Bayesian Q-learning

Richard Dearden
Department of Computer Science
University of British Columbia
Vancouver, BC V6T 1Z4, Canada
dearden@cs.ubc.ca

Nir Friedman *
Computer Science Division
387 Soda Hall
University of California
Berkeley, CA 94720
nir@cs.berkeley.edu

Stuart Russell
Computer Science Division
387 Soda Hall
University of California
Berkeley, CA 94720
russell@cs.berkeley.edu

Abstract

A central problem in learning in complex environments is balancing *exploration* of untested actions against *exploitation* of actions that are known to be good. The benefit of exploration can be estimated using the classical notion of *Value of Information*—the expected improvement in future decision quality that might arise from the information acquired by exploration. Estimating this quantity requires an assessment of the agent's uncertainty about its current value estimates for states. In this paper, we adopt a Bayesian approach to maintaining this uncertain information. We extend Watkins' Q-learning by maintaining and propagating probability distributions over the Q-values. These distributions are used to compute a myopic approximation to the value of information for each action and hence to select the action that best balances exploration and exploitation. We establish the convergence properties of our algorithm and show experimentally that it can exhibit substantial improvements over other well-known model-free exploration strategies.

1 Introduction

Reinforcement learning is a rapidly growing area of interest in AI and control theory. In principle, reinforcement learning techniques allow an agent to become competent simply by exploring its environment and observing the resulting percepts and rewards, gradually converging on estimates of the value of actions or states that allow it to behave optimally. Particularly in control problems, reinforcement learning may have significant advantages over supervised learning: first, there is no requirement for a skilled human to provide training examples; second, the exploration process allows the agent to become competent in areas of the state space that are seldom visited by human experts and for which no training examples may be available.

In addition to ensuring more robust behavior across the state space, exploration is crucial in allowing the agent to discover the reward structure of the environment and to determine the optimal policy. Without sufficient incentive to explore, the agent may quickly settle on a policy of low utility simply because it looks better than leaping into the unknown. On the other hand, the agent should not keep exploring options that it already has good reason to believe are suboptimal. Thus, a good exploration method should balance the expected gains from exploration against the cost of trying possibly suboptimal actions when better ones are available to be exploited.

Optimal solution of the exploration/exploitation tradeoff requires solving a Markov decision problem over *information states*—that is, the set of all possible probability distributions over environment models that can be arrived at by executing all possible action sequences and receiving any possible percept sequence and reward sequence. The aim is to find a policy for the agent that maximizes its expected reward. Although this problem is well-defined, given a prior distribution over possible environments, it is not easy to solve exactly. Solutions are known only for very restricted cases—mostly the so-called *bandit problems* in which the environment has a single state, several actions, and unknown rewards [3].

Section 2 discusses several existing approaches to exploration, as well as the model-free Q-learning algorithm we use as our underlying learning method. This paper presents two new approaches to exploration:

Q-value sampling: Wyatt [17] proposed Q-value sampling as a method for solving bandit problems. The idea is to represent explicitly the agent's knowledge of the available rewards as probability distributions; then, an action is selected stochastically according to the current probability that it is optimal. This probability depends monotonically not only on the current expected reward (exploitation) but also on the current level of uncertainty about the actual reward (exploration). In this work, we extend this approach to multi-state reinforcement learning problems. The primary contribution here is a Bayesian method for representing, updating, and propagating probability distributions over rewards.

Myopic-VPI: Myopic value of perfect information [8] provides an approximation to the utility of an information-gathering action in terms of the expected improvement in decision quality resulting from the new information. This provides a direct way of evaluating the exploration/exploitation tradeoff. Like Q-value sampling, myopic-VPI uses the current probability distributions over rewards to control exploratory behavior.

Section 3 describes these two algorithms in detail, along with the Bayesian approach to computing reward distributions. In Section 4 we prove convergence results for the algorithms, and in Section 5 we describe the results of a

*Current address: Institute of Computer Science, The Hebrew University, Givat Ram, Jerusalem 91904, Israel, nir@cs.huji.ac.il.

1. Let the current state be s.
2. Select an action a to perform.
3. Let the reward received for performing a be r, and the resulting state be t.
4. Update $Q(s, a)$ to reflect the observation $< s, a, r, t >$ as follows:
   $Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{a'} Q(t, a'))$
   where α is the current learning rate.
5. Go to step 1.

Figure 1: The Q-learning algorithm.

number of experiments comparing them against other exploration strategies. In our experiments, myopic-VPI was uniformly the best approach.

2 Q-Learning

We assume the reader is familiar with the basic concepts of MDPs (see, e.g., Kaelbling et al. [9]). We will use the following notation: An MDP is a 4-tuple, $(\mathcal{S}, \mathcal{A}, p_t, p_r)$ where $\mathcal{S}$ is a set of *states*, $\mathcal{A}$ is a set of *actions*, $p_t(s \xrightarrow{a} t)$ is a *transition model* that captures the probability of reaching state t after we execute action a at state s, and $p_r(r|s, a)$ is a *reward model* that captures the probability of getting reward r when executing action a at state s.

In this paper, we focus on infinite-horizon MDPs with a discount factor $0 < \gamma < 1$. The agent's aim is to maximize the *expected discounted total reward* $E[\sum_i \gamma^i r_i]$, where r_i denotes the reward received at step i. Letting $V^*(s)$ denote the optimal expected discounted reward achievable from state s and $Q^*(s, a)$ denote the value of executing a at s, we have the standard Bellman equations [2]:

$$
\begin{aligned}
V^*(s) &= max_a Q^*(s, a) \\
Q^*(s, a) &= \sum_r r \cdot p_r(r|s, a) + \gamma \sum_t p_t(s \xrightarrow{a} t) V^*(t),
\end{aligned}
$$

Reinforcement learning procedures attempt to maximize the agent's expected reward when the agent *does not* know p_t and p_r. In this paper we focus on *Q-learning* [14], a simple and elegant *model-free* method that learns Q-values without learning the model p_t. In Section 6, we discuss how our results carry over to model-based learning procedures.

