Implicit Imitation Learning with Suboptimal Mentors

Robert Timpe
Advisor: Rebecca Fiebrink

Submitted to: Princeton University
Department of Computer Science

April 30, 2012

This Thesis represents my own work in accordance with University Regulations

Robert Timpe

Table of Contents

Abstract

Implicit imitation learning presents an attractive method for speeding up the process of learning an optimal policy in a reinforcement learning situation. Price and Boutilier [Price and Boutilier, 1999] have explored this problem and shown that given an optimal mentor, an imitation agent can converge to an optimal policy more quickly than a non-imitation agent. However, their work requires the assumption of a mentor which always follows an optimal policy. In this work, we show how the agent's behavior can be impacted when the mentor's policy is suboptimal or when the mentor is intentionally adversarial. In particular, certain mentor behaviors can cause an imitation agent to converge to an optimal policy more slowly than a non-imitation agent, and can even cause the imitation agent to converge to a suboptimal policy, depending on the exploration function. Crucially, this behavior occurs in MDPs where a non-imitation agent is capable of discovering an optimal policy via exploration. We also briefly discuss how this behavior might be avoided.

1 Introduction

Reinforcement learning is an attractive approach to machine learning because of the flexibility that it offers. In reinforcement learning situations, the environment is modeled as a Markov decision process (or MDP) consisting of states and actions which cause state transitions. The agent starts in a particular state and is given rewards (based on a reward function) after each state transition. In this context, a policy is a mapping from states to actions that tells the agent what to do in any given state. The goal of the agent is then to learn an optimal policy which maximizes the expected value of the agent's cumulative, discounted rewards. In a reinforcement learning scenario, the reward function and/or transition functions are generally unknown and must be learned by the agent. This means that the agent must do a fair amount of exploration of its environment in order to learn these functions. On the other hand, the agent is always looking to maximize its reward, and any exploration is potentially dangerous because of the possibility of the agent ending up in a state with a very low reward. This extensively studied [Vermorel and Mohri, 2005] tradeoff between exploitation and exploration is a difficult problem but is crucial to the field of reinforcement learning.

One way in which convergence to an optimal policy can be sped up is via imitation learning. In imitation learning the agent is as before, but now there is another agent following a policy of its own. This second agent is generally referred to as the mentor and is almost universally assumed to be following an optimal policy. If the agent can in some way observe the mentor executing its policy, then it can very quickly converge to an optimal policy of its own without having to rely on extensive exploration of the state space.

Implicit imitation learning is a form of imitation learning which requires minimal cooperation from the mentor [Price and Boutilier, 1999]. In this formulation, the agent has no access to the mentor's internal thought process (i.e. its policy). Instead, it observes the mentor's state transitions and uses this information to augment its own learning. This requires the assumption that the mentor and agent have the same (or at least similar) reward functions and state spaces. As long as these conditions are met, the agent can effectively learn from the mentor.

However, one assumption that is made throughout this process is that the mentor is optimal. And while this may be a valid assumption in some cases, it is not difficult to imagine cases where the mentor only knows a suboptimal policy or is deliberately misleading. For example, in robotics (a common application of reinforcement learning), a robot may try to imitate another robot which does not have access to an optimal policy, or even has an incentive to give other agents bad information. One solution would be to simply avoid using imitation agents in such situations, but then we miss out on the possibility of a much quicker learning process. It would be much better to design imitation agents which are resistant to suboptimal mentors.

Our main purpose is to show that suboptimal and adversarial mentors are indeed a problem that needs to be addressed. In [Price and Boutilier, 2003b] Price and Boutilier briefly discuss this problem. They describe the basic way in which a mentor may bias an agent towards a suboptimal policy. But they also conclude that in situations where the use of imitation learning is justified, a non-imitation agent is unlikely to perform better than an imitation agent. Our goal is to show examples of simple MDPs where the mentor biases an imitation agent towards a suboptimal policy, but a

non-imitation agent is still able to discover an optimal policy. These MDPs will be very simple and thus not meet the condition of being sufficiently complicated to justify implicit imitation learning. This simplicity will help to isolate the essential features that make this undesirable behavior possible in an attempt to demonstrate how more complicated MDPs (where implicit imitation learning is justified) can facilitate the same behavior.

The mentor cannot negatively influence the agent's policy directly since the agent will ignore mentor state values worse than its own. But as Price and Boutilier discuss in [Price and Boutilier, 2003b], a suboptimal mentor may bias an imitation agent towards suboptimal policies. Namely, by exploring certain parts of the state space while ignoring others, the mentor can bias the agent's exploration. There are a number of variations on this basic theme, but the most simple can cause the agent to converge to a suboptimal policy in which the agent finds a locally optimal policy but is unable to discover the globally optimal policy. In contrast, a non-imitation agent, which is not subject to this bias, may still find a globally optimal policy.

2 Background

The model used in implicit imitation learning includes two agents—the observer and the mentor. Although the model can be extended to settings with even more agents, we focus only on the two agent case. We also assume that the agents are non-interacting—that is the actions and position of one agent have no impact on the other. Therefore, the MDPs of the mentor and agent can be viewed separately, although they must have the same or similar structures, as we will see.

In general, an MDP is a tuple $M = (S, A, R, Pr, \gamma)$. S is the (finite) state space and A is the set of actions. We assume for simplicity that every action is available in each state. $\Pr(t \mid s, a)$ is the probability that taking action a while in state s will result in a transition to state t. Note that we may have $s = t$. R is the reward function where $R(s)$ gives the reward for state s. Finally, γ is the discount factor which is used in calculating future rewards. We follow [Price and Boutilier, 1999] in choosing a discount factor near 1 since we want policies that take into account long-term strategies. For our experiments, γ is always 0.95 unless otherwise noted.

In their original work on imitation learning, Price and Boutilier also make assumptions about the state and action spaces of the mentor and observer [Price and Boutilier, 2003b]. They assume that $S_m = S_o$ where S_m is the mentor's state space and S_o is the observer's state space. They also assume that A_m is a subset of A_o. However, we will assume that the mentor and the agent have the exact same actions available. Additionally, Price and Boutilier make no assumptions about the relationship between the two reward functions, but we will only look at examples where the mentor and the observer have the same reward functions. Finally, we assume that the observer and the mentor have the same transition probabilities. This in essence means that the mentor and observer share the same MDP.

As in [Price and Boutilier, 2003b], we will assume that the transition model is unknown, but the reward function is known. As Price and Boutilier point out, this may seem like a contrived set-up, but it is fairly common in practice. In many cases it is relatively easy to describe a reward model, but much more difficult to describe the dynamics of the entire state space. We also assume that the observer can perfectly

observe the mentor's state at every point. That is, the observer has access to a number of tuples (*s,t*) where a each tuple represents a state transition made by the mentor from state *s* to state *t*. This may be a strong assumption (especially in a robotics setting), but Price and Boutilier [Price and Boutilier, 2003b] discuss ways to relax it. But for our purposes we will simply assume that the mentor is perfectly observable by the agent.

2.1 Implicit Imitation

Price and Boutilier [Price and Boutilier, 2003b] discuss both model-based and model-free approaches to the problem of implicit imitation learning. We will only discuss model-based approaches here. We are therefore interested in finding the value of each state. The value of a state is the expected sum of discounted cumulative rewards that will be experienced if the agent follows an optimal policy (although we can also consider the value of a state if the agent follows any arbitrary policy). Therefore, the value of state *s* is defined as

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

There are well known algorithms for solving this system of equations when the reward function and dynamics are known - we use value iteration [Bellman 1957] for simplicity. Since we do not know the dynamics directly we must estimate them based on past experience. As in [Price and Boutilier, 2003b], we have access to a number of experience tuples of the form (*s, t, a*), which represents taking action *a* from state *s* and ending up in state *t*. We see a new one of these tuples after every state transition, and we keep track of them by counting how many times each tuple is seen. Then we can estimate

$$(2)\ \Pr_o(t \mid s,a) = \frac{number\,of\,tuples\,(s,t,a)}{\sum_{t \in S} number\,of\,tuples\,(s,t,a)}$$

However, if we just solve the normal Bellman equations, we do not take into account the influence from the mentor. To account for this, Price and Boutilier [Price and Boutilier, 2003b] provide the following "augmented" Bellman backup

$$(3)\ V(s) = R(s) + \gamma \max\left\{ \max_{a \in A_o}\left\{ \sum_{t \in S} \Pr_o(t \mid s,a)V(t)\right\}, \sum_{t \in S} \Pr_m(t \mid s)V(t)\right\}$$

We can estimate $\Pr_m(t \mid s)$ from our observations of the mentor as

$$(4)\ \Pr_m(t \mid s) = \frac{number\,of\,tuples\,(s,t)}{\sum_{t \in S} number\,of\,tuples\,(s,t)}$$

Therefore incorporating imitation learning is fairly simple. We just estimate the mentor and observer dynamics based on counts of observed transitions and perform augmented Bellman backups to calculate values.

The form of imitation learning used here is the simplest version. Price and Boutilier also describe several additions to the basic algorithm [Price and Boutilier, 2003b], but we do not use them here. Some are not relevant to this work (for example, they describe several ways of relaxing various assumptions and handling mentors

which do not occupy the same MDP as the observer). They also describe a method for keeping track of model confidence. The idea is that there may be states which the mentor visits infrequently, in which case the observer may have a bad estimate of the mentor's transition probability. Price and Boutilier propose keeping an estimate of model confidence to deal with these situations. If the model confidence is low, then the agent can ignore the observations of the mentor and do regular Bellman backups.

Since the mentor and observer occupy the same MDP, the only potentially relevant extension would be model confidence testing. However, model confidence testing is not particularly relevant to our experiments. First of all, when the mentor follows a greedy stationary policy, model testing is less important. When actions are deterministic, the mentor will always follow a single path and model confidence will be irrelevant because once the agent observes the mentor once, it will know the mentor's state transitions exactly. Stochastic actions will make things more complicated, but since the mentor and agent occupy the same MDP, their uncertainties will be the same probabilistically. Moreover, we can drive the mentor confidence arbitrarily high by simply having the mentor repeat its policy many times before letting the observer begin. In some practical settings, this may even be a better choice. After all, ignoring the mentor initially due to low confidence would defeat the purpose of imitation learning (i.e. speeding up reinforcement learning in the early stages). For these reasons and other reasons to be discussed below, model confidence is not included in our experiments.

3 Experimental Setup

To test the influence of bad mentors on an imitation agent, a grid world MDP was created. An example configuration of this world is shown below.

