Embedded Teaching of Reinforcement Learners

Ronen I. Brafman
Stanford University
Stanford, CA 94305
brafman@cs.stanford.edu

Moshe Tennenholtz
Technion - Israel Institute of Technology
Haifa 32000
moshet@ie.technion.ac.il

Abstract
Knowledge plays an important role in an agent's ability to perform well in its environment. Teaching can be used to improve an agent's performance by enhancing its knowledge. We propose a specific model of teaching, which we call *embedded teaching*. An embedded teacher is an agent situated with a less knowledgeable "student" in a common environment. The teacher's goal is to lead the student to adopt a particular desired behavior. The teacher's ability to teach is affected by the dynamics of the common environment and may be limited by a restricted repertoire of actions or uncertainty about the outcome of actions; we explicitly represent these limitations as part of our model. In this paper, we address a number of theoretical issues including the characterization of a challenging embedded teaching domain and the computation of optimal teaching policies. We then incorporate these ideas in a series of experiments designed to evaluate our ability to teach two types of reinforcement learners.

1 Introduction

An agent's ability to function in an environment is greatly affected by its knowledge of the environment. In some special cases, we can design agents with sufficient knowledge for performing a task [Gol93], but, in general, agents must acquire information on-line in order to optimize their performance, i.e., they must learn.

One possible approach to improving the performance of learning algorithms is employing a teacher. For example, Lin [Lin92] uses teaching by example to improve the performance of agents, supplying them with examples that show how the task can be achieved. Tan's work [Tan93] can also be viewed as a form of teaching in which agents share experiences. In both methods some non-trivial form of communication or perception is required. We strive to model a broader notion of teaching that encompasses any behavior that can improve another agent's performance. At the same time, we want our model to clearly delineate the limits of the teacher's ability to influence the student.

In this paper, we propose a teaching approach that maintains a situated "spirit" much like that of reinforcement learning [Sut88, Wat89, Kae90], which we call *embedded teaching*. An embedded teacher is simply a "knowledgeable" agent situated with the student in a shared environment. Her[1] goal is to lead the student to adopt some specific behavior. However, the teacher's ability to teach is restricted by the nature of the environment they share: not only is her repertoire of actions limited, but she may also lack full control over the outcome of these actions. As an example, consider two mobile robots without any means of direct communication. Robot 1 is familiar with the surroundings, while Robot 2 is not. In this situation, Robot 1 can help Robot 2 find a goal through certain actions, such as blocking Robot 2 when it is headed in the wrong direction. However, Robot 1 may have only limited control over the outcome of such an interaction because of uncertainty about the behavior of Robot 2.

Our goal is to understand how an embedded teacher can help a student adopt a particular behavior. This paper addresses a number of theoretical questions relating to this problem, and then experimentally explores the ability to teach two types of reinforcement learners. First, we propose two-person games as a framework for studying embedded teaching, and identify one class of games, the iterated prisoner's dilemma, in which teaching is particularly challenging. Then we show how, under certain assumptions, an optimal teaching policy can be derived. Such an optimal teaching policy is valuable not only for its own sake, but also because it supplies us with an upper-bound on any method's ability to influence the behavior of a student. This gives us insight into the properties of the student's learning algorithm.

In the experimental part of this paper we try to teach agents using two versions of the Q-learning algorithm [Wat89]. "Normal" Q-learners (QL) use the full Q-learning algorithm while "blind" Q-learners (BQL) model the world as having only one state. Our choice of Q-learners as the subjects of these experiments is motivated both by our view that reinforcement-learning is one of the most natural learning methods for situated learning and by the popularity of Q-learning within the reinforcement-learning community. Our experiments explore the challenging teaching context mentioned above and use the insight on optimal teaching methods gained in the earlier theoretical discussion. We examine a number of possible teaching strategies that make different assumptions about the teacher's knowledge and compare their success rates. In the case of BQL we observe an interesting phase-transition in the teacher's success rate as the student's inclination to explore is varied. In the case of QL

[1] To differentiate between teacher and student, we use female pronouns for the former and male pronouns for the latter.

	A	B
1	a	b
2	c	d

(A)

	A	B
1	6	5
2	1	2

(B)

	A	B
1	5	1
2	2	6

(C)

	A	B
1	3	-2
2	5	6

(D)

	A	B
1	5	-10
2	10	-5

(E)

Figure 1: Matrices A, B, C, D and E

we show that a simple teaching strategy based on the concepts of punishment and reward often provides an excellent teaching method. We then analyze the conditions under which this strategy is likely to succeed.

2 The basic setting

In the introduction we suggested viewing teaching as a special form of interaction between two agents. In this paper we will focus on a particular type of two-agent interaction, in which the teacher and student are repeatedly engaged in some joint activity. While the student has no prior knowledge pertaining to this activity, the teacher understands its dynamics. In our model, the teacher's goal is to lead the student to adopt a particular behavior in such interactions. For example, teacher and student meet occasionally at the road and the teacher wants to teach the student to drive on the right side. Or perhaps, the teacher and the student share some resource, such as CPU time, and the goal is to teach him judicious use of this resource.