A Q-learning agent works by estimating the values of $Q^*(s, a)$ from its experiences. It then select actions based on their Q-values. The algorithm is shown in Figure 1. If every action is performed in every state infinitely often, and α is decayed appropriately, $Q(s, a)$ will eventually converge to $Q^*(s, a)$ for all s and a [15].

The strategy used to select an action to perform at each step is crucial to the performance of the algorithm. As with any reinforcement learning algorithm, some balance between exploration and exploitation must be found. Two commonly used methods are *semi-uniform random exploration* and *Boltzmann exploration*. In semi-uniform random exploration [16], the best action is selected with some probability p, and with probability $1 - p$, an action is chosen at random. In some cases, p is initially set quite low to encourage exploration, and is slowly increased. Boltzmann exploration [14] is a more sophisticated approach in which the probability of executing action a in state s is:

$$
Pr(a) = \frac{e^{Q(s,a)/T}}{\sum_{a'} e^{Q(s,a')/T}}
$$

where T is a temperature parameter that can be decreased slowly over time to decrease exploration. In this approach, the probability of an action being selected increases with the current estimate of its Q-value. This means that sub-optimal but good actions tend to be selected more often than clearly poor actions.

Both these exploration methods are *undirected*, meaning that no exploration-specific knowledge is used. A number of *directed* methods have also been proposed, of which the best known is *interval estimation* [10]. Most of the directed techniques can be thought of as selecting an action to perform based on the expected value of the action plus some *exploration bonus* [11]. In the case of interval estimation, we assume a normal distribution for the observed future values of each action in each state, and select an action by maximizing the upper bound of a $100(1 - \alpha)\%$ confidence interval (for some confidence coefficient α) over this distribution. The exploration bonus for interval estimation is half the width of the confidence interval. Other exploration bonuses have been proposed, based on the frequency or recency with which each action has been performed, or on the difference between predicted and observed Q-values.

The exploration-specific information in the Interval Estimation algorithm is strictly local in nature. The exploration bonus is calculated only from the future values observed from the current state. Exploration can also be done globally, selecting actions now that we believe will lead us to less-explored parts of the state space in the future. We can do this by backing up exploration specific information along with the Q-values. Meuleau and Bourgine [11], propose IEQL+, which is closely related to interval estimation in that it backs up Q-values and uses them to compute a local exploration bonus. Unlike interval estimation, IEQL+ also backs up an exploration bonus and combines the two to compute the new exploration value of the action.

For a survey of directed and undirected exploration techniques, see [13].

3 Bayesian Q-learning

In this work, we consider a Bayesian approach to Q-learning in which we use probability distributions to represent the uncertainty the agent has about its estimate of the Q-value of each state. As is the case with undirected exploration techniques, we select actions to perform solely on the basis of local Q-value information. However, by keeping and propagating distributions over the Q-values, rather than point estimates, we can make more informed decisions. As we shall see, this results in global exploration, but without the use of an explicit exploration bonus.

3.1 Q-Value Distributions

In the Bayesian framework, we need to consider prior distributions over Q-values, and then update these priors based on the agent's experiences. Formally, let $R_{s,a}$ be a random variable that denotes the *total* discounted reward received when action a is executed in state s and an optimal policy is followed thereafter. What we are initially uncertain about is how $R_{s,a}$ is distributed; in particular, we want to learn the value $Q^*(s,a) = E[R_{s,a}]$.

We start by making the following simplifying assumption:

Assumption 1: $R_{s,a}$ has a normal distribution.

We claim that this assumption is fairly reasonable. The accumulated reward is the (discounted) sum of immediate rewards, each of which is a random event. Thus, appealing to the central limit theorem, if γ is close to 1 and the underlying MDP is ergodic when the optimal policy is applied, then $R_{s,a}$ is approximately normally distributed.

This assumption implies that to model our uncertainty about the distribution of $R_{s,a}$, it suffices to model a distribution over the *mean* $\mu_{s,a}$ and the *precision* $\tau_{s,a}$ of $R_{s,a}$. (The precision of a normal variable is the inverse of its variance, that is, $\tau_{s,a} = 1/\sigma^2_{s,a}$. As it turns out, it is simpler to represent uncertainty over the precision than over the variance.) Of course, the mean, $\mu_{s,a}$, corresponds to the Q-value of (s,a).

Our next assumption is that the *prior* beliefs about $R_{s,a}$ are independent of those about $R_{s',a'}$.

Assumption 2: The prior distribution over $\mu_{s,a}$ and $\tau_{s,a}$ is independent of the prior distribution over $\mu_{s',a'}$ and $\tau_{s',a'}$ for $s \neq s'$ or $a' \neq a$.

This assumption is fairly innocuous, in that it restricts only the form of prior knowledge about the system. Note that this assumption does *not* imply that the *posterior* distribution satisfy such independencies. (We return to this issue below.)

Next we assume that the prior distributions over the parameters of each $R_{s,a}$ are from a particular family:

Assumption 3: The prior $p(\mu_{s,a}, \tau_{s,a})$, is a *normal-gamma* distribution.

We will now define and motivate the choice of the normal-gamma distribution. See [7] for more details.

