Implicit Imitation in Multiagent Reinforcement Learning

Bob Price and Craig Boutilier
Department of Computer Science
University of British Columbia
Vancouver, B.C., Canada V6T 1Z4
{price,cebly}@cs.ubc.ca

Abstract

Imitation is actively being studied as an effective means of learning in multi-agent environments. It allows an agent to learn how to act well (perhaps optimally) by passively observing the actions of cooperative teachers or other more experienced agents its environment. We propose a straightforward imitation mechanism called *model extraction* that can be integrated easily into standard model-based reinforcement learning algorithms. Roughly, by observing a mentor with similar capabilities, an agent can extract information about its own capabilities in unvisited parts of state space. The extracted information can accelerate learning dramatically. We illustrate the benefits of model extraction by integrating it with prioritized sweeping, and demonstrating improved performance and convergence through observation of single and multiple mentors. Though we make some stringent assumptions regarding observability, possible interactions and common abilities, we briefly comment on extensions of the model that relax these.

1 Introduction

The application of reinforcement learning (RL) to multiagent systems offers unique opportunities and challenges. When agents are viewed as independently trying to achieve their own ends, interesting issues in the interaction of agent policies [12] must be resolved (e.g., by appeal to equilibrium concepts). However, the possibility that agents may coordinate their policies for mutual gain [22] or distribute their search for optimal policies (and communicate partial results to one another) [13] offers intriguing possibilities for accelerating RL and enhancing agent performance.

Another way in which individual agent performance can be improved is by having a novice agent learn reasonable behavior from an expert *mentor*. This type of learning can be brought about through explicit *teaching* or demonstration [1, 11, 25], by sharing of privileged information [13] or through a more elaborate psychological theory of *imitation* [2]. In imitation, the agent's own exploration is used to ground its observations of other agent's behaviors in its own capabilities and resolve any ambiguities in observations arising from partial observability and noise. A common thread in all of this work is the use of a mentor to guide the *exploration* of the observer. Guidance is typically achieved through some form of explicit communication between mentor and observer. A less direct form of teaching involves an observer extracting information from a mentor without the mentor making an explicit attempt to demonstrate a specific behavior of interest [15].

In this paper we develop a model of *implicit imitation* through observation of a mentor. Roughly, an agent observes the state transitions induced by the mentor's actions. Assuming that it has actions similar to the (unknown) action taken by the mentor, the agent can use this information to update its value function. In addition, a mentor can provide hints to the observer about the parts of the state space on which it may be worth focusing attention. The observer's attention to an area might take the form of additional exploration of the area or additional computation brought to bear on the agent's prior beliefs about the area. We derive a new technique, *model extraction*, that is independent of any specific RL algorithm, though it is best suited for use with model-based methods. We illustrate its effectiveness empirically by incorporating it into prioritized sweeping.

Our model has several advantages over more direct forms of imitation and teaching. It does not require any agent to explicitly play the role of mentor or teacher. Observers learn simply by watching the behavior of other agents (mentors); if a mentor shares certain subtasks with the observer, the observed behavior can be incorporated (indirectly) by the observer. This is important because there will be many situations in which an observer can learn from a mentor that is unwilling or unable to alter its behavior to teach the observer, or even communicate information to it. For example, common communication protocols may be unavailable to agents designed by different developers (e.g., Internet agents); agents may be competing; or there may simply be no incentive for one agent to provide information to another.[1] Because an agent learns by observation, it can exploit the existence of multiple mentors, essentially distributing its search. The observer is also not constrained to directly imitate the mentor and can decide whether such imitation is worthwhile. Finally, we do not assume that the observer knows the actual actions taken by the mentor, or that the mentor shares a reward function (or goals) with the mentor. While we make some strict assumptions in this paper, the model can be generalized in interesting ways, as we describe below. Many of these generalizations are the subject of ongoing research.

We present the basic model and assumptions underlying model extraction in Section 2. We develop the model extraction algorithm and demonstrate a simple mechanism to focus the agent's updating mechanism on promising areas of the state space in Section 3, and illustrate the performance of the algorithm empirically in Section 4. We conclude in Section 5 with remarks on related work and a discussion of current and future research, especially with respect to the relaxation of certain modeling assumptions.

2 Model and Assumptions

Our model assumes that we have some number of agents acting to achieve their own objectives in a common environment. To keep the notation simple, we present the model using two agents only: a *mentor* m and an *observer* o. The observer is a reinforcement learner that can observe aspects of the mentor's behavior. The extension to multiple agents should be viewed as adding further (potential) mentors to the system, since we consider the RL problem facing a single observer.[2]

The agents act to control a *multiagent Markov decision process (MMDP)*, in which the agents select actions independently, and their *joint actions* determine stochastic state transitions.[3] In order to focus on imitation without getting into issues of strategic reasoning (e.g., game theoretic concepts like equilibria), we assume that the agents' actions are noninteracting; that is, each agent can ignore the behavior of the other in predicting the effect of its own actions on the system and the reward it receives. To do this, we factor the MMDP into two standard (single-agent) MDPs $M_o = \langle S_o, A_o, \Pr_o, R_o \rangle$ and $M_m = \langle S_m, A_m, \Pr_m, R_m \rangle$ pertaining to the observer and mentor, respectively. Here, for $i \in \{o, m\}$, S_i is the state space for agent i, A_i its set of actions, $\Pr_i(t|s,a)$ the stochastic state transition function and $R_i(s)$ its reward function (defined over S_i). For instance, in a grid world, we might assume that the mentor and observer can each move without affecting the other's ability to move or impact the reward it receives by occupying certain grid cells.

