# Robust Bayesian reinforcement learning through tight lower bounds

Christos Dimitrakakis[1]

EPFL, Lausanne, Switzerland
christos.dimitrakakis@epfl.ch

**Abstract.** In the Bayesian approach to sequential decision making, exact calculation of the (subjective) utility is intractable. This extends to most special cases of interest, such as reinforcement learning problems. While utility bounds are known to exist for this problem, so far none of them were particularly tight. In this paper, we show how to efficiently calculate a lower bound, which corresponds to the utility of a *memoryless* policy for the decision problem, which is generally different from both the Bayes-optimal policy and the policy which is optimal for the mean MDP. We then show how these can be applied to obtain robust exploration policies in a Bayesian reinforcement learning setting.

## 1 Setting

We consider decision making problems where an agent is acting in a (possibly unknown to it) environment. By choosing actions, the agent changes the state of the environment and in addition obtains scalar rewards. The agent acts so as to maximise the expectation of the utility function: $U_t \triangleq \sum_{k=t}^{T} \gamma^k r_k$, where $\gamma \in [0, 1]$ is a discount factor and where the instantaneous rewards $r_t \in \mathbb{R}$ are drawn from a Markov decision process (MDP) $\mu$, defined on a state space $\mathcal{S}$ and an action space $\mathcal{A}$, both equipped with a suitable metric and $\sigma$-algebra, with a set of transition probability measures $\left\{\mathcal{T}_\mu^{s,a} \,\middle|\, s \in \mathcal{S}, a \in \mathcal{A}\right\}$ on $\mathcal{S}$ , and a set of reward probability measures $\left\{\mathcal{R}_\mu^{s,a} \,\middle|\, s \in \mathcal{S}, a \in \mathcal{A}\right\}$ on $\mathbb{R}$, such that:

$$r_t \mid s_t = s, a_t = a \sim \mathcal{R}_\mu^{s,a}, \qquad s_{t+1} \mid s_t = s, a_t = a \sim \mathcal{T}_\mu^{s,a}, \tag{1.1}$$

where $s_t \in \mathcal{S}$ and $a_t \in \mathcal{A}$ are the state of the MDP, and the action taken by the agent at time $t$, respectively. The environment is controlled via a *policy* $\pi \in \mathcal{P}$. This defines a conditional probability measure on the set of actions, such that $\mathbb{P}_\pi(a_t \in A \mid s^t, a^{t-1}) = \pi(A \mid s^t, a^{t-1})$ is the probability of the action taken at time $t$ being in $A$, where we use $\mathbb{P}$, with appropriate subscripts, to denote probabilities of events and $s^t \triangleq s_1, \ldots, s_t$ and $a^{t-1} \triangleq a_1, \ldots, a_{t-1}$ denotes sequences of states and actions respectively. We use $\mathcal{P}_k$ to denote the set of $k$-order Markov policies. Important special cases are the set of *blind* policies $\mathcal{P}_0$ and the set of *memoryless* policies $\mathcal{P}_1$. A policy in $\pi \in \bar{\mathcal{P}}_k \subset \mathcal{P}_k$ is *stationary*, when $\pi(A \mid s_{t-k+1}^t, a_{t-k+1}^{t-1}) = \pi(A \mid s^k, a^{k-1})$ for all $t$.

The expected utility, conditioned on the policy, states and actions is used to define a *value function* for the MDP $\mu$ and a stationary policy $\pi$, at stage $t$:

$$Q^\pi_{\mu,t}(s,a) \triangleq \mathbb{E}_{\mu,\pi}(U_t \mid s_t = s, a_t = a), \qquad V^\pi_{\mu,t}(s) \triangleq \mathbb{E}_{\mu,\pi}(U_t \mid s_t = s), \tag{1.2}$$

where the expectation is taken with respect to the process defined jointly by $\mu, \pi$ on the set of all state-action-reward sequences $(\mathcal{S}, \mathcal{A}, \mathbb{R})^*$. The *optimal* value function is denoted by $Q^*_{\mu,t} \triangleq \sup_\pi Q^\pi_{\mu,t}$ and $V^*_{\mu,t} \triangleq \sup_\pi V^\pi_{\mu,t}$. We denote the optimal policy[1] for $\mu$ by $\pi^*_\mu$. Then $Q^*_{\mu,t} = Q^{\pi^*_\mu}_{\mu,t}$ and $V^*_{\mu,t} = V^{\pi^*_\mu}_{\mu,t}$.