A normal-gamma distribution over the mean μ and the precision τ of an unknown normally distributed variable R is determined by a tuple of *hyperparameters* $\rho = \langle \mu_0, \lambda, \alpha, \beta \rangle$. We say that $p(\mu, \tau) \sim NG(\mu_0, \lambda, \alpha, \beta)$ if

$$p(\mu,\tau) \propto \tau^{\frac{1}{2}} e^{-\frac{1}{2}\lambda\tau(\mu-\mu_0)^2} \tau^{\alpha-1} e^{\beta\tau}$$

Standard results show how to update such a prior distribution when we receive independent samples of values of R:

Theorem 3.1: [7] *Let $p(\mu,\tau) \sim NG(\mu_0, \lambda, \alpha, \beta)$ be a prior distribution over the unknown parameters for a normally distributed variable R, and let $r_1, \ldots, r_n$ be n independent samples of R with $M_1 = \frac{1}{n}\sum_i r_i$ and $M_2 = \frac{1}{n}\sum_i r_i^2$. Then $p(\mu, \tau \mid r_1, \ldots, r_n) \sim NG(\mu_0', \lambda', \alpha', \beta')$ where $\mu_0' = \frac{\lambda\mu_0 + nM_1}{\lambda + n}$, $\lambda' = \lambda + n$, $\alpha' = \alpha + \frac{1}{2}n$, and $\beta' = \beta + \frac{1}{2}n(M_2 - M_1^2) + \frac{n\lambda(M_1-\mu_0)^2}{2(\lambda+n)}$*

That is, given a single normal-gamma prior, the posterior after any sequence of independent observations is also a normal-gamma distribution.

Assumption 3 implies that to represent the agent's prior over the distribution of $R_{s,a}$, we only need to maintain a tuple of hyperparameters $\rho_{s,a} = \langle \mu_0^{s,a}, \lambda^{s,a}, \alpha^{s,a}, \beta^{s,a} \rangle$. Given Assumptions 2 and 3, we can represent our prior by a collection of hyperparameters for each state s and action a. Theorem 3.1 implies that, had we had independent samples of each $R_{s,a}$, the same compact representation could have been used for the joint posterior. We now assume that the posterior has this form

Assumption 4: At any stage, the agent's posterior over $\mu_{s,a}$ and $\tau_{s,a}$ is independent of the posterior over $\mu_{s',a'}$ and $\tau_{s',a'}$ for $s \neq s'$ or $a' \neq a$.

In an MDP setting, this assumption is likely to be violated; the agent's observations about the reward-to-go at different states and actions can be strongly correlated—in fact, they are related by the Bellman equations. Nonetheless, we shall assume that we can represent the posterior as though the observations were independent, i.e., we use a collection of hyperparameters $\rho_{s,a}$ for the normal-gamma posterior for the mean and precision parameters of each $R_{s,a}$.

We exploit this compact representation in the Bayesian Q-learning algorithm, which is similar to the standard Q-learning algorithm, except that instead of storing the Q-value $Q_{s,a}$, we now store the hyperparameters $\rho_{s,a}$. In the following sections, we address the two remaining issues: how to select an action based on the current belief state about the MDP, and how to update these beliefs after a transition.

3.2 Action Selection

In every iteration of the Q-learning algorithm we need to select an action to execute. Assuming that we have a probability distribution over $Q(s,a) = \mu_{s,a}$ for all states s and actions a, how do we select an action to perform in the current state? We consider three different approaches, which we call *greedy*, *Q-value sampling*, and *myopic-VPI*.

Greedy selection One possible approach is the *greedy* approach. In this approach, we select the action a that maximizes the expected value $E[\mu_{s,a}]$. Unfortunately, it is easy to show that $E[\mu_{s,a}]$ is simply our estimate of the mean of $R_{s,a}$. Thus, the greedy approach would select the action with the greatest mean, and would not attempt to perform exploration. In particular, it does not take into account any uncertainty about the Q-value.

Q-value sampling Q-value sampling was first described by Wyatt [17] for exploration in multi-armed bandit problems. The idea is to select actions stochastically, based on our current subjective belief that they are optimal. That is, action a is performed with probability given by

$$\begin{aligned} Pr(a = \arg\max_{a'} \mu_{s,a'}) &= Pr(\forall a' \neq a, \mu_{s,a} > \mu_{s,a'}) \\ &= \int_{-\infty}^{\infty} Pr(\mu_{s,a} = q_a) \prod_{a' \neq a} Pr(\mu_{s,a'} < q_a)\, dq_a \quad (1) \end{aligned}$$

The last step in this derivation is justified by Assumption 4 that states that our posterior distribution over the values of separate actions is independent.

To evaluate this expression, we use the marginal density of μ given a normal-gamma distribution.

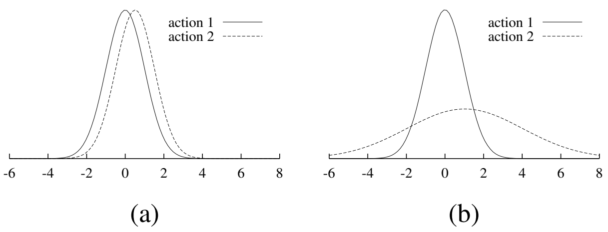


Figure 2: Examples of Q-value distributions of two actions for which Q-value sampling has the same exploration policy even though the payoff of exploration in (b) is higher than in (a).

Lemma 3.2: [7] *If* $p(\mu, \tau) \sim NG(\mu_0, \lambda, \alpha, \beta)$*, then*

$$p(\mu) = \left(\tfrac{\lambda}{2\pi}\right)^{\frac{1}{2}} \beta^{\alpha} \tfrac{\Gamma(\alpha+\frac{1}{2})}{\Gamma(\alpha)} \left(\beta + \frac{1}{2}\lambda(\mu-\mu_0)^2\right)^{-(\alpha+\frac{1}{2})}, \quad (2)$$

and

$$Pr(\mu < x) = T\left((x-\mu_0)\left(\frac{\lambda\alpha}{\beta}\right)^{\frac{1}{2}} : 2\alpha\right)$$

where $T(x : d)$ *is the cumulative* t-distribution *with* d *degrees of freedom. Moreover,* $E[\mu] = \mu_0$*, and* $\mathrm{Var}[\mu] = \frac{\beta}{\lambda(\alpha-1)}$.