If the observer is to learn by observing the mentor, there must be some relationship between the space of potential behaviors they can implement. We make two especially strong assumptions in this regard. First, the agents have identical state spaces; i.e., $S = S_m = S_o$. Second, the observer has (at least) the same capabilities as the mentor; i.e., $A_m \subseteq A_o$ and $\Pr_m(t|s,a) = \Pr_o(t|s,a)$ for any a they have in common. These assumptions make our description of the imitation task somewhat simpler, but can be relaxed. The essential feature is that there be some analogical mapping between state transitions so that the observations of the mentor can be recast in terms the observer understands [19] (see Section 5). Our formulation makes this mapping trivial. However, we do not assume that the reward functions R_o and R_m are related in any specific way.

Initially, the observer will not know the transition model $\Pr_o$; but we assume it knows its reward function R_o. While in typical RL models, the learner does not know R_o, prior knowledge of R_o is consistent with the interpretation of RL as automatic programming: the agent designer can often provide predicates to evaluate the quality of a situation in advance, but cannot provide a model of the state dynamics. To learn from the mentor, the observer must be able to observe certain aspects of its behavior. Unless it communicates its policy or selected actions to the observer, the actual actions taken by the mentor will be unknown to the observer. All it can see is the effects of an action. For instance, in a noisy grid world, the mentor may attempt to move north but actually move east: the observer sees only the actual state transition, not the intended transition or actual action attempted. Therefore, we assume the observer can see the specific state transitions $s \to t$ taken by the mentor. The mentor's state is thus fully observable to the observer. This, too, is a strong assumption—we indicate ways in which it can be relaxed in Section 5.

3 Model Extraction and Implicit Imitation

The observer's problem is to determine an optimal course of action that maximizes expected value, where its value function is given by the Bellman equation:

$$V(s) = R_o(s) + \gamma \max_{a \in A_o} \left\{ \sum_{t \in S} \Pr_o(t|s,a) V(t) \right\} \quad (1)$$

We assume here a discounted infinite horizon context with discount factor γ. For the present, we also assume that the agent's actions do not influence the agent's reward structure, but this assumption will later be relaxed.

[1] For reasons of consistency, we use the term "mentor" to describe any agent from which an observer can learn, even if the mentor is an unwilling participant.

[2] This is not to say that the other "mentors" cannot be learners or even imitators themselves.

[3] MMDPs could be viewed as a form of *stochastic game* [20]; since Shapley's original formulation involved the zero-zum assumption, we use different terminology to avoid confusion.

In an RL context, our observer has access to transition samples of the form $\langle s, a, t\rangle$ (recall that it knows $R_o(s)$). An optimal policy can be determined by directly learning a Q-function $Q(s, a)$ [24], or using model-based methods that estimate the model $\Pr_o$ and solve (generally, asynchronously and incrementally) Equation 1 (e.g., prioritized sweeping [17]). There is a limited scope in which observations of the mentor's state trajectories can influence this computation. Since the observer already knows its reward function, its observations of the mentor cannot influence its reward model. Observations could be used to update its estimate of its own transition model $\Pr_o$, or the order in which to apply Bellman backups to its own estimated value function. We consider both forms of influence in this section. Two other possible (direct) influences are not explored here: the observer could attempt to infer the best action to perform in state s based on the mentor's trajectory; or the observer could use the observations to directly compute the value function, or constraints on the value function for states it has not visited [23].

In Section 3.1, we describe the model extraction algorithm in the context of model-based RL. However, in Section 3.2, we indicate how the same ideas can be incorporated into model-free algorithms such as Q-learning.

3.1 Model Extraction in General Terms

If a mentor is "expert" at achieving its objectives, we can assume it is executing a *stationary*, *deterministic* policy π_m, with $\pi_m(s)$ denoting its action choice at state s. Behavior π induces a Markov chain $\Pr_m(t|s) = \Pr_m(t|s, \pi_m(s))$ over S. By observing mentor transitions, the observer can construct an estimate $\widehat{\Pr}_m$ of this chain: $\widehat{\Pr}_m(t|s)$ can simply be estimated by the relative observed frequency of mentor transitions $s \rightarrow t$ (w.r.t. all transitions taken from s).

Since $A_m \subseteq A_o$, the observer knows that there is some action $a \in A_o$ that is the same as $\pi_m(s)$; that is, there exists an action a such that $\Pr_o(t|s, a) \approx \widehat{\Pr}_m(t|s)$. Unfortunately, since the observer does not know its own transition model—it has only estimates—and cannot observe the mentor's actions directly, it does not know the identity of this action $\pi_m(s)$. However, the mere fact that there *exists some action* with estimated transition probabilities $\Pr_m(\cdot|s)$ can be used to good effect by the observer in value function estimation. Since $A_m \subseteq A_o$, $V(s)$ can be given by the following *augmented Bellman equation*:

$$V(s) = R_o(s) + \gamma \max\left\{\max_{a\in A_o}\left\{\sum_{t\in S}\Pr_o(t|s,a)V(t)\right\},\ \sum_{t\in S}\Pr_m(t|s)V(t)\right\} \quad (2)$$

This is the usual Bellman equation (1) with an extra term added, the second summation, denoting the expected value of duplicating the mentor's action $\pi_m(s)$. Since this (unknown) action is identical to one of the observer's actions, the term is redundant and the augmented value equation is valid. Of course, the observer using the augmented backup operation must rely on estimates of these quantities. If the observer's exploration policy ensures that each state is visited infinitely often, the estimates of the Pr_o terms will converge to their true values. The mentor's policy is not under the observer's control. However, if the mentor's policy is ergodic over state space S, then these terms too will converge to their true values. If the mentor's policy restricts it to a subset of states $S' \subseteq S$ (those forming the basis of its Markov chain), these estimates will converge correctly with respect to S' if the chain is ergodic, and states in $S - S'$ will remain unvisited. An observer can apply the augmented equation only for those states visited by the mentor. In either of these cases, the use Equation (2) does not impact the usual convergence results for RL algorithms.

