Social Reward Shaping in the Prisoner's Dilemma (Short Paper)

Monica Babes
Dept. of Computer Science
Rutgers University
Piscataway, NJ 08854
babes@cs.rutgers.edu

Enrique Munoz de Cote
Politecnico di Milano, DEI
piazza Leonardo da Vinci 32,
I-20133 Milan, Italy
munoz@elet.polimi.it

Michael L. Littman
Dept. of Computer Science
Rutgers University
Piscataway, NJ 08854
mlittman@cs.rutgers.edu

ABSTRACT

Reward shaping is a well-known technique applied to help reinforcement-learning agents converge more quickly to near-optimal behavior. In this paper, we introduce *social reward shaping*, which is reward shaping applied in the multiagent-learning framework. We present preliminary experiments in the iterated Prisoner's dilemma setting that show that agents using social reward shaping appropriately can behave more effectively than other classical learning and non-learning strategies. In particular, we show that these agents can both lead —encourage adaptive opponents to stably cooperate— and follow —adopt a best-response strategy when paired with a fixed opponent— where better known approaches achieve only one of these objectives.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Performance, Experimentation

Keywords

Reinforcement learning, leader/follower strategies, iterated prisoner's dilemma, game theory, subgame perfect equilibrium

1. INTRODUCTION

The field of reinforcement learning [8] is concerned with creating agents that act to maximize received reward in complex, dynamic environments. The majority of work has focused on single agent environments, often conceptualized as Markov decision processes [6], where optimal behavior can be identified independently of the effect of the agent on the environment.

Researchers have developed reinforcement-learning algorithms for general-sum games, with a variety of research goals. In this work, we are concerned with what is called the "prescriptive non-cooperative" agenda [7]. That is, we want to develop techniques for learning that result in agent behaviors that are effective in a variety of settings in which agents act independently and not necessarily cooperatively.

More concretely, the experiments we report below address the following scenario. Our learning agent will be playing an iterated two-player matrix game. This kind of scenario is commonly studied in game theory and it is known that there can be benefits to agents maintaining some history of recent interactions [5]. It has been noted that the amount of history an agent assumes its opponents have is significant [1], so we have decided to restrict all agents to a common, fixed history length.

In this setting, a dichotomy can be made between *leader* and *follower* agents [2]. A follower is an agent that implicitly assumes its opponent is playing a fixed strategy and it attempts to maximize its own reward against this fixed strategy. In contrast, a leader assumes its opponent will adapt to its decisions, and so behaves in a way that encourages the opponent to adopt mutually beneficial behavior. When leaders and followers are paired up, the result is usually harmonious. However, when leaders play leaders or followers play followers, the result is unpredictable and often quite poor. Our interest in this work is in developing an algorithm that can either lead or follow, as appropriate.

In the next section, we describe our experimental setting for studying leaders and followers. Section 3 proposes our new approach, social reward shaping, that biases a follower to display leader-like tendencies. In Section 4, we evaluate the resulting algorithm, along with several well-known alternatives. We find that our approach is able to both lead and follow, depending on the situation. Section 5 concludes with a discussion of how the approach can be generalized and applied in other multiagent scenarios.

2. EVALUATING LEADERS/FOLLOWERS

In this preliminary work, our empirical testbed for studying leading and following is the iterated Prisoner's dilemma. On each round, both players choose to either cooperate (**C**) or defect (**D**). Players receive rewards according to the following scheme:

player	*opponent*	*reward*	*description*
C	**C**	3	mutual cooperation
C	**D**	0	sucker payoff
D	**C**	4	temptation
D	**D**	1	mutual defection

To evaluate a pair of strategies, we have them play 40,000 rounds each run and average 1,000 runs. On each round, players choose **C** or **D** and are given one step of history as

Cite as: Social Reward Shaping in the Prisoner's Dilemma (Short Paper), Monica Babes, Enrique Munoz de Cote and Michael L. Littman, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008),* Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 1389-1392.

state	grim	alwaysC	alwaysD	TFT	Pavlov
CC	**C**	**C**	**D**	**C**	**C**
CD	**D**	**C**	**D**	**D**	**D**
DC	**D**	**C**	**D**	**C**	**D**
DD	**D**	**C**	**D**	**D**	**C**

Table 1: A set of fixed history strategies. State XY means the agent chose X in the previous round and the opponent chose Y.

input. They then are informed of their opponent's choice and their own reward.

