Design for an Optimal Probe

Michael Duff DUFF@GATSBY.UCL.AC.UK
Gatsby Computational Neuroscience Unit, University College London, WC1N 3AR England

Abstract

Given a Markov decision process (MDP) with expressed prior uncertainties in the process transition probabilities, we consider the problem of computing a policy that optimizes expected total (finite-horizon) reward. Implicitly, such a policy would effectively resolve the "exploration-versus-exploitation tradeoff" faced, for example, by an agent that seeks to optimize total reinforcement obtained over the entire duration of its interaction with an uncertain world. A Bayesian formulation leads to an associated MDP defined over a set of generalized process "hyperstates" whose cardinality grows exponentially with the planning horizon. Here we retain the full Bayesian framework, but sidestep intractability by applying techniques from reinforcement learning theory. We apply our resulting actor-critic algorithm to a problem of "optimal probing," in which the task is to identify unknown transition probabilities of an MDP using online experience.

1. Introduction

Suppose we are confronted with a Markov decision process that has unknown transition probabilities, p_{ij}^k, $i = 1, ..., N$, $k = 1, ..., M$, where N and M are the number of states and actions respectively. We wish to identify these unknown parameters.

At our disposal we have a "probe," which we can launch into the MDP. The probe has a finite lifetime, and may be pre-loaded with an adaptive policy for navigating through the MDP. Its goal is to reduce uncertainty as much as possible during its lifetime.

In this paper, we cast this problem of optimal probing as a particular instance of a problem of computing optimal policies for what we term "Bayes-adaptive Markov decision processes" (BAMDP's). A BAMDP is a sequential decision processes that, in incorporating a Bayesian model for evolving uncertainty about unknown process parameters, takes the form of an MDP defined over a set of "hyperstates" whose cardinality grows exponentially with the planning horizon—conventional dynamic programming solution techniques are typically dismissed as being intractable. We propose a computational procedure that retains the full Bayesian formulation, but sidesteps intractability by utilizing reinforcement learning techniques that employ Monte-Carlo simulation and parameterized function approximators.

One way of thinking about the algorithm is that it performs an offline computation of an online, adaptive machine. One may regard the process of computing an optimal policy for the BAMDP as "compiling" an optimal learning strategy, which can then be "loaded" into an agent. With actions dictated by the compiled policy, the agent would behave in a (seemingly adaptive) manner that would be optimal *with respect to its prior*, which describes the distribution of environmental scenarios the agent is likely to encounter.

Given the prior uncertainty over possible environments, the optimal-learning agent must collect and use information in an intelligent way, balancing greedy exploitation of certainty-equivalent world models with exploratory actions aimed at discerning the true state of nature. The optimal policy for a BAMDP implicitly performs this "exploration-versus-exploitation tradeoff" in an optimal way, though in our formulation the agent does not expicitly explore or exploit, rather it takes an optimal action (which may be some subtle mixture of pure explore or exploit) with respect to its full Bayesian model of the uncertain sequential decision process.

By pursuing an approach that is grounded in theory, and that appeals to a complete world model that incorporates a Bayesian framework for characterizing evolving uncertainty, the algorithm we develop produces polices that exhibit performance gains over simple heuristics. Moreover, in contrast to many heuristics, the justification or legitimacy of the policies follows directly from the fact that they are clearly motivated by a complete characterization of the underlying decision problem to be solved.

In the following section, we review the Bayesian framework for modeling MDP's with uncertain transition probabilities. This is followed by the development of our algorithm, which is a Monte-Carlo, gradient-based scheme utilizing particular function approximation architectures for representing policies and value functions—the approach has been advanced in the context of conventional MDP's by Sutton *et al.* (2000) and Konda and Tsitsiklis (2000). Finally, we return to the problem of designing an optimal probe and show how our actor-critic algorithm can be applied.

2. Bayes-adaptive Markov decision processes

Let us refer to a Markov decision process with Bayesian modeling of unknown transition probabilies as a *Bayes-adaptive Markov decision process* (BAMDP) (Bellman (1961) uses the term "adaptive control process"). The status of such a process at a given stage is summarized by its ***hyperstate***, $\langle s, x \rangle$, which consists of the process's Markov chain state, s, which we shall refer to as the ***physical state***, together with its ***information state***, x, which is the collection of distributions describing uncertainty in the transition probabilites. Given the hyperstate, if we apply an action and observe a transition, then Bayes's rule prescribes how the information state is to be revised.

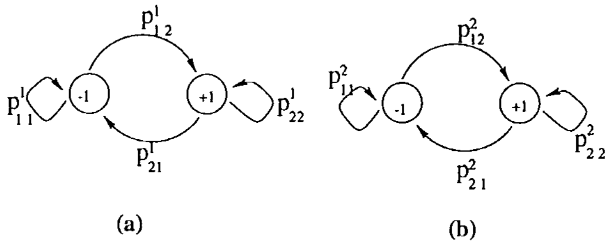

Figure 1. An MDP with uncertain transition probabilities: dynamics under (a) action 1 and (b) action 2 (± 1 denotes the rewards for entering right- and left-hand states).