Our primary interest, however, is the behavior of the system during the initial stages of learning when the advantage of a knowledgeable mentor can make the most difference to an observer. Assuming that the mentor is pursuing a greedy policy, there will be many states for which the observer has much more accurate estimates of $\Pr_m(t|s)$ than it does for $\Pr_o(t|s, a)$ for any specific a. Since the observer is learning, it must explore both state space (causing less frequent visits to s) and action space (thus spreading its experience at s over all actions a), generally ensuring that the sample size upon which $\Pr_m$ is based is greater than that for $\Pr_o$ for any specific action. Apart from being more accurate, the use of $\Pr_m(t|s)$ can often give more informed value estimates at state s, since prior action models are generally "flat" or uniform, and only become distinguishable at a given state when the observer has sufficient experience at state s.

When the mentor's Markov chain is not ergodic, or even if the mixing rate is sufficiently low, the mentor may visit a certain state s relatively infrequently. A state that is rarely (or never) visited by the mentor may provide a very misleading estimate—based on the small sample or the prior for the mentor's chain—of the value of the mentor's (unknown) action at s; and since the mentor's policy is not under the control of the observer, this misleading value may persist for an extended period. This stems in part from the fact that we use maximization to combine values based on the estimates of the mentor's chain and the observer's action models, and in part because we use mean values (estimated probabilities) based on observed samples. The augmented Bellman equation does not consider the reliability of the information sources.

To overcome this, we have incorporated an estimate of model confidence into our augmented backups. For the mentor's Markov chain and the observer's action transitions, we assume a Dirichlet prior over the parameters of each of these multinomial distributions [7]. These reflect the observer's initial uncertainty about the possible transition probabilities. From sample counts of mentor and observer transitions, we update these distributions. With this information, we could attempt to perform an optimal Bayesian estimation of the value function; but when the sample counts are small (and normal approximations are not appropriate), there is no simple, closed form expression for the resultant distributions over values. We therefore employ an approximate method for combining information sources inspired by Kaelbling's interval estimation method [9].

We first compute the observer's optimal action a_o^* using a simple max over the mean value of the state given each possible action (The usual method). With the benefit of the Dirichlet distributions, we then construct a lower bound v_o^- on the value of the state to the observer using the model derived from its own behavior—this is the lower bound of a suitable confidence interval over the expected value of that state. A second lower bound v_m^- is constructed using the model based on observations of the mentor. If $v_m^- < v_o^-$, then either the mentor-inspired model has, in fact, a lower expected value (within a specified degree of confidence) and uses a nonoptimal action (from the observer's perspective), or the mentor-inspired model has lower confidence. In either case, we reject the information provided by the mentor and use the model derived from the observer's observations of itself. Using Tchebycheff's inequality, we can choose a confidence level even though the Dirichlet distributions for small sample counts are highly non-normal.

We note that the reasoning above holds even if the mentor is implementing a (stationary) stochastic policy (since the expected value of stochastic policy for a fully-observable MDP cannot be greater than that of an optimal deterministic policy). While the "direction" offered by a mentor implementing a deterministic policy tends to be more *focussed*, empirically we have found that mentors offer *broader* guidance in moderately stochastic environments or when they implement stochastic policies, since they tend to visit more of the state space. We note that the extension to multiple mentors is straightforward—each mentor model can be incorporated into the augmented Bellman equation without difficulty.

Though the model has assumed no action costs (i.e., rewards that depend only on the state), we can use more general reward functions (e.g., where reward has the form $R(s,a)$). The difficulty lies in backing up action costs when the mentor's chosen action is unknown. We use a simple heuristic method to find the action of the observer that is "closest" to that of the mentor (given current transition probability estimates) and then construct an estimate of the value of the state using the information derived from mentor transitions based on this action. Let $\kappa(s)$ denote that observer's action at state s that is "closest" to the action executed by the mentor at state s. This is defined as the action whose observed transition distribution has minimum Kullback-Leibler distance from the mentor's observed transition distribution:

$$\kappa(s) = \mathrm{argmin_a} \left\{ -\sum_{\mathrm{t}} \mathrm{Pr_o(t|s,a)} \log \mathrm{Pr_m(t|s)} \right\} \quad (3)$$

The augmented Bellman equation 2 can be rewritten using the κ function as follows:

$$\begin{aligned} V(s) \quad = \quad & \max \left\{ \max_{a \in A_o} \left\{ R_o(s,a) + \gamma \sum_{t \in S} \Pr_o(t|s,a) V(t) \right\}, \right. \\ & \left. R_o(s,\kappa(s)) + \gamma \sum_{t \in S} \Pr_m(t|s) V(t) \right\} \end{aligned}$$

We note that Bayesian methods could be used could be used to estimate action costs in the mentor's chain as well.

Our *model extraction algorithm* requires that: (a) the observer maintain an estimate $\widehat{\Pr}_m(t|s)$ of the Markov chain induced by the mentor's policy—this estimate is updated with every observed transition; and (b) that all backups performed to estimate its value function use the augmented backup Equation 2 with confidence testing.[4]

A second way in which an observer can exploit its observations of the mentor is to focus attention on the states visited by the mentor. In a model-based approach, the specific *focusing mechanism* we adopt requires the observer to perform a (possibly augmented) Bellman backup at state s whenever the mentor transitions out of s. This has three effects. If the mentor tends to visit interesting regions of space (e.g., if it shares a certain positive or negative reward structure with the observer in state s), the observer's attention is drawn there. More importantly, this focus of attention directs computational effort toward parts of state space where the estimated model $\widehat{\Pr}_m(t|s)$ changes, hence where the estimated value of one of the observer's actions may change. In addition, computation is focused where the model is likely more accurate (as discussed above).