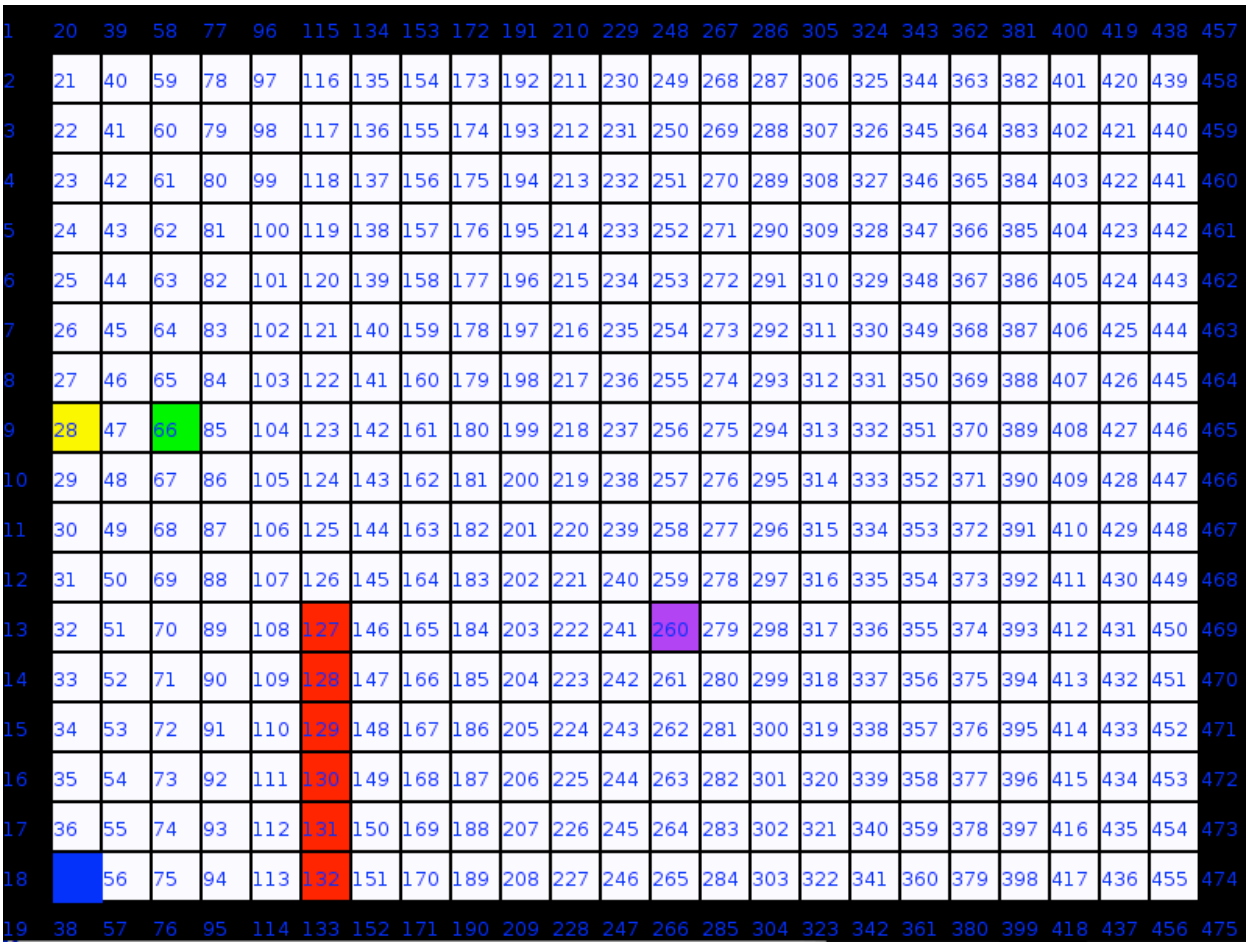

Figure 1 - Sample MDP with states labeled and lines drawn to show grid locations

In **Figure 1**, the states are numbered and extra lines are drawn to show grid locations. The blue square represents the current agent position. The agent can move up, down, left or right as long as it is in a white square. For example, the agent can go up from state 52 to reach state 51, left to reach 33, and so on. Black squares are obstacles and cannot be occupied - trying to move into a black square will have no result (i.e. the agent will transition to its current position). White squares all have a

reward of 0. Purple squares cause transitions to other parts of the state space. That is, entering a purple square will cause a transition to some other square. This allows the agent to “teleport” to different parts of the state space. When purple squares appear in experiments, we will indicate which square they cause a transition to. All other squares are terminal states and entering them causes the agent to return to the starting position (which by default is the one shown above). Red squares have a reward of -1. Yellow squares have a reward of +1 and green squares have a reward of +2. The color information is summarized in **Table 1**.

Color	Type	Reward
White	Ordinary state	0
Black	Obstacle - cannot be occupied	0
Red	Terminal	-1
Yellow	Terminal	+1
Green	Terminal	+2
Purple	Teleporting	0

Table 1 - Summary of different state types and reward values

By default, all transitions are deterministic. However in some experiments we will have stochastic transitions. In these cases, the agent will perform its intended action with some probability *p*. With probability 1 - *p* the agent will move perpendicular to its intended action. For example, the agent might try to go up and succeed with probability 0.8, but fail with probability 0.2. In case of failure it would move either left or right with equal probability.

3.1 Agent Design

Our agent implements implicit imitation learning by performing the augmented Bellman backup described in **Equation 3**. It keeps counts of its own state transitions as well as mentor state transitions which are then used to estimate the various transition probabilities. After entering a terminal state and resetting, the agent performs value iteration based on these estimated probabilities to update its value estimates.

We use epsilon-greedy exploration as our exploration policy. In particular we use the epsilon-decreasing strategy described in [Vermorel and Mohri, 2005]. In this strategy, the agent explores at each simulation step *t* with probability

$$(5)\ \epsilon_t = \min\left\{\frac{\epsilon_0}{t}, 1\right\}$$

where ϵ_0 is a constant. In our experiments we always use $\epsilon_0 = 75$. We chose this value because it allowed for an appropriate amount of exploration for the size and complexity of MDPs used in our experiments. A non-imitation agent with $\epsilon_0 = 75$ will explore almost all of the state space (excluding areas which are very hard to reach). As the experiments below will show, this amount of exploration is more than enough for a non-imitation agent to converge to an optimal policy. Increasing this value results in more exploration than necessary. While doing so would in some cases allow the imitation agent to perform better, simply increasing the amount of exploration is not a true solution to the problem of dealing with a suboptimal mentor (as will be discussed later) and does not scale well to larger state spaces.

When the agent chooses to explore it chooses an action uniformly at random. This means that for the first 75 steps of simulation, the agent will perform pure exploration, after step 150 it will explore with probability less than 0.5, and so on. Also,

if the agent can only take actions which result in transitions to states with value = 0 (i.e. states that it hasn't yet explored) it will also choose an action uniformly at random.

We used epsilon-greedy exploration because it is both simple to implement and commonly used in practice. Additionally, epsilon-greedy exploration is used by Price and Boutilier [Price and Boutilier, 1999] in their original experiments on implicit imitation learning. Despite its simplicity, epsilon-greedy exploration can produce optimal or near optimal results in some cases and is often used in favor of more complex but theoretically justifiable techniques [Vermorel and Mohri, 2005]. And while epsilon-greedy exploration is particularly vulnerable to certain types of suboptimal mentor behavior, some more complex exploration strategies are similarly vulnerable, if not more so (as will be discussed below).

In some experiments we will give the agent some prior knowledge of the transition probabilities. We follow [Price and Boutilier, 2003b] and give this prior knowledge (similar to a prior distribution in Bayesian learning) by initializing the counts of observed transitions to some value greater than 0. For example, to give the agent a prior which indicates deterministic actions, we could initialize the observations of transition (1, 2, "up") to 1. This indicates that the agent has seen the transition from state 1 to 2 after taking action "up" once, which makes sense if state 2 is above state 1. We then repeat for all possible transitions allowed by the MDP. By initializing to a small value such as 1, we allow the agent to quickly "discard" the prior if it turns out to be inaccurate.

During the various experiments, we will also talk frequently about the values computed for various states. To aid this discussion, we will often include images that

demonstrate these values, such as the one in **Figure 2**.

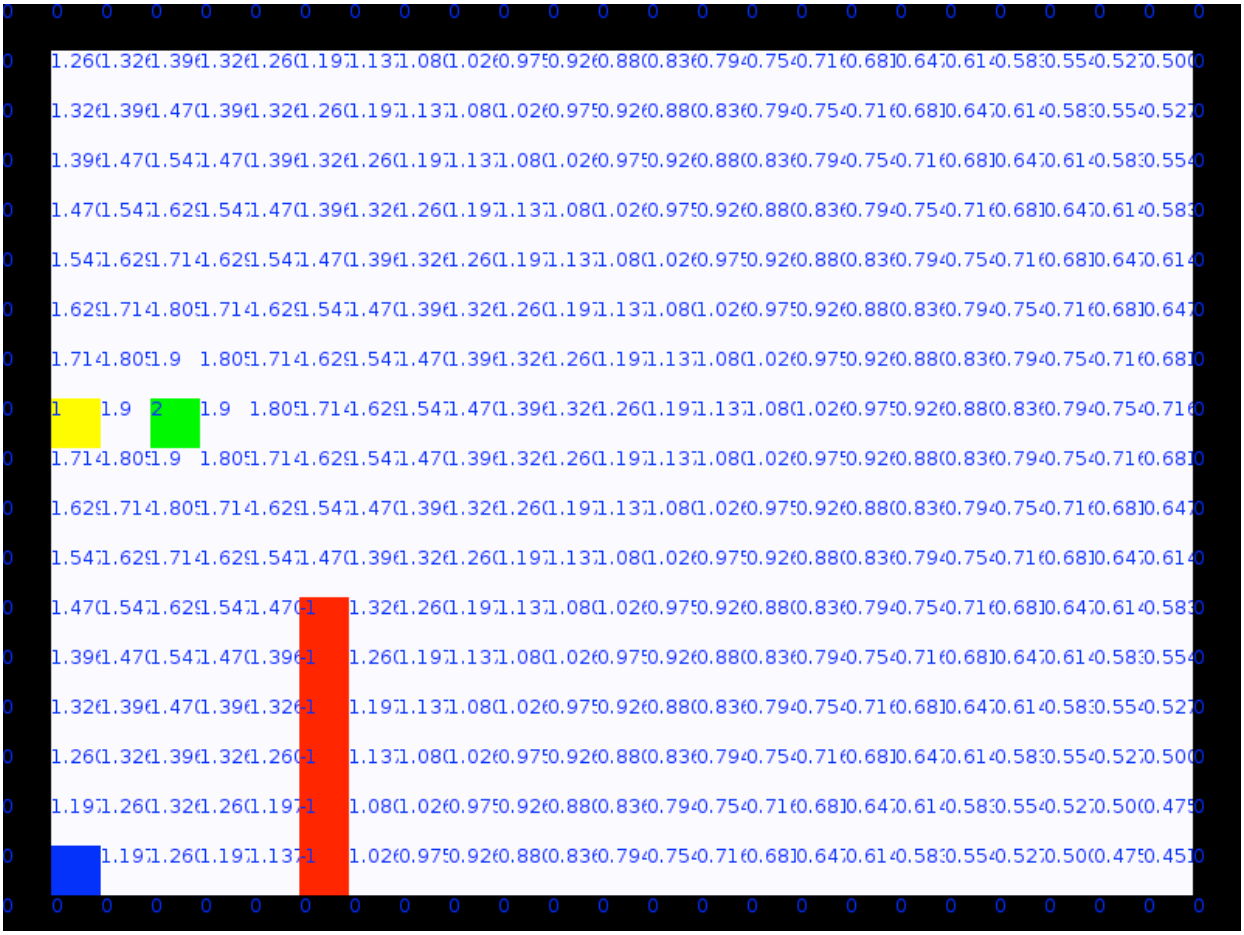

Figure 2 - A sample MDP with computed values shown in blue text. Red states have reward of -1, yellow states have reward +1 and green states have reward +2. The blue square is the agent position.