There are two ways to handle the case when the true MDP is unknown. The first is to consider a set of MDPs such that the probability of the true MDP lying outside this set is bounded from above [e.g. 15, 16, 3, 14, 19, 18]. The second is to use a Bayesian framework, whereby a full distribution over possible MDPs is maintained, representing our subjective belief, such that MDPs which we consider more likely have higher probability [e.g. 9, 6, 22, 1, 8]. Hybrid approaches are relatively rare [11]. In this paper, we derive a method for efficiently calculating near-optimal, robust, policies in a Bayesian setting.

### 1.1 Bayes-optimal policies

In the Bayesian setting, our uncertainty about the Markov decision process (MDP) is formalised as a probability distribution on the class of allowed MDPs. Finding the optimal policy under this type of uncertainty is intractable in most cases of interest [7, 9, 13]. More precisely, assume a probability measure $\xi$ over a set of possible MDPs $\mathcal{M}$, representing our belief. Our goal is to discover a policy $\pi$ maximising the expected utility with respect to the belief $\xi$:

$$\mathbb{E}_{\xi,\pi} U_t = \int_{\mathcal{M}} \mathbb{E}_{\mu,\pi}(U_t) \, \mathrm{d}\xi(\mu). \tag{1.3}$$

Without loss of generality, we may assume that all MDPs in $\mathcal{M}$ share the same state and action space. For compactness, and with minor abuse of notation, we define the following value functions with respect to the belief:

$$Q^\pi_{\xi,t}(s,a) \triangleq \mathbb{E}_{\xi,\pi}(U_t \mid s_t = s, a_t = a), \qquad V^\pi_{\xi,t}(s) \triangleq \mathbb{E}_{\xi,\pi}(U_t \mid s_t = s), \tag{1.4}$$

which represent the expected utility under the belief $\xi$, at stage $t$, of policy $\pi$, conditioned on the current state and action.

**Definition 1 (Bayes-optimal policy).** *A Bayes-optimal policy $\pi^*_\xi$ with respect to a belief $\xi$ is a policy maximising (1.3). Similarly to the known MDP case, we use $Q^*_{\xi,t}, V^*_{\xi,t}$ to denote the value functions of the Bayes-optimal policy.*

It is important to note that a Bayes-optimal policy is not necessarily the same as the optimal policy for the true MDP. Rather, it is the optimal policy given that the true MDP was drawn at the start of the experiment from the distribution $\xi$. All the theoretical development in this paper is with respect to $\xi$.

[1] We assume that there exists at least one optimal policy. If there are multiple optimal policies, we choose arbitrarily among them.

### 1.2 Related work and main contribution

Computation of the Bayes-optimal policy is generally intractable [9]. In this work we provide a simple algorithm for finding near-optimal *memoryless* policies in polynomial time. By definition, for any belief $\xi$, the expected utility under that belief of any policy $\pi$ is a lower bound on that of the optimal policy $\pi^*_\xi$. Consequently, the near-optimal memoryless policy gives us a tight lower bound on the subjective utility.

A similar idea was used in [8], where the stationary policy that is optimal on the *expected MDP* is used to obtain a lower bound. This is later refined through a stochastic branch-and-bound technique that employs a similar upper bound. In a similar vein, [12] uses approximate Bayesian inference to obtain a stationary policy for the current belief. More specifically, they consider two families of expectation maximisation algorithms. The first uses a variational approximation to the reward-weighted posterior of the transition distribution, while the second performs expectation propagation on the first two moments. However, none of the above approaches return the optimal stationary policy.

It is worthwhile to mention the very interesting point-based Beetle algorithm of Poupart et al. [17], which discretised the belief space by sampling a set of future beliefs (rather than MDPs). Using the convexity of the utility with respect to the belief, they constructed a lower bound via a piecewise-linear approximation of the complete utility from these samples. The approach results in an approximation to the optimal non-stationary policy. Although the algorithm is based on an optimal construction reported in the same paper, sufficient conditions for its optimality are not known.