Our assessment of a strategy comes from evaluating it against two opponents: a fixed strategy (leader) and an adaptive strategy (follower). We are interested in whether strategies can follow the leader and lead the follower, that is, can they attain (nearly) the highest reward possible against each kind of opponent.

Before we describe the opponent strategies we used in evaluation, we take a moment to describe fixed history strategies more generally.

2.1 Fixed History Strategies

A fixed history strategy is one that maps the action choices of both players in the past k rounds to a fixed probability distribution over action choices. In our studies, k is set to 1, so a fixed history strategy maps from a pair of actions (those chosen by the two players in the previous round).

Table 1 catalogs several fixed history strategies. The first, "grim" cooperates until defected upon. The next two, "alwaysC" and "alwaysD" pay no attention to their history and always cooperate or defect, respectively. "TFT" (tit for tat) chooses its action to match the opponent's previous action. "Pavlov" cooperates if both players agreed in the previous round and otherwise defects.

When faced with a fixed history strategy, like any of those just described, an agent's environment can be modeled as a Markov decision process (MDP). MDPs are defined by states, actions, rewards, transitions, and a discount factor $0 \leq \gamma \leq 1$. An agent in an MDP chooses actions to maximize its total expected discounted reward over the infinite horizon.

Any fixed history strategy maps to an MDP. Specifically, each history corresponds to a state. Transitions between states are a function of the agent's choice and the behavior of the fixed history strategy. For example, an agent playing alwaysC inhabits the MDP depicted in the top of Figure 1.

In the diagram, circles represent states (histories) consisting of the agent's previous action and the opponent's previous action. Arcs are labeled with a choice of action for the agent along with its immediate reward. An arc points to the next state that results from the agent's choice and the implicit choice of the opponent.

When facing alwaysC, there are only two states that can be reached. Regardless of the state, the agent chooses between **D** (reward +4) and **C** (reward +3). The optimal strategy here is clear—alwaysD.

Against TFT, depicted in the bottom of Figure 1, the situation is more complex. Starting from CC, the agent can choose **C** and get +3 or **D** and get +4. Choosing **D** results in a transition to the state DC. Because TFT repeats the agent's previous choice, it will choose **D** from state DC, leaving the agent with two low-scoring options. If the dis-

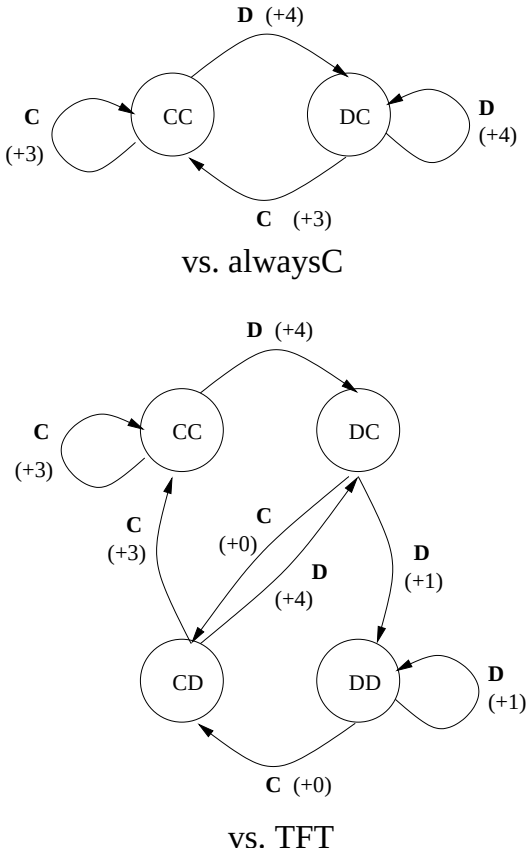

Figure 1: The MDPs induced when an agent faces an opponent playing alwaysC and TFT.