3.2 Model Extraction in Specific RL Algorithms

Model extraction and the focusing mechanism described above can be integrated into model-based RL algorithms with relative ease. In the experiments described in the next section, *prioritized sweeping* is used [17]. Roughly, prioritized sweeping works by maintaining an estimated transition model $\widehat{Pr}$ and reward model $\widehat{R}$. Whenever an experience tuple $\langle s,a,r,t \rangle$ is sampled, the estimated model at state s can change; a Bellman backup is done at s to incorporate the revised model and some (usually fixed) number of additional backups are performed at selected states.

[4] Of course, these backups are implemented using estimated models $\widehat{\Pr}_o(t|s,a)$ and $\widehat{\Pr}_m(t|s)$.

States are selected using a *priority* that estimates the potential change in their values based on the changes precipitated by earlier backups (see [17] for details). Essentially, computational resources (backups) are focused on those states that can most "benefit" from those backups.

Incorporating our ideas into prioritized sweeping simply requires the following changes:

(a) With each transition $\langle s, a, t\rangle$ the observer takes, the estimated model $\widehat{\Pr}_o(t|s,a)$ is updated and an *augmented backup* (Equation 2) is performed at state s. Augmented backups are then performed at a fixed number of states using the usual priority queue implementation.

(b) With each observed mentor transition $\langle s, t\rangle$, the estimated model $\widehat{\Pr}_m(t|s)$ is updated and an augmented backup is performed at s. Augmented backups are then performed at a fixed number of states using the usual priority queue implementation.

The fact that augmented backups are used instead of standard Bellman backups implements model extraction, while the requirement that backups be performed at observed transitions (in addition to experienced transitions) incorporates our focusing mechanism. We can view this RL algorithm as embodying a form of *implicit imitation*. The observer is not forced to "follow" or otherwise mimic the actions of the mentor directly. But it does back up value information along the mentor's trajectory as if it had. Ultimately, the observer must move to those states to discover which actions are to be used; but in the meantime, important value information is being propagated that can guide its exploration (we discuss exploration in more detail below).

We note that the usual convergence results hold for prioritized sweeping (and other algorithms such as certainty equivalence) when model extraction is used. As the models converge, the augmentation of the Bellman equation becomes irrelevant. During initial phases of training, however, it can induce different exploratory and computational behavior in the observer, and in many instances faster convergence. In particular, as we will see below, this implicit imitation often causes the learner to obtain higher accumulated reward during the initial stages of learning.

While model extraction fits naturally with model-based RL methods, some of these ideas can be readily incorporated into model-free algorithms like Q-learning. For example, the observer could augment its Q-function with a "fictitious" action a_m—the action used by the mentor—and update the Q-values of a_m using mentor transitions (imposing its own R). When updating the Q-values of its own actions, the value of any state s would be estimated using the maximum Q-values of all actions, including a_m.

3.3 Action Selection

Exploration is a key part of any RL algorithm, and here we adopt ε-greedy action selection, using a decaying exploration rate ε over time. The following remarks apply equally well to other pseudo-greedy methods such as Boltzmann exploration.

When selecting an action at state s, greedy action selection requires that we choose the action with highest expected value (i.e., the a that maximizes $\sum_t \widehat{Pr}_o(t|s,a)V(t)$). When the mentor's policy dictates an especially appealing action at state s, it may well be that $\sum_t \widehat{Pr}_m(t|s)V(t)$ is greater than the value of any known action, in which case the observer is better off trying to duplicate the action of the mentor. Unfortunately, this action is unknown. Again, we define the "greedy action" for the observer to be that action a whose estimated distribution $\widehat{Pr}_o(t|s,a)$ has minimum Kullback-Leibler distance from $\widehat{Pr}_m(t|s)$. This proves especially effective early in training when value estimates based on actions with small samples vary widely, while confidence in the mentor's model is much higher.[5]

4 Empirical Validation

The following empirical tests report on the incorporation of model extraction and our focusing mechanism with prioritized sweeping. These results illustrate the types of problems and scenarios in which implicit imitation of this form can provide advantages to an RL agent. In each of the experiments, an expert "mentor" is introduced into the experiment to serve as a model for an "observer" agent. In each case, the mentor is following an ϵ-greedy policy with a very small ϵ (on the order of 0.01). This tends to cause the mentor's trajectories to lie within a "cluster" surrounding optimal trajectories (and reflect good if not optimal policies). Even with this (and environment stochasticity), mentors generally do not "cover" the entire state space so confidence testing is important.

In all of these experiments, prioritized sweeping is used with a fixed number of backups per observed or experienced sample.[6] ε-greedy exploration is used with decaying ε. Observer agents are given uniform Dirichlet priors. Observer agents are compared to control agents that do not benefit from a mentor's experience, but are otherwise identical (implementing prioritized sweeping with similar parameters and exploration policies). The tests are all performed on stochastic grid-world domains, since these make it clear

[5] It is easy to construct cases where the action a closest to the mentor's a_m has lower expected value than action b, while the expected value of a_m is greater than that of b. In such a case, it often makes sense to view a as a better (greedy) action than b. However, we recognize that, ultimately, confidence factors should be incorporated into such judgments as well, so that mentor models with relatively small sample sizes are not given such priority.

[6] Generally, the number of backups was set to be roughly equal to the length of the optimal "noise-free" path.

