Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

TD-$\Delta\pi$: A Model-Free Algorithm for Efficient Exploration

Bruno Castro da Silva and **Andrew G. Barto**
{bsilva,barto}@cs.umass.edu
Department of Computer Science, University of Massachusetts, Amherst, MA 01002 USA

Abstract

We study the problem of finding efficient exploration policies for the case in which an agent is momentarily not concerned with exploiting, and instead tries to compute a policy for later use. We first formally define the Optimal Exploration Problem as one of sequential sampling and show that its solutions correspond to paths of minimum expected length in the space of policies. We derive a model-free, local linear approximation to such solutions and use it to construct efficient exploration policies. We compare our model-free approach to other exploration techniques, including one with the best known PAC bounds, and show that ours is both based on a well-defined optimization problem and empirically efficient.

1 Introduction

Balancing exploration and exploitation is a classical problem in Reinforcement Learning (RL). This problem is relevant whenever one has to learn a good actuation policy, while *at the same time* obtaining as much reward as possible. Often, however, it makes sense to assume an initial training phase during which the goal is to just explore efficiently, so that an optimal policy can be learned fast but without necessarily worrying about performing well (Şimşek and Barto 2006). This is useful whenever collecting online samples is costly or when pre-learning a set of skills might help optimizing other tasks later on. In this paper, we are interested in finding a good exploration policy to collect revelant samples from a Markov Decision Process (MDP) such that a reasonable exploitation policy can be quickly constructed.

We first formally define the Optimal Exploration Problem as one of sequential sampling by posing it as an MDP constructed by expanding the state space of the original one that we want to explore. Solutions to this expanded MDP correspond to paths of minimum expected length in the space of policies and describe optimal sequential sampling trajectories. We show an important property of such solutions and a special function that can be constructed based on them. Since directly computing these solutions is not feasible, we derive a local linear approximation to the relevant estimates and present an intuitive geometric interpretation of its meaning. We compare our model-free approach to other exploration techniques, most importantly Delayed Q-Learning (DQL) (Strehl et al. 2006) and ΔV (Şimşek and Barto 2006), and show that ours is both based on a well-defined optimization problem and empirically efficient.

2 Related Work

Efficient exploration in RL has been studied extensively, usually with the objective of maximizing return in an agent's lifetime, thus requiring a trade-off between exploration and exploitation. In this paper, on the other hand, we are concerned with purely exploratory policies. Some of the existing approaches to tackle this are simple techniques such as random exploration, picking actions that were selected the least number of times, visiting unknown states, etc. (Thrun 1992). These are inefficient due to treating the entire state space uniformly, ignoring useful structure provided by the value function. Other approaches for efficiently learning consider the full exploration versus exploitation problem directly. Duff (2003) proposes a Bayesian approach for the case in which prior uncertainties about the transition probabilities are available; Abbeel and Ng (2005) present a method for computing a near-optimal policy assuming that demonstrations by a teacher are available. Another model-based Bayesian approach was proposed by Dearden, Friedman, and Andre (1999), where the Value of Information for exploring states is computed considering a model and uncertainty about its parameters. Finally, Kolter and Ng (2009) present a method for constructing a belief state for the transition probabilities and obtaining a greedy approximation to an optimal Bayesian policy.

Two techniques have been especially influential among researchers studying efficient RL algorithms: R–Max (Brafman and Tennenholtz 2001), and E^3 (Kearns and Singh 1998). Both give polynomial guarantees for the time to compute a near-optimal policy. These techniques differ from ours in at least two important aspects: *(1)* they maintain a complete, though possibly inaccurate model of the environment; *(2)* they perform expensive, full computations of policies (via, e.g., value iteration) over the known model as steps in their algorithms. Therefore, a direct, meaningful comparison with our **model-free** approach would be difficult. Instead, we compare with Delayed Q-Learning (Strehl et al. 2006), a model-free approach which, to the best of our knowledge, has the best known PAC-MDP bounds and

which provably performs near-optimally in all but a polynomial number of timesteps.

Other techniques relevant to this work include the Active RL algorithm (Epshteyn, Vogel, and DeJong 2008) and the ΔV approach (Şimşek and Barto 2006). The former is similar to ours in that it defines an exploration policy based on a type of sensitivity analysis, namely that of the policy with respect to perturbations to a model. It differs from ours in that our analysis focuses, alternatively, on the impact that collecting additional samples has on the expected evolution of Q-values, and therefore on the ranking of actions, and also in that Active RL assumes an initial, complete estimate of a model, while we don't. The latter approach (ΔV) shares with ours the idea of model-free exploration and uses a similar formalization. It focuses exploration on regions of the state space where the magnitude of the value function changes the most, implicitly maximizing the speed with which the value function is fine-tuned. Unfortunately, it has practical shortcomings, mainly because agents following it become "obsessed" with fine-tuning the value of states even when the policy is already correct. Intuitively, the specific values of the states shouldn't matter; the important information to be acquired is the *ranking* of actions. Achieving this type of exploration strategy is the goal of this paper.