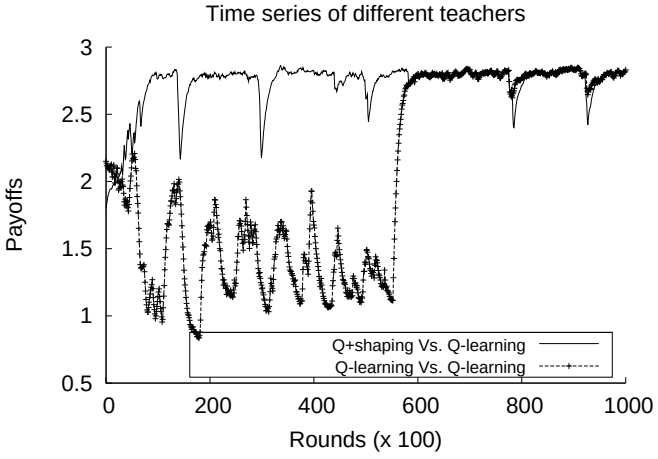

Figure 2: Time series for two runs of Q-learning vs. Q-learning and Q+shaping vs. Q-learning.

count factor is set sufficiently close to 1 ($\gamma = 0.9$ works), the optimal action to take in CC is **C** because the immediate benefit of **D** (+4 vs. +3) is washed out by the decreased rewards the agent receives on future steps. The best response to TFT, therefore, is alwaysC.

2.2 Follower: Q-learning

Q-learning [8] is a family of algorithms for addressing reinforcement-learning problems. A Q-learning agent maintains a data structure, the Q function, that maps each state and action combination to an estimate of the long-term reward received. Symbolically, if s is a state and a is an action, then $Q(s, a)$ is an estimate of the total expected discounted reward the agent will receive if it takes action a in state s, then chooses actions to maximize total expected discounted reward.

A Q-learner updates its Q function after each experience $\langle s, a, s', r\rangle$, where s is the state it was in, a is the action it took, s' is the resulting state, and r is its reward. The update is

$$Q(s, a) := (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a')),$$

where $0 \leq \alpha \leq 1$ is the algorithm's current learning rate.

Variants of Q-learning differ along three dimensions: (1) how they initialize the Q function before beginning learning, (2) how they select actions during learning, and (3) how they change their learning rates over time. In our work, we initialize the Q function to zero, select actions using the ϵ-greedy rule (take the estimated best action with probability $1 - \epsilon$ and a random action otherwise), and use a constant learning rate. We made these decisions to keep the algorithm simple and also to allow it to continue exploring and reacting to changes in the opponent's behavior.

Against a fixed history opponent, Q-learning is an excellent follower. It provably finds an approximation of the optimal strategy.

2.3 Leaders: Tit-for-Tat and alwaysC

The tit-for-tat strategy, the fourth listed in Table 1, is an effective leader in the iterated Prisoner's Dilemma. In particular, the optimal strategy for its opponent is to cooperate, resulting in average rewards of 3 for both players. It is not feasible to reliably score higher against a Q-learner, since a Q-learner will not tolerate acting as a sucker indefinitely.

The alwaysC strategy can also be thought of as a leader in that it adopts a fixed strategy and invites its opponent to optimize against it. In this case, a follower should learn to play alwaysD and exploit the leader. Q-learning will do so, resulting in a consistent reward of 4.

In our experiments, we test strategies by how well they perform against alwaysC (a leader) and Q-learning (a follower). We declare an algorithm a good follower if it obtains average rewards close to 4 against alwaysC and a good leader if it obtains average rewards close to 3 against Q-learning.

3. SOCIAL REWARD SHAPING

In this section, we propose social reward shaping.