Figure 1 depicts the transition diagram for a simple MDP with 2 physical states and 2 feasible actions in each state. Transition probabilities, about which we are uncertain, label arcs. Rewards (± 1) are taken here to be deterministic rewards that are received upon transitioning into a particular state. Our goal is to maximize the expected total (undiscounted) reward received over the entire course of interaction, which we assume has finite duration.

If the process is in a given state and an action is taken, then the result is that the process either remains in its current physical state or jumps to the other complementary state—one observes a Bernoulli process with unknown parameter.

In general, the distributions characterizing uncertainty can be general densities defined over continuous domains; representing these densities and performing the integrations prescribed by Bayes's rule can be a computationally intensive process. In practice, it is convenient to adopt "conjugate families of distributions" to model our uncertainty (DeGroot, 1970). For example, if our uncertainty in p^1_{11} is expressed as a beta distribution parameterized by (α^1_1, β^1_1), then the posterior distribution for p^1_{11}, given an observation (transition from state 1 under action 1), is also a beta distribution, but with parameters that are incremented to reflect the observed datum.

For the process as a whole, we track the progress of *four* Bernoulli processes: the result of taking `action 1 in state 1`, `action 1 in state 2`, `action 2 in state 1`, `action 2 in state 2`. So if the prior probability for remaining in the current state, for each of these state-action pairs, is represented by a beta distribution, then one may construct a corresponding hyperstate transition diagram, in which information-state components are composed of four pairs of parameters specifying the beta distributions describing the uncertainty in the transition probabilities. The full hyperstate may be written as: $(s, (\alpha^1_1, \beta^1_1), (\alpha^1_2, \beta^1_2), (\alpha^2_1, \beta^2_1), (\alpha^2_2, \beta^2_2))$, where s is the physical Markov chain state, and where, for example, (α^1_1, β^1_1) denotes the parameters specifying the beta distribution that represents uncertainty in the transition probability p^1_{11}.

Figure 2 shows part of the hyperstate transition diagram associated with the uncertain Markov decision process of Figure 1. We have written the information state parameters in matrix form to highlight their observed correspondence to transition counts modulo the prior; *i.e.*, if we subtract the information state associated with the prior from any given hyperstate's information state component, the result may be interpreted as the number of transitions of each type observed in transit from the initial hyperstate to the given hyperstate.

An optimality equation may be written in terms of the hyperstates (see Martin (1967) for a rigorous deriva-

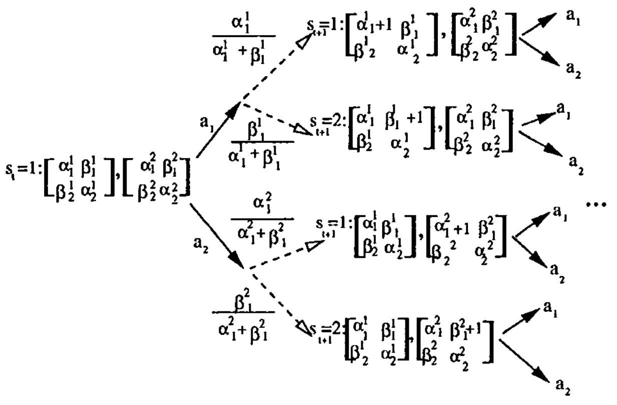
Figure 2. The hyperstate transition diagram associated with the uncertain Markov decision process of Figure 1.

tion); the optimal value function must be consistent with local transitions to successor hyperstates and their values, for example:

$$
\begin{aligned}
&V\left(1;\begin{bmatrix}\alpha_1^1 & \beta_1^1\\ \beta_2^1 & \alpha_2^1\end{bmatrix},\begin{bmatrix}\alpha_1^2 & \beta_1^2\\ \beta_2^2 & \alpha_2^2\end{bmatrix}\right) \quad = \\
&\max\left\{\frac{\alpha_1^1}{\alpha_1^1+\beta_1^1}\left[r_{11}^1 + \right.\right.\\
&\quad\left. V\left(1;\begin{bmatrix}\alpha_1^1+1 & \beta_1^1\\ \beta_2^1 & \alpha_2^1\end{bmatrix},\begin{bmatrix}\alpha_1^2 & \beta_1^2\\ \beta_2^2 & \alpha_2^2\end{bmatrix}\right)\right]\\
&+\frac{\beta_1^1}{\alpha_1^1+\beta_1^1}\left[r_{12}^1 + \right.\\
&\quad\left. V\left(2;\begin{bmatrix}\alpha_1^1 & \beta_1^1+1\\ \beta_2^1 & \alpha_2^1\end{bmatrix},\begin{bmatrix}\alpha_1^2 & \beta_1^2\\ \beta_2^2 & \alpha_2^2\end{bmatrix}\right)\right],\\
&\frac{\alpha_1^2}{\alpha_1^2+\beta_1^2}\left[r_{11}^2 + \right.\\
&\quad\left. V\left(1;\begin{bmatrix}\alpha_1^1 & \beta_1^1\\ \beta_2^1 & \alpha_2^1\end{bmatrix},\begin{bmatrix}\alpha_1^2+1 & \beta_1^2\\ \beta_2^2 & \alpha_2^2\end{bmatrix}\right)\right]\\
&+\frac{\beta_1^2}{\alpha_1^2+\beta_1^2}\left[r_{12}^2 + \right.\\
&\quad\left.\left. V\left(2;\begin{bmatrix}\alpha_1^1 & \beta_1^1\\ \beta_2^1 & \alpha_2^1\end{bmatrix},\begin{bmatrix}\alpha_1^2 & \beta_1^2+1\\ \beta_2^2 & \alpha_2^2\end{bmatrix}\right)\right]\right\}.
\end{aligned}
$$

In principle, one can consider computing an optimal policy via dynamic programming, though the transition diagram suggests 4^{depth} hyperstates at a given depth—roughly a million hyperstates at a horizon of 10. This is Bellman's (1961) "menace of the expanding grid"—the number of hyperstates grows exponentially with the time-horizon.[1] Another feature of the transition diagram is that the transition probabilities are known. Although the underlying transition probabilites between physical states are unknown, the Bayesian model (*a la* beta distribution parameters) leads to a hyperstate process with known transition probabilities (which, it should be stressed, are consistent with the assumed prior over MDP's).

For MDP's with more than two physical states, sampling becomes multinomial, for which the appropriate conjugate family of distributions is Dirichlet, and the current formulation generalizes in a straightforward way. We should also mention that the concept of a conjugate family of distributions can be generalized. One can consider mixtures of conjugate distributions, or "extended natural conjugate distributions" (Martin, 1967), which provide more degrees of freedom for expressing our prior uncertainty and allow for nonzero correlation between rows of the generalized transition matrix.[2] However, the central issue of Bellman's "menace of the expanding grid" remains—the number of distinct reachable hyperstates grows exponentially with the time horizon.

3. An actor-critic algorithm for BAMDP's

3.1. Stochastic policy and value

A stochastic policy for a general MDP with state set S may be specified by: $\pi_i^k \overset{def}{=} Pr\{action = k|state = i\}$, The value function, V_i^π, $i = 1,...,|S|$, is the expected infinite-horizon discounted sum of rewards, starting in state i and applying a fixed policy π. Value function components satisfy a linear system of equations that enforces local consistency:

$$V_i^\pi = \sum_k \pi_i^k \left[\sum_j p_{ij}^k(r_{ij}^k + \gamma V_j^\pi)\right] \quad i = 1,...,|S|.$$

In the context of BAMDP's, the state set, S, is the collection of hyperstates, which even for simple problems, is quite large. Suppose we represent policies by functions that depend upon parameters $\eta_1, \eta_2, ..., \eta_L$, where

[1]Not all 4^{depth} hyperstates at a given depth are distinct. At a depth of 25, for example, there are only 403,702 distinct hyperstates (Duff, 2002), though this number still grows exponentially with the planning horizon.

[2]A Dirichlet prior assumes independence between rows of the generalized transition matrix. For a matrix with N columns, one would like to select Dirichlet parameters associated with each row to match the prior estimates of the means and variances of the N individual p_{ij}^a's. However, this would require $2N - 1$ parameters, and the Dirichlet prior supplies only N. A practical approach for assigning a prior from empirical data proceeds by fitting the means exactly, then by performing a least-squares fit of the remaining Dirichlet parameter to match the variances of the p_{ij}^a's (Silver, 1963).

typically, $L << |S|$: $\pi_i^k(\eta_1, ..., \eta_L)$. One may think of the η's as parameters that, together with a collection of features, determine distributions over actions given a hyperstate as input.

With this reduced flexibility, we cannot specify all possible stochastic policies exactly (in particular, in general we cannot specify the optimal policy), but our hope is that our parameterized family of policies will be rich enough to represent policies that are good or nearly optimal. In exchange, the space required by our policy representation, which is now biased, becomes independent of the number of hyperstates.