In this paper, we obtain a tight lower bound for the *current* belief by calculating a nearly optimal *memoryless* policy. The procedure is computationally efficient, and we show that it can be used in practice to perform robust Bayesian exploration in unknown MDPs. This is achieved by computing a new optimal memoryless policy once our belief has changed significantly. The latter technique was also employed by other approaches [14, 2, 1, 20, 22]. In addition, it can be seen as a principled generalisation of the sampling approach suggested in [20]. The crucial difference is that, unlike previous approaches which use some form of *optimistic* policy, we instead employ a more conservative policy in each stationary interval. This can be significantly better than the policy which is optimal for the expected MDP.

The first problem we tackle is how to compute this policy given a belief over a finite number of MDPs. For this, we provide a simple algorithm based on backwards induction [see 7, for example]. In order to extend this approach to an arbitrary MDP set, we employ Monte Carlo sampling from the current posterior. Unlike other Bayesian sampling approaches [6, 20, 1, 4, 21, 8, 22], we use these samples to estimate a policy that is nearly optimal (within the restricted set of memory policies) with respect to the distribution these samples were drawn from.

## 2 Backwards induction on multi-MDPs

Even when our belief $\xi$ is a probability measure over a finite set of MDPs $\mathcal{M}$, the problem of finding an optimal policy remains intractable. We reduce it to the problem of finding the memoryless policy $\pi$ maximising the expected utility, if the policy will be applied to an MDP randomly chosen with distribution $\xi$.

We can approximate the optimal *memoryless* policy with respect to $\xi$, by setting $\xi(\mu \mid s_t = s, \pi) = \xi(\mu)$, and then using the backwards induction procedure shown below. By definition:

$$Q^\pi_{\xi,t}(s,a) = \mathbb{E}_{\xi,\pi}(r_t \mid s_t = s, a_t = a) + \gamma \, \mathbb{E}_{\xi,\pi}(U_{t+1} \mid s_t = s, a_t = a), \tag{2.1}$$

where the expected reward term can be written as

$$\mathbb{E}_{\xi,\pi}(r_t \mid s_t = s, a_t = a) = \int_{\mathcal{M}} \mathbb{E}_\mu(r_t \mid s_t = s, a_t = a) \, \mathrm{d}\xi(\mu), \tag{2.2a}$$

$$\mathbb{E}_\mu(r_t \mid s_t = s, a_t = a) = \int_{-\infty}^{\infty} r \, \mathrm{d}\mathcal{R}^{s,a}_\mu(r). \tag{2.2b}$$

The next-step utility can be written as:

$$\mathbb{E}_{\xi,\pi}(U_{t+1} \mid s_t = s, a_t = a) = \int_{\mathcal{M}} \mathbb{E}_{\mu,\pi}(U_{t+1} \mid s_t = s, a_t = a) \, \mathrm{d}\xi(\mu), \tag{2.3a}$$

$$\mathbb{E}_{\mu,\pi}(U_{t+1} \mid s_t = s, a_t = a) = \int_{\mathcal{S}} V^\pi_{\mu,t+1}(s') \, \mathrm{d}\mathcal{T}^{s,a}_\mu(s'). \tag{2.3b}$$

Putting those steps together, we obtain Algorithm 1, which greedily calculates a memoryless policy for a $T$-stage problem and returns its expected utility. The

**Algorithm 1** MMBI - Backwards induction on multiple MDPs.

**procedure** MMBI($\mathcal{M}, \xi, \gamma, T$)
  Set $V_{\mu,T+1}(s) = 0$ for all $s \in \mathcal{S}$.
  **for** $t = T, T-1, \ldots, 0$ **do**
    **for** $s \in \mathcal{S}, a \in \mathcal{A}$ **do**
      Calculate $Q_{\xi,t}(s,a)$ from (2.1) using $\{V_{\mu,t+1}\}$ .
    **end for**
    **for** $s \in \mathcal{S}$ **do**
      $a^*_{\xi,t}(s) = \arg\max \{Q_{\xi,t}(s,a) \,|\, a \in \mathcal{A}\}$.
      **for** $\mu \in \mathcal{M}$ **do**
        $V_{\mu,t}(s) = Q_{\mu,t}(s, a^*_{\xi,t}(s))$.
      **end for**
    **end for**
  **end for**
**end procedure**

calculation is greedy, since optimising over $\pi$ implies that at any step $t + k$, we

must condition the belief on past policy steps $\xi(\mu \mid s_{t+k} = s, \pi_t, \ldots, \pi_{t+k-1})$ to calculate the expected utility correctly. Thus, the optimal $\pi_{t+k}$ depends on both future and past selections. Nevertheless, it is easy to see that Alg. 1 returns the correct expected utility for time step $t$. Theorem 1 bounds the gap between this and the Bayes-optimal value function when the difference between $\xi(\mu \mid s_t = s, \pi)$ and $\xi(\mu)$ is small:

**Theorem 1.** *For any $k \in [t, T]$, let $\xi_k \triangleq \xi(\cdot \mid s^k, a^k)$ be the posterior after $k$ observations. If $\|\xi_t - \xi_T\|_{\lambda,1} \leq \epsilon$, where $\lambda$ is a dominating measure on $\mathcal{M}$, then the value function $V_{\xi,t}(s)$ obtained by Algorithm 1 satisfies:*

$$\|V_{\xi,k}(s) - \mathbb{E}^{\pi}_{\xi_t}(U_k \mid s^k, a^k)\|_{\lambda,1} \leq \frac{r_{max}}{(1-\gamma)^2}\gamma^{k-t}\epsilon\lambda(\mathcal{M}). \tag{2.4}$$

*Proof.* For $k > t$, we have:

$$\begin{aligned}|V_{\xi,k}(s) - \mathbb{E}_{\xi}(U_k \mid s^t, a^t)| &= \left|\int_{\mathcal{M}} [\xi_t(\mu) - \xi_k(\mu)(s)] V_{\mu,k}(s)\, \mathrm{d}\lambda(\mu)\right| \\ &\leq \frac{r_{\max}}{1-\gamma}\int_{\mathcal{M}} |\xi_t(\mu) - \xi_k(\mu)(s)|\, \mathrm{d}\lambda(\mu) \leq \frac{r_{\max}}{1-\gamma}\epsilon\lambda(\mathcal{M}).\end{aligned}$$

The final result is obtained via the geometric series. □

Finally, the $\xi$-optimal memoryless policy is generally *different* from the policy which is optimal with respect to the expected MDP $\hat{\mu}_\xi \triangleq \mathbb{E}_\xi \mu$, as can be seen via counterexample where $\mathbb{E}_\xi V^\pi_\mu \neq V^\pi_{\hat{\mu}_\xi}$, or even where $\mathbb{E}_\xi \mu \notin \mathcal{M}$. This will also be seen in the experiments described in Sec. 3, where near-optimal memoryless policies are compared against the $\hat{\mu}_\xi$-optimal policy.

### 2.1 Computational complexity

When $\mathcal{M}$ is finite and $T < \infty$, MMBI (Alg. 1) returns a greedily-optimised policy $\pi_{\mathrm{MMBI}}$ and its value function. When $T \to \infty$, MMBI can be used to calculate an $\epsilon$-optimal greedy approximation by truncating the horizon, as shown below.

**Lemma 1.** *Assuming $r_t \in [0, r_{max}]$, Alg. 1 requires $\mathcal{O}\left(\left[|\mathcal{M}||\mathcal{S}|^2(|\mathcal{A}|+1) + (1+|\mathcal{M}|)|\mathcal{S}||\mathcal{A}|\right]\log_\gamma \frac{\epsilon(1-\gamma)}{r_{max}}\right)$ operations to bound the value function error by $\epsilon$.*

*Proof.* Since $r_t$ *in*$[0, r_{\max}]$, if we look up to some horizon $T$, our value function error is bounded by $\gamma^T c$, where $c = H r_{\max}$ and $H = \frac{1}{1-\gamma}$ is the effective horizon. Consequently, we need $T \geq \log_\gamma(\epsilon/c)$ to bound the error by $\epsilon$. For each $t$, step 5 is performed $|\mathcal{S}||\mathcal{A}|$ times. Each step takes $O(|\mathcal{M}|)$ operations for the expected reward and $O(|\mathcal{S}||\mathcal{M}|)$ operations for the next-step expected utility. The second loop is $\mathcal{O}(|\mathcal{S}|(|\mathcal{A}| + |\mathcal{M}||\mathcal{S}|))$, since it is performed $|\mathcal{S}|$ times, with the max operators taking $|\mathcal{A}|$ operations, while inner loop is performed $|\mathcal{M}|$ times with each local MDP update step 10 takes $|\mathcal{S}|$ operations. □

It is easy to see that the most significant term is $\mathcal{O}(|\mathcal{M}||\mathcal{S}|^2|\mathcal{A}|)$, so the algorithmic complexity scales linearly with the number of MDPs. Consequently, when $\mathcal{M}$ is not finite, exact computation is not possible. However, we can use high probability bounds to bound the expected loss of a policy calculated stochastically through MSBI (Alg.2).

