PAC Optimal Planning for Invasive Species Management: Improved Exploration for Reinforcement Learning from Simulator-Defined MDPs

Thomas G. Dietterich **Majid Alkaee Taleghan** **Mark Crowley**
School of Electrical Engineering and Computer Science
Oregon State University
Corvallis, OR 97331
tgd,alkaee,crowley@eecs.oregonstate.edu

Abstract

Often the most practical way to define a Markov Decision Process (MDP) is as a simulator that, given a state and an action, produces a resulting state and immediate reward sampled from the corresponding distributions. Simulators in natural resource management can be very expensive to execute, so that the time required to solve such MDPs is dominated by the number of calls to the simulator. This paper presents an algorithm, DDV, that combines improved confidence intervals on the Q values (as in interval estimation) with a novel upper bound on the discounted state occupancy probabilities to intelligently choose state-action pairs to explore. We prove that this algorithm terminates with a policy whose value is within ε of the optimal policy (with probability $1-\delta$) after making only polynomially-many calls to the simulator. Experiments on one benchmark MDP and on an MDP for invasive species management show very large reductions in the number of simulator calls required.

Introduction

Consider a Markov Decision Process in which the number of states and actions is small enough that a tabular representation of the MDP action-value function fits in memory, but where the transition function is not provided in explicit form. Instead, it can only be sampled from an expensive simulator. In such cases, we desire an MDP planning algorithm that minimizes the number of calls to the simulator, runs in time polynomial in the relevant problem parameters, and outputs a policy that is approximately optimal with high probability.

Our motivating application involves optimal management of a river system to control an invasive plant species, Tamarisk. A native of the Middle East, Tamarisk has become an invasive plant in the dryland rivers and streams of the western US (DiTomaso and Bell 1996; Stenquist 1996). It out-competes native vegetation through its thirst for water and by introducing salt into the soil. Given a Tamarisk invasion, a land manager must decide how and where to fight the invasion (e.g., eradicate Tamarisk plants? plant native plants? upstream? downstream?). Although approximate or heuristic solutions to this problem would be useful, our collaborating economists tell us that our policy recommendations will carry more weight if they are provably optimal with high probability.

The transition function for Tamarisk involves several phases. First the chosen treatments are applied (which may kill some Tamarisk trees). Then each tree (native or Tamarisk) may die according to a standard mortality probability. If the tree does not die, it then produces a stochastic number of seeds. These disperse (preferentially downstream). Multiple seeds arriving at the same location then compete to take root and become established. Each of these steps is easy to specify in the form of a stochastic simulation (e.g., as a dynamic Bayesian network), but exact computation of the transition probabilities is intractable, even for a single state-action pair.

A large problem instance involves 4.7×10^6 states with 2187 actions in each state. On a modern 64-bit machine, the action-value function for this problem can fit into main memory. However, computing the full transition function to sufficient accuracy to support standard value iteration requires on the order of 3×10^{20} simulator calls.

To minimize the number of simulator calls, we introduce a model-based reinforcement learning algorithm called DDV (a Romanization of $\Delta\Delta V$). Our algorithm incorporates two innovations. First we develop an improved confidence interval for the multinomial transition probabilities. This interval provides tighter estimates when the transition function is sparse, which permits us to stop sampling earlier than previous methods. Second, we introduce a new exploration heuristic that chooses which state-action pair to sample based on an approximate value of information calculation. Specifically, DDV computes a confidence interval on the value of the start state and chooses a state-action pair (s,a) to explore based on an estimate of the degree to which exploring (s,a) will shrink this confidence interval. That estimate is based on a novel upper bound on the *occupancy measure* of the MDP. The occupancy measure for a state in the MDP is the (discounted) probability that the optimal policy will occupy that state (summed over all time) given that it starts in a designated state. Our key observation is that the impact of exploring (s,a) on the confidence interval at the start state can be estimated from the impact on state s of exploring (s,a) multiplied by the occupancy measure.

This paper provides evidence to support two claims:

(a) DDV requires fewer samples than existing methods to achieve good performance and (b) DDV terminates with a policy that is approximately optimal with high probability after only polynomially-many calls to the simulator.

Definitions

We employ the standard formulation of an infinite horizon discounted Markov Decision Process (MDP) with a designated start state distribution (Bellman 1957; Puterman 1994) $\mathscr{M} = \langle S, A, P, R, \gamma, P_0 \rangle$. S is a finite set of states of world; A is a finite set of possible actions that can be taken in each state; $P : S \times A \times S \mapsto [0,1]$ is the conditional probability of entering state s' when action a is executed in state s; $R(s,a)$ is the (deterministic) reward received after performing action a in state s; $\gamma \in (0,1)$ is the discount factor, and P_0 is the distribution over starting states. It is convenient to define a special starting state s_0 and action a_0 and define $P(s|s_0,a_0) = P_0(s)$ and $R(s_0,a_0) = 0$. We assume that $0 \leq R(s,a) \leq R_{max}$. Generalization of our algorithm to (bounded) stochastic rewards is straightforward.