Consider the state-action probabilities to be implicit functions of the parameter vector η; that is, $\pi_i^k = \pi_i^k(\eta_1, \eta_2, ..., \eta_L)$, $i = 1, ..., |S|$, $k = 1, ..., M$. Taking partial derivatives of both sides of the value-function equation with respect to a generic component of the policy parameter vector leads to:

$$\begin{aligned} \frac{\partial V_i^\pi}{\partial \eta_l} &= \sum_k \frac{\partial \pi_i^k}{\partial \eta_l} \sum_j p_{ij}^k (r_{ij}^k + \gamma V_j^\pi) \\ &\quad + \gamma \sum_j \sum_k \pi_i^k p_{ij}^k \frac{\partial V_j^\pi}{\partial \eta_l}. \end{aligned} \tag{1}$$

This describes an $|S| \times |S|$ linear system of equations for the components $\frac{\partial V_i^\pi}{\partial \eta_l}$, $i = 1, 2, ..., |S|$. It has the form $Ax = b$, where

$$\begin{aligned} A_{ij} &= \delta_{ij} - \gamma \sum_k \pi_i^k p_{ij}^k \\ b_i &= \sum_k \frac{\partial \pi_i^k}{\partial \eta_l} \sum_j p_{ij}^k (r_{ij}^k + \gamma V_j). \end{aligned} \tag{2}$$

There are L such linear systems, one corresponding to each η_l.

3.2. Monte-Carlo gradient computation

A is a square matrix with dimension size equal to the number of hyperstates. Solving for the vector of gradient components directly (an $O(|S|^3)$ proposition) or iteratively ($O(|S|^2$ per iteration) is unthinkable in general However, a Monte-Carlo approach leads to a tractable computational procedure.

Note that the matrix A is of the form $A = I - \gamma P_\pi$, where P_π is the "policy-averaged" transition matrix; *i.e.*, it is the average transition matrix associated with the stochastic policy π. Let us re-express the solution of the linear system in terms of the Neumann series:

$$x = A^{-1}b = (I - \gamma P_\pi)^{-1} b = \sum_{k=0}^{\infty} (\gamma P_\pi)^k b.$$

Our goal is to maximize the value associated with the initial hyperstate, i_0. Therefore, we are interested in determining the gradient of this value component with respect to controller parameters. Notice that $\frac{\partial V_{i_0}}{\partial \eta_l}$ is the i_0th element of $A^{-1}b$, or equivalently, it is the (i_0th row of A^{-1}) $\times b$. A Monte-Carlo scheme can make use of this fact. For example, for an episodic, undiscounted ($\gamma = 1$) process, the Neumann series expression above implies that the $(i_0, j)th$ component of A^{-1} is just the expected number of visits to hyperstate j, given that the process starts in i_0. This implies that we may obtain an unbiased sample of $\frac{\partial V_{i_0}}{\partial \eta_l}$ by simulating a hyperstate trajectory starting in i_0 and accumulating components of the $b-$vector corresponding to the trajectory's hyperstates:

$$\sum_{j \text{ on trajectory}} b_j.$$

Note that the A matrix is the same for all gradient components. Only the b-vector depends upon the parameter, η_l. Therefore a single trajectory yields unbiased estimates for *all* gradient components, $\frac{\partial V_{i_0}}{\partial \eta_l}$ $\forall l$.

Conceptually, Equation 1 defines L linear systems of equations; each linear system has the form $Ax = b$, where the solution, x, is the vector of gradient components, $\frac{\partial V_i}{\partial \eta_l}$, $i = 1, \ldots, N$ for fixed choice of l. Each of the L linear systems has the same A-matrix, but the b-vectors vary with l as specified by Equation 2. In our Monte-Carlo approach, we simulate a hyperstate trajectory starting from i_0, which determines the element indices of b that are to be accumulated to form an unbiased estimate of $\frac{\partial V_{i_0}}{\partial \eta_l}$ $\forall l$. This method for selecting element indices of b is valid for *all* of the b-vectors corresponding to different choices of η_l, and so, operationally, a single hyperstate trajectory is sufficient for providing an unbiased estimate of gradient components with respect to all η_l.

3.3. Function approximation

We shall employ distinct parameterized function approximators for representing value functions and stochastic policies for the BAMDP. First, with regard to value-function approximation, we propose that our approximator be a linear combination of hyperstate features; *i.e.*, $V(s, x) \approx \sum_l \theta_l \varphi_l (s, x)$. We choose this architecture for its simplicity, and useful analytical convergence results exist for this case.

We proceed under the assumption that value is a relatively smooth function of information state.[3] Without further (problem-specific) assumptions, little can be said in general about behavior as a function of physical

[3]It is known that the value function, V, is a continuous function of information state parameters, x (Martin, 1967, p.44).

state. Therefore, we propose using separate parameterized value-function approximators for each physical state. For our feature set, we propose using the components of information state, x, together with a "bias" feature grounded at a value of 1—the total number of feature-vector components is thus $N^2M + 1$, where N is the number of physical states. We then write:

$$V(s, x) \approx V_s(x) = \sum_l \theta_l[s] x_l,$$

where V_s denotes the function-approximator associated with physical state s, and $x_0 = 1$ while x_l for $l > 0$ ranges over information state components.