A natural model for two-agent interactions is a two-agent game [Owe82]. A two-agent game consists of two players; associated with player i is a set of possible actions A_i. Each play of the game involves a choice of an action by each player. A *joint action* is a tuple of the form (a_1, a_2), where $a_i \in A_i$ is player i's action. For each joint action the game specifies a payoff (or reward) vector (p_1, p_2), where p_i is the payoff received by agent i following this joint action. Since the teacher and the student are repeatedly interacting, we model them as players of a two-agent *iterated game*, which is simply a sequence of plays of a two player game. To capture the idea that the teacher is more knowledgeable than the student, we assume that she knows the structure of the game, i.e., she knows the payoff function, and that she recognizes the actions taken at each play. On the other hand, the student does not know the payoff function, although he can perceive the payoff he receives.

In this paper, we make the simplifying assumption that both teacher and student have only two actions from which to choose. Furthermore, we do not care about the cost of teaching and thus will ignore the teacher's payoff in our description. This provides a basic setting in which to take this first step towards understanding the teaching problem.

The teaching model can be concisely modeled by a 2×2 matrix. The teacher's actions are designated by the letters A and B, while the student's actions are designated by the numbers 1 and 2. Each entry corresponds to a joint action and represents the student's payoff when this joint action is played. We will suppose that we have matrix A in Figure 1 and that we wish to teach the student to use action **1**. At this stage all we assume about the student is that if he *always* receives a better payoff following action 1 he will learn to play it.

We can see that in some situations teaching is trivial. Assume that the first row dominates the second row, i.e., $a > c$ and $b > d$. In that case, the student will naturally prefer to take action 1, and teaching is not very challenging, although it might be a useful way to speed the learning process. For example, if $a - c > b - d$, as in matrix B in Figure 1, the teacher can make the advantage of action 1 more noticeable to the student by always playing action A.

Now suppose that only one of $a > c$ or $b > d$ holds. In this case, teaching is still easy. We use a basic teaching strategy, which we call *preemption*. In preemption the teacher chooses an action that makes action 1 look better than action 2. For example, when the situation is described by matrix C in Figure 1, the teacher will always choose action A.

Next, assume that both c and d are greater than both a or b, as in matrix D in Figure 1. Regardless of which action the teacher chooses, the student receives a higher payoff by playing action 2 (since $\min\{5, 6\} > \max\{3, -2\}$). Therefore, no matter what the teacher does, the student will learn to prefer action 2. Teaching is hopeless in this situation.

All other types of interactions are isomorphic to the case where $c > a > d > b$, as in matrix E in Figure 1. This is still a challenging situation for the teacher, because action 2 dominates action 1 (because $10 > 5$ and $-5 > -10$). Therefore, preemption cannot work. If teaching is possible it will be more complex than always choosing the same action. Since this seems to the most challenging teaching situation, we will devote our attention in Sections 5 and 6 to teaching a reinforcement learner to choose action 1 in this class of games.

It turns out that the above situation is quite important in game-theory and multi-agent interaction. It is a projection of a very famous game, the prisoner's dilemma, which can be represented as:

	teacher	
student	Coop	Defect
Coop	(a,a)	(b,c)
Defect	(c,b)	(d,d)

or more commonly

	teacher	
student	Coop	Defect
Coop	(a,a)	(-c,c)
Defect	(c,-c)	(d,d)

where $c > a > d > b$. The actions in the prisoner's dilemma are called Cooperate (Coop) and Defect; we identify Coop with actions 1 and A, and Defect with actions 2 and B. The prisoner's dilemma captures the essence of many important social and economic situations; in particular, it encapsulates the notion of cooperation. It has thus motivated enormous discussion among game-theorists and mathematical economists [EMN89]. In the prisoner's dilemma, whatever the choice of one player, the second player can maximize its payoff by playing Defect. It thus seems "rational" for each player to defect. However, when both players defect, their payoffs are much worse than if they both cooperate.

As an example, suppose two agents will be given \$10 each for moving some object. Each agent can perform the task alone, but it will take an amount of time and energy which they value at \$20. However, together, the effort each will make is valued at \$5. We get the following instance of the prisoner's dilemma:

	Agent 1	
Agent 2	Move	Rest
Move	(5,5)	(-10,10)
Rest	(10,-10)	(0,0)

In the experimental part of this paper the teacher's task will be to teach the student to cooperate in the prisoner's dilemma. We will measure the success of a teaching strategy by looking at the cooperation rate it induces in students over some period of time, i.e., the percentage of the student's actions which are Coop. All results shown in this paper are with respect to the following matrix:

	Teacher	
Student	Coop	Defect
Coop	(10,10)	(-13,13)
Defect	(13,-13)	(-6,-6)

We have observed qualitatively similar results in other instantiations of the prisoner's dilemma, although the precise cooperation rate varies.