3 Summary of the Method

We first informally describe our method and in later sections show that it constitutes a principled approximation to a well-defined optimization problem. The algorithm that we propose, which we call $\Delta\pi$, focuses exploration not on regions where the value function is changing the most, or in which a model is being made more accurate, as several of the works described in Section 2, but on regions where the *likelihood of a change in the policy is high*. In other words, exploration is based on how sensitive the policy is at a given state, given that we continue to gather information about it.

The indicator of policy sensitivity that we use is based on a simple linear extrapolation of the behavior of the Q-function at a state, both for the action currently considered to be optimal and for some other recently tried action. Specifically, given any unbiased estimate of how a Q-value changes as more samples are collected (e.g., a temporal difference error), we can estimate if and *when* the value of some action with a lower Q-value will surpass the value of the one currently considered to be optimal. When exploring, one should find desirable those actions whose Q-values are soon likely to surpass that of the action currently considered to be optimal; the sooner this crossing point is predicted to occur, the more attractive the action should be. If, on the other hand, the evolution of Q-values indicates that the ordering of actions is likely to remain the same, then we shouldn't find those states attractive (Figure 1). We denote an approximation to the *expected time until a policy change* in a given state by $d(s, a)$. This value serves as a guide to how valuable it is to explore certain parts of the MDP based on how likely it is that new samples from them will lead to changes in the policy. The derivation of $d(s, a)$ as a principled approximation and its precise definition are given in Sections 4 and 5.

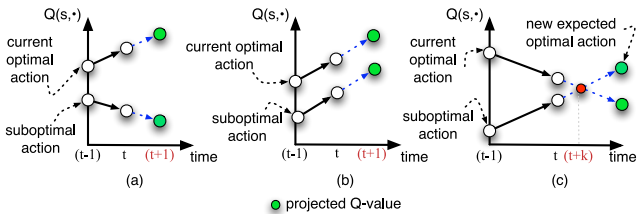

Figure 1: Geometric interpretation of the expected time until a *policy change*, $d(s, a)$, represented by the intersection between lines. (a) Q-values predicted to diverge; no change expected; (b) Q-values seem to evolve at same rate; no change expected; (c) change expected in k steps.

Notice that negative expected times until a policy change have a natural interpretation: a change has *already* happened and now the Q-values seem to be diverging (Figure 1a). Our exploration algorithm considers small values of $|d(s, a)|$ attractive, since they either indicate that a policy change is expected soon or that one has happened recently. In the latter case, it might be important to continue exploring the corresponding states to ensure that the change was not caused by noise in the sampling of rewards and next states. In order to model this, we do not use $d(s, a)$ directly as an indication of policy sensitivity; instead, we define the following quantity:

$$r(s,a) = \begin{cases} \exp\left(-\frac{d^2(s,a)}{\sigma}\right) & |d(s,a)| < \lambda \\ -p & |d(s,a)| \geq \lambda, \end{cases} \quad (1)$$

where p is a small penalty given when the action's values seem to have stabilized, λ quantifies how rigorous we are when deciding whether this is the case, and σ controls the maximum horizon of time during which we trust the predictions made by our local linear approximation. In practice we have observed that many functions other than the Gaussian can be used to define $r(s, a)$, as long as they are monotonically increasing and decreasing in the same intervals as a Gaussian and the resulting $r(\cdot, \cdot)$ is bounded. In systems where noise does not play a crucial role, one might want to favor exploration of expected *future* policy changes by adding a penalty (e.g, -1) to $r(s, a)$ in case $d(s, a) < 0$.

We point out that the direct use of $r(s, a)$ as a guide for exploration provides just a myopic perspective, since it reflects only the value of exploring *one specific state and action*. In general, though, the choice of which regions to explore is a sequential decision problem: states that do not look promising now might allow the agent to reach regions where several corrections in the policy are expected. This can be dealt with by using $r(s, a)$ as a *new*, surrogate reward function for the MDP that we want to explore, in which case its solutions approximately minimize the sum of times until all policy corrections are performed. This corresponds to executing an exploration policy that tries to correctly rank actions as fast as possible (see Section 4). Notice that because $r(s, a)$ is used as a surrogate reward function, we need to store the original Q-function (the one related to the exploitation policy being estimated) separately. Specifically, we keep track of two separate Q-functions: one related to the *exploration*

policy and one to the *exploitation policy*. The latter is constructed based on samples collected by the former, and the former is updated given new estimates from the latter. This is also the approach taken by Şimşek and Barto (2006).