Turning now to the issue of function approximation for policies, we propose that stochastic policy functions, which map hyperstates and collections of admissible actions to probability distributions over admissible actions, be relatively smooth functions of information state, but allow for arbitrary and discontinuous variation with respect to physical state and prospective action. We therefore propose using a separate parameterized function approximator for each physical state and each admissible action. Since we have committed to using parameterized function approximation for representing policies, we choose an architecture that produces only valid distribution functions over candidate actions. In anticipation of utilizing the gradient formula developed in Section 3.1, we also require that our approximator be differentiable with respect to its parameters. Many policy approximation architectures satisfy these criteria—here we consider one possibility, an exponentiated, normalized function involving exponential terms that are linear in the feature components:

$$\pi((s, x), a) \approx \pi_{(s,a)}(x) = \frac{e^{\sum_l \eta_l[s][a] x_l}}{\sum_{a'} e^{\sum_l \eta_l[s][a'] x_l}},$$

where $\pi_{(s,a)}$ signifies the function approximator associated with state s and action a, and where l ranges over the $N^2M + 1$ components of information state and bias.

3.4. BAMDP policy and value gradients

We now combine the gradient formula (Equation 1) with the policy representation parameterized as in the previous section. The main analytical step consists of calculating the various components $\frac{\partial \pi_i^k}{\partial \eta_l}$ appearing in Equation 1's formula for value-gradient. Applying the quotient rule, considering the numerator of the result for possible cases of i and k, and using our definition of π results in:

$$\begin{aligned} \frac{\partial}{\partial \eta_j[i][k]} \pi_{(s,a)}(x) &= \frac{\partial}{\partial \eta_j[i][k]} \frac{e^{\sum_l \eta_l[s][a] x_l}}{\sum_{a'} e^{\sum_l \eta_l[s][a] x_l}} \\ &= \begin{cases} \pi_{(s,a)}(x)\left[\delta_{ka} - \pi_{(s,k)}(x)\right] x_j & i = s \\ 0 & i \neq s \end{cases} \end{aligned}$$

where δ is the Kronecker delta.

Recall that the Monte-Carlo scheme estimates gradient components via $\frac{\partial V_{i_0}}{\partial \eta_l} \approx \sum_{i \in traj} b_i$, where $b_i = \sum_k \frac{\partial \pi_i^k}{\partial \eta_l} \sum_j p_{ij}^k (r_{ij}^k + \gamma V_j)$.

Previously, we provided a Monte-Carlo interpretation of our value-gradient formula. With hyperstates (s, x) sampled by following policy π from the initial state, $(s(t_0), x(t_0))$, the incremental contribution to the gradient estimate (reflecting one hyperstate transition, $(s, x) \rightarrow (s', x')$) becomes:

$$\begin{aligned} &\Delta \frac{\partial}{\partial \eta} V^\pi(s(t_0), x(t_0)) \propto \\ &\sum_a \frac{\partial \pi_{(s,a)}(x)}{\partial \eta} \sum_{s'} p_{ss'}^a(x)\left[r_{ss'}^a + V^\pi(s', x'(s, a, s', x))\right]. \end{aligned}$$

Here, $p_{ss'}^a(x)$ denotes the Bayesian point estimate for a physical state transition from s to s' under action a given information state x, and $x'(s, a, s', x)$ denotes the Bayes-rule-updated information state induced by a physical-state transition in the underlying Markov chain from s to s' under action a, given the prior information-state, x. Substituting our expression for the policy gradient, $\frac{\partial \pi}{\partial \eta}$, leads to:

$$\begin{aligned} &\Delta \frac{\partial}{\partial \eta_j[s]][k]} V^\pi(s(t_0), x(t_0)) \propto \\ &\qquad \sum_a \{\, \pi_{(s,a)}(x)\left[\delta_{ka} - \pi_{(s,k)}(x)\right] \\ &\qquad\quad \times \sum_{s'} p_{ss'}^a(x)\left[r_{ss'}^a + V^\pi(s', x'(s, a, s', x))\right] \,\} x_j \\ &= \pi_{(s,k)}(x) \{\, \sum_{s'} p_{ss'}^k(x)\left[r_{ss'}^k + V^\pi(s', x'(s, k, s', x))\right] \\ &\qquad - \sum_a \pi_{(s,a)}(x) \sum_{s'} p_{ss'}^a(x) \\ &\qquad\qquad \times \left[r_{ss'}^a + V^\pi(s', x'(s, a, s', x))\right] \,\} x_j. \end{aligned}$$

