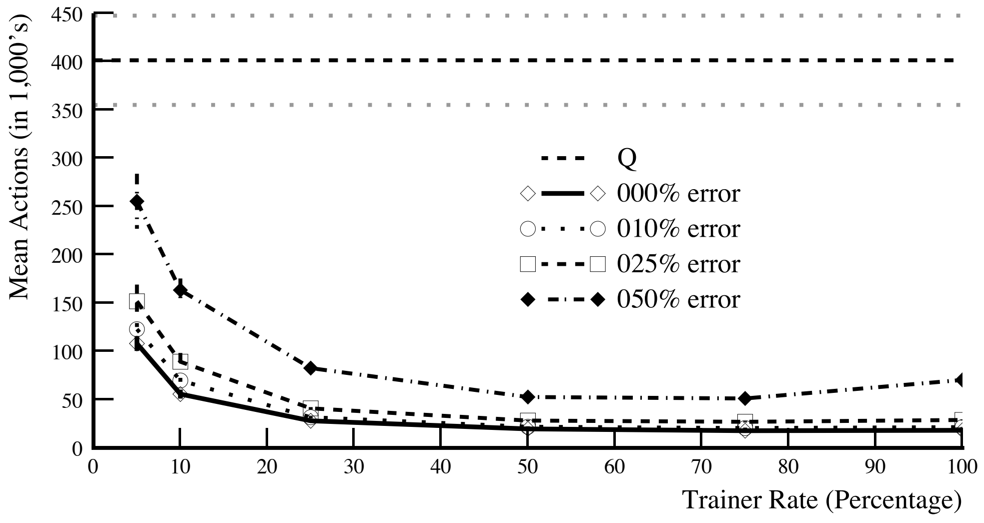

Figure 3. Fixed Schedule results

Now consider Figure 4, which represents the results of the experiments with the Introspection Approach. The horizontal axis in this graph represents the different widths, ranging from 0.0 up to 1.6 (which is $\ln 5.0$), and, as in the previous graph, the vertical axis represents the average number of actions to reach the stopping criterion. At all levels of trainer expertise and interval width, the learning agent performed almost identically (that is, the lines are very close together), and much better than not having instruction from the trainer.

It is difficult to compare the two approaches directly based on the results in Figure 3 and Figure 4 because the dependent axes represent different quantities. To make the comparison between the two approaches easier, examine Figure 5, which plots the number of trainer responses versus the number of actions necessary to achieve the stopping criterion for each of the experiments, given the perfect trainer. The plots look similar for the other 3 trainers.

As depicted in the graph, the same number of trainer responses lead to satisfying the stopping criterion more quickly with the Introspection Approach. For example, the cluster of IA points near 5000 trainer responses lead to the learner's developing an almost optimal

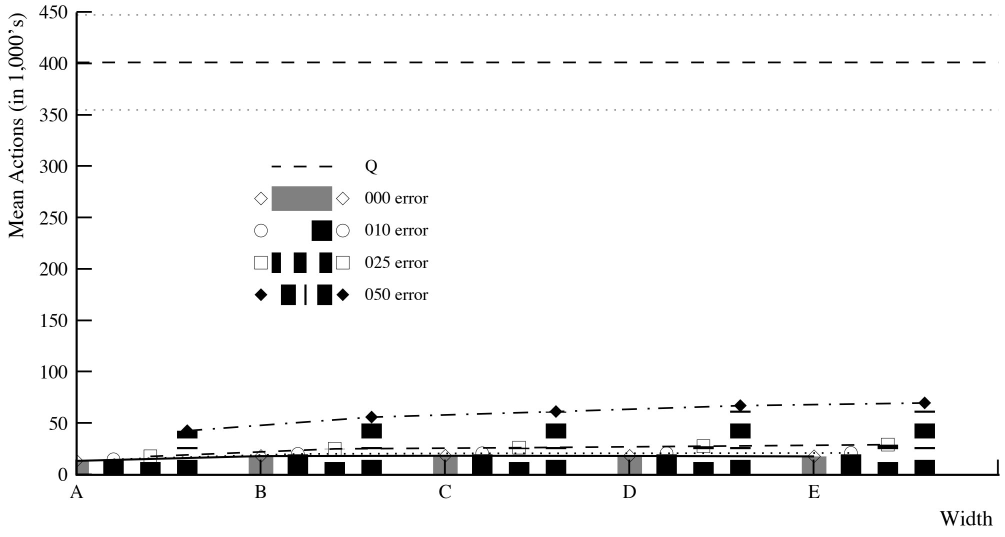

Figure 4. Introspection Approach results

policy in, on average, 13,600 total actions. With the random approach, approximately the same number of trainer responses were used to learn the task in anywhere between 110,000 and 40,000 total actions. Not only do the trainer's actions from IA provide better feedback, allowing the learner to develop its policy more quickly, there is much less variance.

Figure 5 also shows that the performance of the two methods, IA and random, approach each other as the number of training examples increases. One would expect that they both perform similarly when the trainer responds 100% of the time, which is depicted on the graph by the clusters of points on the far right (near 19,000 trainer responses).

Also note in Figure 5 that, with the IA, more trainer actions are not necessarily better. Although it is hard to see in the graph, the curve representing IA *rises* slightly as the number of trainer responses rises. This seems to indicate that the learner can gain a better understanding of its environment by being allowed to explore on its own, as well as by learning from a trainer. Given that the curve is almost flat, indicates that we can minimize the trainer's usage with the IA, learning to perform the task in almost the same amount of time that we would require if we relied solely on the trainer.