4 Optimal Exploration

Let an MDP M be a tuple $\left(S_M, A_M, R_M, T_M, \gamma_M\right)$, where S_M is a finite set of states, A_M is a finite set of actions, $R_M : S_M \rightarrow \Re$ is a reward function, $T_M : S_M \times A_M \times S_M \rightarrow [0, 1]$ is a transition function, and γ_M is a discount factor. Solving M consists of finding an optimal policy π^*_M, i.e., a mapping from states to actions that maximizes the expected discounted sum of future rewards. Let $Q^\pi_M(s, a)$ be the function that gives the expected total discounted reward obtained when taking action a in s and following π thereafter. The optimal Q-function for an MDP M is denoted by Q^*_M, and an estimate of it at time t by Q^t_M. A greedy policy with respect to a Q-function can be derived by taking the action that maximizes the Q-function at a given state; let $\pi_{[Q]}$ be this deterministic greedy policy, obtained when using Q to rank actions and breaking ties randomly. Let $V^\pi(s)$ be the value of state s when following policy π, and $V_D(\pi)$ be the value of a policy π given an initial state distribution: $V_D(\pi) = \sum_{s \in S} D(s)V^\pi(s)$, where $D(s)$ is the probability of the MDP starting at state s.

For the problem of *Optimal Exploration*, we wish to find a (possibly non-stationary) policy such that the samples it collects allow for the identification of π^*_M as quickly as possible; we note that this is different from calculating V^*_M as quickly as possible. Specifically, an optimal exploration policy might correctly rank all *optimal* actions even though the values of some (or all) states are still inaccurate. Formally, we define the Optimal Exploration Problem as one of **sequential sampling** by posing it as an MDP constructed by expanding the state space of the process we originally want to explore. Solutions to this expanded MDP correspond to paths of minimum expected length in the space of policies and describe optimal sequential sampling trajectories. Based on the original MDP M, we define a new MDP, M', such that any optimal policy for M', by construction, induces an optimal exploration strategy for M. As will become clear shortly, optimality is defined in terms of the minimum expected number of actions (or steps) needed until enough information is collected and π^*_M can be found. We construct M' in a way so that trajectories in it correspond to sequences of joint evolution of states in M and estimates Q^t_M; this evolution satisfies the Markov property and encodes trajectories in the space of policies for M. M' is defined by:

- a state space $S' = S_M \times \Re^{|S_M||A_M|}$. S' corresponds to the same state space of M, but augmented with the current estimate of the optimal Q-function for M. We denote the state $s'_t \in S'$ in which M' is at time t as a tuple $s'_t \equiv \left(s^t_M, Q^t_M\right)$;
- an action space $A' = A_M$, i.e., the same as the action space of the original MDP;
- Q^0_M, an initial estimate of Q^*_M;
- L, an off-policy, deterministic learning mechanism that converges to an optimal policy. Given an action a taken in M', we can imagine also executing a in the original MDP, M, and observing a sample experience $\left(s^t_M, a, r^t_M, s^{t+1}_M\right)$. L then takes this information, along with Q^t_M, and returns an updated estimate of the Q-function: $Q^{t+1}_M \leftarrow L(s^t_M, a, r^t_M, s^{t+1}_M, Q^t_M, \rho)$, where ρ is the set of any other data structures or parameters required by L, such as learning rates, models, etc;
- T', a transition function based on T_M and L. Given the current state s'_t of M', T' describes the distribution over possible next states s'_{t+1}. Since s'_{t+1} is a tuple of the form $\left(s^{t+1}_M, Q^{t+1}_M\right)$, we can think of T' as computing each of those components independently: s^{t+1}_M probabilistically according to $T_M(s^t_M, a)$, and Q^{t+1}_M by applying L to the last sample experience obtained when executing a in M. Also, T' is such that all states with zero instantaneous reward (i.e., goal states, as defined below) are absorbing;
- $0 < \gamma' < 1$, a (fixed) discount rate;
- R', a reward function mapping states of M' to the reals:

$$R'(\left(s^t_M, Q^t_M\right)) = \begin{cases} -1 & \text{if } V_D(\pi_{[Q^t_M]}) \neq V_D\left(\pi^*_M\right) \\ 0 & \text{otherwise.} \end{cases}$$