The number in each square represents the value of that state. To form this example, the agent was given a prior that perfectly describes the state space and computed the values based on that information. So the value of the green state is 1, the neighboring states have value 1.9 (because of the discount factor of 0.95) and so on. In this case all of the values are correct, but the values will often be incorrect/incomplete depending on the agent's current knowledge of the MDP.

4 Types of Suboptimal Mentor Behavior

As Price and Boutilier briefly discuss [Price and Boutilier, 2003b], it is difficult for the mentor to negatively influence the agent. This stems from the outer max function in the augmented Bellman backup in **Equation 3** above. This max function prevents the mentor from negatively influencing the agent's concept of value at any state. That is, if the mentor follows a policy that would assign lower values to certain states, the agent simply ignores the mentor's input for those states.

At first glance, this method appears to be the optimal way to include information from the mentor. By only including mentor data in this way, the agent relies primarily on its own experience except in states where it hasn't explored (or at least hasn't discovered an optimal policy). Mentor data can then be used to help focus the agent's policy towards more rewarding sections of the state space if the agent hasn't already discovered these areas on its own. And indeed, this behavior prevents certain types of mentor behavior from negatively influencing the agent. For example, if mentor observations were incorporated in some other way, it might be possible for the mentor to execute a very bad policy in an otherwise highly rewarding segment of the state space and thus cause the agent to focus elsewhere. Taking the maximum of the agent's own experiences and mentor data prevents this type of behavior.

However, there are still ways in which the mentor can negatively influence the agent. Such situations tend to arise in state spaces with different types of positive rewards. For example, in a state space with a terminal state having reward +1 and another terminal state having reward +2, the mentor can bias the agent towards a policy

that always ends in the +1 state. This is accomplished by the mentor executing a policy which travels near the +2 state but ends in the +1 state. By doing so, the mentor biases the agent's policy towards the +1 state. If the agent has no prior knowledge about the state space, then it will be totally unaware of the +2 state and will only know of the +1 state because of the mentor. Unless it discovers the +2 state in its own exploration, it will converge to a policy that always ends in the +1 state and is thus suboptimal. Even if the agent happens to be exploring near the +2 state, its value estimates (which are based on the policy of the mentor) will bias it away from the +2 state and back towards the +1 state. This is the key aspect of the mentor behavior that causes suboptimal behavior by the agent. Because the information from the mentor is better than the agent's own information (since the agent has no prior knowledge, its initial policy is to wander randomly) it will be accepted. And if the mentor visits states near the +2 state before returning to the +1 state, the agent's policy will make it return to the +1 state rather than explore randomly, as it would if it ignored the mentor. Because of this, a non-imitation agent will perform better in such a setting because it will not be biased by the mentor. Its exploration will be more random, so it will be more likely to discover the +2 state. The exact details depend greatly on the layout of the state space, and such an example is shown in the first experiment.

Note that the above setup only works if the agent has no prior knowledge of the state space. In practice, reinforcement learning agents are often given some sort of prior knowledge about the state space to help improve the learning process. In the limiting case of a perfect prior knowledge, it is not possible for the mentor to influence the agent at all. In fact, perfect prior knowledge renders reinforcement learning

irrelevant. Since the agent already knows the reward function (by assumption) and the transition probabilities (based on its prior knowledge) it can immediately compute an optimal policy. However, perfect prior knowledge is rarely available, so we explore several ways in which a suboptimal mentor can negatively influence an agent with imperfect prior knowledge.

There are two types of imperfect prior distributions we will explore. The first is a prior which simply does not describe certain types of states. The example we will use is the "teleporting" state described above, but many other types are possible. Such states may be common in practical applications with any type of nonlinearity in the state space. Our basic state space consisting of a grid with transitions to neighboring cells is extremely linear, while many practical state spaces may not be. The second type of prior we will discuss is one which does not accurately reflect transition probabilities. In this case, the agent may have a prior which indicates that state transitions are highly stochastic, whereas in reality the probability of an action failing is fairly small. In both of these cases, suboptimal mentor behavior can cause the imitation agent to perform poorly compared to a non-imitation agent. Even if the mentor does not cause the imitation agent to converge to a suboptimal policy, it may cause it to converge to an optimal policy more slowly than a non-imitation agent.

5 Experiments

Our experiments can be divided into two sections: experiments where the agent has a prior and experiments where the agent does not have a prior. We start off with experiments with no priors because not having a prior makes it easier for the mentor to

mislead the agent. The first two experiments demonstrate the effects of a suboptimal mentor in deterministic MDPs. We attempted to keep these first MDPs simple to show various features that can cause a suboptimal mentor to negatively influence an imitation agent. In the third experiment the agent still has no prior but actions are now stochastic, so it is possible for the imitation agent to converge to a suboptimal policy even though there is only one positive reward state. The final two experiments feature an agent with a prior. In each case, the prior is somehow incomplete or incorrect which can lead to the mentor negatively influencing the imitation agent. All of the MDPs used in our experiments were specifically designed to demonstrate how learning from a suboptimal mentor can hurt an imitation agent. As a result they may seem contrived or unrealistic. However, there are plenty of more realistic examples where the effects of a suboptimal mentor are less pronounced but still problematic.

5.1 Experiments without Priors

5.1.1 Experiment 1

Our first experiment demonstrates the basic strategy to be followed by the mentor in subsequent experiments. The environment layout is shown in **Figure 4**.

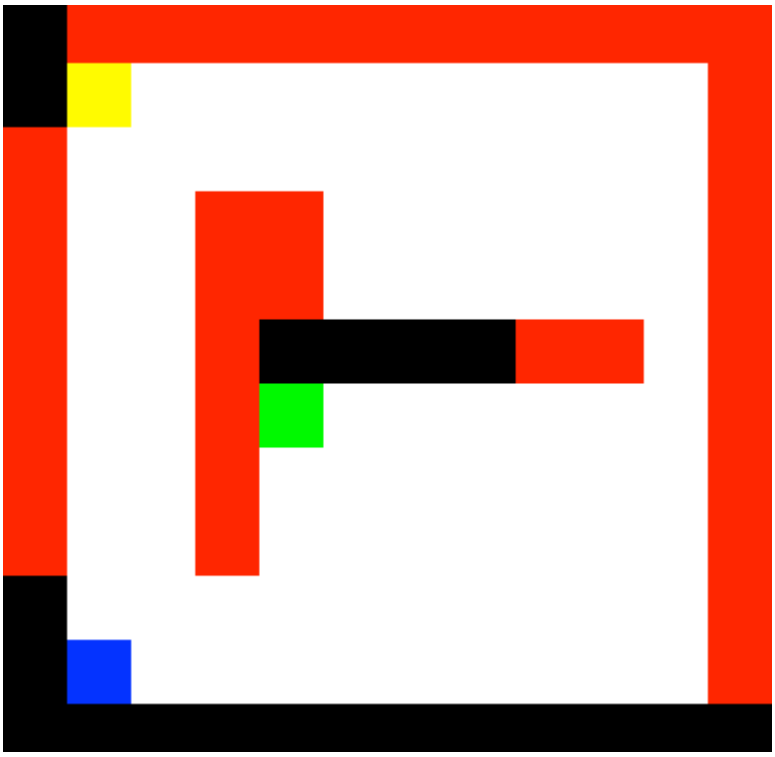

Figure 4 - MDP used in experiment 1. The agent (blue square) starts in the lower left. An optimal policy (discovered by the non-imitation agent) ends in the green +2 state, while a suboptimal policy learned by the imitation agent ends in the yellow +1 state.

In **Figure 4**, the agent is shown in its initial position (see **Table 1** for a description of the different states). In this first experiment, the agent has no prior and all actions are deterministic. **Figure 5** shows the mentor's policy in this environment.

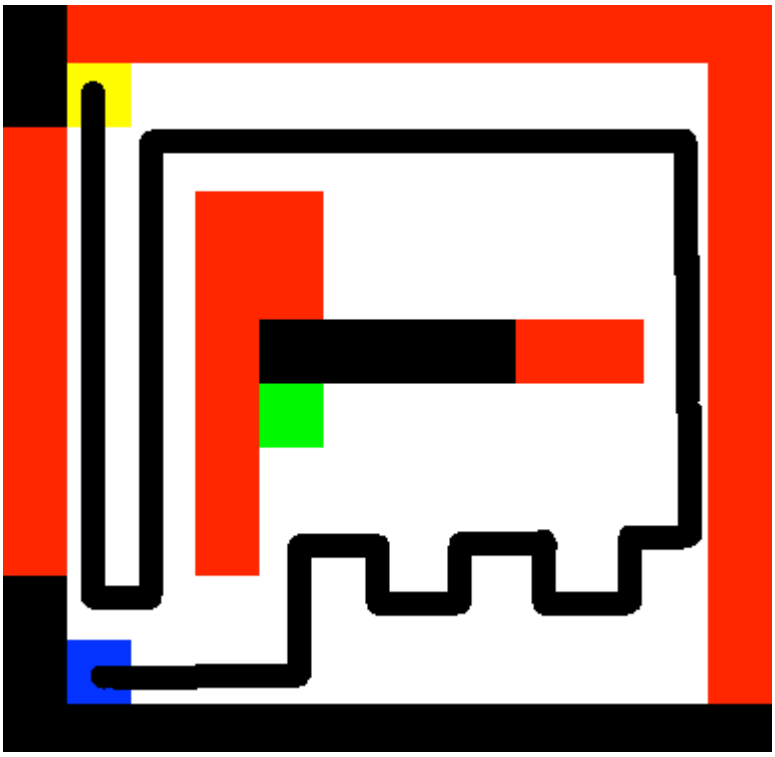

Figure 5 - The black line represents the mentor's policy. The mentor tries to get the imitation agent to converge to a policy ending in the yellow +1 state, rather than the green +2 state.

Not only does this policy end in the yellow +1 state rather than the green +2 state, but it also takes a longer route in order to visit many of the states near the +2 state. **Figure 6** shows a set of example values that the agent would compute based on this input.