3.1 Reward Shaping

Simply put, shaping rewards are imaginary rewards that are presented to a reinforcement-learning algorithm as an addition to the actual environmental reward. Their purpose is to encourage desirable behavior during the learning process.

Potential-based shaping has been shown to provide a principled approach to reward shaping [4]. Here, the idea is that every state s is assigned a potential, $\Phi(s)$. The reward obtained from a transition from state s to state s' is augmented by $\gamma\Phi(s') - \Phi(s)$. Because of the (telescoping) form of this function, the added rewards can speed learning but do not interfere with the final learned behavior—after learning, the best action choice in each state is unchanged.

Although the choice of potential can be arbitrary, assigning $\Phi(s)$ to be the maximum total expected discounted reward from s can make learning optimal behavior nearly immediate. In the context of Q-learning, adding potential-based shaping rewards via potential function $\Phi(s)$ has the same effect as initializing the Q function to $\Phi(s)$ [9]. In our algorithm, we derive a good choice of potential function from an analysis of subgame perfect equilibria, as described next.

3.2 Subgame Perfect Equilibria

A fixed history strategy provides an action choice for all possible histories. In an iterated matrix game, two fixed history strategies constitute a *subgame perfect equilibrium* [5] if each strategy, for each possible history, chooses the action that maximizes its total expected discounted reward with respect to its opponent.

It is instructive to consider the case of TFT. Although TFT vs. TFT results in mutual cooperation, the combination is not a subgame perfect equilibrium. The unique best response to TFT is alwaysC. Any other choice is suboptimal for some history. In particular, from the state CD, TFT chooses **D**, but **C** leads to higher total expected discounted reward (for a sufficiently high discount factor).

Note that two Q-learners with ϵ-greedy action selection cannot converge to anything other than a subgame perfect equilibrium—for every state, the Q-learner will (nearly always) choose the best action according to its estimates. Therefore, it is useful to consider the set of subgame perfect equilibria under ϵ-greedy action selection. That is, we are interested in any strategy that, when it plays against a copy of itself, will choose the same actions as what ϵ-greedy action selection would choose given that the Q functions contain the actual total expected discounted reward for each state–action pair.

We enumerated all sixteen deterministic 1-step history strategies and checked each to determine whether it constitutes a subgame perfect equilibrium if paired with itself. Three are: alwaysD, grim, and Pavlov. Of these, Pavlov has, by far, the highest average reward.

3.3 Social Reward Shaping

Our proposal is to use a mutually beneficial subgame perfect equilibrium, in this case Pavlov, as the basis for a potential-based shaping function. The combination, which we call Q+shaping, is produced by first computing the state values (the potentials) for Pavlov, and initializing the function estimate with this potential-based function. This initialization is equivalent to adding shaping rewards during the learning process [9].

4. RESULTS

	follower (Q-learning)	*leader* (alwaysC)
alwaysC	[0.100, 0.105, 0.110]	[3.000, 3.000, 3.000]
Q-learning	[1.507, 2.781, 2.923]	[3.963, 3.965, 3.967]
TFT	[2.926, 2.930, 2.933]	[3.000, 3.000, 3.000]
Q+shaping	[2.253, 2.786, 2.811]	[3.963, 3.965, 3.967]]

Table 2: Comparison of the ability of four strategies to lead (score nearly 3 against Q-learning) and follow (score nearly 4 against alwaysC) in at least 75% of the runs. Boxed results meet the criterion. Of the four, only Q+shaping is able to both lead and follow consistently.

In our experiments, all Q-learners used ϵ-greedy exploration ($\epsilon = 0.1$) and used a discount factor of $\gamma = 0.9$.

We compared the performance of four algorithms: alwaysC, Q-learning ($\alpha = 0.01$), TFT, and Q+shaping ($\alpha = 0.01$) against two test opponents: Q-learning ($\alpha = 0.02$) and alwaysC. The reason we included alwaysC and TFT in the comparison is that these strategies can arise naturally if shaping rewards are added carelessly to Q-learning in an attempt to encourage more cooperation or retaliation for defection. We ran each algorithm against each opponent for 40,000 rounds each run and averaged over the last 5,000 rounds. Table 2 shows the percentiles obtained in a collection of 1000 runs. These results demonstrate the ability of our algorithm to teach and follow another algorithm consistently.