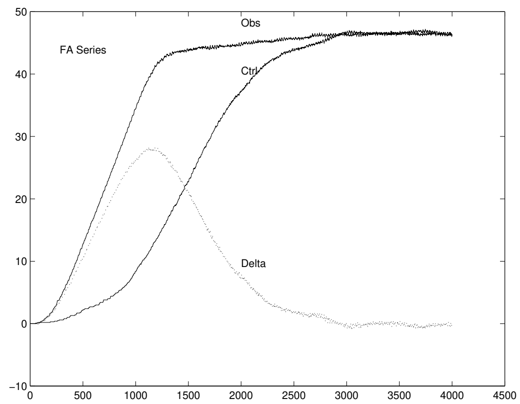

Figure 1: Basic Observer and Control Agent Comparisons

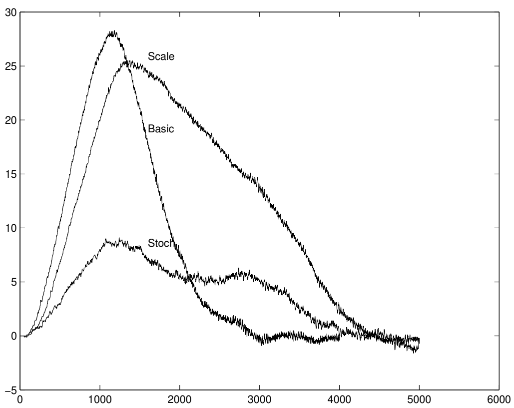

Figure 2: Influence of Domain Size and Noise

to what extent the observer's and mentor's optimal policies overlap (or fail to). We assume an eight-connectivity between cells so that any state in the grid has 9 neighbors including itself, but agent's use only the North, South, East and West actions. We focus primarily on whether imitation improves performance during learning, since the learner will converge to an optimal policy whether it uses imitation or not.

First we show a typical profile of an observer using model extraction (with an expert mentor) vs. a control agent which does not. Both agents attempt to learn a policy that maximizes discounted return on a 10x10 grid world. They start in the upper-left corner and seek a goal with value 1.0 in the lower corner. Upon reaching the goal, the agents are restarted at the top. Actions are noisy, resulting in an unintended transition 10% of the time. The discount factor is 0.9. In Figure 1 we plot the cumulative number of goals obtained over the previous 1000 time steps for the observer "Obs" and control "Ctrl" (results are averaged over 10 runs). The observer is able to quickly incorporate a policy learned from the mentor into its value estimates. This results in a steeper learning curve. In contrast, the control agent slowly explores the space to build a model first. The "Delta" curve shows the difference in performance between the agents. Both agents converge to the same optimal value function.

The next experiment illustrates the sensitivity of imitation to the size of the state space and action noise level. In Figure 2 we plot the delta curves for the "Basic" scenario just described, the "Scale" scenario in which the state space size is increased 69% 13x13, and the "Stoch" scenario in which the noise level is increased to 40% (results are averaged over 10 runs). The difference between the observer and the non-imitating prioritized sweeping agent increases with the state space size. This reflects Whitehead's observation for grid worlds that exploration needs can increase quickly with state space size, but that the optimal path length increases only linearly [25]. Here we see that the guidance of the mentor can help more in larger state spaces.

Increasing the noise reduces the observer's ability to act upon the information received from the mentor and therefore erodes its advantage over the control agent. We note, however, that the benefit of imitation degrades gracefully with increased noise and is present even at this relatively extreme noise level.

Sometimes the observer's prior beliefs about the transition probabilities of the mentor can mislead the observer and cause it to generate inappropriate values. The confidence mechanism proposed in the previous section can prevent the observer from being fooled by misleading priors on the mentor's transition probabilities. This experiment is based on the scenario illustrated in Figure 3. Again the agent's task is to navigate from the top-left corner to the bottom-right corner of a 10x10 grid in order to attain a reward of +1. We have created a pathological scenario in which islands of high value (+5) are enclosed by obstacles. Since the observer's priors reflect eight-connectivity and are uniform, the high-valued cell in the middle of each island are believed to be reachable from the states diagonally adjacent with some small prior probability. In reality, however, the agent's action set precludes this and the agent will therefore never be able to realize this value. The four islands in this scenario thus create a fairly large region in the center of the space with a high value which could potentially trap an observer.

To test, the confidence mechanism, the mentor follows a path around the outside of the obstacles so that its path cannot lead the observer out of the trap. The combination of a high initial exploration rate and the ability of prioritized sweeping to spread value across large distances then virtually guarantees that the observer will be "lead" to the trap. Given this scenario, we ran two observer agents and a control. The first observer used a confidence interval with width given by 5σ which according to the Tchebycheff rule should cover approximately 96 percent of an arbitrary distribution. The second observer was given a 0σ

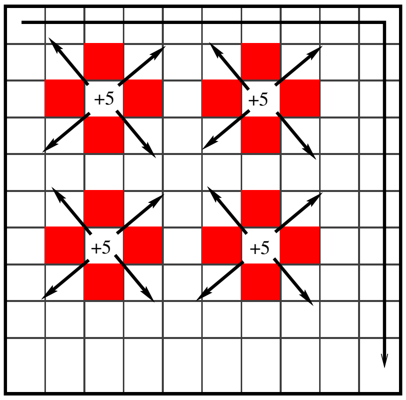

Figure 3: An Environment with Misleading Priors

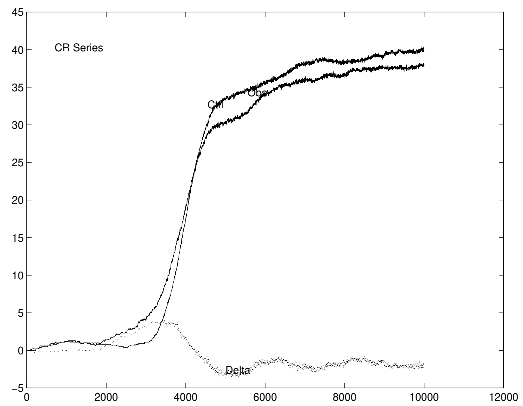

Figure 4: Misleading priors may degrade performance

interval which effectively disables confidence testing. The observer with no confidence testing consistently became stuck. Examination of the value function revealed consistent peaks within the trap region and observation of the agent showed that it was stuck in the trap. The observer with confidence testing consistently escapes the trap. Observation of its value function over time shows that the trap forms, but fades away as the observer gains enough experience to throw out erroneous priors. In Figure 4, the performance of the observer with confidence testing is shown with the performance of the control agent (results are averaged over 10 runs). We see that the observer's performance is only slightly degraded from that of the unaugmented control agent even in this pathological case.