**Algorithm 2** MSBI: Multi-Sample Backwards Induction

**procedure** MSBI($\xi, \gamma, \epsilon$)
  $n = \left(\frac{3r_{\max}}{\epsilon(1-\gamma)}\right)^3$.
  $\mathcal{M} = \{\mu_1, \dots, \mu_n\}$, $\mu_i \sim \xi$.
  MMBI($\mathcal{M}, p, \gamma, \log_\gamma \frac{\epsilon(1-\gamma)}{r_{\max}}$), with $p(\mu_i) = 1/n$ for all $i$.
**end procedure**

MSBI simply takes a sufficient number of samples of MDPs from $\xi$, so that in $\xi$-expectation, the loss relative to the optimal stationary policy is bounded according to the following lemma.

**Lemma 2.** *The expected utility $\mathbb{E}_\xi U$ is within $3cn^{-1/3}$ of the optimal stationary policy. Thus, $n = (3c/\epsilon)^3$ is sufficient to bound the loss by $\epsilon$, where $c = Hr_{max}$ and $H = \frac{1}{1-\gamma}$ is the effective horizon.*

*Proof.* Let $\hat{\mathbb{E}}^n U = \frac{1}{n}\sum_{i=1}^n \mathbb{E}_{\mu_i} U$ denote the empirical expected utility over the sample of $n$ MDPs, where the policy subscript $\pi$ is omitted for simplicity. Since $\mathbb{E}_\xi \hat{\mathbb{E}}^n U = \mathbb{E}_\xi U$, we can use the Hoeffding inequality to obtain:

$$\xi\left(\left\{\mu^n \,\middle|\, \hat{\mathbb{E}}^n U \geq \mathbb{E}_\xi U + \epsilon\right\}\right) \leq e^{-2n\epsilon^2/c^2}.$$

This implies a bound

$$\mathbb{E}_\xi(\hat{\mathbb{E}}^n U - \mathbb{E}_\xi U) \leq c\delta + c\sqrt{\frac{\ln(1/\delta)}{2n}} \leq c(8n)^{-1/3} + c\sqrt{\frac{(8n)^{1/3}}{2n}} = 3cn^{-1/3}.$$

Let $\bar{\mathcal{P}}_1 \subset \mathcal{P}_1$ be the set of stationary policies. Since the bound holds uniformly (for any $\pi \in \mathcal{P}$), the policy $\hat{\pi}^* \in \bar{\mathcal{P}}_1$ maximising $\hat{\mathbb{E}}^n$ is within $3cn^{-1/3}$ of the $\xi$-optimal policy in $\bar{\mathcal{P}}_1$. □

Finally, we can combine the above results to bound the approximation error of MSBI with respect to expected loss:

**Theorem 2.** MSBI *(Alg. 2) requires $\mathcal{O}\left(\left(\frac{6r_{max}}{\epsilon(1-\gamma)}\right)^3 |\mathcal{S}|^2|\mathcal{A}| \log_\gamma \frac{\epsilon(1-\gamma)}{2r_{max}}\right)$ operations to be $\epsilon$-close to the best MMBI policy.*

*Proof.* From Lem. 2, we can set $n = (6c/\epsilon)^3$ to bound the regret by $\epsilon/2$. Using the same value in Lem. 1, and setting $|\mathcal{M}| = n$, we obtain the required result. □

MMBI can be used to obtain a much tighter value function bound than the mean-MDP-optimal policy. This can be seen in Fig. 1, where the lower bounds are compared with a simple upper bound.

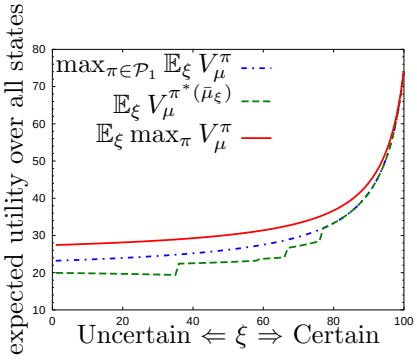



**Fig. 1.** Value function bound.

### 2.2 Application to robust Bayesian reinforcement learning

While MSBI can be used to obtain a stationary policy which is in expectation close to the optimal stationary policy for a given belief, the question is how to extend the procedure to online reinforcement learning. The simplest possible approach is to simply recalculate the stationary policy after some interval $B > 0$. This is the approach followed by MCBRL (Alg. 3), shown below.