A *strong simulator* (also called a *generative model*) is a function $F : S \times A \mapsto S \times \Re$ that given (s,a) returns (s',r) where s' is sampled according to $P(s'|s,a)$ and $r = R(s,a)$.

A (deterministic) policy is a function from states to actions, $\pi : S \mapsto A$. The value of a policy π at the starting state is defined as $V^{\pi}(s_0) = \mathbb{E}[\sum_{t=1}^{\infty} \gamma^t R(s_t, \pi(s_t))]$, where the expectation is taken with respect to the stochastic transitions. The maximum possible $V^{\pi}(s_0)$ is denoted $V_{max} = R_{max}/(1-\gamma)$. An optimal policy π^* maximizes $V^{\pi}(s_0)$, and the corresponding value is denoted by $V^*(s_0)$. The action-value of state s and action a under policy π is defined as $Q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^{\pi}(s')$. The optimal action-value is denoted $Q^*(s,a)$.

Define $pred(s)$ to be the set of states s^- such that $P(s|s^-,a) > 0$ for at least one action a.

Definition 1. *(Fiechter 1994). A learning algorithm is PAC-RL if for any discounted MDP (S,A,P,R,γ,P_0), $\varepsilon > 0$, $1 > \delta > 0$, and $0 \leq \gamma < 1$, the algorithm halts and outputs a policy π such that*

$$\mathbb{P}[|V^*(s_0) - V^{\pi}(s_0)| \leq \varepsilon] \geq 1 - \delta,$$

in time polynomial in $|S|$, $|A|$, $1/\varepsilon$, $1/\delta$, $1/(1-\gamma)$, and R_{max}.

The *occupancy measure* $\mu^*(s)$ is the expected discounted number of times that the optimal policy π^* visits state s,

$$\mu^*(s) = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t I[s_t = s] \,\middle|\, s_0, \pi^* \right], \qquad (1)$$

where $I[\cdot]$ is the indicator function and the expectation taken is with respect to the transition distribution. This can be computed via dynamic programming on the Bellman flow equation (Syed, Bowling, and Schapire 2008): $\mu^*(s) = P_0(s) + \gamma \sum_{s^-} \mu(s^-) P(s|s^-, \pi^*(s^-))$. This says that the (discounted) probability of visiting state s is equal to the sum of the probability of starting in state s (as specified by the starting state distribution $P_0(s)$) and the probability of reaching s by first visiting state s^- and then executing an action that leads to state s.

As a learning algorithm explores the MDP, it collects the following statistics. Let $N(s,a)$ be the number of times the simulator has been called with state-action pair (s,a). Let $N(s,a,s')$ be the number of times that s' has been observed as the result. Let $R(s,a)$ be the observed reward.

Previous work on sample-efficient MDP Planning

Fiechter (1994) first introduced the notion of PAC reinforcement learning in Definition 1 and presented a PAC-RL algorithm. His algorithm assumed a discounted MDP with a "reset" action available during learning, so that the learner could return to the start state at any point. This matches our application setting where an ecosystem is in state s_0 now, and we seek a good policy for managing it for the coming years.

Subsequent work criticized Fiechter's approach because it posits a separation between "explore time" and "exploit time". Within the reinforcement learning framework, virtually all subsequent work has focused on algorithms that manage the exploit-explore tradeoff. These algorithms all follow a single (potentially-infinite) trajectory and seek to converge to optimal *behavior*. Optimality has been defined in three different ways.

Under the first definition, the algorithm announces (at some time t^* in state s_{t^*}) that it has found a policy $\hat{\pi}$ such that from this point forward, the expected return $V^{\hat{\pi}}(s_{t^*})$ is within ε of the optimal return $V^*(s_{t^*})$ with probability $1-\delta$. Examples of this approach include the Explicit Explore-Exploit (E^3) algorithm of Kearns and Singh (1998) and the R_{max} algorithm of Brafman, et al. (2003).

The second definition of optimal behavior is in terms of the total (infinite horizon) regret of the algorithm, which is the difference between the reward that would have been received using the optimal policy and the reward that is actually received by the algorithm as it explores and exploits. Jaksch, Ortner, and Auer (2010) introduce UCRL2, which achieves a total regret of no more than $34D|S|\sqrt{|A|t\log\frac{t}{\delta}}$ for any time $t > 1$, where D is a parameter, called the "diameter" of the MDP, that quantifies the difficulty of getting from any state s_i to any other state s_j.

The third definition of optimal behavior is in terms of the total number of times that the algorithm executes a non-ε-optimal action (Kakade 2003). Model-based Interval Estimation (MBIE) (Strehl and Littman 2008) introduces a practical algorithm satisfying Kakade's optimality criterion.

From the perspective of sustainability problems, all of these critera based on optimal behavior are not useful. The key difficulty is that in most cases, it is likely that the starting state s_0 is a transient state that we expect (indeed, we hope) will not be visited again if the ecosystem is being managed optimally. For example, in our invasive species problem, s_0 will correspond to a situation in which an invading species is spreading rapidly, whereas we hope that the optimal policy can eradicate the species and quickly eliminate it if it re-appears. Because s_0 is a transient state, algorithms that follow a single trajectory will not revisit it very often, and hence, will not learn how to behave ε-optimally in that state.