3 Optimal teaching policies

In the previous section we concentrated on modeling the teaching context and determining which particular problems are interesting. In this section we start exploring the question of how a teacher should teach. First we will define what an optimal policy is. Then we will define Markov decision processes (MDP) [Bel62], and show that under certain assumptions teaching can be viewed as an MDP. This will allow us to tap into all the vast knowledge that has accumulated on solving these problems. In particular, we can use well known methods, such as value iteration [Bel62], to find the optimal teaching policy.

We start by defining an optimal teaching policy. A *teaching policy* is a function that returns an action at each iteration; possibly, it may depend on a complete history of the past joint actions. There is no "right" definition for an optimal policy, as the teacher's motivation may vary. However, in this paper, the teacher's objective is to maximize the number of games in which the student's action are "good", e.g., Coop in the prisoner's dilemma. The teacher does not know the precise number of iterations she will be playing, so she slightly prefers earlier success to later success. Formally, we assume that each teaching policy induces a probability distribution $\mu(\cdot)$ over S, the student's space of possible action sequences, and define the following: Let $\{\sigma_n\} = (\sigma_1, sigma_2, \ldots)$ be an infinite sequence of actions and let $u(a)$ be the value the teacher places on a student's action, a. Then

$$v(\{\sigma_n\}) \stackrel{\text{def}}{=} \sum_{k=0}^{\infty} \left(\gamma_0^k \cdot u(\sigma_k)\right)$$

The coefficient γ_0 is called the *discount factor*; it captures the idea of caring less about events that are further in the future. The value of a teaching policy π is defined to be

$$val(\pi) = \int_S v(s) d\mu(s)$$

Thus, an optimal policy maximizes val, the expected value of v.[2]

Next, we define MDPs. In an MDP a decision maker is continually moving between different states. At each point in time she observes her current state, receives some payoff (which depends on this state), and chooses an action. Her action and her current state determine (perhaps stochastically) her next state. The goal is to maximize some function of the payoffs. Formally, an MDP is a four-tuple $\langle S, A, P, r\rangle$, where S is the state-space, A is the decision-maker's set of possible actions, $P : S \times S \times A \to [0,1]$ is the probability of a transition between states given the decision-maker's action, and $r : S \to \mathbb{R}$ is the reward function.

Now suppose that the student can be in a set Σ of possible states, that his set of actions is A_s, and that the teacher's set of actions is A_t. Moreover, suppose that the following properties are satisfied

(1) The student's new state is a function of his old state and the current joint-action, denoted by $\tau : \Sigma \times A_s \times A_t \to \Sigma$;

(2) The student's action is a stochastic function of his current state, where the probability of choosing a at state s is $\rho(s, a)$;

(3) the teacher knows the student's state. (The most natural way for this to happen is that the teacher knows the student's initial state and the function τ and uses them to simulate the agent.)

Notice that under these assumptions a teaching policy should be a a function of Σ. The only information relevant to the agent's future actions are his future states. The only information relevant to his future states are his current state (which also determines his current action) and the teacher's action.

[2] Formally defining μ over S requires defining an appropriate σ-algebra. However, we note that because the set of actions is finite, S can be treated much like $\mathbb{R}$ and the σ-algebra of Borel sets can be used. In particular, u is a continuous function on this space, and the integral is well defined.

Therefore, the only knowledge relevant to the teacher's choice is the student's current state. It is easy to see that we have the makings of the following MDP.

Define the teacher's MDP to be TMDP= $\langle \Sigma, A_t, P, U \rangle$, where

$$P(s, s', a_t) \stackrel{\text{def}}{=} \sum_{a_s \in A_s} \rho(s, a_s) \cdot \delta_{s', \tau(s, a_s, a_t)}$$

($\delta_{i,j}$ is defined as 1 when $i = j$, and 0 otherwise). That is, the probability of a transition from s to s' under a_t is the sum of probabilities of a student action that will induce this transition. The reward function is the expected value of u, i.e.,

$$U(s) \stackrel{\text{def}}{=} \sum_{a_s \in A_s} \rho(s, a_s) \cdot u(a_s)$$

Theorem 1 *The optimal teaching policy is given by the γ_0 optimal policy in TMDP.*

The γ_0-optimal policy in an MDP is the policy that maximizes at each state the expected value of $\sum_{n=1}^{\infty} [\hat{u}_{(n)} \cdot \gamma_0^{n-1}]$, where $\hat{u}(n)$ is the payoff received at the n^{th} future state. There are well-known ways of finding this policy. In the experiments below we use a method based on value-iteration [Bel62].

The optimal policy can be used for teaching, when the teacher possess sufficient information to determine the current state of the student. But even when that is not the case, it allows us to calculate an upper bound on the success $val(\pi)$ of any teaching policy π. This number is a property of the *learning* algorithm, and measures the degree of influence any agent can have over the given student.