**Algorithm 3** MCBRL: Monte-Carlo Bayesian Reinforcement Learning

**procedure** MCBRL($\xi_0, \gamma, \epsilon, B$)
  Calculate $\xi_t(\cdot) = \xi_0(\cdot \mid s^t, a^{t-1})$.
  Call MSBI($\xi_t, \gamma, \epsilon$) and run returned policy for $B$ steps.
**end procedure**

## 3 Experiments

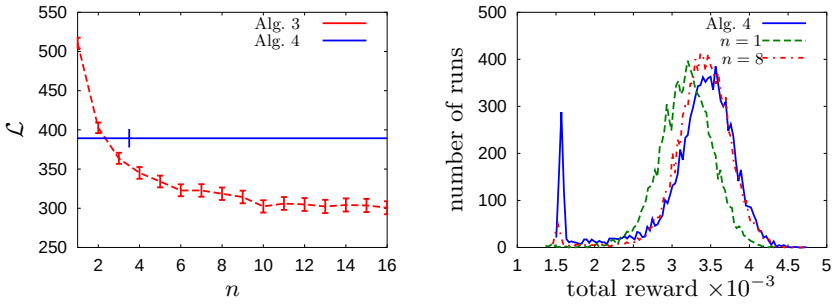
Selecting the number of samples $n$ according to $\epsilon$ for MCBRL is computationally prohibitive. In practice, instead of setting $n$ via $\epsilon$, we simply consider increasing values of $n$. For a single sample ($n = 1$), MCBRL is equivalent to the sampling method in [20], which at every new stage, samples a single MDP from the current posterior and then uses the policy that is optimal for the sampled MDP.

We also compared MCBRL against the common heuristic of acting according the policy that is optimal with respect to the *expected* MDP $\hat{\mu}_\xi \triangleq \mathbb{E}_\xi \mu$. The algorithm, referred to as the Exploit heuristic in [17], is shown in detail in Alg. 4. At every step, this calculates the expected MDP by obtaining the expected transition kernel and reward function under the current belief. It then acts according to the optimal policy with respect to $\hat{\mu}_\xi$. This policy may be much worse than the optimal policy, even within the class of stationary policies $\bar{\mathcal{P}}_1$.

We compared the algorithms on the Chain task [5], which is commonly used to evaluate the quality of exploration in reinforcement learning algorithms. Traditionally, the task has a horizon of 1000 steps, a discount factor $\gamma = 0.95$ is

**Algorithm 4** EXPLOIT: Expected MDP exploitation [17]

**procedure** EXPLOIT($\xi_0, \gamma$)
  **for** $t = 1, \dots$ **do**
    Calculate $\xi_t(\cdot) = \xi_0(\cdot \mid s^t, a^{t-1})$.
    Estimate $\hat{\mu}_{\xi_t} \triangleq \mathbb{E}_{\xi_t} \mu$.
    Calculate $Q^*_{\hat{\mu}_{\xi_t}}(s, a)$ using discount parameter $\gamma$.
    Select $a_t = \arg\max_a Q^*_{\hat{\mu}_{\xi_t}}(s, a)$
  **end for**
**end procedure**




(a) Expected regret estimate (b) Empirical performance distribution

**Fig. 2.** Performance on the chain task. Figure 2(a) shows the expected regret during the first $10^3$ steps of the chain task, relative to the optimal (oracle) policy. The *sampling* curve shows the estimated expected regret of Alg. 3, as the number of samples increases, averaged over $10^4$ runs, with 95% confidence interval calculated via a $10^4$-boostrap. The *expected* curve shows the performance of an algorithm acting greedily with respect to the mean MDP. Figure 2(b) shows the empirical distribution of total rewards obtained in $10^4$ runs, for: the *expected* MDP approach and MCBRL with $n = 1$ and $n = 8$ samples.

used, and the expected total reward $\mathbb{E}_{\mu,\pi} \sum_{t=1}^{T} r_t$ is compared. We also report the expected utility $\mathbb{E}_{\mu,\pi} U_t$, which depends on the discount factor. All quantities are estimated over $10^4$ runs with appropriately seeded random number generators to reduce variance.[2] The initial belief about the state transition distribution was set to be a product-Dirichlet prior [see 7] with all parameters equal to $|\mathcal{S}|^{-1}$, while a product-Beta prior with parametrs $(1, 1)$ was used for the rewards.