Although virtually all work in model-based reinforcement learning does not address transient-state behavior, many important innovations have been made since Fiechter's 1994 paper. The goal of the present paper is to build on these innovations to develop a "next generation" algorithm for model-based MDP planning for a designated start state.

Building Blocks

This section provides the definitions, results, and algorithm components needed to define DDV.

Value Iteration for Confidence Intervals

Suppose we have collected a sample of $\langle s_t, a_t, r_t, s_{t+1} \rangle$ data points and recorded the results using $N(s,a,s')$ and $R(s,a)$. MBIE employs the following confidence interval for multinomial distributions, introduced by Weissman, et al. (2003). Let $\hat{P}(s'|s,a) = N(s,a,s')/N(s,a)$ be the maximum likelihood estimate for $P(s'|s,a)$, and let $\hat{P}$ and $\tilde{P}$ denote $\hat{P}(\cdot|s,a)$ and $\tilde{P}(\cdot|s,a)$. Let

$$CI(\hat{P}|N(s,a),\delta) = \left\{ \tilde{P} \;\middle|\; \|\tilde{P} - \hat{P}\|_1 \leq \omega(N(s,a),\delta) \right\}, \quad (2)$$

(where $\|\cdot\|_1$ is the L_1 norm and $\omega(N(s,a),\delta) = \sqrt{\frac{2[\ln(2^{|S|}-2) - \ln\delta]}{N(s,a)}}$) denote a set of probability distributions. Weismann et al. (2003) prove that with probability $1-\delta$, $P(\cdot|s,a) \in CI(\hat{P}(\cdot|s,a)|N(s,a),\delta)$. The confidence interval is an L_1 "ball" of radius $\omega(N(s,a),\delta)$ around the maximum likelihood estimate for P.

Given confidence intervals for all (s,a), we wish to obtain confidence intervals on the Q and V values for the optimal value function of the MDP. For any state where $N(s,a) = 0$, define $Q_{lower}(s,a) = 0$ and $Q_{upper}(s,a) = V_{max}$. Then, generalizing Strehl and Littman (2008), we perform the following Q iterations to convergence:

$$Q_{upper}(s,a) = R(s,a) + \max_{\tilde{P}(s,a) \in CI(P(s,a),\delta_1)} \gamma \sum_{s'} \tilde{P}(s'|s,a) \max_{a'} Q_{upper}(s',a') \quad (3)$$

$$Q_{lower}(s,a) = R(s,a) + \min_{\tilde{P}(s,a) \in CI(P(s,a),\delta_1)} \gamma \sum_{s'} \tilde{P}(s'|s,a) \max_{a'} Q_{lower}(s',a') \quad (4)$$

At convergence, define $V_{upper}(s) = \max_a Q_{upper}(s,a)$ and $V_{lower}(s) = \max_a Q_{lower}(s,a)$.

Proposition 1. *If $\delta_1 = \delta/(|S||A|)$, then with probability $1-\delta$, $Q_{lower} \leq Q^*(s,a) \leq Q_{upper}$ for all (s,a) and $V_{lower} \leq V^*(s) \leq V_{upper}$ for all s.*

Proof. Strehl and Littman (2008) prove that (3) is a contraction mapping, and hence, value iteration will converge. By symmetry, the same holds for (4). Dividing δ by $|S||A|$ ensures that the $|S||A|$ separate confidence intervals hold simultaneously with probability $1-\delta$. The result follows. □

Strehl and Littman provide Algorithm UPPERP (Algorithm 1) for solving the optimization over $CI(P(s,a),\delta_1)$ in (3) and (4) efficiently. Let us consider (3). If the radius of the confidence interval is ω, then we can solve for $\tilde{P}$ by shifting $\Delta\omega = \omega/2$ of the probability mass from outcomes s' for which $V_{upper}(s') = \max_{a'} Q_{upper}(s',a')$ is low ("donor states") to outcomes for which it is maximum ("recipient states"). This will result in creating a $\tilde{P}$ distribution that is at L_1 distance ω from $\hat{P}$. The algorithm repeatedly finds a pair of successor states $\underline{s}$ and $\overline{s}$ and shifts probability from one to the other until it has shifted $\Delta\omega$.

Note that in most cases, $\overline{s}$ will be a state for which $N(s,a,\overline{s}) = 0$—that is, a state we have never visited. In such cases, $V_{upper}(\overline{s}) = V_{max}$.

An analogous algorithm solves (4).