4 The learning algorithm

We experiment with two types of students. One uses a reinforcement learning algorithm which can be viewed as Q-learning with one state, and the other uses Q-learning. In choosing parameters for these students we tried to emulate choices made in the reinforcement learning literature.

The first student, which we call a *Blind Q-learner* (BQL), can perceive rewards, but cannot see how the teacher has acted or remember his own past actions. He only keeps two values, one value for each action: q(Coop) and q(Defect). His update rule is the following: if he performed action a and received a reward of R then

$$q_{new}(a) = (1 - \alpha) \cdot q_{old}(a) + \alpha \cdot R$$

The parameter α, the *learning-rate*, is fixed to 0.1 in our experiments.

The second student is a "real" Q-learner (QL). He can observe the teacher's actions and has a number of possible states. The QL maintains a Q-values for each state-action pair. His states encode his recent experiences, i.e., the past joint actions. The update rule is:

$$q_{new}(s, a) = (1 - \alpha) \cdot q_{old}(s, a) + \alpha \cdot (R + \gamma V(s'))$$

Here R is the reward received upon performing a at state s; s' is the state of the student following the performance of a at s; γ is called the *discount factor*, and will be 0.9, unless otherwise noted; and $V(s')$ is the current estimate of the value of the best policy on s', defined as $\max_{a \in A_s} q(s', a)$. All Q-values are initially set to zero.

The student's update rule tells us how his Q-values change as a result of new experiences. We must also specify how these Q-values determine his behavior. Both types of students choose their actions based on the Boltzmann distribution. This distribution associates a probability $P_s(a)$ with the performance of an action a at a state s ($P(a)$ for the BQL).

$$P_s(a) \stackrel{\text{def}}{=} \frac{\exp(q(s,a)/T)}{\sum_{a' \in A} \exp(q(s,a')/T)} \quad (QL) \qquad P(a) \stackrel{\text{def}}{=} \frac{\exp(q(a)/T)}{\sum_{a' \in A} \exp(q(a')/T)} \quad (BQL)$$

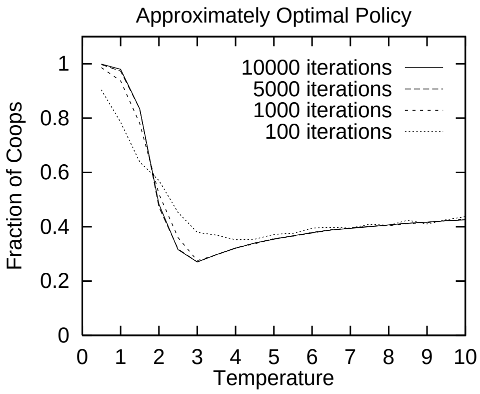
Here T is called the *temperature*. Usually, one starts with a high value for T, which makes the action choice more random, inducing more exploration on the part of the student. T is slowly reduced, making the Q-values play a greater role in the student's choice of action. We use the following schedule: $T(0) = 75$ and $T(n+1) = T(n) * 0.9 + 0.05$. This schedule has the characteristic properties of fast initial decay and slow later decay. We also experiment with fixed temperature.

5 Blind Q-Learners

This section describes our experimental results with BQL. We examined a policy that approximates the optimal policy, and two teaching methods that do not rely on a student model.

Optimal policy. First we show that BQLs fit the student model of Section 3. For their state space, we use the set of all possible assignment for their Q-values. This is a continuous subspace of $\mathbb{R}^2$, and we discretize it (in order to be able to compute a policy), obtaining a state space with approximately 40,000 states. Next, notice that transitions are a stochastic function of the current state (current Q-values) and the teacher's action. To see this notice that Q-value updates are a function of the current Q-value and the payoff; the payoff is a function of the teacher's and student's actions; and the student's actions are a stochastic function of the current Q-value. In the left side of Figure 2 we see the success of teaching using the policy generated by using dynamic programming to solve this optimization problem. Each curve represents the fraction of Coops as a function of the temperature for some fixed number of iterations. The values are means over 100 experiments.

Two Q-Learners. We also ran experiments with two identical BQLs. This can be viewed as "teaching" using another Q-learner. The results are shown in the right side of Figure 2. At all temperatures the optimal strategy performs better than Q-learning as a "teaching" strategy. The fact that at temperatures of 1.0 or less the success rate approaches 1 will be beneficial when we later add temperature decay. However, we also see that there is an inherent limit



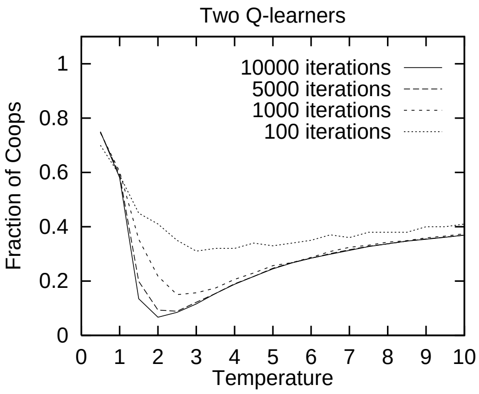


Figure 2: Fraction of Coops as a function of temperature for the approximately optimal policy (left) and for "teaching" using an identical Q-learner (right). Each curve corresponds to Coop rate over some fixed number of iterations. In the approx. optimal policy the curves for 1000, 5000 and 10000 iterations are nearly identical.