The next experiment demonstrates how the potential gains of imitation can increase with the (qualitative) difficulty of the problem. In the "maze" scenario we introduce obstacles in order to increase the difficulty of the learning problem. The maze is set on a 25x25 grid (Figure 5) with 286 obstacles complicating the agent's journey from the top-left to the bottom-right corner. The only solution takes the form of a snaking 133-step path, with distracting paths (up to length

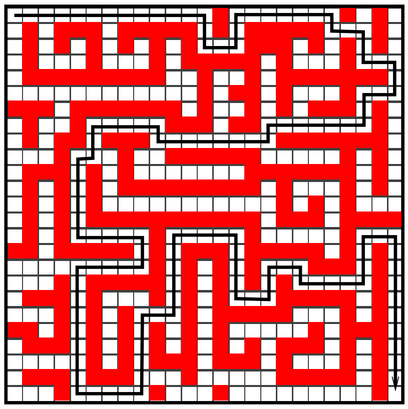

Figure 5: A complex maze

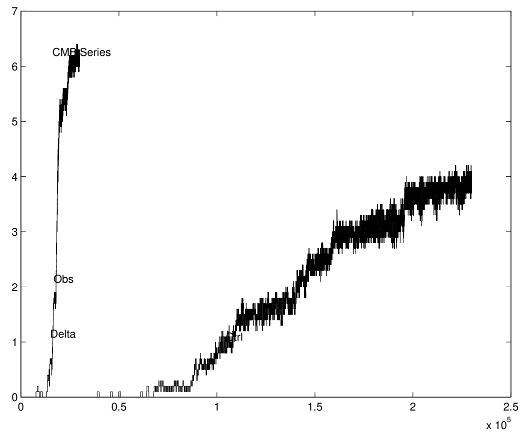

Figure 6: Imitation in a complex space

22) branching off from the solution path necessitating frequent backtracking. The discount factor is 0.98. With 10% noise, the optimal goal-attainment rate is about 6 goals per 1000 steps.

From the graph in Figure 6 (with results averaged over 10 runs), we see that the control agent takes on the order of 150,000 steps to build a decent value function that reliably leads to the goal. At this point, it is only achieving 4 goals per 1000 steps on average as its exploration rate is still reasonably high (unfortunately, decreasing exploration more quickly does not lead to faster value function formation). The imitation agent is able to take advantage of the mentor's expertise to build a reliable value function in about 20,000 steps. Since the control agent has been unable to reach the goal at all in the first 20,000 steps, the delta between the control and the imitator is simply equal to the imitator's performance. The imitator can quickly achieve the optimal 6 goals per 1000 steps as its exploration rate decays much more quickly.

The augmented backup rule does not require that the reward

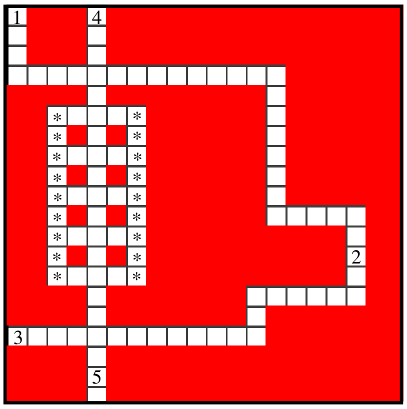

Figure 7: A maze with a perilous shortcut

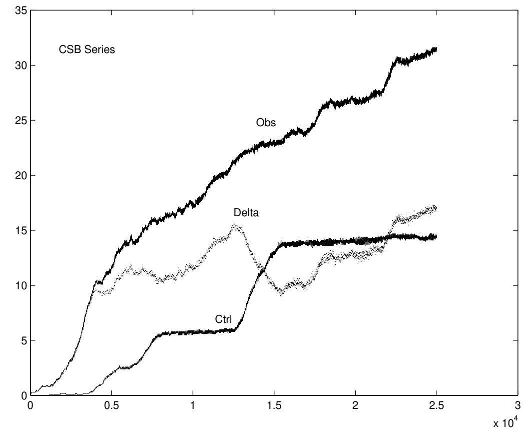

Figure 8: Transfer with non-identical rewards

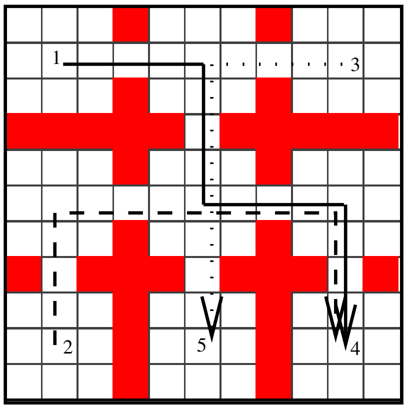

Figure 9: Multiple mentors scenario

structure of the mentor and observer be identical. There are many useful scenarios, where rewards are dissimilar but induce value functions and policies that share some structure. In this experiment, we demonstrate one interesting scenario in which it is comparatively easy to find a suboptimal solution, but difficult to find the optimal solution. Once the observer finds this suboptimal path, however, the observer agent is able to exploit its observations of the mentor to see that there is a shortcut that significantly shortens the path to the goal. The structure of the scenario can be seen in Figure 7. The suboptimal solution lies on the path from 1 around the scenic route at 2 to the goal at 3. The mentor takes the vertical path from 4 to 5 through the shortcut. To discourage the use of the shortcut by novice agents, it is lined with cells (marked "*") that jump back to the start state. It is therefore difficult for a novice agent executing many random exploratory moves to make it all the way to the end of the shortcut and obtain the value which would reinforce its future use. Both the observer and control therefore generally find the scenic route first.