Figure 2 summarises the results in terms of total reward. The left hand side (2(a)) shows the expected difference in total reward between the optimal policy $\pi^*$ and the used policy $\pi$, over $T$ steps, otherwise known as the regret: $\mathcal{L} = \mathbb{E}_{\mu,\pi} \sum_{t=1}^{T} r_t - \mathbb{E}_{\mu,\pi} \sum_{t=1}^{T} r_t$. The averages are calculated over $10^4$ runs, while the error bars denote 95% confidence intervals obtained via a $10^4$-bootstrap [10]. We can see that for $n = 1$, MCBRL performs worse than the expected MDP approach, in terms of *total reward*. On the other hand, as the number of samples increase, the performance of the multi-sample estimation monotonically improves.

Some more detail on the overall behaviour of the algorithms can be seen in Figure 2(b), which shows the empirical performance distribution in terms of total reward. It is clear that the expected MDP approach has a high probability of getting stuck in a sub-optimal regime. On the other hand MCBRL, for $n = 1$, results in significant over-exploration of the environment. However, as $n$ increases, MCBRL explores significantly less, while the number of runs where we are stuck in the sub-optimal regime remains small ($< 1\%$ of the runs).

| Model | $\sum_{t=1}^{1000} r_t$ ($\mathbb{E}\, U$) | 80% percentile | confidence interval |
|---|---|---|---|
| Alg. 4 | 3287 (26.64) | 2518 – 3842 | 3275 – 3299 |
| $n = 1$ | 3166 (28.50) | 2748 – 3582 | 3159 – 3173 |
| $n = 8$ | 3358 (29.65) | 2932 – 3800 | 3350 – 3366 |
| $n = 16$ | 3376 (29.95) | 2946 – 3830 | 3368 – 3384 |
| Model | $\sum_{t=1}^{1000} r_t$ | Standard interval | |
| Beetle [17] | 1754 | 1712–1796 | |
| AMP-EM [12] | 2180 | 2108–2254 | |
| SEM [12] | 2052 | 2000 –2111 | |

**Table 1.** Comparative results on the chain task. The 80% percentile interval is such that no more than 10% of the runs were above the maximum or below the minimum value respectively. The confidence interval shows the accuracy of the mean estimate, calculated as the 95% bootstrap interval. The results for Beetle and the EM algorithms were obtained from the cited papers, with and the interval based on the reported standard deviation.

Table 1 presents comparative results on the chain task for Alg. 4 and for MCBRL for $n \in \{1, 8, 16\}$ in terms of the total reward received in $10^3$ steps. This enables us to compare against the results reported in [17, 12]. While the

[2] In both cases this expectation is with respect to the distribution induced by the actual MDP $\mu$ and policy $\pi$ followed, rather than with respect to the belief $\xi$.

performance of Alg. 4 may seem surprisingly good, it is actually in line with the results reported in [17]. Therein, BEETLE only outperformed Alg. 4 in the *Chain* task when stronger priors were used. In addition, we would like to note that while the case $n = 1$ is worse than Alg. 4 for the total reward metric, this no longer holds when we examine the expected utility, where an improvent can already be seen for $n = 1$.

## 4 Discussion

This paper introduced MMBI, a simple backwards induction procedure, to obtain the optimal stationary policy with respect to a belief over a finite number of MDPs. We subsequently generalised this to MSBI, a stochastic procedure, which finds a policy whose loss is in expectation close to that of the optimal stationary policy, with a gap that depends polynomially on the number of samples, for a belief on arbitrary set of MDPs. This is then applied to reinforcement learning problems by using the MCBRL algorithm to sample a number of MDPs at regular intervals. This can be seen as a principled generalisation of [20], which only draws one sample at each interval. Then MSBI is used to calculate a near-optimal stationary policy within each interval. This performs significantly better than the simple method of following the policy which is optimal with respect to the expected MDP. It is also shown that the performance increases as we make the bound tighter by increasing the number of samples taken.

Compared to results reported for other Bayesian reinforcement learning approaches on the *Chain* task, this rather simple method performs surprisingly well. This can be attributed to the fact that at each stage, the algorithm selects actions according to a nearly-optimal stationary policy.