Note that rewards in M' are nonnegative only in states in which the use of the best actions, according to the ranking induced by Q^t_M, yields a greedy exploitation policy for M whose value is optimal. This ensures that maximizing cumulative rewards in M' is equivalent to efficiently reaching a Q-function for M that allows all *optimal* actions to be correctly ranked. This is made rigorous in Proposition 1:

Proposition 1 *An optimal policy for M' specifies a path of minimum expected length in the space of policies for M, starting from an arbitrary initial policy and reaching an optimal policy for M. Paths between policies are specified by sequences of sample experiences in M.*

Proposition 1 follows from the facts that *(1)* S_M and A_M are finite and L is deterministic, and thus from any $s' \in S'$ there exists only a finite number of possible next states in M'; *(2)* since R' is bounded and $0 < \gamma' < 1$, the value function for M' is bounded, specifically in $[\frac{1}{\log \gamma'}, 0]$; and finally *(3)* because L is a learning algorithm that converges to an optimal policy for M (even if asymptotically), there exists *at least one* proper policy for M', that is, one that reaches the goal state with probability 1 regardless of the initial state. This is true because otherwise M would not be solvable. Taken together, these observations imply that there exists a nonempty, possibly uncountable number of proper policies for M', which form a totally ordered set with respect to the value of each policy. Because this set is bounded above, its supremum is well-defined and there exists an optimal policy for M'. This policy, by construction, minimizes the expected number of samples needed in order to compute π^*_M. All above-mentioned expectations are taken over all possible trajectories in M'. This result is similar to the more general problem of Stochastic Shortest Paths (SSP) (Bertsekas and

Tsitsiklis 1991) — the main diference being that SSPs require MDPs with finite state spaces. Finally, note that M' is constructed in such a way that an optimal policy for M is reached whenever the greedy policy induced by the current estimate Q^t_M correctly ranks all optimal actions, *even if the values of the states themselves are still inaccurate*.

It should be clear that directly solving M' is not feasible, since R' assumes prior knowledge of an optimal policy for M. This impossibility is not surprising: one cannot find a *truly* minimal sequence of exploration actions without knowing beforehand T_M and R_M, which would make exploration unnecessary. However, M' is useful since we can observe general properties of its solutions and use them to construct a principled technique for efficient exploration. In what follows, we discuss some of these properties and derive a local linear approximation which allows us to construct a principled exploration strategy called $\Delta\pi$.

Let $\phi^\pi_{s,a}(t)$ be the expected value of $Q_M(s,a)$ after a trajectory of length t in M', starting from some given state $s' \in S'$ and following a fixed policy π for M'. In order to simplify the notation, we suppress the dependence on s':

$$\phi^\pi_{s,a}(t) = E[Q^t_M(s,a)]. \tag{2}$$

The above expectation is taken with respect to trajectories in M'; the probabilities involved depend on π and T'. ϕ encodes how Q-value estimates are expected to evolve if updated with samples collected by π. Let π_{expl} be any policy for M'; this policy induces an exploration policy for M. Let us analyze the expected length k of the shortest trajectory in M', when following π_{expl}, such that we expect a change in the greedy policy (for M) induced by the expected Q-values: $\arg\min_k \left[\exists s \in S \quad \pi_{[\phi^{\pi_{expl}}(t+k)]}(s) \neq \pi_{[\phi^{\pi_{expl}}(t)]}(s)\right]$. If this is generated by an optimal policy for M', then k is the expected minimum number of samples from M needed to cause a change in the current greedy policy. Similarly, we can define the expected minimum number of samples until the induced policy changes *in a given state* $s \in S$:

$$\arg\min_k \left[\pi_{[\phi^{\pi_{expl}}(t+k)]}(s) \neq \pi_{[\phi^{\pi_{expl}}(t)]}(s)\right]. \tag{3}$$

Let us now assume we have taken an arbitrary step in M' and observed a next state $s'_{t+1} \in S'$. This state contains an updated estimate of the Q-function, namely Q^{t+1}_M. If the ranking of actions induced by Q^t_M changes with respect to Q^{t+1}_M, we say a *crossing* has occurred. For example, if a_1 and a_2 are actions and $Q^t_M(s,a_1) > Q^t_M(s,a_2)$ but $Q^{t+1}_M(s,a_1) \leq Q^{t+1}_M(s,a_2)$, then a crossing has occurred.