to our ability to affect the behavior at higher temperatures. An interesting phenomenon is the phase transition observed around $T = 2.5$. A qualitative explanation of this phenomenon is that high temperature adds randomness to the student's choice of action, because it makes the probabilities $P(a)$ less extreme. Consequently, the ability to predict the student's behavior lessens, and with it the probability of choosing a good action. However, while randomness serves to lower the success rate initially, it also guarantees a level of effective cooperation, which should approach 0.5 as the temperature increases. Finally, notice that although (Coop,Coop) seems like the best joint-action for a pair of agents, two interacting Q-learners never learn to play this joint strategy consistently, although they approach 80% Coops at low temperatures.

Teaching without a model. When the teacher does not have a precise model of the student, we cannot use the techniques of Section 3 to derive an optimal policy, because it is assumed that the teacher can "observe" the student's current state (i.e. that it knows the student's Q-values). We therefore explore two teaching methods that only exploit knowledge of the game and the fact that the student is a BQL.

Both methods are motivated by a basic strategy of *countering* the student's move. The basic idea is to try and counter good actions by the student with an action that will lead to a high payoff, and to counter bad actions with an action that will give him a low payoff. Ideally, we would like to play Coop when the student plays Coop, and Defect when the student plays Defect. Of course, we don't know what action the student will choose, so we try to predict from his past actions.

If we assume that the Q-values change very little from one iteration to the

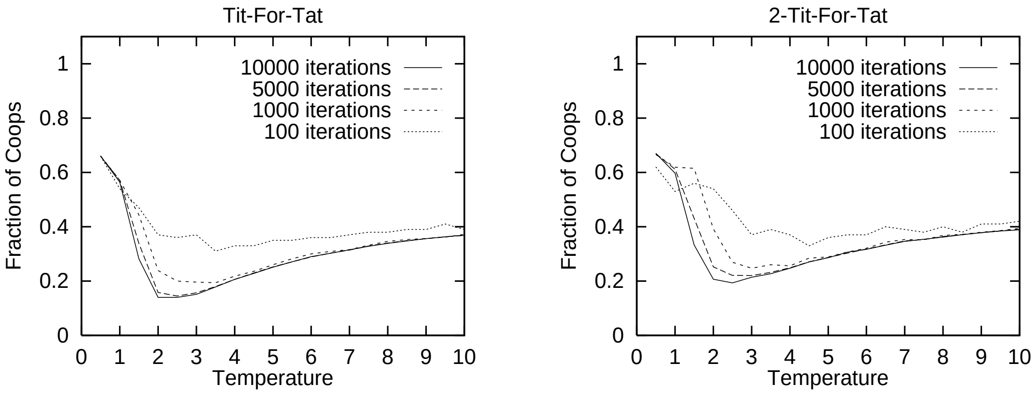


Figure 3: Fraction of Coops as a function of temperature for the teaching strategy based on TFT (left) and 2TFT (right).

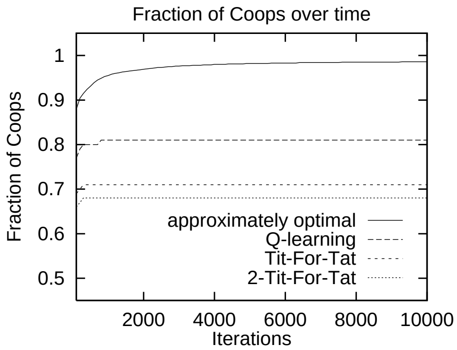
other, the student's most likely action in the next game is the same action that he took in the most recent game. Therefore, if we saw the student play Coop in the *previous* turn, we will play Coop *now*. Similarly, the teacher will follow a Defect by the student with a Defect on her part. This strategy, called Tit-Tor-Tat (TFT for short), is well known [EMN89]. Our experiments show that it is not very successful in teaching a BQL (see Figure 3).

We also experimented with a variant of TFT, which we call 2TFT. In this strategy the teacher plays Defect only after observing two consecutive Defects on the part of the student. It is motivated by our observation that in certain situations it is better to let the student enjoy a free lunch (i.e., match his Defect with a Coop) than to make Coop look bad to him, because that may cause his Q-value for Coop to be so low that he is unlikely to try it again. Two consecutive Defects indicate that the probability of the student playing a Defect as his next action is quite high. The results, shown in Figure 3, indicate that this strategy worked better than TFT, and in some ranges of temperature, better than Q-learning. However, in general, both TFT and 2TFT gave disappointing results.[3]

Finally, Figure 4 shows the performance of all four teaching strategies discussed so far when we incorporate temperature decay. We can see that the optimal policy is very successful. As we explained before, teaching is easier when the student is more predictable, which is the case when temperature is lower. With temperature decay the student spends most of his time in relatively low temperature and behaves similarly to the case of fixed, low temperature. While an initial high-temperature phase could have altered this behavior, we did not observe such effects.