We found that Q-learning and Q+shaping both follow the fixed opponent, scoring nearly 4 on average against alwaysC. (The observed value is slightly lower than 4 because the ϵ-greedy exploration causes the agents to sometimes choose **C**, against their better judgment.) In contrast, TFT and alwaysC both cooperate when they play against alwaysC, which results in a lower score of 3. On the other hand, TFT is able to lead the Q-learner, reaching nearly the mutual cooperation reward of 3. Of course, alwaysC cooperates with the Q-learner, which correctly learns to exploit it, leaving alwaysC with nearly the sucker reward of 0. The most interesting cases are when learners play against each other. As Table 2 shows, Q+shaping leads Q-learning to consistent convergence to cooperation (average reward around 3 for all percentiles), whereas two "plain" Q-learners stumble around for quite a bit longer, often ending in suboptimal performance (average reward closer to 1.5 at the 25th percentile). To convey a sense of the behavior of the two different strategies, Figure 2 plots a learning curve with the performance of Q-learning and Q+shaping both playing against Q-learning. These curves are somewhat typical.

5. CONCLUSION AND FUTURE WORK

We have shown that providing shaping rewards derived from a mutually beneficial subgame perfect equilibrium yields a promising multiagent-learning algorithm. In particular, these Q+shaping agents effectively lead a learning opponent to adopt mutually beneficial behavior. However, against a fixed history agent, the shaping bias is overcome and the algorithm finds a reward-maximizing behavior.

One concern with the current algorithm is that its success appears to be sensitive to the selected learning rate. If the opponent's learning rate is too low, Q+shaping will end up acting as a follower, and the advantage of shaping is lost. We feel this issue is an unavoidable consequence of simultaneously acting as a leader and a follower, but we hope to increase the robustness of Q+shaping without greatly increasing its complexity.

These results show how Q+shaping is able to efficiently lead a learning algorithm consistently. This characteristic of being able to guide the early experience of a learning process has many benefits. For example, a teacher can help guide convergence to a specific beneficial equilibrium point in games with asymmetric equilibria like chicken or Bach & Stravinsky. In fact, we have performed other experiments that suggest social reward shaping is able to accomplish this goal. Note that on the test case reported here (PD), we found that two plain Q-learners are able to converge to mutual cooperation, although they do so less reliably.

Future work will examine other iterated matrix games that require longer fixed history strategies. For example, Prisoner's Dilemma with a temptation reward of 5 requires 2 steps of history for a mutually beneficial subgame perfect equilibrium. This extension should be relatively simple, since polynomial-time algorithms for finding subgame perfect equilibria exist [3]. We will consider extending our approach to more general stochastic games, which appears more challenging.

6. REFERENCES

[1] Y.-H. Chang and L. P. Kaelbling. Playing is believing: The role of beliefs in multi-agent learning. In *Advances in Neural Information Processing Systems 14*, 2002.

[2] M. L. Littman and P. Stone. Implicit negotiation in repeated games. In *Eighth International Workshop on Agent Theories, Architectures, and Languages (ATAL-2001)*, pages 393–404, 2001.

[3] M. L. Littman and P. Stone. A polynomial-time Nash equilibrium algorithm for repeated games. *Decision Support Systems*, 39(1):55–66, 2005.

[4] A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 278–287, 1999.

[5] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.

[6] M. L. Puterman. *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, 1994.

[7] Y. Shoham, R. Powers, and T. Grenager. If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 2007. Special issue on the foundations of research in multi-agent learning.

[8] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[9] E. Wiewiora. Potential-based shaping and Q-value initialization are equivalent. *J. Artif. Intell. Res. (JAIR)*, 19:205–208, 2003.