Note that ϕ is defined only in the domain of integer timesteps. For our purposes, however, it is advantageous to embed it in a continuous process by assuming that updated Q-values change *linearly and continuously* between timesteps. Viewing ϕ as a function of continuous time is useful for the following reason: if $Q^t_M(s,a_1) > Q^t_M(s,a_2)$ but $Q^*_M(s,a_1) \leq Q^*_M(s,a_2)$, then for some (not necessarily integer) k, $\phi^\pi_{(s,a_1)}(t+k) = \phi^\pi_{(s,a_2)}(t+k)$, assuming that π is a proper policy for M'. This proposition is trivially true because of the Intermediate Value Theorem. It allows us to say that a crossing occurs precisely at the time k when the Q-values of two actions are momentarily equal, before one surpasses the other. It also helps us to interpret non-integer values of ϕ, which might occur since it is an expectation. Finally, it makes it easier to meaningfully compare non-integer expected crossing times in terms of the *rate* with which the ranking of actions seems to be changing. This becomes particularly clear if using Boltzmann policies with high temperatures, in which case the rate of change in action probabilities of two actions, as new samples are collected, can be shown to cross exactly when the derivatives of their Q-values becomes equal. This connection between the rate of change in action preferences and the derivative of their Q-values appears again as part of the solution of Equation 5.

5 Deriving an Efficient Exploration Policy

We would now like to use the definition of ϕ (or an approximation of it) to derive an efficient, though not necessarily optimal, exploration strategy for M. We first observe that because updates to the Q-function are generally not independent, the *minimum time to rank actions in all states* (the quantity minimized by $\pi^*_{M'}$) is not equal to the sum of the minimum times to rank *actions at each state* in turn. However, we propose that a policy that minimizes the latter is also a good approximation of the former. We further note that because $d(s,a)$ is an estimate of the minimum time until a change in ranking *at a given state*, it is possible to minimize that latter quantity by solving a sequential decision process in which $d(s,a)$ (or a related quantity) serves as a surrogate reward function for M. Under this new reward structure, π^*_M defines an *efficient exploration policy* which quickly improves the ranking of actions at each state. For more details, see Algorithm 1. We empirically show this to be an effective approximation in Section 6 and discuss when it might perform poorly in Section 7. Finally, note that minimizing $d(s,a)$ is equivalent to *minimizing the time until the nearest crossing*. Let us build on this last observation and define $c_{(s,a_1,a_2)}(t)$, the expected difference between the Q-values of any actions a_1 and a_2, for any given state $s \in S$:

$$c_{(s,a_1,a_2)}(t) = \phi^\pi_{(s,a_1)}(t) - \phi^\pi_{(s,a_2)}(t). \tag{4}$$

The smallest root of $c_{(s,a_1,a_2)}(t)$ corresponds to the *minimum expected time* at which a_1 and a_2 cross, and therefore represents exactly the information required for estimating $d(s,a)$. However, ϕ^π (and therefore c as well) is hard to describe analytically since the precise understanding of how Q-values evolve requires knowing the structure of M and of the learning algorithm. Although we do not have a closed form for ϕ^π, we can use a Taylor expansion around the time of the *last* sample experience, $t-1$:

$$\hat{\phi}^\pi_{(s,a)}(t) \approx \phi^\pi_{(s,a)}(t-1) + \frac{\partial \phi^\pi_{(s,a)}(t-1)}{\partial t}. \tag{5}$$

We expand the series around the time of the last experience since we need to approximate the terms in Equation 5 by using sampled values; it should be clear that any statistics of interest will be the most accurate if we allow the use of all $t-1$ samples observed so far. Also, note that we *do*

have unbiased samples for both terms in Equation 5: a sample of $\phi^{\pi}_{(s,a)}(t-1)$ is simply $Q^{t-1}_{\pi}(s,a)$, and a sample of $\frac{\partial \phi^{\pi}_{(s,a)}(t-1)}{\partial t}$ is $\alpha_M \delta_{(s,a)}(l)$, where $\delta_{(s,a)}(l)$ is the TD error[1] for the last time $Q_\pi(s,a)$ was updated, at time l; α_M is the learning rate used in L. For any given π, these are unbiased estimators: $Q^{t-1}_{\pi}(s,a)$, directly because of the definition of $\phi^{\pi}_{(s,a)}(t-1)$; and $\alpha_M \delta_{(s,a)}(l)$, by a similar argument and by noticing that *(1)* it can computed by subtracting consecutive Q-values, and *(2)* expectation is a linear operator. Better, lower-variance estimates of the derivative of $\phi^{\pi}_{(s,a)}(t)$ can be obtained and are useful in highly stochastic problems: one could estimate them via finite differences, by averaging past updates to the Q-function, or by propagating updates to other Q-values through a model. In what follows, we use just the simplest estimates possible, as described above, and instantiate a model-free version of $\Delta\pi$ called TD(0)-$\Delta\pi$.