In practice, we can avoid the computation of (1). Instead, we sample a value from each $p(\mu_{s,a})$, and execute the action with the highest sampled value. It is straightforward to show that this procedure selects a with probability given by (1). Of course, sampling from a distribution of the form of (2) is non-trivial and requires evaluation of the cumulative distribution $P(\mu < x)$. Fortunately, $T(x : d)$ can be evaluated efficiently using standard statistical packages. In our experiments, we used the library routines of Brown *et al.* [5].

Q-value sampling resembles, to some extent, Boltzmann exploration. It is a stochastic exploration policy, where the probability of performing an action is related to the distribution of the associated Q-values. One drawback of Q-value sampling is that it only considers the *probability* that a is best action, and does not consider the *amount* by which choosing a might improve over the current policy. Figure 2 show examples of two cases where Q-value sampling would generate the same exploration policy. In both cases, $Pr(\mu_{a_2} > \mu_{a_1}) = 0.6$. However, in case (b) exploration seems more useful than in case (a), since the potential for larger rewards is higher for the second action in this case.

Myopic-VPI selection This method considers quantitatively the question of policy improvement through exploration. It is based on *value of information* [8]. Its application in this context is reminiscent of its use in tree search [12], which can also be seen as a form of exploration. The idea is to balance the expected gains from exploration—in the form of improved policies—against the expected cost of doing a potentially suboptimal action.

We start by considering what can be gained by learning the true value $\mu^*_{s,a}$ of $\mu_{s,a}$. How would this knowledge change the agent's future rewards? Clearly, if this knowledge does not change the agent's policy, then rewards would not change. Thus, the only interesting scenarios are those where the new knowledge does change the agent's policy. This can happen in two cases: (a) when the new knowledge shows that an action previously considered sub-optimal is revealed as the best choice (given the agent's beliefs about other actions), and (b) when the new knowledge indicates that an action that was previously considered best is actually inferior to other actions. We now derive the value of the new information in both cases.

For case (a), suppose that a_1 is the best action; that is, $E[\mu_{s,a_1}] \geq E[\mu_{s,a'}]$ for all other actions a'. Moreover suppose that the new knowledge indicates that a is a better action; that is, $\mu^*_{s,a} > E[\mu_{s,a_1}]$. Thus, we expect the agent to gain $\mu^*_{s,a} - E[\mu_{s,a_1}]$ by virtue of performing a instead of a^*.

For case (b), suppose that a_1 is the action with the highest expected value and a_2 is the second-best action. If the new knowledge indicates that $\mu_{s,a_1} < E[\mu_{s,a_2}]$, then the agent should perform a_2 instead of a_1 and we expect it to gain $E[\mu_{s,a_2}] - \mu^*_{s,a_1}$.

To summarize this discussion, we define the gain from learning the value of $\mu^*_{s,a}$ of $\mu_{s,a}$ as:

$$Gain_{s,a}(\mu^*_{s,a}) = \begin{cases} E[\mu_{s,a_2}] - \mu^*_{s,a} & \text{if } a = a_1 \text{ and } \mu^*_{s,a} < E[\mu_{s,a_2}] \\ \mu^*_{s,a} - E[\mu_{s,a_1}] & \text{if } a \neq a_1 \text{ and } \mu^*_{s,a} > E[\mu_{s,a_1}] \\ 0 & \text{otherwise} \end{cases}$$

where, again, a_1 and a_2 are the actions with the best and second best expected values respectively. Since the agent does not know in advance what value will be revealed for $\mu^*_{s,a}$, we need to compute the *expected* gain given our prior beliefs. Hence the expected value of perfect information about $\mu_{s,a}$ is:

$$VPI(s,a) = \int_{-\infty}^{\infty} Gain_{s,a}(x) Pr(\mu_{s,a} = x) dx$$

Using simple manipulations we can reduce $VPI(s,a)$ to a closed form equation involving the cumulative distribution of $\mu_{s,a}$ (which can be computed efficiently).

Proposition 3.3: *$VPI(s,a)$ is equal to $c + (E[\mu_{s,a_2}] - E[\mu_{s,a_1}])Pr(\mu_{s,a_1} < E[\mu_{s,a_2}])$ when $a = a_1$, and it is equal to $c + (E[\mu_{s,a}] - E[\mu_{s,a_1}])Pr(\mu_{s,a} > E[\mu_{s,a_1}])$ when $a \neq a_1$, where*

$$c = \frac{\alpha_{s,a}\Gamma(\alpha_{s,a}+\frac{1}{2})\sqrt{\beta_{s,a}}}{(\alpha_{s,a}-\frac{1}{2})\Gamma(\alpha_{s,a})\Gamma(\frac{1}{2})\alpha_{s,a}\sqrt{2\lambda_{s,a}}} \left(1 + \frac{E^2[\mu_{s,a}]}{2\alpha_{s,a}}\right)^{-\alpha_{s,a}+\frac{1}{2}}.$$

The value of perfect information gives an upper bound on the myopic value of information for exploring action a. The expected *cost* incurred for this exploration is given by the difference between the value of a and the value of the current best action, i.e., $\max_{a'} E[Q(s,a')] - E[Q(s,a)]$. This suggests we choose the action that maximizes

$$VPI(s,a) - (\max_{a'} E[Q(s,a')] - E[Q(s,a)]).$$

Clearly, this strategy is equivalent to choosing the action that maximizes:

$$E[Q(s,a)] + VPI(s,a).$$

We see that the value of exploration estimate is used as a way of boosting the desirability of different actions. When the agent is confident of the estimated Q-values, the VPI of each action is close to 0, and the agent will always choose the action with the highest expected value.[1]

3.3 Updating Q-values

Finally, we turn to the question of how to update the estimate of the distribution over Q-values after executing a transition. The analysis of the updating step is complicated by the fact that a distribution over Q-values is a distribution over *expected, total* rewards, whereas the available observations are instances of *actual, local* rewards. Thus, we cannot use the Bayesian updating results in Theorem 3.1 directly.