Algorithm 1: UPPERP(s,a,δ,M_0)

Input: s,a
δ: Confidence parameter
M_0: missing mass limit
Lines marked by **GT:** are for the Good-Turing extension
$N(s,a) := \sum_{s'} N(s,a,s')$
$\hat{P}(s'|s,a) := N(s,a,s')/N(s,a)$
$\tilde{P}(s'|s,a) := \hat{P}(s'|s,a)$
$\Delta\omega := \omega(N(s,a),\delta)/2$
GT: $N_0(s,a) := \{s' | N(s,a,s') = 0\}$
while $\Delta\omega > 0$ **do**
  $S' := \{s' : \hat{P}(s'|s,a) < 1\}$ recipient states
  GT: **if** $M_0 = 0$ **then** $S' := S' \setminus M(s,a)$
  $\underline{s} := \text{argmin}_{s' : \tilde{P}(s'|s,a) > 0} V_{upper}(s')$ donor state
  $\overline{s} := \text{argmax}_{s' \in S', \tilde{P}(s'|s,a) < 1)} V_{upper}(s')$ recipient state
  $\xi := \min\{1 - \tilde{P}(\overline{s}|s,a), \tilde{P}(\underline{s}|s,a), \Delta\omega\}$
  $\tilde{P}(\underline{s}|s,a) := \tilde{P}(\underline{s}|s,a) - \xi$
  $\tilde{P}(\overline{s}|s,a) := \tilde{P}(\overline{s}|s,a) + \xi$
  $\Delta\omega := \Delta\omega - \xi$
  GT: **if** $\overline{s} \in N_0(s,a)$ **then** $M_0 := M_0 - \xi$
return $\tilde{P}$

An Improved Confidence Interval for Sparse MDPs

A drawback of the Weissman et al. confidence interval is that $\omega(N,\delta)$ scales as $O(\sqrt{|S|/N})$, so the intervals are very wide for large state spaces. In many real-world MDPs, the transition probability distributions are sparse in the sense that there are only a few states s' such that $P(s'|s,a) > 0$. We would like a tighter confidence interval for sparse distributions.

Our approach to this is to intersect the Weissman et al. confidence interval with a confidence interval based on the Good-Turing estimates of the missing mass (Good 1953). For a given state-action pair (s,a), let $N_k(s,a) = \{s' | N(s,a,s') = k\}$ be the set of all result states s' that have been observed exactly k times. We seek to bound the total probability of those states that have never been observed: $M_0(s,a) = \sum_{s' \in N_0} P(s'|s,a)$. The Good-Turing estimate of $M_0(s,a)$ is

$$\widehat{M}_0(s,a) = \frac{|N_1(s,a)|}{N(s,a)}.$$

In words, Good and Turing count the number of successor states that have been observed exactly once and divide by

the number of samples.

Proposition 2. *With probability* $1-\delta$,

$$M_0(s,a) \leq \widehat{M}_0(s,a) + (1+\sqrt{2})\sqrt{\frac{\ln(1/\delta)}{N(s,a)}}. \quad (5)$$

Proof. Let $S(M_0(s,a),x)$ be the Chernoff "entropy", defined as

$$S(M_0(s,a),x) = \sup_{\beta} x\beta - \ln Z(M_0(s,a),\beta),$$

where $Z(M_0(s,a),\beta) = \mathbb{E}[e^{\beta M_0(s,a)}]$. McAllester and Ortiz (2003) prove (Theorem 16) that

$$S(M_0(s,a),\mathbb{E}[M_0(s,a)]+\varepsilon) \geq N(s,a)\varepsilon^2.$$

From Lemmas 12 and 13 of Kearns and Saul (1998),

$$\mathbb{E}[M_0(s,a)] \leq \widehat{M}_0(s,a) + \sqrt{\frac{2\log 1/\delta}{N(s,a)}}.$$

Combining these results yields

$$S\left(M_0(s,a), \widehat{M}_0(s,a) + \sqrt{\frac{2\log 1/\delta}{N(s,a)}} + \varepsilon\right) \geq N(s,a)\varepsilon^2. \quad (6)$$

Chernoff (1952) proves that

$$\mathbb{P}(M_0(s,a) \geq x) \leq e^{-S(M_0(s,a),x)}.$$

Plugging in (6) gives

$$P\left(M_0(s,a) \geq \widehat{M}_0(s,a) + \sqrt{\frac{2\log 1/\delta}{N(s,a)}} + \varepsilon\right) \leq e^{-N(s,a)\varepsilon^2}. \quad (7)$$

Setting $\delta = e^{-N(s,a)\varepsilon^2}$ and solving for ε gives $\varepsilon = \sqrt{(\log 1/\delta)/N(s,a)}$. Plugging this into (7) and simplifying gives the result. □

Define $CI^{GT}(\hat{P}|N(s,a),\delta)$ to be the set of all distributions $\tilde{P} \in CI(\hat{P}|N(s,a),\delta/2)$ such that $\sum_{s' \in N_0(s,a)} \tilde{P}(s'|s,a) < \widehat{M}_0(s,a) + (1+\sqrt{2})\sqrt{\frac{\ln(2/\delta)}{N(s,a)}}$.

We can incorporate the bound from (5) into UpperP by adding the lines prefixed by "GT:" in Algorithm 1. These limit the amount of probability that can be shifted to unobserved states according to (5). Note that since we are intersecting two confidence intervals, we must compute both (2) and (5) using $\delta/2$ so that they will simultaneously hold with probability $1-\delta$.