3.5. Algorithm summary

The Monte-Carlo algorithm for approximating optimal policies for BAMDP's is sample-based with separate parameterized function approximators for value and for policy. It proceeds by simulating trajectories of the BAMDP: starting from the the initial hyperstate—the physical state $s(t_0)$ and information state $x(t_0)$, which defines the prior—it follows the hyperstate trajectory under policy π. At each step of the trajectory,

- Let the current hyperstate be (s, x). Suppose the value function parameters are $\theta_l[s]$, $s = 1, 2, ..., N$, $l = 0, 1, 2, ..., L$, and the policy parameters are $\eta_l[s][a]$, $s = 1, 2, ..., N$, $a = 1, 2, ..., M$, $l = 0, 1, 2, ..., L$, where $L = N^2M$ is the number of information state components.

- Generate action a via random sampling from the action distribution:

$$\pi_{(s,a)}(x) = \frac{e^{\sum_l \eta_l[s][a]x_l}}{\sum_{a'} e^{\sum_l \eta_l[s][a']x_l}},$$

- Observe $(s,x) \to (s',x')$ and reward r.

- Update value function parameters (via a TD update): for $j = 0,1,2,...L$,

$$\theta_j[s] \leftarrow \theta_j[s] + \alpha\left(r + \sum_l \theta_l[s']x'_l - \sum_l \theta_l[s]x_l\right)x_j,$$

where α is some small step size.

- Update policy parameters; for $k = 1,2,...M$ and $j = 0,1,2,...,N^2M$:

$$\begin{aligned}\eta_j[s][k] &\leftarrow \eta_j[s][k] \\ &+\beta\pi_{(s,k)}(x)\{\textstyle\sum_{s'} p^k_{ss'}(x) \\ &\quad (r^k_{ss'} + \textstyle\sum_l \theta_l[s']x'_l(s,k,s',x)) \\ &\quad - \textstyle\sum_a \pi_{(s,a)}(x) \sum_{s'} p^a_{ss'}(x) \\ &\quad (r^a_{ss'} + \textstyle\sum_l \theta_l[s']x'_l(s,a,s',x))\ \}\,x_j\end{aligned}$$

where π is specified by our exponentiated, normalized approximator, and β is some small step-size (smaller than α makes sense and works well).

3.6. Example

We applied our algorithm to the horizon=25 case of the example presented in Figures 1 and 2. We chose to accumulate the parameter changes suggested by our algorithm over the course of each simulated trajectory, and then scaled those changes by total change-magnitude before making a true parameter change; *i.e.*, we performed a "batch" update of the parameters —results are plotted in Figure 3.

Each point plotted represents an estimate of expected total reward under the evolving policy—each point is the sample average, over 100,000 Monte-Carlo trials, of total reward obtained for a given policy. Performance is not quite that of the optimal tabular policy (which has value 5.719 and was computed by a single backward Gauss-Seidel sweep over the 2 million hyperstates—see (Duff, 2002) for the details of this computation). Our compressed policy representation does not have sufficient degrees-of-freedom to reproduce the optimal policy, which can tailor actions to every specific hyperstate, but its performance is within approximately 4% of optimal.

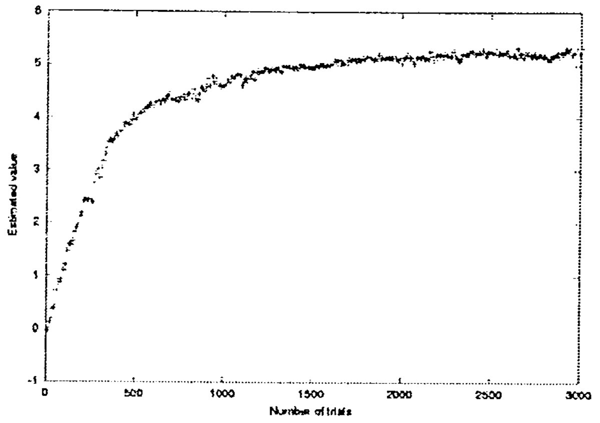

Figure 3. Performance of simulation-based stochastic policy iteration ($\alpha = .05$, $\beta = .01$) on the 2-state, 2-action example, with uniform priors for unknown transition probabilities. For this 2-state/2-action horizon=25 problem, there are nearly 2 million distinct reachable hyperstates. Value and policy function approximators for this case used only 18 and 36 parameters, respectively, but achieved performance that is within 4% of optimal.

4. Design for an optimal probe

To make matters concrete, consider the specific example depicted in Figure 4, an MDP with seven states and two actions. `Action 1` tends to move the probe left, while `action 2` tends to move the probe to the right.

We express our prior uncertainty in state-transition probabilities under these actions in terms of beta distribution parameters, which in Figure 4 label arcs; for example, our uncertainty in transitions (left move) from `state 2` under `action 1` is defined by the beta-distribution parameter pair $(6,2)$.