Suppose that the agent is in state s, executes action a, receives reward r, and lands up in state t. We would like to know the complete sequence of rewards received from t onwards, but this is not available. Let R_t be a random variable denoting the discounted sum of rewards from t. If we assume that the agent will follow the apparently optimal policy, then R_t is distributed as R_{t,a_t}, where a_t is the action with the highest expected value at t.

We might hope to use this distribution to substitute in some way for the unknown future experiences. We now discuss two ways of going about this.

Moment updating The idea of moment updating is, notionally, to randomly sample values $R_t^1, \ldots, R_t^n$ from our distribution, and then update $P(R_{s,a})$ with the sample $r + \gamma R_t^1, \ldots, r + \gamma R_t^n$, where we take each sample to have weight $\frac{1}{n}$. Theorem 3.1 implies that we only need the first two moments of this sample to update our distribution. Assuming that n tends to infinity, these two moments are:

$$\begin{aligned} M_1 &= E[r + \gamma R_t] = r + \gamma E[R_t] \\ M_2 &= E[(r + \gamma R_t)^2] = E[r^2 + 2\gamma r R_t + \gamma^2 R_t^2] \\ &= r^2 + 2\gamma r E[R_t] + \gamma^2 E[R_t^2] \end{aligned}$$

Now, since our estimate of the distribution of R_t is a normal-gamma distribution over the mean and variance of R_t, we can use standard properties of normal-gamma distributions to compute the first two moments of R_t.

Lemma 3.4: *Let R be a normally distributed variable with unknown mean μ and unknown precision τ, and let $p(\mu, \tau) \sim NG(\mu_0, \lambda, \alpha, \beta)$. Then $E[R] = \mu_0$, and $E[R^2] = \frac{\lambda+1}{\lambda} \cdot \frac{\beta}{\alpha-1} + \mu_0^2$.*

Now we can update the hyperparameters $\rho_{s,a}$ as though we had seen a collection of examples with total weight 1, mean M_1, and second moment M_2.

This approach results in a simple closed-form equation for updating the hyperparameters for $R_{s,a}$. Unfortunately, it quickly becomes too confident of the value of the mean $\mu_{s,a}$. To see this, note that we can roughly interpret the parameter λ as the confidence in our estimate of the unknown mean. The method we just described updates μ_0 and λ with the mean of the unknown reward, which is just $r + \gamma E[R_t]$, as if we were confident of this being a true sample. Our uncertainty about the value of R_t is represented by the second moment M_2, which mainly affects the estimate of the variance of $R_{s,a}$. Thus, our uncertainty about R_t is not directly translated to uncertainty about the mean of $R_{s,a}$. Instead, it leads to higher estimate of the variance of $R_{s,a}$.The upshot of all this is that the precision of the mean increases too fast, leading to low exploration values and hence to premature convergence on sub-optimal strategies.

One ad-hoc way of dealing with this problem is to use *exponential forgetting*. This method reduces the impact of previously seen examples on the priors by a constant (which is usually close to 1) at each update. Due to space considerations, we do not review the details of this forgetting operation.

Mixture updating The problem described in the preceding section can be avoided by using the distribution over R_t in a slightly different way. Let $p(\mu_{s,a}, \tau_{s,a} \mid R)$ be the posterior distribution over $\mu_{s,a}, \tau_{s,a}$ after observing discounted reward R. If we observed the value $R_t = x$, then the updated distribution over $R_{s,a}$ is $p(\mu_{s,a}, \tau_{s,a} \mid r + \gamma x)$. We can capture our uncertainty about the value x by weighting these distribution by the probability that $R_t = x$. This results in the following *mixture* posterior:

$$p_{r,t}^{mix}(\mu_{s,a}, \tau_{s,a}) = \int_{-\infty}^{\infty} p(\mu_{s,a}, \tau_{s,a} \mid r + \gamma x) p(R_t = x) dx$$

Unfortunately, the posterior $p_{r,t}^{mix}(\mu_{s,a}, \tau_{s,a})$ does not have a simple representation, and so updating this posterior would lead to a more complex one, and so on. We can avoid this complexity by approximating $p_{r,t}^{mix}(\mu_{s,a}, \tau_{s,a})$ with a normal-gamma distribution after each update.

We compute the best normal-gamma approximation by minimizing the KL-divergence [6] from the true distribution.

Theorem 3.5: *Let $q(\mu, \tau)$ be some density measure over μ and τ and let $\epsilon > 0$. If we constrain α to be greater than $1+\epsilon$, the distribution $p(\mu, \tau) \sim NG(\mu_0, \lambda, \alpha, \beta)$ that minimizes the divergence $KL(q, p)$ is defined by the following equations:*

$$\begin{aligned} \mu_0 &= E_q[\mu\tau]/E_q[\tau] \\ \lambda &= \left(E_q[\mu^2\tau] - E_q[\tau]\mu_0^2\right)^{-1} \\ \alpha &= \max(1 + \epsilon, f(\log E_q[\tau] - E_q[\log \tau])) \\ \beta &= \alpha / E_q[\tau] \end{aligned}$$

where $f(x)$ is the inverse of $g(y) = \log y - \psi(y)$, and $\psi(x) = \frac{\Gamma'(x)}{\Gamma(x)}$ is the digamma *function.*

The requirement that $\alpha \geq 1+\epsilon$ is to ensure that $\alpha > 1$ so that the normal-gamma distribution is well defined. Although this theorem does not give a closed-form solution for α, we can find a numerical solution easily since $g(y)$ is a monotonically decreasing function [1].