In Figure 8, the performance (goals per 1000 steps) of the control and observer are compared (averaged over 10 runs), indicating the value of these observations. Observations show that the observer and control agent take about the same time on average to find the long scenic route to the goal, however, the observer instantly recognizes the shortcut and jumps to almost double the goal rate. This experiment shows that mentors can improve observer policies even when the observer's goals are not on the mentor's path.

The final experiment illustrates how model extraction can be readily extended so that the observer can extract models from multiple mentors and exploit the most valuable parts of each. In Figure 9, the learner must move from start location 1 to goal location 4. Two expert agents with different start and goal states serve as potential mentors. One mentor repeatedly moves from location 3 to location 5 along the dotted line, while a second mentor departs from location 2 and ends at location 4 along the dashed line. In this experiment, the observer must combine the information from the examples provided by the two mentors with independent exploration of its own in order to solve the problem.

In Figure 10, we see that the observer successfully pulls together these information sources in order to learn much more quickly than the control agent (results are averaged over 10 runs).

Though we have not explored this empirically, "negative" transfer can also be accommodated in our framework. Naturally, one might want an observer to learn about "negative" rewards. A mentor which is unsuccessful and ends up in a particularly bad state (from the observer's point of view) may nonetheless be profitable to study.

5 Concluding Remarks

5.1 Related Work

Imitation is a common strategy for information transfer in natural agents, from the octopus [8] to the primate, with many forms of imitation occurring in (and across) other

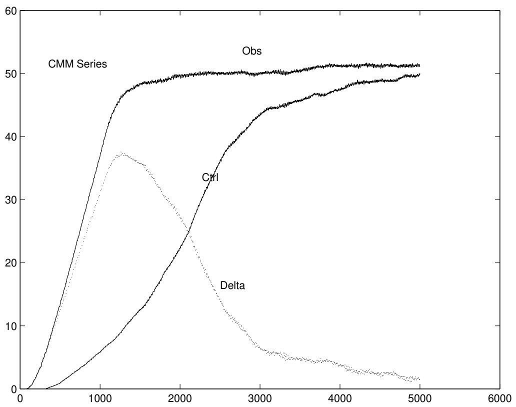

Figure 10: Learning from multiple mentors

species. Human imitation has been studied in contexts as varied as motor development and language acquisition [21]. Many researchers have considered imitation to be an important means of enhancing learning in multiagent systems [2, 4, 5, 13], and many interesting ideas have been pursued in the literature on imitation and teaching.

Much of this work relies on the observer *sharing* a common or similar (possibly perceptual) state with the mentor. In physical systems (e.g., mobile robots) this is often achieved by forcing the observer to follow the mentor [4, 5] or arranging for communication to occur when agents are in close proximity [13]. This can be viewed as ensuring both full observability of the mentor's state and resolving the "state analogy" problem. This approach is useful when partial observability is at issue, though it can be plagued by synchronization and alignment problems [3]. More principled approaches to partial observability should incorporate POMDP models (see below). The "action analogy" problem is addressed formally in [19], where an algebraic formulation of the implications of different abilities on the part of the mentor and observer is provided. We discuss this further below, but at present it is not clear how to use the specific formulation of [19] with RL.

Much work on imitation requires that the mentor directly communicate its intended action or reward, or provide some other feedback to the observer [4, 13, 25], or that complete state, reward and action information is directly accessible [11]. Research on explicit *teaching* or *demonstration* of behaviors includes work focused on abstract domains [11, 25] and on specific robotics tasks [1, 10, 16], where behaviors are learned by replicating observed human behaviors. This work relies on a cooperative relationship between the mentor and observer; in contrast, our model makes no such assumption. The middle ground—where a mentor must decide whether (and how) to facilitate the learning of another agent—should prove to be an interesting research topic. Tan [22] explores multiagent RL where agents share reward information and pool their perceptions—the focus is on accelerating learning. Knowledge transfer occurs through direct communication. While not imitation or teaching, this type of transfer could enhance imitative capabilities.

Work on "learning apprentices" [15, 18] can be seen as a form of learning by observation as well. The goal is to observe a human "mentor" and extract convenient rules of behavior to recommend to the mentor. This work differs from ours in that the observer is not attempting to solve its own decision problem, but is simply trying to improve the abilities of the mentor. Finally, some work has been done on using knowledge of other agent's decisions to place partial order constraints on the value function over states [23].

5.2 Possible Extensions of the Model

For our model to be more widely applied, a number of restrictive assumptions must be relaxed. We are currently exploring some of these. Though our agents have different reward structures, we have assumed that they have common state and action spaces. While requiring that the observer have the same abilities as the mentor is crucial to the simple augmented Bellman equation presented, it is often unrealistic. We are currently extending our algorithms to deal with agents with differing abilities, using methods that allow an observer to reason about how it can approximate a given mentor *trajectory* (rather than a specific action), and methods for reasoning about the similarity of the observer's actions to the observed mentor transition distribution at a given state. Preliminary work in model-free settings has shown remarkable transfer even when the mentor and observer have rather different actions at their disposal. We are also exploring the use of analogical mappings between state spaces to relax the assumption that the agents have a common state space. See [19] for more on such mappings.