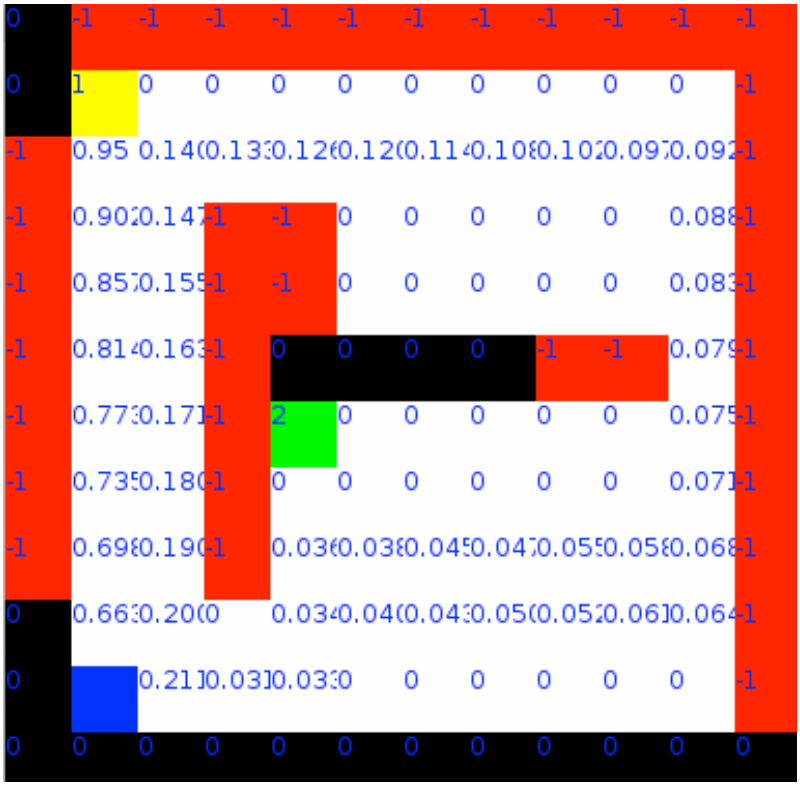

Figure 6 - Sample values computed based on observations of the mentor policy shown in Figure 5. Following the policy suggested by these values will lead the agent towards the yellow +1 state rather than the green +2 state.

The values in **Figure 6** represent the agent's computations before doing any exploration of its own. If the agent were to follow a policy based on these values without doing any exploration of its own, it would never discover the +2 state. Even if the agent does exploration of its own, the mentor's policy will bias it away from the +2 state. In the initial period of pure exploration the agent will frequently transition into a red -1 state. Since these states are terminal, the agent will be reset to the starting position. And since there are many of these states near the agent's starting position, much of the initial period of exploration will be spent in or near these states. After the initial period of pure exploration ends (i.e. after 75 steps of simulation) the agent will be biased towards the +1 state. It will still chose actions uniformly at random when

exploring, but when it is not exploring it will choose actions that bring it closer to the +1 state and away from the +2 state. As the probability of exploration decays over time the agent will be less and less likely to discover the +2 state and will tend to converge to a policy that ends in the +1 state. The superiority of the non-imitation agent's policy is shown in **Figure 7**.

A non-imitation agent, on the other hand, will not be biased by the mentor's policy. During its period of initial exploration it will behave no differently from the imitation agent and will frequently end up in the red -1 states. But unlike the imitation agent it will keep choosing actions uniformly at random after the initial period of pure exploration. This is because until it transitions to either the +1 state or the +2 state, it will assign all non-terminal states a value of 0 and thus chose actions uniformly at random. This means that a non-imitation agent will have an extended period of exploration compared to an imitation agent. Given the layout of this experiment, this also means that the non-imitation agent will be more likely to discover the +2 state. Discovering the +1 state would require the agent to randomly explore a 2x6 grid surrounded by -1 states, whereas finding the +2 state is comparatively easier. Indeed, this is exactly what happened, as shown in **Figure 7**.

The data in **Figure 7** is the sum of cumulative rewards experienced by each agent over the past 200 steps of simulation for the first 1500 steps of simulation, averaged over 5 different runs. Both agents initially experience a period of negative rewards during the initial exploration phase. However, as the simulation progresses the non-imitation agent is able to converge to a more optimal policy than the imitation agent and thus experiences higher rewards. Note that the this behavior is not guaranteed -

the imitation agent might happen to discover the +2 state (and therefore a more optimal policy) or the non-imitation agent might converge to a policy that ends in the +1 state. However, due to the layout of the environment, and more importantly, the behavior of the mentor, such outcomes are unlikely.

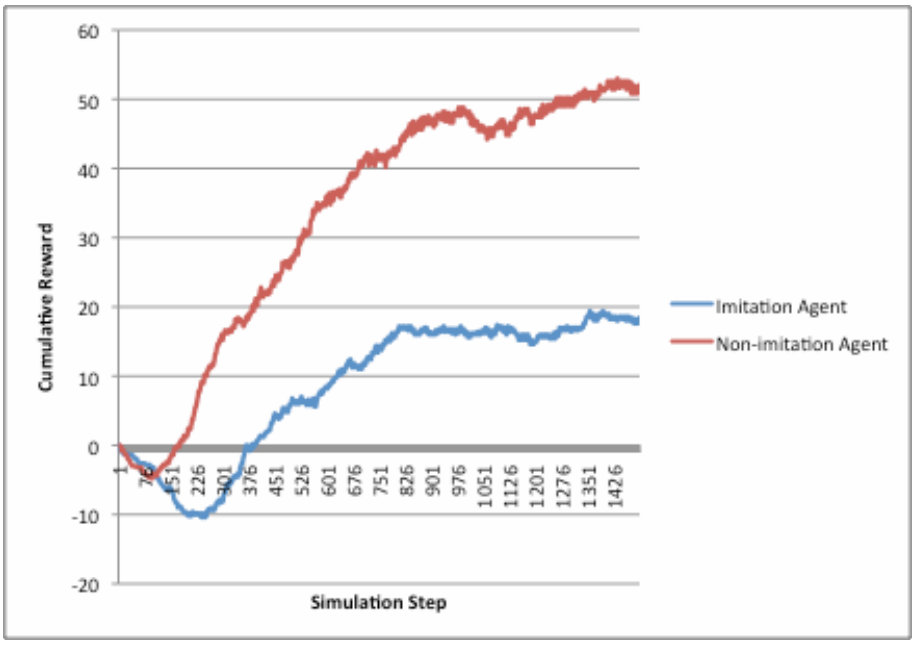

Figure 7 - Cumulative rewards over last 200 steps as a function of step number, averaged over 5 runs.

5.1.2 Experiment 2

The second experiment uses a slightly different layout but with the same parameters. The goal here is to demonstrate a different type of layout in which the same behavior (i.e. an imitation agent which performs worse than an imitation agent) is observed. We also highlight other features of the environment that can contribute to an imitation agent learning a suboptimal policy. Again, actions are deterministic and the agent is not given a prior. The layout is shown in **Figure 8** and the mentor's policy is shown in **Figure 9**.

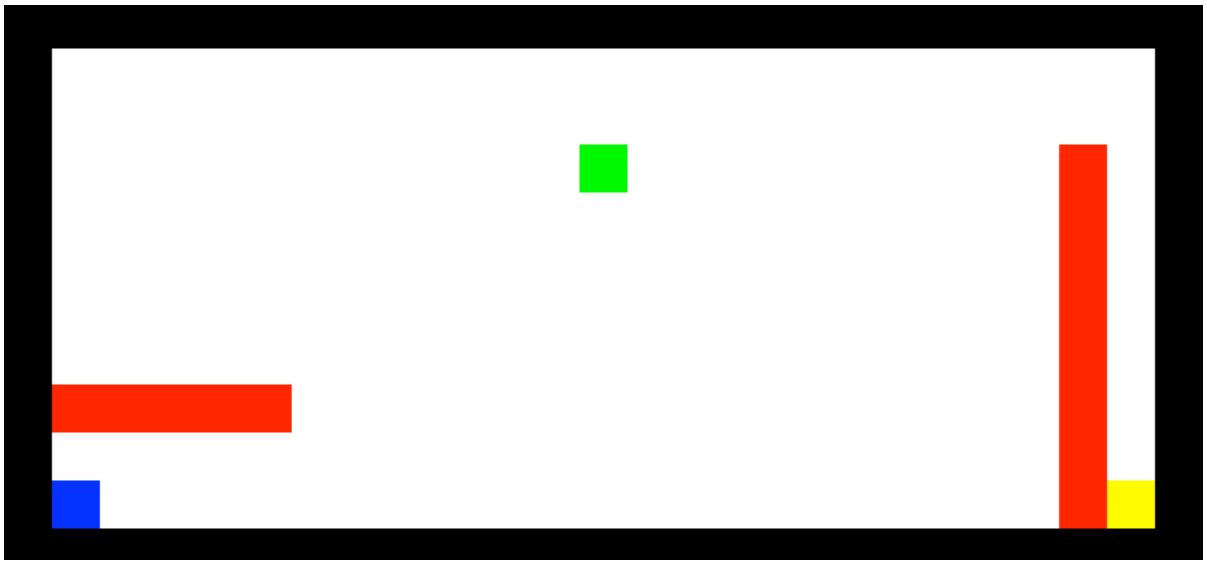

Figure 8 - MDP layout for experiment 2. The optimal policy ends in the green +2 state, but the mentor can cause the imitation agent to converge to a suboptimal policy ending in the yellow +1 state.

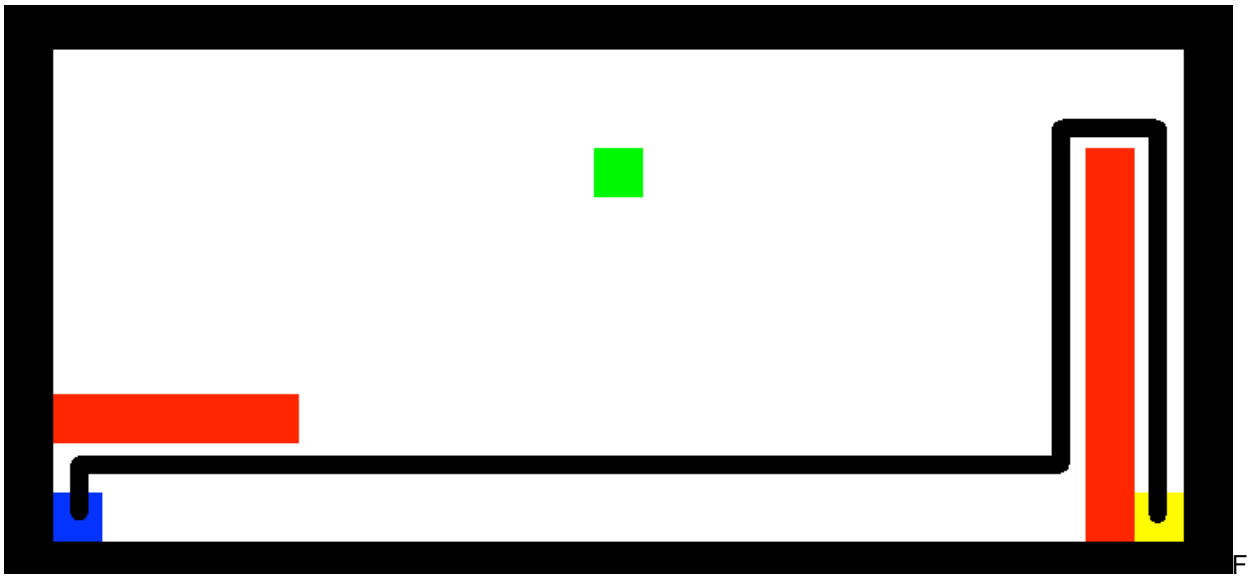

Figure 9 - Mentor policy for experiment 2 shown in black. By visiting the yellow +1 state and not the green +2 state, the mentor biases the imitation agent's policy towards the yellow state.

Again, the goal is for the mentor to cause the agent to converge to a suboptimal policy that ends in the yellow +1 state, rather than the green +2 state. However, in this case the mentor has a simpler policy. Instead of intentionally going near the +2 state only to ultimately end in the +1 state, the mentor simply goes directly to the +1 state.