6 Summary

This paper introduces the Introspection Approach, a method by which a learning agent employing reinforcement learning can decide when to ask a training agent for instruction.

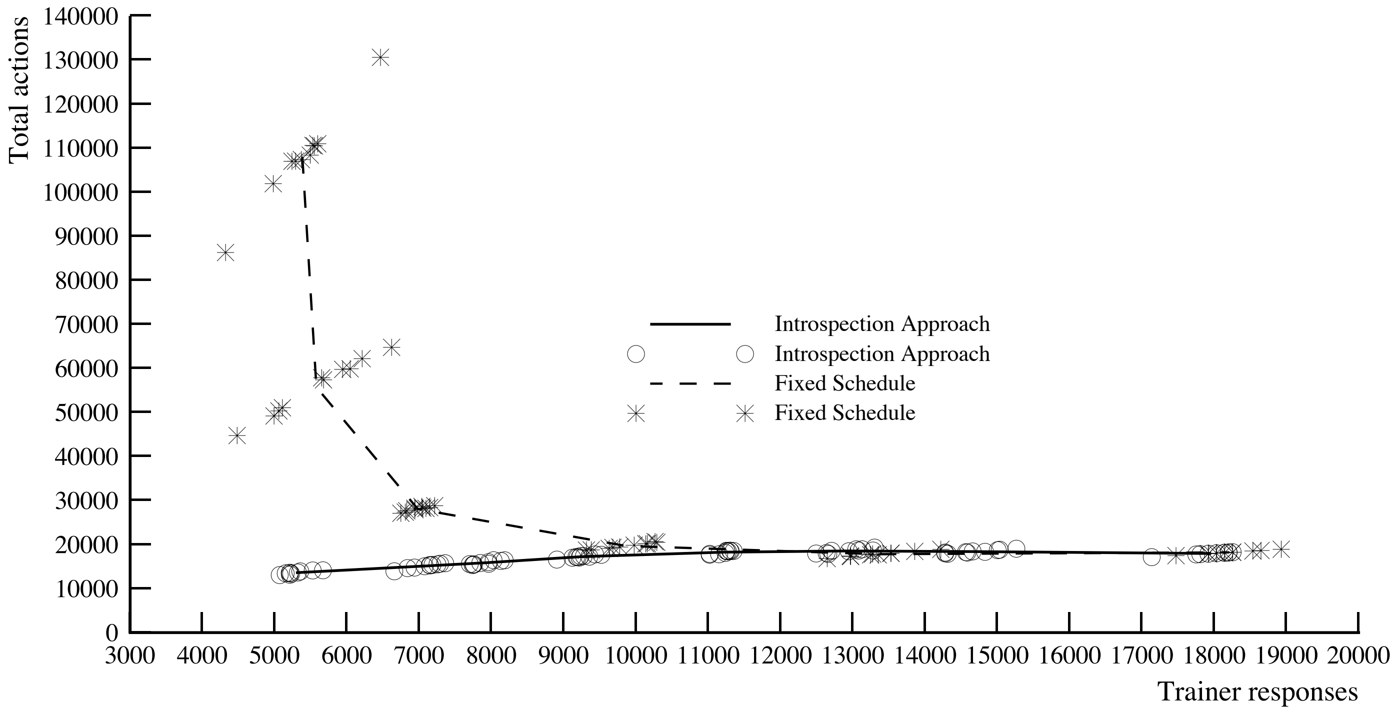

Figure 5. Trainer's actions versus Total actions

We compared the IA to a random approach of asking and found that the same number of trainer's responses produced significantly faster learners using the IA. Thus, guidance received via IA is more informative than random guidance, because the same level of trainer instruction leads to satisfying the stopping criterion more quickly. Thus, we can reduce the interaction that the training agent has with the learning agent without reducing the speed with which the learner develops its policy. In fact, by being intelligent about when the learner asks for help, we can even increase the learning speed for the same level of trainer interaction.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. IRI-9222766. We thank Paul Utgoff and Gunnar Blix for many helpful comments.

References

Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence, 72*, 81-138.

Clouse, J. A., & Utgoff, P. E. (1992). A teaching method for reinforcement learning. *Machine Learning: Proceedings of the Ninth International Conference* (pp. 92-101). San Mateo, CA: Morgan Kaufmann.

Clouse, J. A. (1995). Learning from an automated training agent. *Proceedings: ML95 Workshop on 'Agents that Learn from Other Agents'.*

Gordon, D., & Subramanian, D. (1994). A multistrategy learning scheme for agent knowledge acquisition. *Informatica, 17*, 331-346.

Lin, Long-Ji (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning, 8*, 293-321.

Lin, Long-Ji (1993). Scaling up reinforcement learning for robot control. *Machine Learning: Proceedings of the Tenth International Conference* (pp. 182-189). Amherst, MA: Morgan Kaufmann.

Maclin, R., & Shavlik, J. W. (1994). Incorporating advice into agents that learn from reinforcements. *Proceedings of the Twelfth National Conference on Artificial Intelligence* (pp. 694-699). Seattle, WA: MIT Press.

Sutton, R. S. (1984). *Temporal credit assignment in reinforcement learning.* Doctoral dissertation, Department of Computer and Information Science, University of Massachusetts, Amherst, MA.

Watkins, C.J.C.H. (1989). *Learning with delayed rewards.* Doctoral dissertation, Psychology Department, Cambridge University.