We have sculpted this example in such a way that our prior defines the *expected* probability of moving left under `action 1` as .75 for each state, but with uncertainty that increases with distance from the middle state, `state 4`, which we assume to be the start state. The prior uncertainty for `action 2` transitions is defined similarly (but with mean probability of moving *right* equal to .75).

We shall adopt the variance of the beta distribution as our measure of uncertainty:

$$Var(\alpha,\beta) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}.$$

Let us denote the full set of beta-distribution parameters, which express our uncertainty in state transition

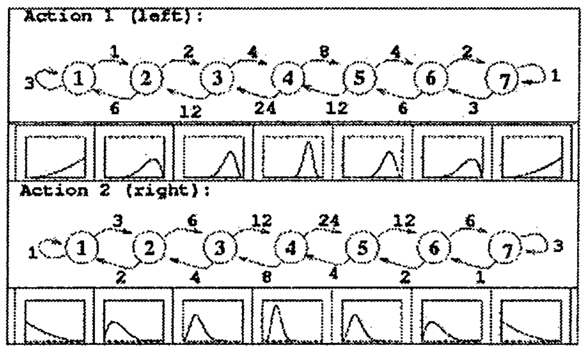

Figure 4. Beta distribution parameters label transition arcs for a simple optimal probing problem. Under action 1, the process moves left with a mean probability .75, but with uncertainty that increases with distance from the middle (start) state. Under action 2, the process moves *right* with mean probability .75, and uncertainty increases in the same way as for action 1. Density graphs plot prior uncertainty in the probability of moving left.

probabilities, as $M = \{m^k_{ij}\}$ $i = 1, 2, \ldots, 7$, $k = 1, 2$, and $j \in \{L, R\}$, where "L" and "R" denote indices associated with left and right transitions, respectively. Thus 28 parameters suffice to specify our uncertainty in the MDP. The prior uncertainty may be assigned subjectively, or may be informed by the results reported from previous probes.

If we are in a particular state, s, and take action a, then experience a transition to the left, then our uncertainty in transition probability associated with state s / action a becomes: $Var(m^a_{sL} + 1, m^a_{sR})$.

We propose defining the reward associated with this transition as the reduction in variance; *i.e.*, $reward(s^a \to L|M) = Var(m^a_{sL}, m^a_{sR}) - Var(m^a_{sL} + 1, m^a_{sR})$, and the probe's goal is to maximize the expected sum of such rewards over its lifetime.

We note that, if we repeatedly sample a fixed state and action, the expected reward (variance-decrement) diminishes with the number of samples (Figure 5).

This, together with the assumed structure of prior uncertainty, should lead to non-trivial probing strategies. For example, one would think that the probe should drive toward one of the ends of the chain where variance is high, squash out uncertainty there, then at some point turn toward the opposite end, passing through low-variance states en route.

We applied our actor-critic algorithm to this BAMDP

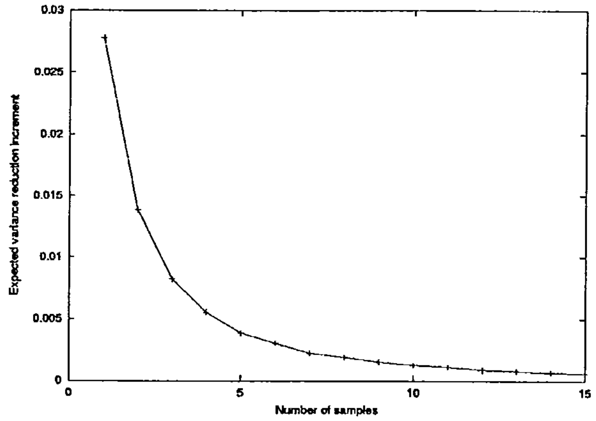

Figure 5. Diminishing variance reduction with repeated sampling. The graph plots the expected incremental reduction in variance for a Bernoulli process with unknown parameter and initial beta distribution parameters $(\alpha, \beta) = (1, 1)$.

(lifetime=15) and its performance is plotted in Figure 6 (each point is the expected performance as gauged by the average of 10,000 Monte-Carlo trials). As a comparison, we also show the expected performance of a probe that chooses its actions randomly, and the performance of a greedy probe that, at each step, chooses the action that maximizes the expected one-step reduction in variance. The actor-critic probe yields approximately a 23% improvement over the random probe and an 11% improvement over the greedy probe (or, over twice as much improvement from greedy to actor-critic as from random to greedy).