The imitation agent is again biased towards the +1 state, but in a slightly different way. In this layout, the mentor observations have less of an impact on the imitation agent's policy than the environment layout. During the early stages of exploration, the agent almost always transitions into one of the nearby -1 states. Once the imitation agent stops exploring as much and is able to make it past the first few states, it will mostly be following the mentor's policy and will be unlikely to discover the +2 state. The non-imitation agent, on the other hand, will be more likely to discover the +2 state eventually. It will also spend a lot of time transitioning into -1 states, but once it happens to make its way out of the first few states, it will have a decent chance of finding the +2 state. It may take a long time to do so - it won't make it out into the more open section of the state space until it happens to go right 4 or 5 times without going up once, but once it does it is unlikely to discover the +1 state because of all the nearby -1 states. The average cumulative rewards over 5 runs are shown in **Figure 10**.

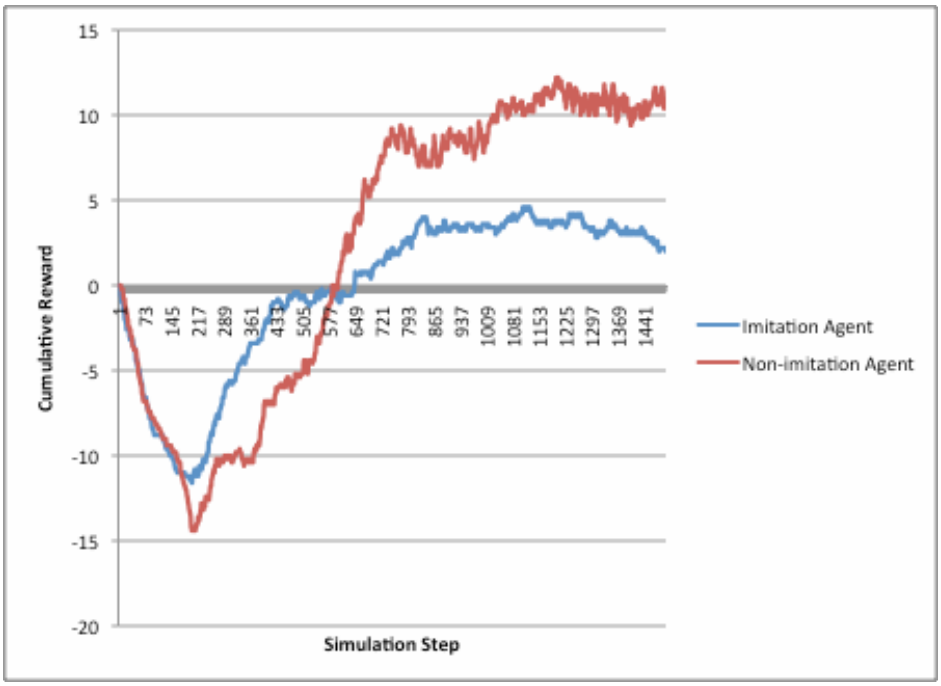

Figure 10 - Cumulative rewards over last 200 steps as a function of step number, averaged over 5 runs.

Again, the non-imitation agent is able to outperform the imitation agent. Initially both agents experience a long streak of negative rewards corresponding to the initial period of blind exploration. And while the imitation agent converges relatively quickly to a suboptimal policy of its own, the non-imitation agent spends more time exploring since until it finds a state with positive reward, it is essentially forced to explore randomly. However, this exploration pays off as it converges to an optimal policy ending in the +2 state. Again, the differences here are probabilistic - the non-imitation agent is not guaranteed to find an optimal policy (of the 5 trials conducted, the non-imitation agent found the +2 state within the first 2000 steps 4 out of 5 times) and the imitation agent is not guaranteed to converge to a suboptimal policy.

These two examples share several important features that allow the non-imitation agent to outperform the imitation agent. First, there are two states with differing

magnitudes of positive rewards. This means that there is a globally optimal policy (which ends in the +2 state) and a locally, but not globally, optimal policy (which ends in the +1 state). And secondly, the state space is arranged in such a way that the mentor is able to bias the imitation agent towards a suboptimal policy but the non-imitation agent is still able to discover an optimal policy. The optimal policy is "easier" to discover because a policy ending in the +2 state does not pass as close to the terminal -1 states as a policy ending in the +1 state. Finally, because the agent has no prior, it is easy for the mentor to improve the agent's value estimations (since the agent initially assigns all non-terminal states a value of 0). Later examples will demonstrate that this behavior is still possible as long as the agent does not have a perfect prior.

5.1.3 Experiment 3

Our third experiment shows how a suboptimal mentor can negatively influence an imitation agent in a stochastic setting. Actions succeed with probability 0.8 and fail with probability 0.2, as described above. Again, the agent is not given a prior. The environment is shown in **Figure 11** and the mentor's policy is shown in **Figure 12**.

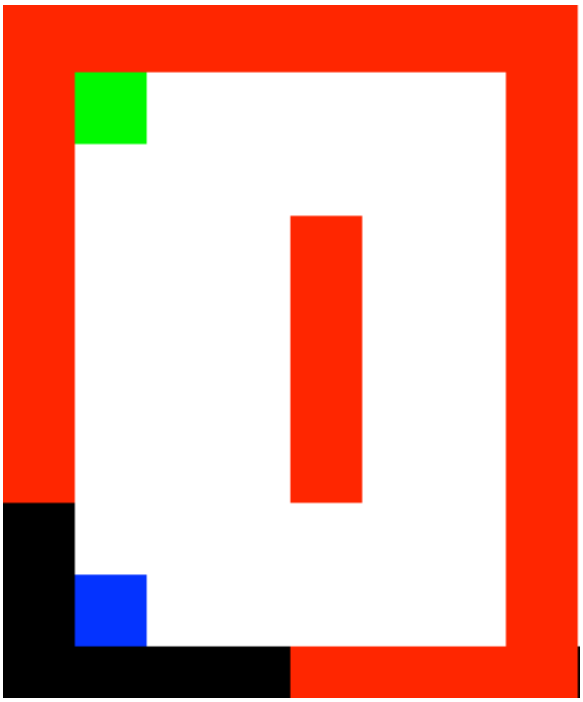

Figure 11 - Layout for experiment 3 with agent in start position. The agent can reach the green +2 state either by going straight up, or taking the more dangerous, roundabout path to the right.

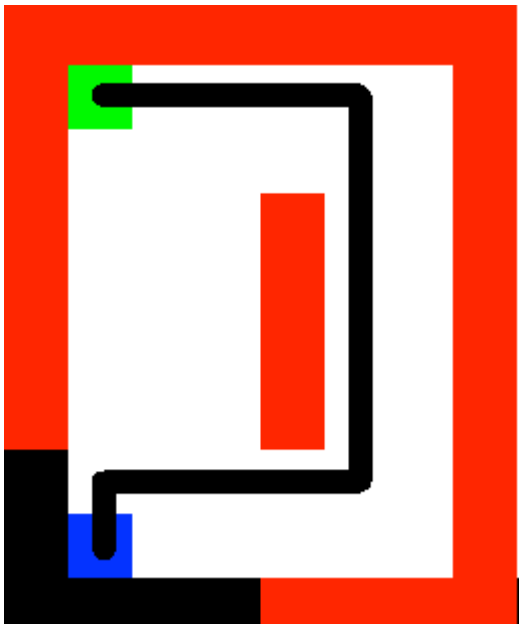

Figure 12 - Mentor policy for experiment 3. The mentor's policy is suboptimal because it takes the longer and more dangerous path to the green +2 state. Doing so biases the imitation agent towards a similarly suboptimal policy.

By taking the longer route, the mentor biases the imitation agent towards this route. And since actions are stochastic, this route is more dangerous (i.e. the agent has a higher probability of transitioning into a -1 state) as well. The numerous -1 states tend to prevent the imitation agent from discovering the shorter route during its own exploration. As in previous examples, the mentor's policy will bias the agent towards a suboptimal policy. The imitation agent may still discover the shorter path, but this is unlikely because of the large number of -1 states. A non-imitation agent will also spend a lot of time running into -1 states, but it will eventually discover the +2 state, most likely via the shorter path. The cumulative rewards averaged over 5 experiments are shown in **Figure 13**. Note that because of the stochastic nature of the MDP, the data is somewhat noisier than in the previous experiments.

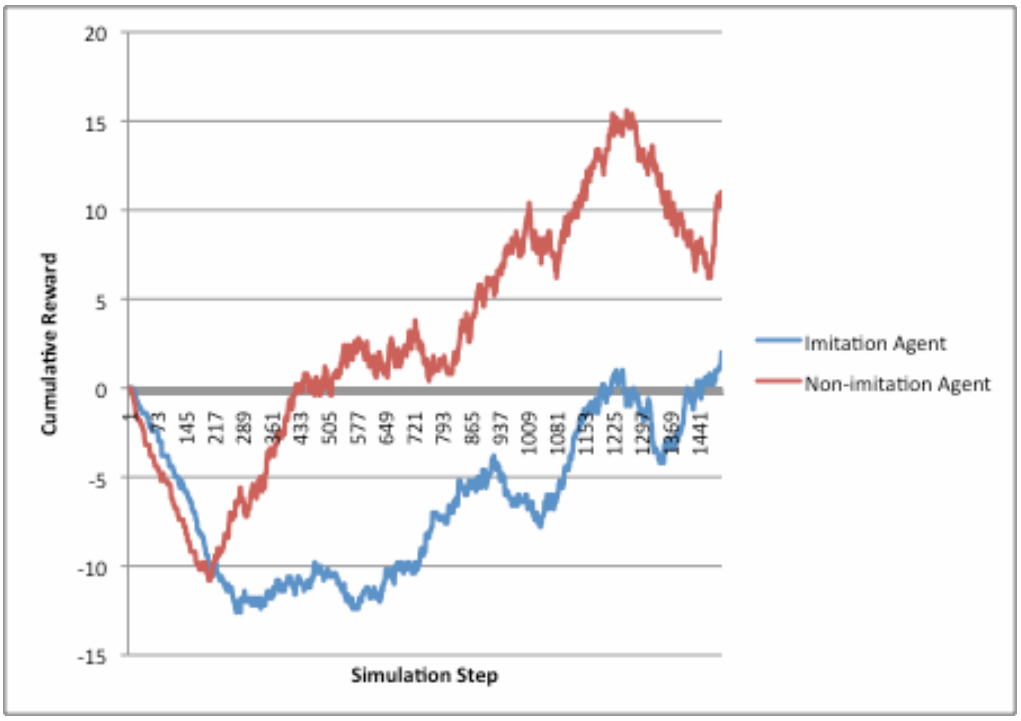

Figure 13 - Cumulative rewards over last 200 steps as a function of step number, averaged over 5 runs.

Again, the non-imitation agent is able to outperform the imitation agent. This result is interesting because unlike the previous experiments, there is only one positive reward state. This shows that it is possible for a suboptimal mentor to negatively influence an imitation agent in MDPs with different "paths" to the goal state, not just MDPs with multiple positive reward states of different magnitudes.