Proposition 2 *A local linear approximation to $\phi^{\pi}(s,\cdot)(t)$ induces a family of approximations for the functions $c_{(s,\cdot,\cdot)}(t)$, whose* smallest roots *correspond to approximations of the minimum expected time until a crossing between any pair of actions.*

Proposition 2 follows from simple geometric reasoning based on $\hat{\phi}^{\pi}$ being a linear approximation. Specifically, we can show that a local linear approximation to the expected time until the value of an action a crosses the value of the one currently considered optimal, $\hat{a}^*$, for some $s \in S$, is:

$$d(s,a) = \left(\frac{1}{\alpha_M}\right) \frac{Q^t(s,\hat{a}^*) - Q^t(s,a)}{\delta_{(s,a)}(T_{s,a}) - \delta_{(s,\hat{a}^*)}(T_{s,\hat{a}^*})} \quad (6)$$
$$\approx \arg\min_t \left(c_{(s,a,\hat{a}^*)}(t) = 0\right)$$

where T_{s,a_i} is the last time at which $Q(s,a_i)$ was updated. $d(s,a)$ is a valid approximation unless its denominator is zero, which occurs if both Q-values seem to be changing at the same rate — in this case, it correctly concludes that no crossings are expected. Note also how it implements the type of policy sensitivity indicator described in Section 3.

6 Experiments

We now compare our approach to other algorithms for efficient exploration. Our main comparisons are with ΔV (Şimşek and Barto 2006) and Delayed Q-Learning (DQL) (Strehl et al. 2006). ΔV is a principled, model-free way of finding efficient, purely-exploratory policies. DQL is, to the best of our knowledge, the model-free technique with best PAC-MDP bounds, and provably performs near-optimally in all but a polynomial number of timesteps. We also compare with two baseline algorithms: *(1)* a Constant-Penalty (CP) technique, which gives small penalties to each visited state and thus implements a *least-visited* strategy (Thrun 1992); and *(2)* ϵ-greedy Q-Learning (QL), for several values of ϵ; this includes random exploration ($\epsilon = 1$).

[1]TD errors are not required, though; any observed difference between consecutive estimates of a Q-value suffice.

Algorithm 1 TD(0)-$\Delta\pi$

for all (s,a) **do**
  $Q^0_{exploit}(s,a) \leftarrow 0$; $Q^0_{explore}(s,a) \leftarrow 0$;
  $\delta_{(s,a)}(0) \leftarrow 0$; $T_{s,a} \leftarrow 0$; visited$(s,a) \leftarrow False$
end for
for $t = 1, 2, 3, \ldots,$ **do**
  Let s_t be state of M at time t
  Choose action $a_t := \arg\max_{a' \in A_M} Q^t_{explore}(s,a')$
  Take a_t in M, observe reward r^t_M, next state s'
  Let $\hat{a}^* := \arg\max_{a' \in A_M} Q^t_{exploit}(s,a')$
  if not visited(s_t,a_t) or not visited$(s_t,\hat{a}^*)$ **then**
    $r(s_t,a_t) := 1$
  else
    if $|\delta_{(s_t,a_t)}(T_{s_t,a_t}) - \delta_{(s_t,\hat{a}^*)}(T_{s_t,\hat{a}^*})| < \lambda$ **then**
      $r(s_t,a_t) := -p$
    else
      Compute $r(s_t,a_t)$ according to Eq. 1 and 6
    end if
  end if
  $Q^{t+1}_{exploit} \leftarrow L(s^t, a_t, r^t_M, s', Q^t_{exploit}, \rho_{\mathbf{exploit}})$
  $Q^{t+1}_{explore} \leftarrow L(s^t, a_t, r(s_t,a_t), s', Q^t_{explore}, \rho_{\mathbf{explore}})$
  $T_{s_t,a_t} \leftarrow t$; visited$(s_t,a_t) \leftarrow True$;
  $\delta_{s_t,a_t}(t) \leftarrow Q^{t+1}(s_t,a_t) - Q^t(s_t,a_t)$;
end for

The first domain in which we evaluate TD(0)-$\Delta\pi$ consists of a simple discrete 25×25 maze with four exits. The four usual actions are available (N,S,E,W), and each has a 0.9 probability of taking the agent to the intended direction, and 0.1 of taking it to another uniform random direction. Rewards are -0.001 for each action, and 1, 2 or 5 when transitioning into one of the terminal states. Q-functions in Algorithm 1 are learned using Q-Learning with learning rate $\alpha = 0.1$ and discount rate $\gamma = 0.99$.