Another complication with this approach is that it requires us to compute $E[\tau_{s,a}]$, $E[\tau_{s,a}\mu_{s,a}]$, $E[\tau_{s,a}\mu_{s,a}^2]$ and

[1] It is clear that the value of perfect information is an optimistic assessment of the value of performing a; by performing a once, we do not get perfect information about it, but only one more training instance. Thus, we might consider weighting the VPI estimate by some constant. We leave this for future work.

$E[\log \tau_{s,a}]$ with respect to $p_{r,t}^{mix}(\mu_{s,a}, \tau_{s,a})$. These expectations do not have closed-form solutions, but can be approximated by numerical integration, using formulas derived fairly straightforwardly from Theorem 3.5.

To summarize, in this section we discussed two possible ways of updating the estimate of the values. The first, *moment update* leads to an easy closed form update, but might become overly confident. The second, *mixture update*, is more cautious, but requires numerical integration.

4 Convergence

We are interested in knowing whether our algorithms converge to optimal policies in the limit. It suffices to show that the means $\mu_{s,a}$ converge to the true Q-values, and that the variance of the means converges to 0. If this is the case, then both the Q-value sampling and the myopic-VPI strategies will, eventually, execute an optimal policy.

Without going into details, the standard convergence proof [15] for Q-learning requires that each action is tried infinitely often in each state in an infinite run, and that $\sum_{n=0}^{\infty} \alpha(n) = \infty$ and $\sum_{n=0}^{\infty} \alpha(n)^2 < \infty$ where α is the learning rate. If these conditions are met, then the theorem shows that the approximate Q-values converge to the real Q-values.

Using this theorem, we can show that when we use moment updating, our algorithm converges to the correct mean.

Theorem 4.1: *If each action a is tried infinitely often in every state, and the algorithm uses moment updating, then the mean $\mu_{s,a}$ converges to the true Q-value for every state s and action a.*

Moreover, for moment updating we can also prove that the variance will eventually vanish:

Theorem 4.2: *If each action a is tried infinitely often in every state, and the algorithm uses the moment method to update the posterior estimates, then the variance* $\mathrm{Var}[\mu_{s,a}]$ *converges to 0 for every state s and action a.*

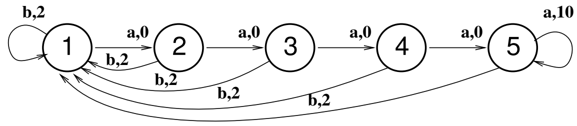
Combining these two results, we see that with moment updating, the procedure will converge on an optimal policy if all actions are tried eventually often. This is the case when we select actions by Q-value sampling.

If we select actions using myopic-VPI, then we can no longer guarantee that each action is tried infinitely often. More precisely, myopic VPI might starve certain actions and hence we cannot apply the results from [15]. Of course, we can define a "noisy" version of this action selection strategy (e.g., use a Boltzmann distribution over the adjusted expected values), and this will guarantee convergence.

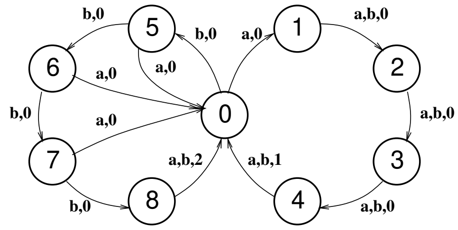
At this stage, we do not yet have counterparts to Theorems 4.1 and 4.2 for mixture updating. Our conjecture is that the estimated mean does converge to the true mean, and therefore similar theorems holds.

5 Experimental Results

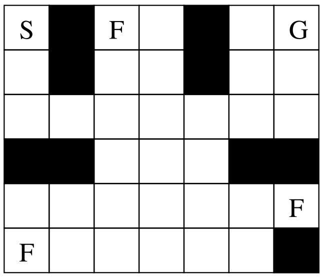
We have examined the performance of our approach on several different domains and compared it with a number of different exploration techniques. The parameters of each algorithm were tuned as well as possible for each domain. The algorithms we have used are as follows:



(a) Task 1 [11].



(b) Task 2 [14].



(c) Task 3. A navigation problem. S is the start state. The agent receives a reward upon reaching G based on the number of flags collected.

Figure 3: The three domains used in our experiments.

- **Semi-Uniform** Q-learning with semi-uniform random exploration.
- **Boltzmann** Q-learning with Boltzmann exploration.
- **Interval** Q-learning using Kaelbling's interval-estimation algorithm [10].
- **IEQL+** Meuleau's IEQL+ algorithm [11].
- **Bayes** Bayesian Q-learning as presented above, using either Q-value sampling or myopic-VPI to select actions, and either Moment updating or Mixture updating for value updates. These variants are denoted QS, VPI, Mom, Mix, respectively. Thus, there are four possible variants of the Bayesian Q-Learning algorithm, denoted, for example, as VPI+Mix.

We tested these learning algorithms on three domains:

- **Chain** This domain consists of the chain of states shown in Figure 3(a). It consists of six states and two actions a and b. With probability 0.2, the agent "slips"and actually performs the opposite action. The optimal policy for this domain (assuming a discount factor of 0.99) is to do action a everywhere. However, learning algorithms can get trapped at the initial state, preferring to follow the b–loop to obtain a series of smaller rewards.
- **Loop** This domain consists of two loops, as shown in Figure 3(b). Actions are deterministic. The problem here is that a learning algorithm may have already converged on action a for state 0 before the larger reward available in state 8 has been backed up. Here the optimal policy is to do action b everywhere.