[3] However, in some sense an identical Q-learner embodies a model of the student.

Figure 4: Fraction of Coops as a function of time for BQL using the temperature decay scheme of Section 3. Teaching strategies shown: approximately optimal strategy, Q-learning, TFT, and 2TFT.



6 Teaching Q-learners

Unlike BQL, Q-learners (QL) have a number of possible states which encode the joint actions of previous games played. A QL with memory one has four possible states, corresponding to the four possible joint actions in the prisoner's dilemma; a QL with more memory will have more states, encoding a sequence of joint actions.

More complex learning architectures have more structure, which brings with it certain problems. One possible problem may be that this structure is more "teaching-resistant." A more real threat is added computational complexity. As we mentioned, to approximate the optimal teaching policy for BQL we had to compute over a space of approximately 40,000 discretized states. While representing the state of a BQL requires only two numbers, one for each Q-value, representing the state of a QL with m states requires $2m + 1$ numbers: one for the Q-value of each state/action pair, and one encoding the current state. The size of the corresponding discretized state-space for the teacher's Markov decision process grows exponentially in m. For the simplest case of memory one (a student with four states) this would be about 10^{18} states. Since solving the problem with 40,000 states took 12 hours on a Sun SPARCstation-10, we were not able to approximate optimal teaching policies for even the simplest QL.

But all is not lost. More structure may mean more complexity, but it also means more properties to exploit. We can reach surprisingly good results by exploiting the structure of Q-learners. Moreover, we can do this using a teaching method introduced in the previous section. However, in QL this method takes on a new meaning that suggests the familiar notions of reward and punishment.

In choosing their actions, QLs "care" not only about immediate rewards, but also about the current action's effect on future rewards. This makes them suitable for a reward and punishment scheme. The idea is the following: suppose the QL did something "bad" (Defect in our case). Although we cannot reliably counter such a move with a move that will lower his reward, we can punish him later by choosing an action that always gives a negative payoff,

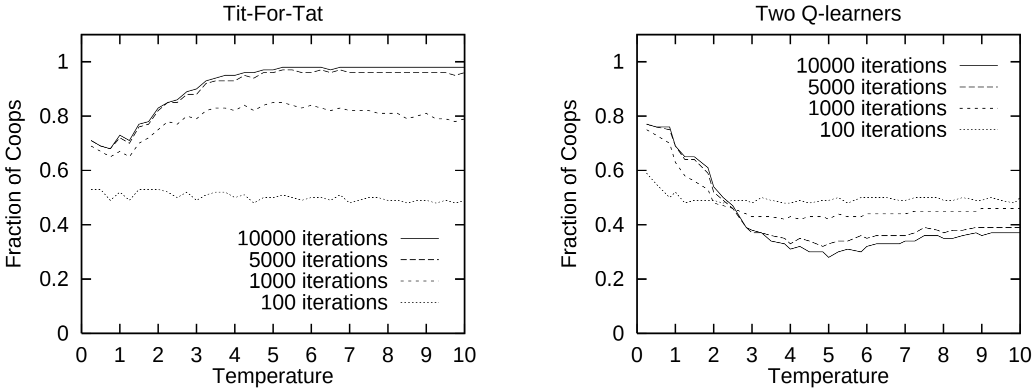


Figure 5: Each curve shows the fraction of Coops as a function of temperature for a fixed number of iterations when TFT was used to teach (left) and when Q-learning was used to teach (right). Values are means over 100 experiments.

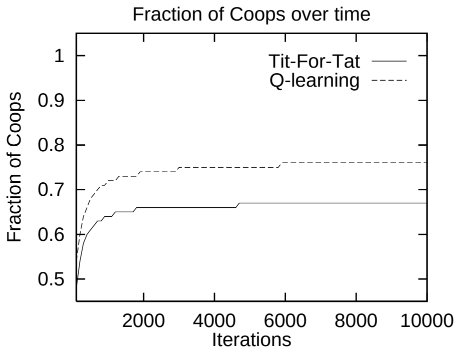
no matter what the student plays. We achieve this by following a student's Defect with a Defect by the teacher. While the immediate reward obtained by a QL playing Defect may be high, he will also learn to associate a subsequent punishment with the Defect action. Thus, while it may be locally beneficial to perform Defect, we may be able to make the long-term rewards of Defect less desirable. Similarly, we can follow a student's Coop with a reward in the form of a Coop by the teacher, since it guarantees a positive payoff to the student.

This suggests using Tit-Tor-Tat again. Notice that for BQLs, TFT *cannot* be understood as a reward/punishment strategy because BQLs care only about the immediate outcome of an action; the value they associate with each action is a weighted average of the immediate payoffs generated by playing this action.