Results for the value of the learned exploitation policy as a function of the amount of exploration allowed are shown in Figure 2 and 3, and are averages over 20 runs. Both our approach, ΔV and DQL perform significantly better than the baseline algorithms. We searched the space of values of ϵ, for QL, and present only some sample results. ΔV initally performs better than our approach, mainly because the random walk it performs during its initial phase finds one of the goals faster; however, a closer look reveals that it gets "obsessed" with fine-tuning the value of states even when the policy for reaching them is already correct. At this moment, on the other hand, TD(0)-$\Delta\pi$ notices that no other policy changes are expected and proceeds to other regions of the state space. TD(0)-$\Delta\pi$ finds the optimal exploitation policy almost 100,000 steps before ΔV. DQL takes even longer to learn: in principle it requires (for this domain) $m \approx$*1 billion* samples before updating the value of any given state–action pair, in order for its bounds to guarantee convergence in polynomial time. In our experiments we used more reasonable values for m, which removed its PAC-MDP properties but made it comparable to other approaches. DQL's bounds also require Q-values to be ini-

tialized optimistically, which for this domain means setting $Q_0(s,a) = 500$. However, we noticed that only values of $Q_0(s,a) \leq 9$ were capable of generating reasonable learning curves. Furthermore, the only way we could make DQL perform similarly to TD(0)-$\Delta\pi$ was to initialize its Q-values faily close to the *optimal* ones, and even then it became stuck in a local minimum 20% less efficient than the optimal exploitation policy. We searched the space of parameters of DQL to make it perform as well as possible; a representative sample of the learning curves is shown in Figure 3.

The second domain in which we evaluate TD(0)-$\Delta\pi$ is a rod positioning task (Moore and Atkeson 1993), which consists of a discretized space containing a rod, obstacles, and a target. The goal is to maneuver the rod by moving its base and angle of orientation so that its tip touches the target, while avoiding obstacles. We discretize the state space into unit x and y coordinates and 10° angle increments; actions move the rod's base one unit in either direction along its axis or perform a 10° rotation in either direction. Rewards are -1 for each action and 1000 when the tip of the rod touches the goal. We used the same learning method and parameters as in the previous domain. Results for the value of the learned exploitation policy as a function of the amount of exploration allowed are shown in Figures 4 and 5. $\Delta\pi$ again performed better than other methods; interestingly, simple approaches like ϵ-greedy QL and CP performed *better* than specialized ones such as ΔV and DQL — the reason being that this domain contains only *one* source of positive reward, which, when found, can be aggressively exploited without risking overlooking others. ΔV again kept fine-tuning the value function even when the policy was already correct, and often got stuck in local minima 25% worse than the optimal exploitation policy. $\Delta\pi$, on the other hand, explored a region only while it had evidence that the policy could still change. We searched the space of parameters of DQL to optimize its performance; representative learning curves are shown in Figure 5. DQL only performs well if initialized with a Q-function fairly close to the optimal and if m is set much lower than required to guarantee its PAC-MDP bounds. After 1.8 million timesteps, it learned an exploitation policy 50% worse than the optimal one.

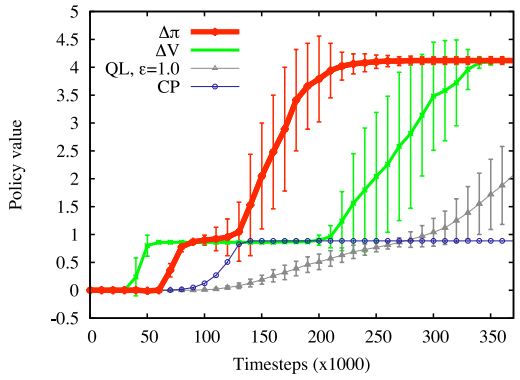

Figure 2: Performance in the maze domain.

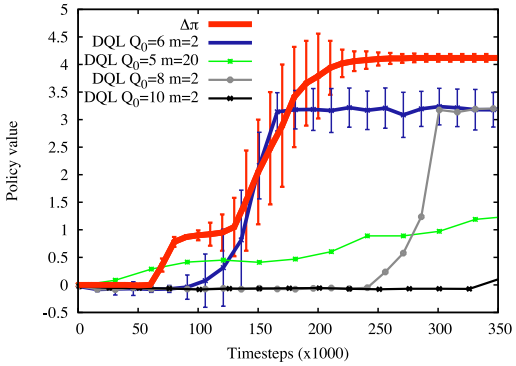

Figure 3: Performance in the maze (vs. DQL).