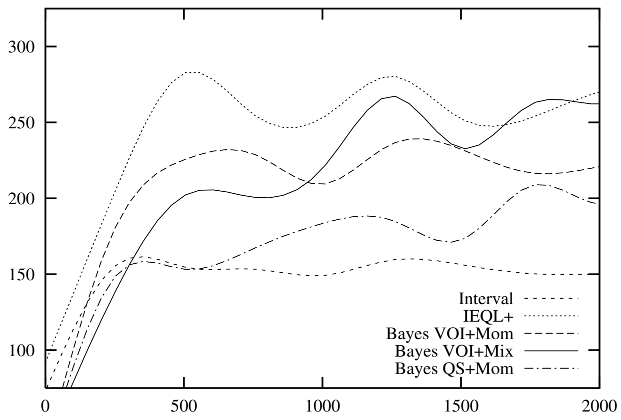
Domain	Method	1st Phase Avg.	1st Phase Dev.	2nd Phase Avg.	2nd Phase Dev.
chain	Uniform	1519.0	37.2	1611.4	34.7
	Boltzmann	1605.8	78.1	1623.4	67.1
	Interval	1522.8	180.2	1542.6	197.5
	IEQL+	2343.6	234.4	2557.4	271.3
	Bayes QS+Mom	1480.8	206.3	1894.2	364.7
	Bayes QS+Mix	1210.0	86.1	1306.6	102.0
	Bayes VPI+Mom	1875.4	478.7	2234.0	443.9
	Bayes VPI+Mix	1697.4	336.2	2417.2	650.1
loop	Uniform	185.6	3.7	198.3	1.4
	Boltzmann	186.0	2.8	200.0	0.0
	Interval	198.1	1.4	200.0	0.0
	IEQL+	264.3	1.6	292.8	1.3
	Bayes QS+Mom	190.0	19.6	262.9	51.4
	Bayes QS+Mix	203.9	72.2	236.5	84.1
	Bayes VPI+Mom	316.8	74.2	340.0	91.7
	Bayes VPI+Mix	326.4	85.2	340.0	91.7
maze	Uniform	105.3	10.3	161.2	8.6
	Boltzmann	195.2	61.4	1024.3	87.9
	Interval	246.0	122.5	506.1	315.1
	IEQL+	269.4	3.0	253.1	7.3
	Bayes QS+Mom	132.9	10.7	176.1	12.2
	Bayes QS+Mix	128.1	11.0	121.9	9.9
	Bayes VPI+Mom	403.2	248.9	660.0	487.5
	Bayes VPI+Mix	817.6	101.8	1099.5	134.9

Table 1: Average and standard deviation of accumulated rewards over 10 runs. A phase consists of 1,000 steps in chain and loop, and of 20,000 steps in maze.

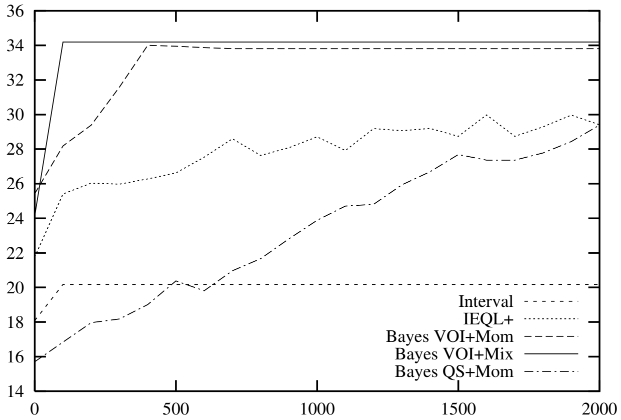
Maze This is a maze domain where the agent attempts to "collect" flags and get them to the goal. In the experiments we used the maze shown in Figure 3(c). In this figure, S marks the start state, G marks the goal state, and F marks locations of flags that can be collected. The reward received on reaching G is based on the number of flags collected. Once the agent reaches the goal, the problem is reset. There are a total of 264 states in this MDP. The agent has four actions—up, down, left, and right. There is a small probability, 0.1, that the agent will slip and actually perform an action that goes in a perpendicular direction. If the agent attempts to move into a wall, its position does not change. The challenge is to do sufficient exploration to collect all three flags before reaching the goal.

The first two domains are designed so that there are sub-optimal strategies that can be exploited. Thus, if the learning algorithm converges too fast, then it will not discover the higher-scoring alternatives. The third domain is larger and less "tricky" although it also admits inferior policies. We use it to evaluate how the various exploration stratgies scale up.

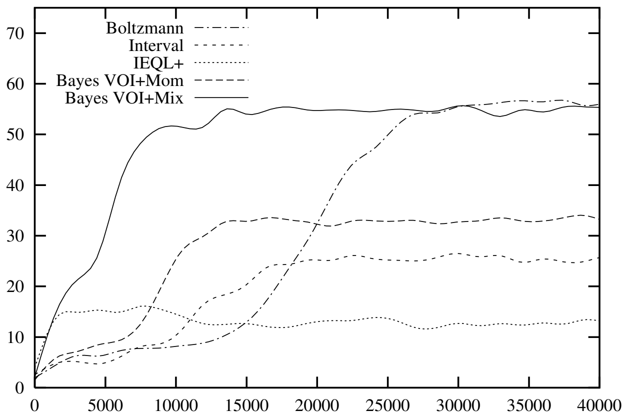
There are several ways of measuring the performance of learning algorithms. For example, we might want to measure the quality of the policy they "recommend" after some number of steps. Unfortunately, this might be misleading, since the algorithm might recommend a good exploiting policy, but might still continue to explore, and thus receive much smaller rewards. We measured the performance of the learning algorithms by the total reward collected during a fixed number of time steps (Table 1). Additionally, we measured the discounted total reward-to-go at each point in the run. More precisely, suppose the agent receives rewards $r_1, r_2, \ldots, r_N$ in a run of length N. Then we define the reward-to-go at time t to be $\sum_{t' \geq t} r_{t'} \gamma^{t'-t}$. Of course, this estimate is reliable only for points that are far enough from the end of the run. In Figure 4, we plot the average reward-to-go as a function of t by averaging these values over 10 runs with different random



(a) Results for the chain domain.