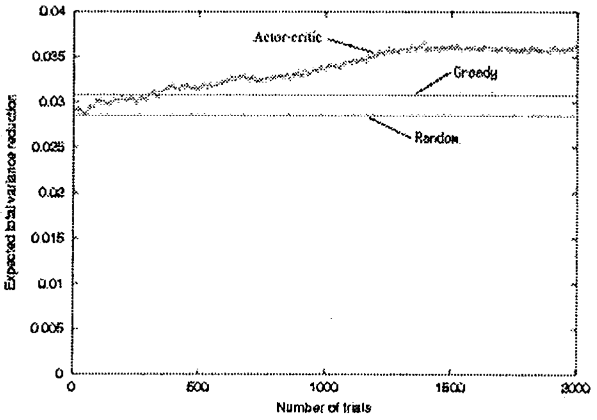

Figure 6. Probe performance (lifetime= 15).

5. Summary

For classical dynamic programming approaches, the complexity of computing optimal BAMDP policies scales exponentially with the time horizon. Even for two physical states and two physical actions, the horizon=25 case examined in our first example is rapidly approaching the limits of tractability. However, the complexity of our simulation-based algorithm does not scale in this way. Space requirements are $O(N^3)$ for representing the value function and policy—space complexity is independent of the time horizon. It is difficult to be precise regarding the time complexity of our algorithm. We simply observe that it is a trial-based algorithm, involving repeated estimation of V^π through simulation of controlled trajectories of length equal to the time horizon. In *very* simple terms (neglecting the nonstationarity of the value function as the policy improves), estimation error scales inversely with the square root of the number of trials (and squared error is proportional to the variance, which may be significant). Sutton *et al.* (2000) and Konda and Tsitsiklis (2000) have considered issues of convergence for actor-critic, policy-gradient algorithms of the type we have proposed and applied to BAMDP's. One of their main conclusions is that features employed by the value function approximation architecture (the "critic") should span a subspace prescribed the choice of parameterization of the controller (the "actor")—the critic need only compute a certain projection of the value function (a high-dimensional object) onto a low-dimensional subspace spanned by a set of basis functions that are *completely* determined by the parameterization of the actor. In practice, if one assumes a particular parameterization of a policy, a "compatability condition" (Sutton *et al.*, 2000) suggests the functional form of parameterization for the value function approximator. For example, if we adopt an exponentiated, normalized functional form for stochastic policies, as we have done in Section 3.3, then the compatability condition would lead us to propose a linear architecture for the value-function approximator, which is exactly what we have done naively.

In summary, the actor-critic algorithm for BAMDP's avoids the massive memory requirements inherent in conventional dyanamic programming (Bellman's "menace of the expanding grid"), and is applicable to general problems of optimal learning, of which the optimal probe example is a special case. The algorithm is practical and it works, at least to the extent that the function approximators invoked adequately represent value functions and policy mappings, and to the extent that the gradient scheme employed leads to improved policies. Our current research seeks alternative general, flexible, concise function approximators. In executing its adaptive strategy, the agent simply tracks the evolving hyperstate (*i.e.*, maintains state-transition counts) and applies actions as prescribed by its hyperstate-to-action policy mapping; that is, the agent is not required to perform significant amounts of computation online. The resulting policy may be viewed as implementing a form of far-sighted (hyperopic) active learning. In contrast to many heuristics, the legitimacy of policies computed by the algorithm follows from the fact that the algorithm is grounded in Bellman's optimality equation defined over hyperstates—we acknowledge long-range effects of information gain by retaining the complete Bayesian formulation of the problem to be solved.

Acknowledgements

The author wishes to thank Andy Barto, Peter Dayan, and Zoubin Ghahramani for helpful comments. This work was supported in part by a grant from the Gatsby Charitable Foundation.

References

Bellman, R. (1961) *Adaptive Control Proceses: A Guided Tour.* Princeton Univ. Press, Princeton, NJ.

DeGroot, M. H. (1970) *Optimal Statistical Decisions*, McGraw-Hill, New York.

Duff, M. O. (2002) *Optimal learning: Computational procedures for Bayes-adaptive Markov decision processes.* Ph.D. Thesis. Dept. of Computer Science, Univ. of Massachusetts, Amherst.

Konda, V. & Tsitsiklis, J. (2000). Actor-critic algorithms. In S.A. Solla, T. K. Leen, & K. R. Muller (Eds.), *Advances in Neural Information Processing Systems—12.* MIT Press, Cambridge, MA.

Martin, J. J. (1967). *Bayesian Decision Problems and Markov Chains.* John Wiley & Sons, New York.

Silver, E. A. (1963). *Markovian decision processes wih uncertain transition probabilities or rewards* (Technical Report No. 1). Research in the Control of Complex Systems, Operations Research Center, MIT.

Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In S.A. Solla, T. K. Leen, & K. R. Muller (Eds.), *Advances in Neural Information Processing Systems—12* (pp. 1057-1063). MIT Press, Cambridge, MA.