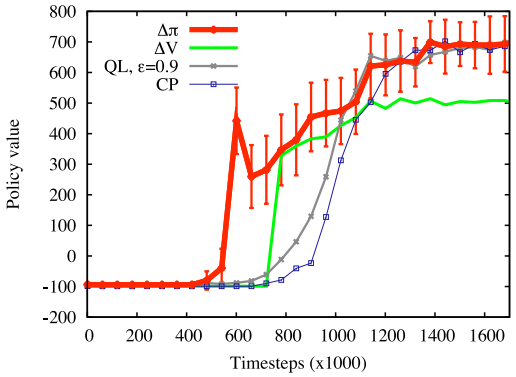

Figure 4: Performance in the rod positioning domain.

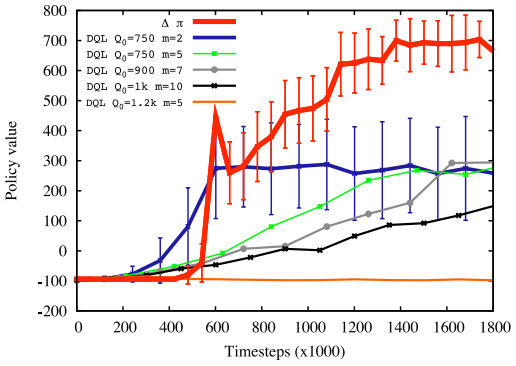

Figure 5: Performance in the rod domain (vs. DQL).

7 Discussion and Conclusions

We have presented a derivation of a local linear approximation to the expected time until a policy change and used it to construct an efficient, model-free exploration technique. The specific approximation used might have practical shortcomings. It is possible, for instance, to construct MDPs in which TD(0)-$\Delta\pi$ performs poorly by initializing it in a region of the state space where many crossings are likely to occur but which is not part of any optimal trajectory. We believe, however, that these cases are not common in practice. In fact, ΔV seems much more sensitive to small changes in

the formulation of the MDP, since simply rescaling the reward function can make it perform arbitrarily slowly. DQL, even with provably polynomial sample complexity, is a good example of how such guarantees don't necessarily correspond to algorithms that are feasible in practice. For future work, we would like to study model-based estimations of Equation 5, which could have lower variance. We also believe there might be a relevant connection between Equation 6 and Advantage functions, and that PAC-MDP bounds can be obtained. Finally, an interesting open problem is that of deciding when to safely terminate the exploration process.

References

Abbeel, P., and Ng, A. 2005. Exploration and apprenticeship learning in reinforcement learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, 1–8. New York, NY, USA: ACM.

Bertsekas, D., and Tsitsiklis, J. N. 1991. An analysis of stochastic shortest path problems. *Mathematics of Operations Research* 16(3):580–595.

Brafman, R., and Tennenholtz, M. 2001. R-MAX - A general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research* 3:213–231.

Şimşek, O., and Barto, A. 2006. An intrinsic reward mechanism for efficient exploration. In *Proceedings of the 23rd International Conference on Machine learning (ICML 2006)*, 833–840. New York, NY, USA: ACM.

Dearden, R.; Friedman, N.; and Andre, D. 1999. Model based bayesian exploration. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI 1999)*, 150–159. Stockholm, Sweden: Morgan Kaufmann.

Duff, M. 2003. Design for an optimal probe. In *Proceedings of the 20th International Conference on Machine learning (ICML 2003)*, 131–138. AAAI Press.

Epshteyn, A.; Vogel, A.; and DeJong, G. 2008. Active reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, 296–303. Omnipress.

Kearns, M., and Singh, S. 1998. Near-optimal reinforcement learning in polynominal time. In *Proceedings of the 15th International Conference on Machine Learning (ICML 1998)*, 260–268. Morgan Kaufmann.

Kolter, J., and Ng, A. 2009. Near-Bayesian exploration in polynomial time. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, 513–520. Omnipress.

Moore, A., and Atkeson, C. 1993. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning* 13:103–130.

Strehl, A.; Li, L.; Wiewiora, E.; Langford, J.; and Littman, M. 2006. PAC model-free reinforcement learning. In *Proceedings of the 23rd International Conference on Machine learning (ICML 2006)*, 881–888. New York, NY, USA: ACM.

Thrun, S. 1992. Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie Mellon University, Pittsburgh, PA, USA.