In Figure 5 we see the success rates of TFT as a function of temperature, as well as the rates for Q-learning as a teaching strategy, and Figure 6 shows the results when we introduce temperature decay. It is apparent that TFT is extremely successful, especially in higher temperatures. Interestingly, the behavior is quite different than that of two QLs. The phase-change noticed in BQL still exists (to a lesser extent) when we look at two QLs. However, TFT exhibits completely different behavior: Coop levels increase with temperature, reaching almost 100% above 3.0. It seems that TFT works better when the QL exhibits a certain level of experimentation.

In these experiments the QL remembers only the last joint action. We experimented with QL with more memory and performance was worse. This can be explained as follows. For a QL with memory one or more, the problem is a fully observable Markov decision process *once* the teacher plays TFT, because TFT is a deterministic function of the previous joint action. We know that Q-learning converges to the optimal policy under such conditions [WD92]. Adding more memory effectively adds irrelevant attributes, which, in turn,

Figure 6: Fraction of Coops as a function of time with temperature decay.



causes a slower learning rate. We have also examined whether 2TFT would be successful when agents have a memory of two. The results are not shown here, but the success rate was considerably lower than for TFT, although better than for two QLs.

TFT performed well as a teaching strategy, and we explained the motivation for using it. We now want to produce a more quantitative explanation, one that can be used to predict its success when we vary various parameters, such as the payoff matrix.

Let the student's payoff matrix be as in matrix A of Figure 1; let p be the probability that the student plays Coop, and let $q = 1 - p$ be the probability that the student plays Defect. These probabilities are a function of the student's Q-values (see the description in Section 4). Let us assume that the probabilities p and q do not change considerably from one iteration to the next. This seems especially justified when the learning rate, α, is small.

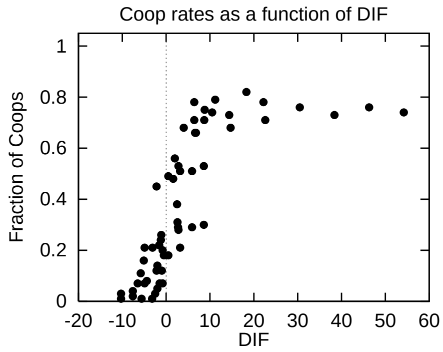
Given this information, what is the student's expected reward for playing Coop? In TFT, the teacher's current action is the student's previous action, so we can also assume that the teacher will play Coop with probability p. Thus, the student's expected payoff for playing Coop is $(p \cdot a + q \cdot b)$. Since Q-learners care about their discounted future reward (not just their current reward), what happens next is also important. Since we assumed that the student cooperated, the teacher will cooperate in the next iteration, and if we still assume p to be the probability that the student will cooperate next, the student's expected payoff in the next step is $(p \cdot a + q \cdot c)$. If we ignore higher order γ terms the expected reward of playing Coop becomes: $p \cdot a + q \cdot b + \gamma(p \cdot a + q \cdot c)$. The expected reward of Defect is thus: $p \cdot c + q \cdot d + \gamma(p \cdot b + q \cdot d)$. Therefore, TFT should succeed as a teaching strategy when:

$$p \cdot a + q \cdot b + \gamma(p \cdot a + q \cdot c) > p \cdot c + q \cdot d + \gamma(p \cdot b + q \cdot d).$$

Since initially $p = q = 0.5$, and it is the behavior at the stage where p and q are approximately equal that will determine whether TFT succeeds, we can attempt to predict the success of TFT based on whether:

$$\mathrm{DIF} = a + b + \gamma(a + c) - [(c + d + \gamma(b + d))] \geq 0$$

Figure 7: Coop rates as a function of $\mathrm{DIF} = a + b + \gamma(a + c) - (c + d + \gamma(b + d))$. The means are for 100 experiments, 10000 iterations each. Student's memory is 1.



To test this hypothesis we ran TFT on a number of matrices using Q-learners with different discount factors. The results in Figure 7 show the fraction of Coops over 10000 iterations as a function of DIF for a teacher using TFT, and with temperature decay. We see that DIF is a reasonable predictor of success. When it is below 0, almost all rates are below 20%, and above 8 most rates are above 65%. However, between 0 and 8 it is not successful.

7 Summary and related work

A number of authors have discussed reinforcement learning in multi-agent systems. Yanco and Stein [YS93] examine the evolution of communication among cooperative reinforcement learners. Sen et al. [SSH94] use Q-learning to induce cooperation between two block pushing robots. Shoham and Tennenholtz [ST92] examine the evolution of conventions in a society of reinforcement learners. Kittock [Kit94] investigates the effects of societal structure on multi-agent learning. Littman [Lit94] develops reinforcement learning techniques for agents whose goals are opposed, and Tan [Tan93] examines the benefit of sharing information among reinforcement learners. Finally, Whitehead [Whi91] has shown that n reinforcement learners that can observe everything about each other can decrease learning time by a factor of n. However, the above work is not concerned with teaching, or with the question of how much influence one agent can have over another.