An Upper Bound on the Occupancy Measure

The exploration heuristic for DDV is based on an occupancy measure μ_{upper}. This section defines this measure and presents a dynamic programming algorithm to compute it.

At each point during the execution of DDV, the states S of the unknown MDP can be partitioned into three sets: (a) the *unobserved states* s (i.e., $N(s^-,a^-,s) = 0$ for all s^-,a^-); (b) the *observed but unexplored states* s (i.e., $\exists(s^-,a^-)N(s^-,a^-,s) > 0$ but $N(s,a) = 0$ for all a), and (c) *the (partially) explored states* s (i.e., $N(s,a,s') > 0$ for some a). Consider the set $\widetilde{\mathscr{M}} = \langle \tilde{S}, \tilde{A}, \tilde{T}, \tilde{R}, s_0 \rangle$ of MDPs satisfying the following properties:

- $\tilde{S}$ consists of all states s that have been either observed or explored,
- $\tilde{A} = A$, the set of actions in the unknown MDP,
- $\tilde{T}$ consists of any transition function T such that for explored states s and all actions a, $T(s,a,\cdot) \in CI^{GT}(\hat{P}(s,a),\delta)$. For all observed but not explored states s, $T(s,a,s) = 1$ for all a, so they enter self-loops.
- $\tilde{R}$: For explored (s,a) pairs, $\tilde{R}(s,a) = R(s,a)$. For unexplored (s,a) pairs, $\tilde{R}(s,a) \in [0, R_{max}]$.
- s_0 is the artificial start state s_0.

$\widetilde{\mathscr{M}}$ contains all MDPs consistent with the observations with the following restrictions. First, the MDPs do not contain any of the unobserved states. Second, the unexplored states contain self-loops and so do not transition to any other states.

Define $P_{upper}(s'|s,a)$ as follows:

$$P_{upper}(s'|s,a) = \max_{\tilde{P}(s,a) \in CI^{GT}(P,\delta)} \tilde{P}(s'|s,a).$$

Define μ_{upper} as the solution to the following dynamic program. For all states s,

$$\mu_{upper}(s) = \sum_{s^- \in pred(s)} \max_{a^-} \gamma P_{upper}(s|s^-,a^-)\mu_{upper}(s^-). \quad (8)$$

The intuition is that we allow each predecessor s^- of s to choose the action a^- that would send the most probability mass to s and hence give the biggest value of $\mu_{upper}(s)$. These action choices a^- do not need to be consistent for multiple successors of s^-. We fix $\mu_{upper}(s_0) = \mu(s_0) = 1$. (Recall, that s_0 is an artificial start state. It is not reachable from any other state—including itself—so $\mu(s_0) = 1$ for all policies.)

Proposition 3. *For all MDPs* $\widetilde{M} \in \widetilde{\mathscr{M}}$, $\mu_{upper}(s) \geq \mu^{\pi^*(\widetilde{M})}(s)$, *where* $\pi^*(\widetilde{M})$ *is any optimal policy of* $\widetilde{M}$.

Proof. By construction, $P_{upper}(s'|s,a)$ is the maximum over all transition distributions in $\widetilde{\mathscr{M}}$ of the probability of $(s,a) \to s'$. And according to (8), the probability flowing to s is the maximum possible over all policies executed in the predecessor states $\{s^-\}$. Finally, all probability reaching a state s must come from its known predecessors $pred(s)$, because all observed but unexplored states only have self-transitions and hence cannot reach s or any of its predecessors. □

In earlier work, Smith and Simmons (2006) employed a less general path-specific bound on μ as a heuristic for focusing Real-Time Dynamic Programming (a method that assumes a full model of the MDP is available).

The DDV Algorithm

Algorithm 2 presents the pseudo-code for DDV. In each iteration, DDV performs dynamic programming to update the bounds on Q, V, and μ. If $\Delta V(s_0) < \varepsilon$, then it computes the optimal policy based on the maximum likelihood estimate of the MDP and exits. Otherwise, it computes

Algorithm 2: DDV $(s_0, \gamma, F, \varepsilon, \delta)$

Input: s_0:start state
γ: discount rate
F: a simulator
ε, δ: accuracy and confidence parameters
$\tilde{S} = \{s_0\}$ // observed and/or explored states
$N(s,a,s') = 0$ for all (s,a,s')
repeat forever
  update Q_{upper}, Q_{lower}, V_{upper}, V_{lower}, μ_{upper} by iterating equations 3,4, and 8 to convergence
  if $V_{upper}(s_0) - V_{lower}(s_0) \leq \varepsilon$ **then**
    // compute optimal policy and terminate
    compute $\hat{Q}$
    $\hat{\pi}(s) = \arg\max_a \hat{Q}(s,a)$
    return $\hat{\pi}$
  forall the *explored or observed states s* **do**
    forall the *actions a* **do**
      compute $Q'_{upper}(s,a)$ and $Q'_{lower}(s,a)$ (see text)
      $\Delta\Delta Q(s,a) := [Q_{upper}(s,a) - Q_{lower}(s,a)] - [Q'_{upper}(s,a) - Q'_{lower}(s,a)]$
      $\Delta\Delta V(s_0|s,a) := \mu_{upper}(s)\Delta\Delta Q(s,a)$
  $(s,a) := \text{argmax}_{(s,a)} \Delta\Delta V(s_0|s,a)$
  $(s,a,r,s') \sim F(s,a)$ // draw sample
  $\tilde{S} := \tilde{S} \cup \{s'\}$
  update $N(s,a,s')$, $N(s,a)$, and $R(s,a)$