5.2 Equivalence of a Suboptimal Mentor and a Bad Prior

All of the above examples work because the agent has no prior knowledge of the state transition probabilities. If the agent has a good prior, then the mentor will have a much harder time confusing the agent. In particular, the simple grid world used above could be accurately described by a prior which indicates that the agent can go up, down, left or right from each state. When such a prior is used in either of the first two experiments listed above, the mentor will have no effect on the agent. This is because the prior perfectly describes the dynamics of the state space, so the agent can immediately compute an optimal policy. Such a prior would also have a similar effect in the third experiment, although the policy computed might not be optimal if the prior failed to take into account the stochastic nature of the state space.

Given the importance of a prior, it is interesting to consider the extent to which a suboptimal mentor is equivalent to a bad prior. For example, both of the first two examples could be simulated by removing the mentor and replacing it with a misleading prior. The mentor observations are just recorded as a set of counts of observed transitions, and in this framework we implement priors by initializing the agent's own transition counts to values other than 0. And since actions are deterministic and the

mentor follows a stationary policy, the mentor will always undergo the same transitions. Therefore, if the agent is given a prior with counts corresponding to the mentor's policy, it will experience the same behavior shown above. The agent will have knowledge of the state transitions that the mentor would have taken and will have the same value calculations as it would if it were observing a mentor.

This works in the deterministic case because only one transition is possible per state and per action. In a stochastic MDP, on the other hand, a misleading mentor and a misleading prior will have slightly different effects. Because multiple transitions per state and per action may be possible, the mentor's influence may decrease in the long term. The prior will still have some effect early on when the agent has experienced relatively few state transitions, but its influence will decrease over time. This is of course different from the effects of observing a mentor where the observations are stored separately, but since mentor observations are most important early on anyway, it is an interesting comparison.

5.3 Experiments with Priors

5.3.1 Experiment 4

As mentioned previously, all of the above examples rely on the agent not having any sort of prior over state transition probabilities. However, it is also possible for a mentor to negatively influence an agent which has a prior if that prior is not completely accurate. We first focus on priors which fail to account for different types of states. In the following experiment, the mentor is given a prior which correctly describes white states (i.e. it indicates that the agent can make up, down, left or right transitions) but

ignores all other states. In particular, we include “teleporting” states as described previously. Entering one of these states causes the agent to transition to some other, non-neighboring state. The layout is shown in **Figure 14**. Actions are deterministic.

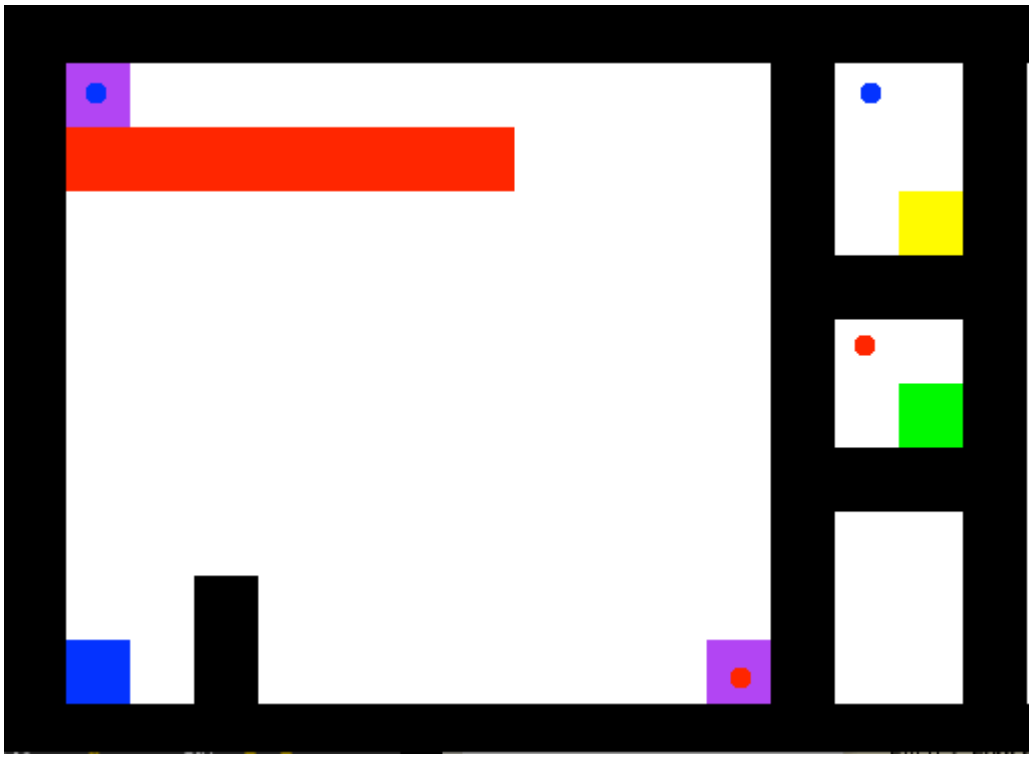

Figure 14 - Layout for experiment 4. The blue and red dots show where the purple states transition to. As in previous examples, there is the optimal policy ending in the green +2 state and a suboptimal policy ending in the yellow +1 state.

In **Figure 14** the purple squares represent teleporting states, and the red and blue dots show the transitions that occur when an agent enters the respective purple state. So entering the upper left purple state causes the agent to transition to the blue dot near the yellow +1 state and entering the lower right purple square causes the agent to transition to the red dot near the green +2 state. Since the prior does not give any information about these states (i.e. the counts are initialized to 0) the agent does not

initially know about those transitions. The mentor policy for this experiment is shown in **Figure 15**.

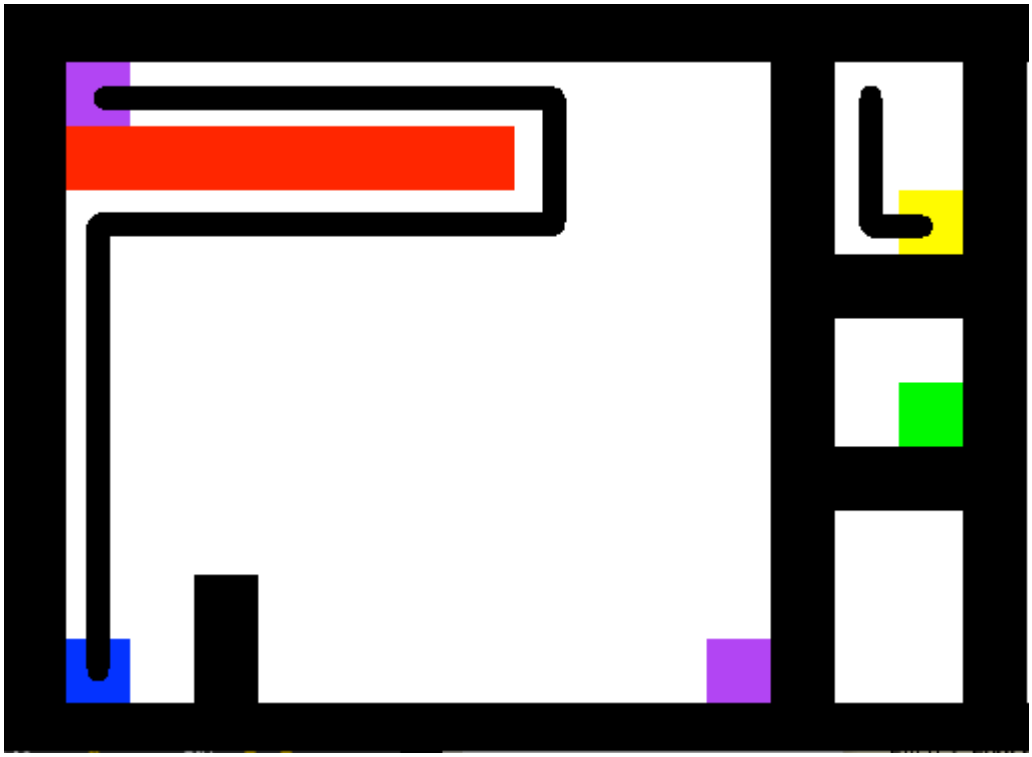

Figure 15 - The mentor's policy for experiment 4. As in previous examples, the mentor follows a suboptimal policy. Since the agent's prior does not include information about transitions from the purple teleporting state, the imitation agent is likely to converge to this policy.

This policy will bias the agent towards a policy which ends in the yellow +1 state, rather than the green +2 state. The values computed by the imitation agent after one observation of the mentor are shown in **Figure 16**.

Figure 16 - Values calculated by the imitation agent based on mentor observations. A policy based on these values will be suboptimal with the agent ending in the yellow +1 state rather than the green +2 state.

As **Figure 16** shows, the imitation agent will be biased towards the yellow +1 state. Although it may discover that the lower right purple state causes a transition to a state near the +2 state, this is unlikely because of the values shown in **Figure 16**. This example uses principles which are very similar to the first two examples. However, the addition of a prior actually makes the agent behave worse than it would without a prior. This is because without a prior, the agent initially only computes values for states that were visited by the mentor. However, the addition of a prior means that the agent computes values for all states connected to those that the mentor visited. So even if the agent strays from the path followed by the mentor, it will still be biased by the values it computed from the mentor observations.

As in earlier examples, the non-imitation agent is able to outperform the imitation agent because it is not biased by the mentor. Since, according to the given prior, no states with positive rewards are reachable from the start state, the non-imitation agent will initially assign most states a value of 0. Then during the course of exploration, it is much more likely to discover the lower right teleporting state than the upper left one because of the series of red -1 states. The cumulative rewards for both agents averaged over 5 runs are shown in **Figure 17**.

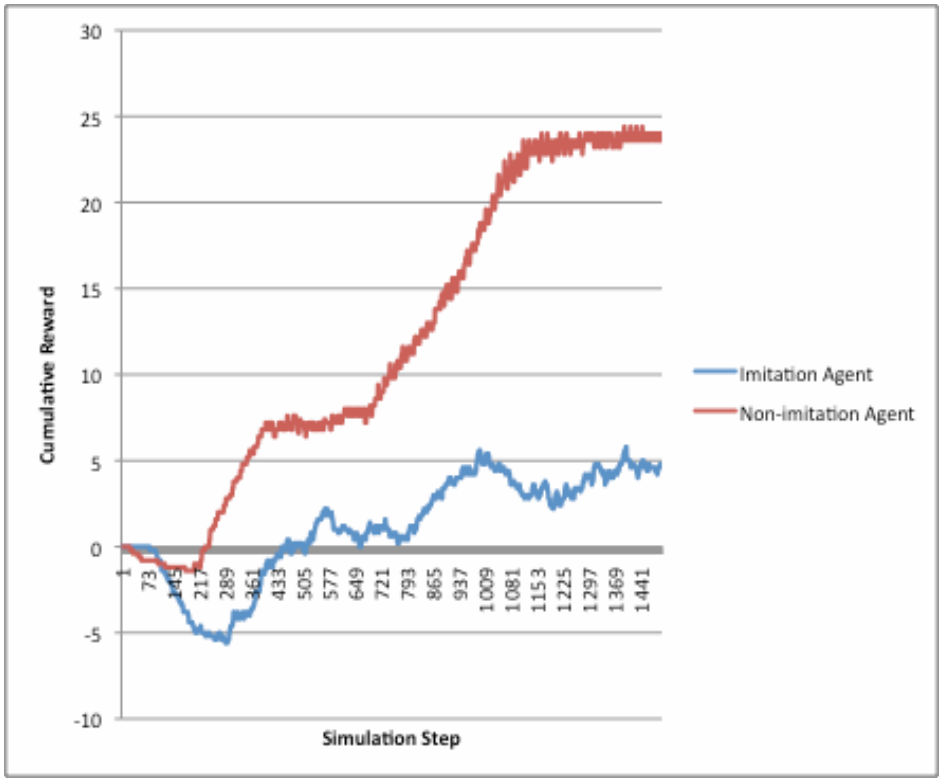

Figure 17 - Cumulative rewards over last 200 steps as a function of step number, averaged over 5 runs.