Lin [Lin92] is explicitly concerned with teaching as a way of accelerating learning of enhanced Q-learners. He uses experience replay and supplies students with examples of how the task can be achieved. As we remarked earlier, this teaching approach is different from ours, since the teachers are not embedded in the student's domain.

Within game theory there is an extensive body of work that tries to understand the evolution of cooperation in the iterated prisoner's dilemma and to find good playing strategies for it[EMN89]. In that work both players have the same knowledge, and teaching is not an issue.

Finally, our work has important links to work on conditioning and especially operant conditioning in psychology [Mac83]. In conditioning experiments an experimenter tries to induce changes in its subjects by arranging that certain relationships will hold in their environment, or by explicitly (in operant conditioning) reinforcing the subjects' actions. In our framework the embedded teacher plays a similar role to that of the experimenter. However, there are a number of important differences. We assume that we have a (possibly partial) model of the student, e.g., we know that it is a QL and we may even know the parameters it is using. Most importantly, we are investigating artificial *computational* entities and we are interested in the *efficiency* of the teaching process. However, since the student model adopted in this paper is known to capture human behavior in recurrent situations (e.g., [Thr98, Bla, RE93]), computational techniques may actually have relevance to our understanding of biological systems.

Summary. We proposed embedded teaching as a situated teaching paradigm, suitable for modeling a wide range of teaching instances. We modeled the teacher and the student as players in a particular iterated two-player game, which we showed to be the most challenging game of its type. In our model, the dynamics of the teacher-student interaction is made explicit, and it clearly delineated the limits placed on the teacher's ability to influence the student. We then showed that with a detailed model of the student, optimal teaching policies can be theoretically generated by viewing the teaching problem as a Markov decision process. The performance of the optimal teaching policy serves as a bound on any agent's ability to influence this student. Finally, we examined our ability to teach two types of reinforcement learners. In particular, we showed that when an optimal policy cannot be used, we can use TFT as a teaching method. In the case of Q-learners this policy was very successful. Consequently, we proposed a model that explains this success. In the future we hope to examine other learning architectures and see whether the lessons learned in this domain can be generalized, and whether we can use these methods to accelerate learning in other domains.

References

[Bel62] R.E. Bellman. *Dynamic Programming.* Princeton University Press, 1962.

[Bla] J. M. Blackburn. Acquisition to Skill: An Analysis of Learning Curves. IHRB Report No. 73, 1936.

[EMN89] J. Eatwell, M. Milgate, and P. Newman, editors. *The New Palgrave: Game Theory.* W.W.Norton & Company, Inc., 1989.

[Gol93] K.Y. Goldberg. Orienting parts without sensors. *Algorithmica*, 10(2):201–225, 1993.

[Kae90] L.P. Kaelbling. *Learning in embedded systems.* PhD thesis, Stanford University, 1990.

[Kit94] James E. Kittock. The impact of locality and authority on emergent conventions. In *AAAI '94*, 1994.

[Lin92] L.J. Lin. Self-improving reactive agents based on reinforcement learning, planning, and teaching. *Machine Learning*, 8(3–4), 1992.

[Lit94] M.L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proc. of the 11th Int. Conf. on Mach. Learn.*, 1994.

[Mac83] N.J. Mackintosh. *Conditioning and Associative Learning*. Oxford University Press, 1983.

[Owe82] G. Owen. *Game Theory (2nd Ed.)*. Academic Press, 1982.

[RE93] A.E. Roth and I. Erev. Experimental Data and Simple Dynamics Models in the Intermediate term. Nobel Symposium on Game Theory, 1993.

[SSH94] S. Sen, M. Sekaran, and J. Hale. Learning to coordinate without sharing information. In *Proc. of AAAI-94*, pages 426–431, 1994.

[ST92] Y. Shoham and M. Tennenholtz. Emergent Conventions in Multi-Agent Systems: initial experimental results and observations. In *KR-92*, 1992.

[Sut88] R.S. Sutton. Learning to predict by method of temporal differences. *Mach. Lear.*, 3(1):9–44, 1988.

[Tan93] Ming Tan. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In *Proceedings of the 10th International Conference on Machine Learning*, 1993.

[Thr98] Edward L. Thronkide. Animal intelligence: An experimental study of the associative processes in animals. *Psychological Monographs*, 2, 1898.

[Wat89] C.J.C.H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, 1989.

[WD92] C.J.C.H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3/4):279–292, 1992.

[Whi91] S.D. Whitehead. A complexity analysis of cooperative mechanisms in reinforcement learning. In *AAAI-91*, pages 607–613, 1991.

[YS93] H. Yanco and L.A. Stein. An Adaptive Communication Protocol for Cooperating Mobile Robots. In *From Animal to Animats: Proc. 2nd Intl. Conf. on the Sim. of Adap. Behavior*, pages 478–485, 1993.