$\Delta\Delta Q(s,a)$ for each (s,a) and chooses the (s,a) that maximizes $\Delta\Delta V(s_0|s,a)$. Then it draws a sample from $P(s'|s,a)$. To make the dynamic programming efficient, we employ prioritization methods (Wingate and Seppi 2006).

To compute $\Delta\Delta Q(s,a)$, we must compute the expected values $Q_{upper}(s,a)$ and $Q_{lower}(s,a)$ after drawing one more sample (s,a,r,s'). We denote these by $Q'_{upper}(s,a)$ and $Q'_{lower}(s,a)$. We consider two cases.

Case 1: $N(s,a) = 0$. In this case, our current bounds are $Q_{lower}(s,a) = 0$ and $Q_{upper}(s,a) = V_{max}$. After we sample (s,a,r,s'), we will observe the actual reward $R(s,a) = r$ and we will observe one of the possible successor states s'. For purposes of deriving our heuristic, we will assume a uniform prior on $R(s,a)$ so that the expected value of R is $\overline{R} = R_{max}/2$. We will assume that s' will be a "new" state that we have never observed before, and hence $V_{upper}(s') = V_{max}$ and $V_{lower}(s') = 0$. This gives us

$$Q'_{upper}(s,a) = \overline{R}(s,a) + \gamma R_{max}/(1-\gamma) \quad (9a)$$

$$Q'_{lower}(s,a) = \overline{R}(s,a), \quad (9b)$$

If a more informed prior is known for $R(s,a)$, then it could be employed to derive a more informed exploration heuristic.

Case 2: $N(s,a) > 0$. In this case, we have observed $R(s,a)$, so it is no longer a random variable. Hence, the expectation is only over s'. For purposes of deriving our exploration heuristic, we will assume that s' will be drawn according to our current maximum likelihood estimate $\hat{P}(s'|s,a)$ but that $N_1(s,a)$ will not change, so the Good-Turing estimate will not change. Under this assumption, the expected value of Q will not change, $M_0(s,a)$ will not change, so the only change to Q_{upper} and Q_{lower} will result from replacing $\omega(N(s,a),\delta)$ by $\omega(N(s,a)+1,\delta)$.

Note that DDV explores a state-action pair (s,a) even if a is not currently the optimal action in s. That is, even if $Q_{upper}(s,a) < Q_{upper}(s,a')$ for some $a' \neq a$. An alternative rule would be to only explore (s,a) if it would reduce the expected value of $\Delta V(s) = V_{upper}(s) - V_{lower}(s)$. However, if there are two actions a and a' such that $Q_{upper}(s,a) = Q_{upper}(s,a')$, then exploring only one of them will not change $\Delta V(s)$. We have studied another variant in which we defined $V_{upper}(s) = \text{softmax}(\tau)_a Q_{upper}(s,a)$ (the softmax with temperature τ). This gave slightly better results, but it requires that we tune τ, which is a nuisance.

We now present the main theoretical result of the paper.

Theorem 1 (DDV is PAC-RL)**.** *There exists a sample size m polynomial in $|S|$, $|A|$, $1/\varepsilon$, $1/\delta$, $1/(1-\gamma)$, R_{max}, such that* $\text{DDV}(s_0, F, \varepsilon, \delta/(m|S||A|))$ *terminates after no more than $m|S||A|$ calls on the simulator and returns a policy π such that $|V^{\pi}(s_0) - V^*(s_0)| < \varepsilon$ with probability $1-\delta$.*

Proof. We begin with sample complexity. At the point where DDV terminates, let $\tilde{P}_{upper}$ be the set of probability transition functions chosen in equation (3), $R(s,a)$ be the learned reward function, and Q_{upper} be the computed upper bound on the Q function. We can view this as defining an MDP M_{upper} whose optimal policy π_{upper} selects actions $\pi_{upper}(s) := \text{argmax}_a Q_{upper}(s,a)$. Define M_{lower} and π_{lower} analogously. Then Lemma 1 of Strehl and Littman (2008) establishes that for all (s,a),

$$Q^{\pi_{upper}}_{upper}(s,a) - Q^{\pi_{lower}}_{lower}(s,a) \leq \frac{2\gamma R_{max}\omega}{(1-\gamma)^2},$$

where ω is the radius of the confidence interval from (2). We want this to be $\leq \varepsilon$. Solving for ω gives

$$\omega \leq \frac{\varepsilon(1-\gamma)^2}{2\gamma R_{max}}.$$

Now we substitute the definition of ω and solve for the sample size m to obtain

$$m \geq \frac{8[\ln(2^{|S|} - 2) - \ln\delta]\gamma^2 R^2_{max}}{\varepsilon^2(1-\gamma)^4}.$$

Now let's consider s_0. Given that $Q_{upper}(s_0,a) - Q_{lower}(s_0,a) \leq \varepsilon$ for all a, it follows that $V_{upper}(s_0) - V_{lower}(s_0) \leq \varepsilon$. To see this, let $a_{upper} = \text{argmax}_a Q_{upper}(s_0,a)$. Then $V_{upper}(s_0) = Q_{upper}(s_0, a_{upper})$. Clearly $V_{lower}(s_0) \geq Q_{lower}(s_0, a_{upper})$, so $\Delta V(s_0) < \varepsilon$.