As in the previous examples, the non-imitation agent is able to significantly outperform the imitation agent. Although it takes the non-imitation agent a little longer to find the +2 state than it does for the imitation agent to find the +1 state, the extra time

spent exploring pays off easily in the long run. Also note that while this example deals with a specific type of special state not covered by the prior, it is easy to imagine other types of environments where this effect could occur. Any time the agent is given a prior that does not accurately describe all states, it is possible to get the kind of discontinuity shown above, and therefore possible for the agent to be negatively affected by the mentor.

5.3.2 Experiment 5

Our final experiment deals with a different kind of misleading prior. In addition to priors that do not correctly account for certain types of states, it is also possible to have a prior which does not correctly reflect the stochastic nature of the MDP. In particular, we will focus on an MDP which has a small chance of actions failing, but where the agent is given a prior that indicates a larger chance of actions failing. Then the agent will underestimate the value of states that are close to states with a negative reward because of the chance of accidentally ending up in one of those states. This fact can then be exploited by the mentor in much the same way as before. If there are two states with different magnitudes of positive rewards and the mentor always transitions to the state with smaller positive reward, the agent will be biased towards that lower reward state.

To demonstrate this, we create the MDP shown in **Figure 18**. In this MDP, the probability of an action failing is 0.1. However, the agent is given a prior which indicates that the probability of an action failing is 0.5.

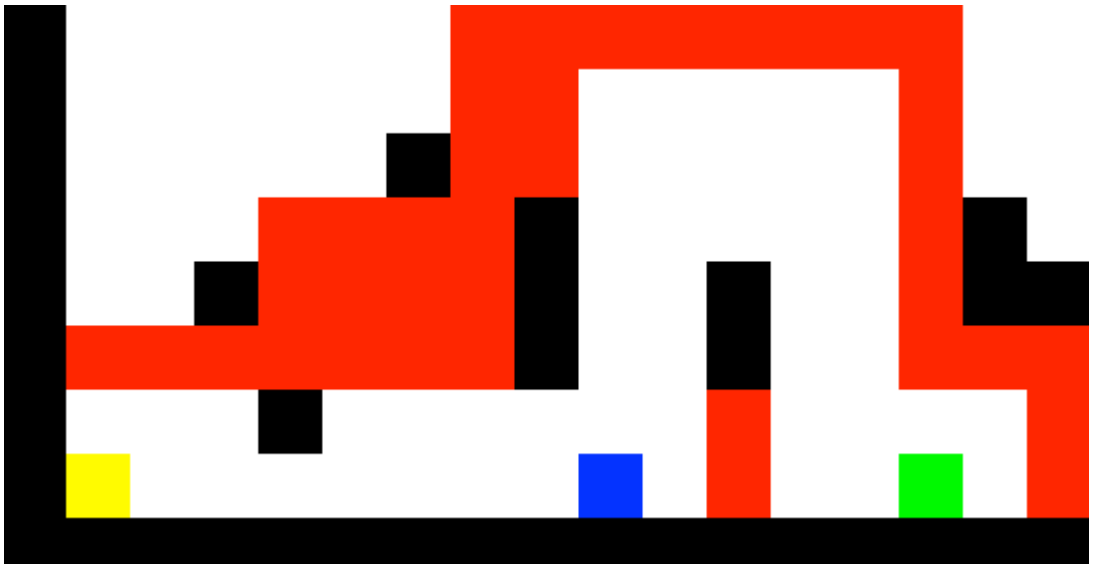

Figure 18 - MDP layout for experiment 5. As in previous examples, there is a suboptimal policy ending in the yellow +1 state and an optimal policy ending in the green +2 state. There is only a small amount of stochasticity, but the agent's prior indicates that actions are highly stochastic. As a result, it will slightly favor a suboptimal policy ending in the yellow state, but can discover an optimal policy through exploration.

Note that the start position of the agent is changed compared to previous experiments. The layout is such that based on values calculated from the prior, the value of going to the green +2 state compared to the yellow +1 state will be very similar. The values calculated based solely on the prior are shown in **Figure 19**.

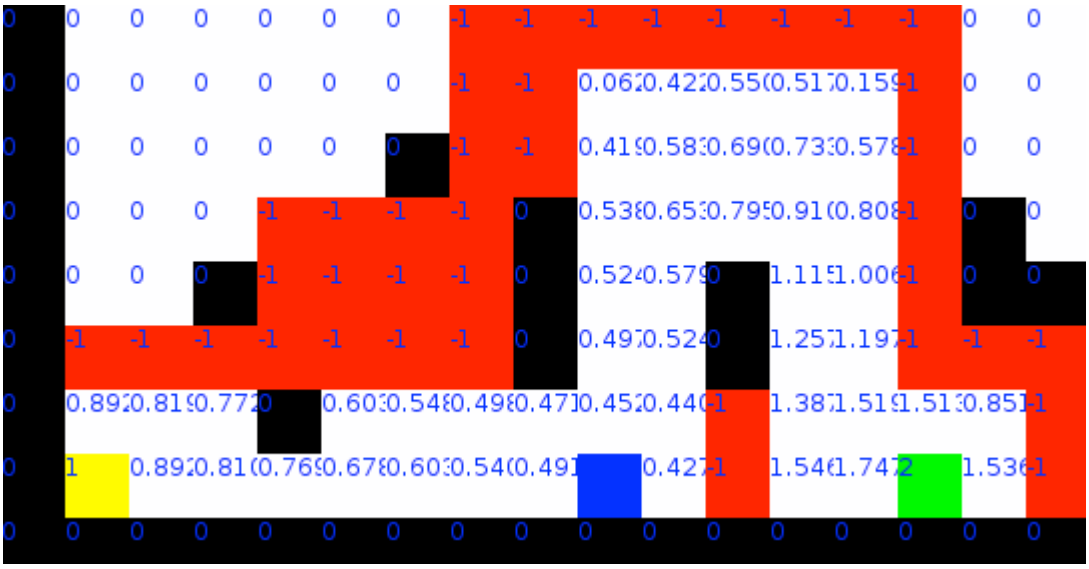

Figure 19 - Values calculated from the prior. The agent slightly favors a policy ending in the yellow +1 state, but if it explores one state up, it can discover an optimal policy.

Since the prior does not correctly capture transition probabilities, these values are slightly wrong. Importantly, the agent will be drawn towards the yellow +1 state rather than the green +2 state from its starting position. If the agent happens to transition to the state above the start state (either by chance or during exploration) it will instead be drawn towards the green +2 state. However, the mentor can easily change this. Since the agent doesn't have an accurate description of transition probabilities, the mentor can have a large impact on the agent's value calculations. In our experiments, the mentor follows the policy shown in **Figure 20**.

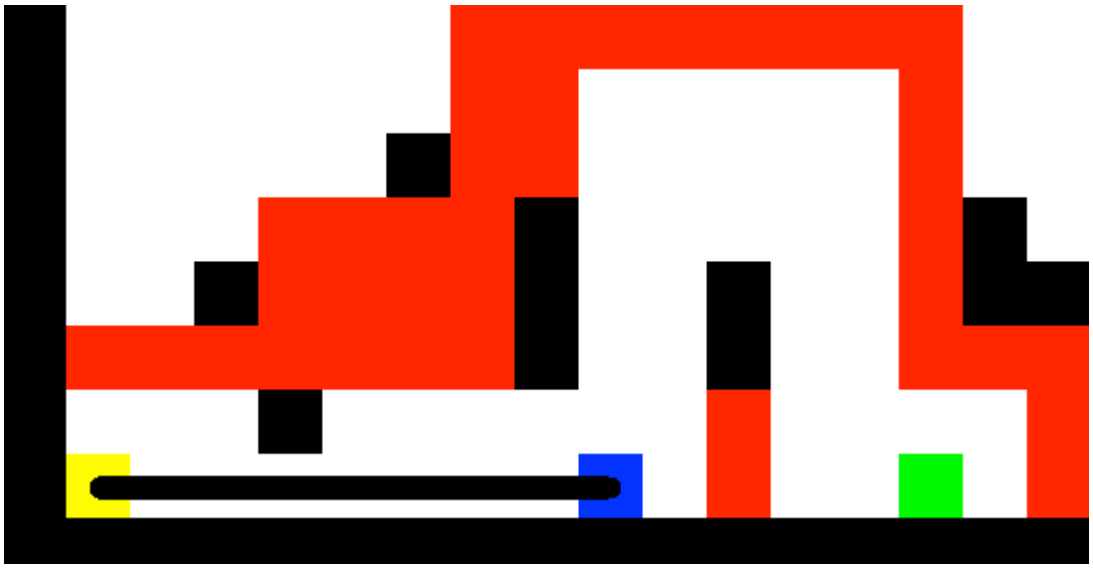

Figure 20 - Mentor policy shown in black. The mentor will bias the imitation agent towards a suboptimal policy ending in the yellow +1 state. Since the agent's policy suggests highly stochastic actions, the observations of the mentor's transitions will significantly increase the values of states visited by the mentor, thus biasing the imitation agent towards a suboptimal policy.

If the mentor happens to transition to one of the states above its desired route, it immediately takes the action "down" to return to its route. By using this policy, the mentor will bias the agent towards the yellow +1 state. To demonstrate this, **Figure 21** shows the agent's value calculations based on the prior and a single observation of the mentor.

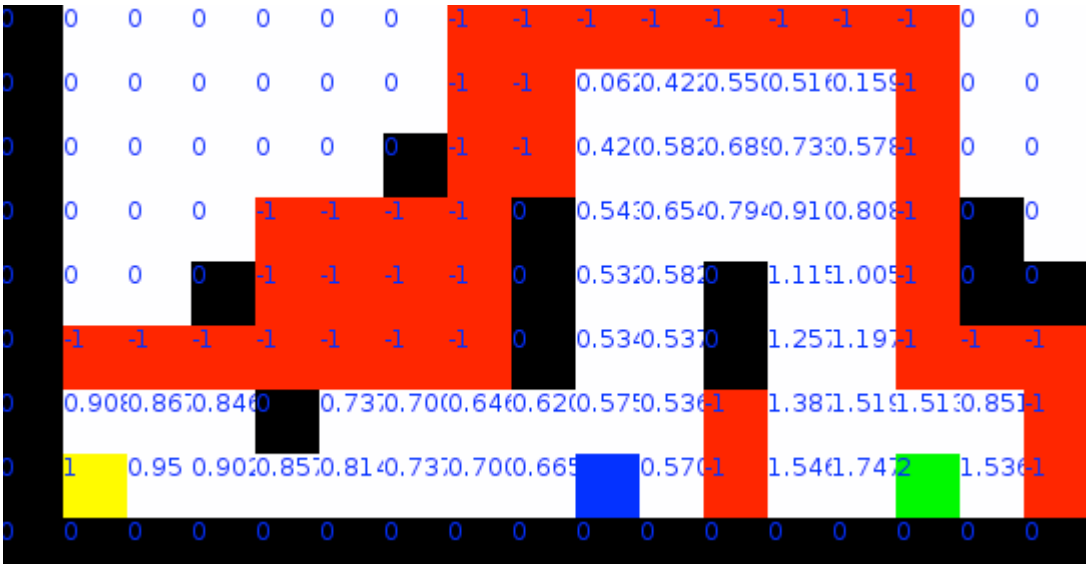

Figure 21 - Values after mentor observations. Observations of the mentor transitions have greatly increased the values of states leading to the yellow +1 state. As a result, the agent is less likely to discover an optimal policy. Even if it explores in states above its starting position, the values shown in the figure will bias it towards the yellow +1 state.