(b) Results for the loop domain.



(c) Results for the maze domain.

Figure 4: Plots of actual discounted reward (y-axis) as a function of number of steps (x-axis) for several methods in three domains. The curves are avarege of 10 runs for each method. The curves for chain and maze were smoothed.

seeds.[2]

Our results show that in all but the smallest of domains our methods are competitive with or superior to state of the art exploration techniques such as IEQL+. Our analysis suggests that this is due to our methods' more effective use of small numbers of data points. Results from the maze domain in particular show that our VPI-based methods begin directing the search towards promising states after making significantly fewer observations than IEQL+ and interval estimation. Overall, we have found that using mixture updating combined with VPI for action selection gives the best performance, and expect these to be the most valuable techniques as we expand this work to model-based learning.

One weakness of our algorithms is that they have significantly more parameters than IEQL+ or interval estimation. In the full version of the paper we analyze the dependence of these results on various parameters. The main parameters that seem to effect the performance of our method is the variance of the initial prior, that is, the ratio $\frac{\beta}{\lambda(\alpha-1)}$. Priors with larger variances usually lead to better performance.

6 Conclusion

We have described a Bayesian approach to Q-learning in which exploration and exploitation are directly combined by representing Q-values as probability distributions and using these distributions to select actions. We proposed two methods for action selection — Q-value sampling and myopic-VPI. Experimental evidence has shown that (at least for some fairly simple problems) these approaches explore the state space more effectively than conventional model-free learning algorithms, and that their performance advantage appears to increase as the problems become larger. This is due to an action selection mechanism that takes advantage of much more information than previous approaches.

A major issue for this work is that the computational requirements are greater than for conventional Q-learning, both for action selection and for updating the Q-values. However, we note that in most applications of reinforcement learning, performing actions is more expensive tha computation time.

We are currently investigating ways to use a Bayesian approach such as this with model-based reinforcement algorithms. In this case, we explicitly represent our uncertainty about the dynamics of the system to estimate the usefulness of exploration. We are also investigating alternative action selection schemes, and approximations that could be used to reduce the computational requirements of this algorithm. Finally, it should be possible to use function approximators to extend this work to problems with large and/or continuous state spaces. There is a well-understood theory of Bayesian neural network learning [4, Ch. 10] that allows posterior means and variances to be computed for each point in the input space; these can be fed directly into our algorithm.

[2] We performed parameter adjustment to find the best-performing parameters for each method. Thus the results reported for each algorithm are probably somewhat optimistic. In the full version of the paper we intend to also show the sensitivity of each method to changes in the parameters.

Acknowledgments

We are grateful for useful comments from David Andre, Craig Boutilier, Daphne Koller and Ron Parr. We thank Nicolas Meuleau for help in implementing the IEQL+ algorithm. Nir Friedman and Stuart Russell were supported in part by ARO under the MURI program "Integrated Approach to Intelligent Systems", grant number DAAH04-96-1-0341, and by ONR under grant number N00014-97-1-0941.

References

[1] M. Abramowitz and I. A. Stegun, editors. *Handbook of Mathematical Functions*. Dover, 1964.

[2] R. E. Bellman. *Dynamic Programming*. Princeton Univ. Press, 1957.

[3] D. A. Berry and B. Fristedt. *Bandit Problems: Sequential Allocation of Experiments*. Chapman and Hall, 1985.

[4] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford Univ. Press, 1995.

[5] B. W. Brown, J. Lovato, and K. Russell. Library of routines for cumulative distribution functions, inverses, and other parameters, 1997. `ftp://odin.mdacc.tmc.edu/pub/source/dcdflib.c-1.1.tar.gz`.

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, 1991.

[7] M. H. Degroot. *Proability and Statistics*. Addison-Wesley, 1986.

[8] R. A. Howard. Information value theory. *IEEE Trans. Systems Science and Cybernetics*, SSC-2:22–26, 1966.

[9] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *J. Artificial Intelligence Research*, 4:237–285, 1996.

[10] L. P. Kaelbling. *Learning in Embedded Systems*. MIT Press, 1993.

[11] N. Meuleau and P. Bourgine. Exploration of multi-states environments: Local measures and back-propogation of uncertainty. *Machine Learning*, 1998. To appear.

[12] S. J. Russell and E. H. Wefald. *Do the Right Thing: Studies in Limited Rationality*. MIT Press, 1991.

[13] S. B. Thrun. The role of exploration in learning control. In D. A. White and D. A. Sofge, eds., *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*. Van Nostrand Reinhold, 1992.

[14] C. J. Watkins. *Models of Delayed Reinforcement Learning*. PhD thesis, Psychology Department, Cambridge University, 1989.

[15] C. J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[16] S. D. Whitehead and D. H. Ballard. Learning to percieve and act by trial and error. *Machine Learning*, 7:45–83, 1991.

[17] J. Wyatt. *Exploration and Inference in Learning from Reinforcement*. PhD thesis, Department of Artificial Intelligence, University of Edinburgh, 1997.