In addition, MSBI itself could be particularly useful for *inverse* reinforcement learning problems where the underlying dynamics are unknown. Then it would be possible to obtain good stationary policies that take into account the uncertainty over the dynamics, which should be better than using the expected MDP heuristic. In addition, MSBI could be used in the inner loop of some more sophisticated method than MCBRL. For example, it could be employed to obtain tight lower bounds for the leaf nodes of a planning tree. Finally, in future work, we hope to obtain nearly-Bayes performance by considering semi-stationary policies.

## Acknowledgments

Many thanks to Matthijs Snel and Shimon Whiteson for extensive discussions on conditions for the optimality of the MMBI algorithm, which were instrumental in discovering an error, and on its applicability to multi-task problems. This work was partially supported by the EU-Project IM-CLeVeR, FP7-ICT-IP-231722, and the Marie Curie Project ESDEMUU, Grant Number 237816.

# Bibliography


[1] J. Asmuth, L. Li, M. L. Littman, A. Nouri, and D. Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *UAI 2009*, 2009.

[2] Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. In *Proceedings of NIPS 2008*, 2008.

[3] R.I. Brafman and M. Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 3:213–231, 2003. ISSN 1532-4435.

[4] P. Castro and D. Precup. Smarter sampling in model-based bayesian reinforcement learning. *Machine Learning and Knowledge Discovery in Databases*, pages 200–214, 2010.

[5] Richard Dearden, Nir Friedman, and Stuart J. Russell. Bayesian Q-learning. In *AAAI/IAAI*, pages 761–768, 1998. URL `citeseer.ist.psu.edu/dearden98bayesian.html`.

[6] Richard Dearden, Nir Friedman, and David Andre. Model based Bayesian exploration. In Kathryn B. Laskey and Henri Prade, editors, *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 150–159, San Francisco, CA, July 30–August 1 1999. Morgan Kaufmann, San Francisco, CA.

[7] Morris H. DeGroot. *Optimal Statistical Decisions*. John Wiley & Sons, 1970.

[8] Christos Dimitrakakis. Complexity of stochastic branch and bound for belief tree search in Bayesian reinforcement learning. In *2nd international conference on agents and artificial intelligence (ICAART 2010)*, pages 259–264, Valencia, Spain, 2009. ISNTICC, Springer.

[9] Michael O'Gordon Duff. *Optimal Learning Computational Procedures for Bayes-adaptive Markov Decision Processes*. PhD thesis, University of Massachusetts at Amherst, 2002.

[10] Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*, volume 57 of *Monographs on Statistics & Applied Probability*. Chapmann & Hall, November 1993. ISBN 0412042312.

[11] Mahdi Milain Fard and Joelle Pineau. PAC-Bayesian model selection for reinforcement learning. In *NIPS 2010*, 2010.

[12] Thomas Furmston and David Barber. Variational methods for reinforcement learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9 of *JMLR : W&CP*, pages 241–248, Chia Laguna Resort, Sardinia, Italy, 2010.

[13] C. J. Gittins. *Multi-armed Bandit Allocation Indices*. John Wiley & Sons, New Jersey, US, 1989.

[14] Thomas Jacksh, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600, 2010.

[15] Leslie Pack Kaelbling. *Learning in Embedded Systems*. PhD thesis, ept of Computer Science, Stanford, 1990.

[16] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. In *Proc. 15th International Conf. on Machine Learning*, pages 260–268. Morgan Kaufmann, San Francisco, CA, 1998. URL `citeseer.ist.psu.edu/kearns98nearoptimal.html`.

[17] P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement learning. In *ICML 2006*, pages 697–704. ACM Press New York, NY, USA, 2006.

[18] A.L. Strehl and M.L. Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008. ISSN 0022-0000.

[19] A.L. Strehl, L. Li, and M.L. Littman. Reinforcement learning in finite MDPs: PAC analysis. *The Journal of Machine Learning Research*, 10:2413–2444, 2009. ISSN 1532-4435.

[20] Malcolm Strens. A bayesian framework for reinforcement learning. In *ICML 2000*, pages 943–950. Citeseer, 2000.

[21] Tao Wang, Daniel Lizotte, Michael Bowling, and Dale Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *ICML '05*, pages 956–963, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: http://doi.acm.org/10.1145/1102351.1102472.

[22] J. Wyatt. Exploration control in reinforcement learning using optimistic model selection. In A. Danyluk and C. Brodley, editors, *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.