After the mentor observations, the agent has higher value estimates for the states visited by the mentor (as well as nearby states). However, the agent is still unaware of the correct state transition probabilities in the other part of the state space (i.e. near the green +2 state). As a result, the agent is more strongly biased towards the yellow +1 state. The agent now needs to go up three times before its policy will dictate going towards the green +2 state. As a result, the imitation agent will perform poorly compared to the non-imitation agent. The cumulative rewards averaged over 5 runs are shown in **Figure 22**.

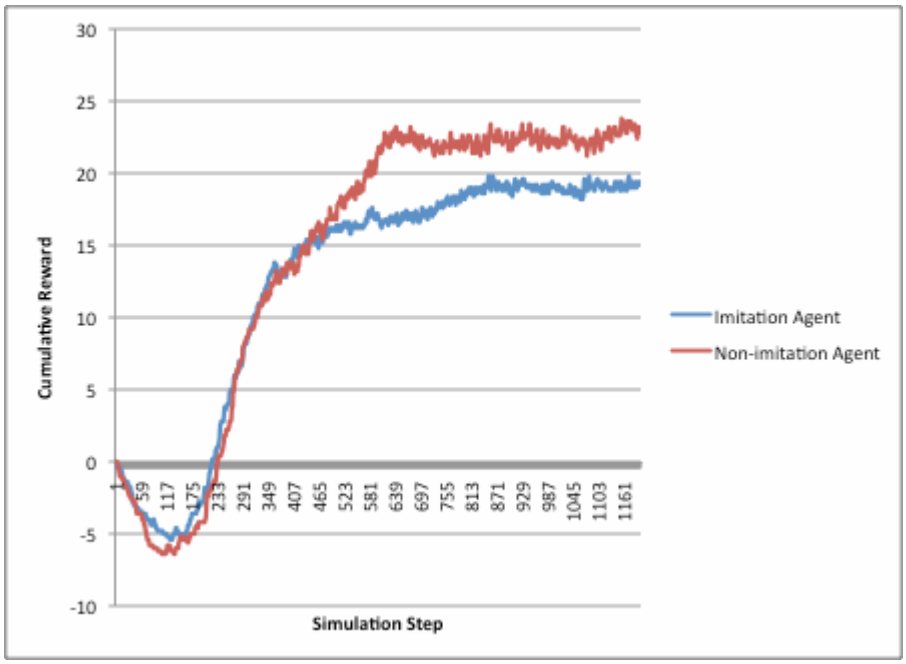

Figure 22 - Cumulative rewards over last 200 steps as a function of step number, averaged over 5 runs.

In general, the non-imitation agent will converge to a better policy which exploits the +2 state while the imitation agent will converge to a policy which only exploits the +1 state.

6 Discussion

Overall, these examples show that it is possible for a mentor to negatively impact an imitation agent compared to a non-imitation agent. Some of the examples may seem contrived or unrealistic, but they were specifically designed to demonstrate ways in which an imitation agent can be misled by its mentor. There are many related, more realistic versions of the above examples where similar phenomena can be observed, albeit possibly with less consistency.

In general, there are several conditions needed for a suboptimal mentor to mislead the observer. First, the MDP must allow for different levels of optimal policies.

All of the above examples rely on convincing the agent to follow a policy which is locally optimal but not globally optimal. In most of the above examples this involved states with different levels of rewards (i.e. +1 compared to +2) but it could also involve other types of optimality such as paths of different length or level of danger, as in experiment 3 above. Secondly, the mentor needs to be able to be able to bias the agent towards one of these suboptimal policies. Since the augmented Bellman backup includes a "max" operator, this can only be accomplished by improving the agent's value estimates of some subset of the states. This is particularly easy to accomplish when the agent has no prior and thus no initial knowledge of state transitions, but it is also possible when the agent has an inaccurate or incomplete prior. And finally, the MDP needs to be laid out in such a way that a non-imitation agent will find an optimal policy by simple exploration. In all of the examples above, the globally optimal policy was easier to discover than the suboptimal policy followed by the mentor. The mentor was able to bias the agent towards a policy that it otherwise was not likely to discover. And since this policy was suboptimal, the imitation agent performed worse than the non-imitation agent.

6.1 Protecting Against a Suboptimal Mentor

Having shown that it is possible for suboptimal mentors to negatively impact an imitation agent, we turn our attention to how an agent can protect itself from a suboptimal mentor. The first defense is a good prior. It is much easier for a mentor to cause suboptimal behavior in an agent with no prior than in an agent with a good prior. A basic prior that describes connectivity between states will prevent many types of

negative mentor influence. However, any prior which causes the agent to compute values that can be improved by mentor observation is potentially vulnerable to a suboptimal mentor. Of course, having a good prior is helpful regardless of concerns about the mentor. Moreover, having a perfect or even very good prior is often impossible, especially in larger state spaces.

An imitation agent can also be made more resistant to a suboptimal mentor by using a more sophisticated exploration policy. Epsilon-greedy exploration fails because the agent explores less frequently as time goes on, regardless of its knowledge of the state space. The agent relies on the mentor to help determine its policy, and as a result does less exploration than it might otherwise. If the non-imitation agent does not discover any positive reward states, it will continue exploring and thus potentially find a more optimal policy. Of course, epsilon-greedy exploration is always vulnerable to finding locally optimal but globally suboptimal policies. However, the mentor can exploit this weakness by essentially highlighting a suboptimal policy and reducing the amount of exploration done by the agent.

Any exploration strategy in which the amount of exploration is proportional to time is vulnerable to this type of behavior. In fact, some common exploration policies are even more vulnerable than the basic epsilon-greedy approach. For example, Price and Boutilier [Price and Boutilier, 2003b] mention the Boltzmann approach as an alternative to simple epsilon-greedy exploration. In the Boltzmann approach, when the agent chooses to explore it chooses an action with probability proportional to its Q-value, rather than uniformly. But this kind of exploration will result in the imitation agent

mirroring the mentor's policy even more closely. The agent will be less likely to stray from states visited by the mentor and thus less likely to discover an optimal policy.

To combat this weakness, a more robust exploration strategy is needed. Price and Boutilier [Price and Boutilier, 2003a] discuss Bayesian exploration [Dearden *et al*., 1999] as one alternative. Bayesian exploration attempts to keep track of the agent's knowledge of different states and explore in such a way as to increase the agent's knowledge of unknown states. This strategy would be an improvement over epsilon-greedy exploration in all of the above examples. In each case, the imitation agent stops exploring before it has visited every state in the state space. If the agent did more exploration based on its knowledge of the state space, it would be more likely to find an optimal policy and less vulnerable to a suboptimal mentor. Depending on how such a strategy is implemented, it might still be vulnerable to a bad mentor. If the agent included mentor observations in determining its knowledge of a state, then the mentor could still prevent the agent from exploring certain areas of the state space. But this remains an interesting question and a potential area for future research.

7 Conclusion

Implicit imitation learning works extremely well when the agent is given a mentor that follows an optimal policy. By observing such a mentor, the agent can greatly speed up the process of learning an optimal policy of its own. Since the agent only needs to observe the mentor's state transitions, this approach can be applied even in cases where the mentor is unaware of the observer. However, care needs to be taken when learning from an arbitrary mentor. In many types of MDPs, attempting to learn from a

suboptimal mentor may delay or even prevent the agent from converging to an optimal policy, while a non-imitation agent may be able to find an optimal policy through exploration. Some example MDPs and mentor policies were illustrated above, but there are many similar MDPs in which the same types of behavior could be observed.

Given that it is possible for a suboptimal mentor to cause an imitation agent to perform poorly compared to a non-imitation agent, the next question is how to avoid this behavior. If an agent tries to learn from an arbitrary mentor it may not be practical to ensure that the mentor is following an optimal policy. Moreover, the agent can potentially benefit from observing a suboptimal mentor so long as it does not substitute mentor observations for exploration.

The first defense against suboptimal mentors is a good prior distribution over state transition probabilities. An agent with a good prior will be much harder to fool and will also converge more quickly, regardless of the mentor's policy. But having a good prior is always helpful, and good priors may be difficult to create. The second and more interesting defense is a good exploration policy. Any exploration policy which does not simply reduce exploration over time is a potential improvement over epsilon-greedy exploration. Bayesian exploration [Dearden *et al*., 1999] is particularly interesting and has already been applied to implicit imitation learning [Price and Boutilier, 2003a]. However, further study is needed to determine exactly how Bayesian exploration (and potentially other exploration policies) fare when given a suboptimal mentor.

One of the greatest benefits of implicit imitation learning is that the agent can potentially learn from any other agent; all that is required is the ability to observe. And while it may be tempting to design agents which can learn from arbitrary mentors, one

must be careful. This work has shown that attempting to learn from a suboptimal mentor can cause undesirable behavior in many common cases. We have also attempted to begin a discussion of what steps should be taken to prevent such behavior. With more research into applications of sophisticated exploration policies, it should be possible to design agents which take advantage of all the benefits of implicit imitation learning but avoid the drawbacks of trying to learn from a suboptimal mentor.

Acknowledgments

Many thanks to my advisor Prof. Rebecca Fiebrink for her guidance and advice over the course of this project. Especially when my ideas didn't work out, her help was invaluable. Thanks also go to Alex and my family for their love and support. Without them I would never have made it this far.

References

[Bellman, 1957] R. Bellman. Dynamic Programming. Princeton University Press.

[Dearden *et al*., 1999] R. Dearden, N. Friedman, D. Andre. Model based Bayesian Exploration. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence,* pp. 150-159, Stockholm, 1999.

[Price and Boutilier, 1999] B. Price and C. Boutilier. Implicit imitation learning in multiagent reinforcement learning. In *Proceedings of the Sixteenth International Conference on Machine Learning,* pp. 325-334, Bled, SI, 1999.

[Price and Boutilier, 2003a] B. Price and C. Boutilier. A Bayesian approach to imitation in reinforcement learning. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence* Acapulco.

[Price and Boutilier, 2003b] B. Price and C. Boutilier. Accelerating reinforcement learning through implicit imitation. *Journal of Artificial Intelligence Research 19 (2003)* pp 569-629.

[Vermorel and Mohri, 2005] J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. *Machine Learning: ECML* pp. 437-448, 2005.