We must also allow for partially observable models, especially with respect to the observer obtaining information about the true state (and trajectories) of the mentor. Our model can easily be extended to allow partial observability and we are currently tackling extensions of our model extraction algorithm in this framework. We hope to provide a principled approach to devising strategies that facilitate learning in this framework (in contrast to heuristics such as explicit following).

A more challenging extension would be the application of model extraction to models that allow interactions between the observer and the mentor. If both are self-interested, a game-theoretic model will be required to account for their strategic reasoning. An observer may need to account for the possibility of deception on the part of the mentor. Even in cooperative settings, issues such as coordination must be resolved. On the other hand, new opportunities for facilitation emerge (e.g., the mentor may alter the environment so that the observer can learn more readily, or learn in a risk-free environment).

Under the current modeling assumptions, other extensions of the algorithm can be explored. We are currently extending our explicit confidence measures in order to address reasoning about mentors with different abilities than the observer. More sophisticated exploration techniques [6, 14] can also exploit confidence factors such as these. Finally, we hope to explore generalization and value function approximation methods within this framework.

5.3 Summary

We have described a formal and principled approach to imitation called model extraction. For stochastic problems in which explicit forms of communication are not possible, the underlying model-based framework combined with model extraction provides an alternative to other imitation and learning-by-observation systems. We have shown model extraction to offer significant transfer capability on several test problems, where it proves to be robust in the face of noise, capable of integrating subskills from multiple mentors, and to provide benefits that increase with the difficulty of the problem. Our initial assumptions are somewhat restrictive—the observer and mentor have similar state and action spaces, for instance—but we are pursuing techniques that relax these.

Since model-extraction is a simple addition to model-based reinforcement learning we expect to be able to generalize the technique to many of the domains to which model-based techniques have been applied. Given its compact form and principled framework, we expect model extraction will be a sound basis on which to build a variety of algorithms for implicit forms of imitation.

Acknowledgements Thanks to the anonymous referees for their suggestions. Bob also thanks Gord McCalla for his early support of research into imitation. This research was supported by NSERC Research Grant OGP0121843, and IRIS Phase-III Project BAC.

References

[1] C. G. Atkeson and S. Schaal. Robot learning from demonstration. *ICML-97*, 1997.

[2] Paul Bakker and Yasuo Kuniyoshi. Robot see, robot do : An overview of robot imitation. *AISB96 Workshop on Learning in Robots and Animals*, pages 3–11, 1996.

[3] Aude Billard and Kerstin Dautenhahn. Grounding communication in autonomous robots: an experimental study. *Robotics and Autonomous Systems, Special Issue on Scientific Methods in Mobile Robotics, M. Recce and U. Nehmzow (eds.)*, 1-2(24):71–81, 1998.

[4] Aude Billard and Gillian Hayes. Learning to communicate through imitation in autonomous robots. *ICANN 97*, pages 763–68, 1997.

[5] Kerstin Dautenhahn. Getting to know each other – artificial social intelligence for autonomous robots. *Robotics and Autonomous Systems*, 16:333–356, 1995.

[6] R. Dearden, N. Friedman, and S. Russell. Bayesian Q-learning. In *AAAI-98*, pages 761–68. AAAI Press, 1998.

[7] M. H. Degroot. *Probability and Statistics*. Addison-Wesley, New York, 1986.

[8] G. Fiorito and P. Scotto. Observational learning in octopus vulgaris. *Science*, 256:545–47, 1992.

[9] Leslie Pack Kaelbling. *Learning in Embedded Systems*. MIT Press, Cambridge, 1993.

[10] Yasuo Kuniyoshi, Masayuki Inaba, and Hirochika Inoue. Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation*, 10(6):799–822, 1994.

[11] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8:293, 1992.

[12] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163, New Brunswick, NJ, 1994.

[13] Maja J. Mataric. Using communication to reduce locality in distributed multi-agent learning. *Journal of Experimental and Theoretical Artificial Intel.*, 10(3):357–369, 1998.

[14] Nicolas Meuleau and Paul Bourgine. Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*, 1998. to appear.

[15] T. M. Mitchell, S. Mahadevan, and L. Steinberg. LEAP: A learning apprentice for VLSI design. *IJCAI-85*, pages 573–580, 1985.

[16] H. Miyamoto, F. Gandolfo, H. Gomi, S. Schaal, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato. A kendama learning robot based on a dynamic optimization theory. *IEEE Int. Workshop on Robot and Human Comm.*, 1995.

[17] Andrew W. Moore and Christopher G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less real time. *Machine Learning*, 13, 1993.

[18] Hiroshi Motoda and Kenichi Yoshida. Machine learning techniques to make computers easier to use. *Artificial Intelligence*, 103(1-2):295–321, 1998.

[19] Chrystopher Nehaniv and Kerstin Dautenhahn. Mapping between dissimilar bodies: Affordances and the algebraic foundations of imitation. In *Proc. European Workshop on Learning Robots*, Edinburgh, 1998.

[20] L. S. Shapley. Stochastic games. *Proc. National Academy of Sciences*, 39:327–332, 1953.

[21] Gisela E. Speidel and Keith E. Nelson. *A Fresh Look at Imitation in Language Learning*, pages 1–19. Springer-Verlag, London, UK, 1989.

[22] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. *ICML-93*, pages 330–37, 1993.

[23] Paul E. Utgoff and Jeffrey A. Clouse. Two kinds of training information for evaluation function learning. *Proceedings of Ninth National Conference on Artificial Intelligence*, pages 596–600, 1991.

[24] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[25] Steven D. Whitehead. Complexity analysis of cooperative mechanisms in reinforcement learning. *AAAI-91*, pages 607–613, 1991.