If DDV samples every (s,a) at least $m = \tilde{O}(|S|\gamma^2(1/\varepsilon)^2(1/(1-\gamma))^4)$ times, then it will terminate with $\Delta V(s_0) \leq \varepsilon$. Hence, the sample complexity of DDV is $\tilde{O}(|S|^2|A|\gamma^2(1/\varepsilon)^2(1/(1-\gamma))^4)$, which is polynomial in the relevant quantities.

To prove the approximate correctness of the result, we know from Proposition 1 that if we had $1-\delta/|S||A|$ confidence intervals on all (s,a), then with probability $1-\delta$,

$$V_{upper}(s_0) \leq V^*(s_0) \leq V_{lower}(s_0).$$

However, DDV may not have explored all (s,a). Given that Q_{upper} and Q_{lower} assume V_{max} and 0 in unexplored states (respectively), and given that DDV did not explore them, this implies that exploring those states would not increase $\Delta V(s_0)$.

Because DDV recomputes the confidence interval for one $Q(s,a)$ after every sample (s,a,r,s'), it computes no more than $|S||A|m$ confidence intervals. So by invoking DDV with confidence parameter $\delta/|S||A|m$, we ensure that the final confidence interval contains $V^*(s_0)$ with probability $1-\delta$. □

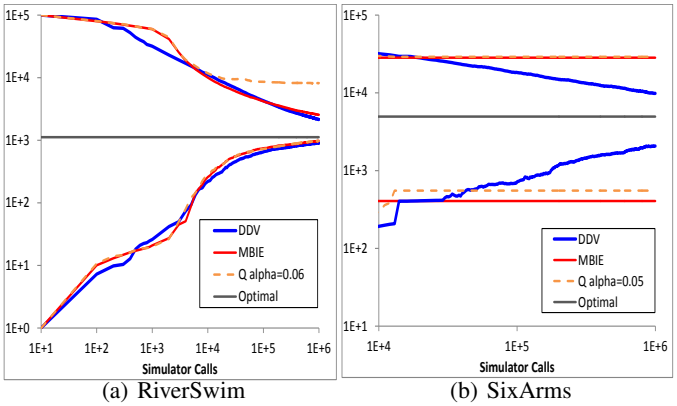
Experimental Evaluation

We report three experiments. First, we tested the effectiveness of the $\Delta\Delta V(s_0)$ exploration heuristic on two benchmark MDPs, RiverSwim and SixArms, that have been studied by Strehl and Littman (2004; 2008). They showed that MBIE out-performs R_{max}, E^3, and ε-greedy exploration on these problems, so we compare against MBIE and Q learning (with greedy exploration based on optimistic initialization).

In this experiment, we do not employ the Good-Turing confidence intervals in order to focus on the $\Delta\Delta V$ heuristic. Figures 1(a) and 1(b) show the computed confidence intervals as a function of the number of simulator calls for the first 10^6 calls. On RiverSwim, DDV rapidly shrinks the bounds (at around 10^4 calls). But in general, both MBIE and DDV do well. Q learning gets stuck exploring in the wrong part of the MDP. In SixArms, MBIE and Q learning do well at first, but then they both get stuck, while DDV continues to converge.

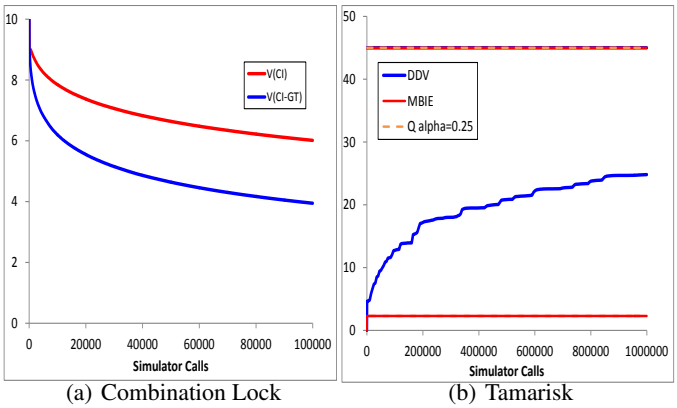
In the second experiment, we tested the effectiveness of incorporating the Good-Turing bound into the confidence interval by running DDV with and without the improved confidence interval. We employed a "combination lock" MDP with 500 states. In each state i, there are two possible actions. The first makes a transition to state $i+1$ with reward 0 except for state 500, where the reward is 1. The second makes a transition (uniformly) to one of the states $1,\ldots,i-1$ with reward 0. The optimal policy is to choose the first action in every state, even though it doesn't provide a reward until the final state. Figure 2(a) compares the confidence intervals for the two configurations. Note that even though the two configurations use different confidence intervals during exploration, the figure reports $V_{upper}(s_0)$ and $V_{lower}(s_0)$ computed using CI^{GT}. We see that in this sparse MDP, the Good-Turing bound substantially accelerates learning.

In the final experiment, we compare MBIE, Q learning, and DDV on a small instance (3 river segments, 4 actions, 216 states) of our Tamarisk management MDP. Figure 2(b) shows that the lower bound for DDV is improving, but DDV's upper bound and all of the bounds for MBIE and Q learning are not changing. We don't know the optimal solution for this problem.



(a) RiverSwim (b) SixArms

Figure 1: Learning curves for MBIE, Q-learning, and DDV as measured by confidence bounds on $V(s_0)$



(a) Combination Lock (b) Tamarisk

Figure 2: Left: Learning curve for DDV with and without incorporating Good-Turing confidence bounds. Right: Learning curves for MBIE, Q-learning, and DDV on a Tamarisk management MDP.

Concluding Remarks

This paper has developed a new algorithm for MDP planning (for a fixed start state) by building on recent advances in reinforcement learning and introducing two improvements. The first is a tighter confidence bound for sparse MDPs that incorporates Good-Turing estimates of the probability of transitioning to unseen states. The second is a new exploration heuristic based on an upper bound on the state occupancy measure. The resulting algorithm, DDV, is proved to be PAC-RL. Experiments show that both innovations contribute improved learning speed.

With these improvements, we are able to solve moderate-sized instances of our Tamarisk MDP. However, additional improvements in the algorithm (e.g., action elimination, adapting the partitioning of δ to the number of explored states) will be required to solve problems of significant size.

Acknowledgments

The authors thank Kim Hall and H. Jo Albers for collaborating with us on the Tamarisk problem. This material is based upon work supported by the National Science Foundation under Grant No. 0832804.

References

Bellman, R. 1957. *Dynamic Programming*. New Jersey: Princeton University Press.

Brafman, R. I., and Tennenholtz, M. 2003. R-max: A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning. *The Journal of Machine Learning Research* 3:213–231.

Chernoff, H. 1952. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations. *The Annals of Mathematical Statistics* 23(4):493–507.

DiTomaso, J., and Bell, C. E. 1996. *Proceedings of The Saltcedar Management Workshop*. Rancho Mirage, CA: www.invasivespeciesinfo.gov/docs/news/workshopJun96/index.html.

Fiechter, C.-N. 1994. Efficient Reinforcement Learning. In *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, 88–97. ACM Press.

Good, I. J. 1953. The Population Frequencies of Species and the Estimation of Population Parameters. *Biometrika* 40(3):237–264.

Jaksch, T.; Ortner, R.; and Auer, P. 2010. Near-optimal Regret Bounds for Reinforcement Learning. *Journal of Machine Learning Research* 11:1563–1600.

Kakade, S. M. 2003. *On the Sample Complexity of Reinforcement Learning*. Doctoral dissertation, University College London.

Kearns, M., and Saul, L. 1998. Large Deviation Methods for Approximate Probabilistic Inference, with Rates of Convergence. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 311–319.

Kearns, M., and Singh, S. 1998. Near-optimal Reinforcement Learning in Polynomial Time. *Machine Learning* 49:260–268.

McAllester, D., and Ortiz, L. 2003. Concentration Inequalities for the Missing Mass and for Histogram Rule Error. *Journal of Machine Learning Research* 4:895–911.

Puterman, M. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Mathematical Statistics. Wiley.

Smith, T., and Simmons, R. 2006. Focused Real-Time Dynamic Programming for MDPs: Squeezing More Out of a Heuristic. In *AAAI 2006*, 1227–1232.

Stenquist, S. 1996. *Saltcedar Management and Riparian Restoration Workshop*. Las Vegas, NV: www.invasivespeciesinfo.gov/docs/news/workshopSep96/index.html.

Strehl, A. L., and Littman, M. L. 2004. An Empirical Evaluation of Interval Estimation for Markov Decision Processes. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)*, 128–135.

Strehl, A., and Littman, M. 2008. An Analysis of Model-Based Interval Estimation for Markov Decision Processes. *Journal of Computer and System Sciences* 74(8):1309–1331.

Syed, U.; Bowling, M.; and Schapire, R. 2008. Apprenticeship Learning Using Linear Programming. In *International Conference on Machine Learning*.

Weissman, T.; Ordentlich, E.; Seroussi, G.; Verdu, S.; and Weinberger, M. J. 2003. Inequalities for the L1 Deviation of the Empirical Distribution. Technical report, HP Labs.

Wingate, D., and Seppi, K. 2006. Prioritization Methods for Accelerating MDP Solvers. *Journal of Machine Learning Research* 6:851–881.